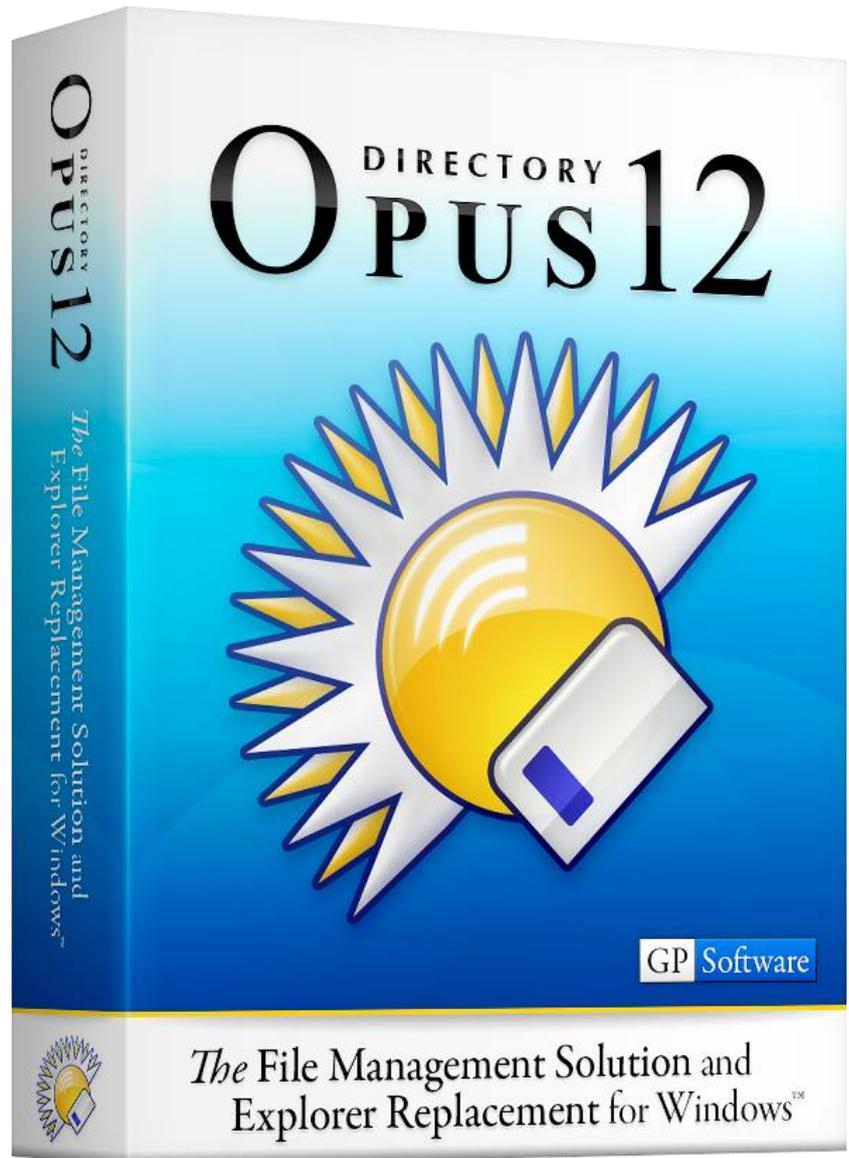


# Reference Manual



(c) GPSoftware, Brisbane 2021

# Contents

<b>Copyright Notice</b>	<b>1</b>
<b>Program License</b>	<b>4</b>
<b>Acknowledgements</b>	<b>6</b>
<b>Introduction</b>	<b>9</b>
Installing and Registering .....	11
Starting Opus .....	16
<b>Basic Concepts</b>	<b>17</b>
The Lister .....	18
Opening a Lister .....	20
Navigation .....	21
Tabs .....	35
View Modes .....	41
Dual Display .....	45
Toolbars .....	47
Find-as-you-type Field .....	57
Viewer Pane .....	61
Metadata Pane .....	65
Utility Panel .....	65
Layouts .....	66
Styles .....	69
Themes .....	71
Calculating Folder Sizes .....	74
Status Bar .....	75
Explorer Replacement .....	77
Source and Destination .....	78
Selecting Files .....	80
Selecting files with the mouse and keyboard .....	80
Simple Wildcard Selection .....	86
Advanced Selection .....	86
Searching and Filtering .....	89
Filter Bar .....	89
Show Everything .....	91
Toolbar Filter Field .....	92
Windows Search .....	94
Find Files .....	95
Sorting and Grouping .....	101
Folder Options .....	107
Folder Options Dialog .....	108
Folder Formats .....	119
Content Types .....	120
Locking the Format .....	121
Identifying the current format .....	122
Flat View .....	124
Virtual File System .....	128
System virtual folders .....	128

File Collections.....	129
Libraries .....	133
Archives .....	135
FTP.....	136
MTP.....	136
Compatibility Files .....	137

## **File Operations 139**

Copying, Moving and Deleting Files.....	140
Copy and Paste .....	140
Drag and drop .....	141
Copying using the toolbar buttons.....	143
Copy Queues .....	146
The Jobs Bar.....	150
The Confirm File Replace Dialog .....	151
Copying Updated Files.....	154
Filtered Operations .....	161
Deleting Files .....	174
Renaming Files .....	177
Inline Rename.....	177
Simple Wildcard Rename.....	178
Advanced Rename.....	179
Creating Folders.....	201
Creating Archives .....	204
Adding to Archives.....	205
Add to Archive Dialog .....	206
Zip Files.....	211
Tracking and Undoing File Operations.....	215
Changing Attributes.....	217
Labels.....	221
Editing Metadata.....	225
Document Properties .....	228
Picture Properties .....	229
Time Shifting.....	230
Music Properties.....	232
Video Properties .....	235
Extended Properties.....	235
Programmatic setting of Metadata.....	236
File Descriptions.....	239
UAC and Administrator Mode .....	242

## **FTP 245**

FTP Address Book.....	246
Default Settings .....	248
Site Page .....	248
Network Page .....	250
Display Page.....	251
Index Page .....	252
Sounds Page .....	253
Misc Page .....	254
Speed Page .....	255
Special Page .....	256
Proxy Page.....	256
Adding a new Site.....	258
FTP Connect .....	260
Site Properties.....	262
FTP Log.....	264

<b>Additional Functionality</b>	<b>267</b>
Viewing Images .....	268
Viewer Keys and Toolbar.....	270
Configurable Toolbar .....	272
Control Bar .....	273
Image Marking .....	274
Playing Sounds .....	276
Image Conversion.....	277
Automated image conversion tasks .....	279
Print Folder.....	280
Duplicate File Finder .....	283
Flickr Synchronization.....	286
Splitting Files.....	291
Joining Files.....	292
Making Links and Junctions .....	293
Floating Toolbars.....	296
Controlling Floating Toolbars .....	297
System-wide Hotkeys .....	299
Exporting to USB .....	301
Update Checker .....	305
CLI.....	308
<b>Preferences</b>	<b>311</b>
Backing up and Restoring Preferences .....	313
Preferences Categories.....	316
Display .....	317
Favorites and Recent .....	328
File Displays.....	335
File Display Modes.....	340
File Operations .....	349
Folders.....	357
Folder Tabs.....	368
Folder Tree .....	374
Internet .....	379
Launching Opus .....	383
Layouts and Styles.....	389
Miscellaneous .....	392
Toolbars.....	415
Viewer .....	422
Zip and Other Archives .....	428
<b>Customize Mode</b>	<b>433</b>
The Customize Dialog .....	435
Commands.....	435
Toolbars.....	444
Keys.....	446
Context Menus .....	449
Creating your own buttons.....	452
Editing the Toolbar.....	452
Command Editor .....	487
User-defined Commands .....	498
Synchronous and Asynchronous functions.....	502
Internal Command Arguments .....	503
Passing files to external programs .....	507
Command modifiers .....	508
MS-DOS Batch commands .....	509
Embedding Rename Scripts.....	512
DDE Functions .....	513

Embedded functions .....	514
<b>File Types</b>	<b>517</b>
Directory Opus File Types .....	520
File Type Groups .....	521
The Open With editor .....	523
File Type Editor .....	526
Actions .....	527
Events .....	529
Context Menu .....	532
Drop Menu .....	536
Replace Menu .....	538
Info Tip .....	540
Tiles Mode .....	543
<b>Scripting</b>	<b>545</b>
Rename Scripts .....	547
Custom Fields in the Rename Dialog .....	549
Script Functions .....	552
Script Dialogs .....	554
Creating Script Dialogs .....	555
Script Dialog Editor .....	556
The Dialog Message Loop .....	570
Reading Dialog Control Values .....	575
Interacting with Dialog Controls .....	577
Script Add-ins .....	581
Script Package .....	582
Script Resources .....	584
String Resources .....	585
Example Scripts .....	588
Example Rename Script .....	588
Simple Script Function .....	589
Adding a new Internal Command .....	590
Adding a new Column .....	593
Adding a new Column from Shell Properties .....	595
Simple Dialogs and Popup Menus .....	596
Script Dialog Example .....	600
Responding to Events .....	603
<b>Reference</b>	<b>607</b>
Wildcard Reference .....	608
Pattern Matching Syntax .....	608
Regular Expression Syntax .....	611
Status Bar Codes .....	616
Codes for file and folder counts .....	616
Codes for disk space .....	620
Codes for music and video duration .....	621
Codes for graphical elements .....	621
Other Codes .....	622
Bar graphs and Percentages .....	633
Hiding sections on the status bar .....	638
Padding sections on the status bar .....	640
Command Reference .....	642
Argument Types .....	642
Internal Commands .....	643
External control codes .....	907
Command modifier reference .....	923
Scripting Reference .....	946

Scripting Objects .....	946
Scripting Events .....	1100
DOpusRT Reference.....	1119
External Manipulation of File Collections .....	1123
Retrieving File and Folder Information.....	1131
Metadata Keywords.....	1134
Keywords for Columns.....	1134
Keywords for SetAttr META .....	1139
Icon Sets .....	1153
Icon Set XML Definition File .....	1153
Icon Sizes .....	1154
Icon Names.....	1154
Icon Display Names .....	1155
Icon Categories.....	1156
DPI aware Icon Sets .....	1157
Localization .....	1159
Icon Images .....	1160
Rename Macro Language .....	1161

## **Release History** **1164**

Welcome to Directory Opus 12!.....	1165
Summary of major new features.....	1165
Rename.....	1167
Image Viewer .....	1176
High DPI support.....	1181
File and folder labels .....	1183
Manual sorting.....	1187
File displays.....	1190
Folder Formats and Folder Options.....	1195
Script dialogs.....	1203
File copying.....	1206
File operations .....	1208
Find.....	1209
Toolbars.....	1211
FAYT / Filter Bar .....	1214
Status Bar .....	1215
Miscellaneous things .....	1216
Script Miscellaneous .....	1219
Reference.....	1222
12.2 16th September, 2016.....	1246
12.3 5th December, 2016.....	1249
12.4 15th March, 2017.....	1257
12.5 27th April, 2017.....	1263
12.6 7th June, 2017.....	1266
12.7 23rd November, 2017 .....	1267
12.8 24th April, 2018.....	1274
12.9 30th May, 2018.....	1284
12.10 3rd October, 2018.....	1286

## **Index** **1299**

# Copyright Notice

Directory Opus 12 for Windows and this manual are Copyright © GPSSoftware, Brisbane 2001-2018. All rights reserved. Title, ownership rights and intellectual property rights in and to the SOFTWARE shall at all times remain the property of GPSSoftware.

No part of this publication or the accompanying SOFTWARE may be copied or distributed, transmitted, transcribed, stored in a retrieval system or translated into any human or computer language, in any form or by any means, electronic, mechanical, magnetic, manual or otherwise, or disclosed to any third-party without the express written permission of GPSSoftware, Australia.

## Disclaimer

GPSSoftware and any associated dealers or distributors make no representation or warranties with respect to the performance of the SOFTWARE nor of the contents of this publication and specifically disclaim any implied warranties of merchandisable quality or fitness for any particular purpose. Further, GPSSoftware reserves the right to revise the SOFTWARE and this publication and to make changes to them from time to time without obligation of GPSSoftware to notify any person or organization of such revisions or changes.

Illustrations in this publication are intended to be representations and may not be exact duplicates of the screen layouts generated by the SOFTWARE.

## License

GPSSoftware provides this program and any updates to the original purchaser under the terms set out below and licenses its use worldwide. You assume responsibility for the selection of the program to achieve your intended results, and for the installation, use and results obtained from the program.

## Warranty

All care has been taken to ensure that the program performs the functions as set out in this manual. However, GPSSoftware provides the program “AS IS” and makes no express or implied warranties with respect to the SOFTWARE, its documentation, performance, fitness for a particular purpose, or merchantability. The entire risk as to the quality and performance of the SOFTWARE is borne by you. Should the SOFTWARE prove defective, you and not GPSSoftware shall assume the entire cost of any service and/or repair.

To the maximum extent permitted by applicable law, in no event shall GPSSoftware, its agents or suppliers be liable for any special, incidental, indirect, or consequential damages whatsoever (including, without limitation, damages for loss of business profits, business interruption, loss of business information, loss of data, or any other pecuniary loss) arising out of the use of or

inability to use the SOFTWARE product, even if GPSoftware has been advised of the possibility of such damages.

At the discretion of GPSoftware, any and all warranty or technical support is available only to the original purchaser of this package. Please contact your local GPSoftware distributor in your local country for Technical Support in the first instance.



# Program License

Directory Opus 12 is supplied under the following license conditions, is valid for the original purchaser only and is not transferable. You accept responsibility for any and all Directory Opus Program Certificates issued by GP Software to you.

## Directory Opus Light Version License

The Opus Light version license is a single-installation license which allows the installation of the SOFTWARE on one physical machine only, for exclusive use on that machine only. Installation on more than one machine requires a separate license and license certificate to be purchased for each machine.

*Lifetime priority support is not available* for the Opus Light version. Depending on the exact nature of the license purchased, the license grants you the right to limited technical support via the Directory Opus Resource Centre, bug fixes and updates but not upgrades to the SOFTWARE for the duration of this license.

## Directory Opus Professional Version License

The Opus Professional version license is available for single or multiple installations under the one license. The SOFTWARE may only be installed on the number of machines specified by the program license and associated certificate. Additionally, you may install the SOFTWARE on ONE personal laptop, irrespective of the number of installs allowed by your license. For example, this means that if you have purchased a two install license you may install the SOFTWARE on two computers of ANY type PLUS ONE personal laptop.

The standard license is a single-installation license which allows the installation of the SOFTWARE on one physical machine only, for exclusive use on that machine only, plus one personal laptop owned by and in exclusive use by the registered purchaser. Installation on more than one machine (not including one personal laptop) requires a multiple license to be purchased to include each extra machine. *(NOTE: The free extra laptop license applies for domestic use only and does not apply for Business usage or for GPSoftware Open License Plan licensees.)*

*Lifetime priority support via the Directory Opus Resource Centre is included* as part of the Opus Professional license. Depending on the exact nature of the license purchased, the license grants you the right to priority technical support, bug fixes and rights to updates and enhancements including discounted upgrades to the SOFTWARE for the duration of this license.

## General

You may use the SOFTWARE on the licensed computer(s) at a specific site only (excluding laptops), unless a professional multiple site license has been negotiated at time of purchase from GPSoftware. Where the SOFTWARE is executed from a common disk shared by multiple CPU's, you must ensure that one authorized copy of the SOFTWARE has been licensed from GPSoftware or authorized agent or resellers for each USER / CPU executing the SOFTWARE.

On a machine for which you have a license, you may install the software on multiple operating systems or virtual machines for use by one user at the same time.

You may not redistribute this SOFTWARE and/or any accompanying serial numbers or registration certificates to any person, organization or third party without the prior written consent of GPSoftware.

**YOU MAY NOT USE, COPY, OR TRANSFER THE PROGRAM OR MANUAL, OR A COPY, IN WHOLE OR IN PART, EXCEPT AS EXPRESSLY PROVIDED. IF YOU TRANSFER POSSESSION OF ANY COPY OF THE PROGRAM TO ANOTHER PARTY, YOUR LICENCE IS AUTOMATICALLY TERMINATED. YOU MAY NOT HIRE OR LEASE THE SOFTWARE TO A THIRD PARTY.**

### **Advanced FTP Subsidiary License** (*Opus Professional Version Only*)

A subsidiary license must be purchased to activate the Advanced FTP functionality in the SOFTWARE. This is issued on a per program license basis and subsidiary licenses must be purchased for all machines in a specific program license. For example, a dual software license requires the purchased of a dual FTP subsidiary license.

### **USB Export Subsidiary License**

A subsidiary license must be purchased to activate the USB Export feature in the SOFTWARE. This license entitles you to install the SOFTWARE on ONE USB Device irrespective of the number of install licenses purchased. Once installed on a USB device the SOFTWARE may be executed from that specific USB device only, on any computer.

### **Limited Term Evaluation License**

If you are using an evaluation version you may install and trial the program for a period of 30 days (either version) or for a maximum of 60 days (Professional version) if an evaluation license and certificate have been granted from the GPSoftware web site. After this time you must either purchase a valid license, or cease using the SOFTWARE and uninstall any and all copies of the SOFTWARE and any related material from your computer.

### **Termination**

This license is effective unless terminated. This license may be terminated immediately without notice from GPSoftware if you fail to comply with any provision of this license. Upon knowledge or notification of termination you must immediately uninstall and destroy the SOFTWARE and any and all associated materials and all copies thereof, and any and all Program License Certificates that have been issued to you. The licensee may terminate the license at any time by destroying the SOFTWARE and all copies thereof. Unauthorized distribution of an issued program certificate immediately voids your license and results in forfeiture of all your rights to continue to use the SOFTWARE.

# Acknowledgements

Directory Opus 12 for Windows and this manual are Copyright© GPSoftware, Brisbane, 2001-2018. All rights reserved.

The software was written by Jonathan Potter, Leo Davidson and Dr Greg Perry, in 100% C++/Win32 using Microsoft Visual Studio.

Cris van Minnen and Trevor Morris designed and produced most of the icons and images.

This manual was written by Jonathan Potter.

GPSoftware extends its grateful thanks to the many people who have encouraged and provided assistance and feedback in the development and extensive beta testing of this software.

Opus® and Directory Opus® are registered trademarks of **GPSoftware / Redbrook Pty Ltd**. The Opus logo is a registered trademark of GPSoftware. DOpus™, DirOpus™ and Opus File Manager™ are trademarked 1991 by GPSoftware; Opus Magellan™ is trademarked 1996 GPSoftware; OpusPC™, PCOpus™, Smart Favorites™ are trademarked 1998 by GPSoftware. Flat View™ is trademarked by GPSoftware 2003. Opus6™, Opus8™, Opus9™, Opus10™, Opus11™ and Opus12™ are trademarked by GPSoftware 2000-2016.

Microsoft® Windows, Windows XP, Windows Vista, Windows 7, Windows 8, Windows 10 and other related terms are copyright© or registered® trademarks of Microsoft Corporation.

Flickr is a trademark and copyright of Yahoo Inc.

Photoshop is a trademark of Adobe Systems, Inc.

Any other products mentioned in this manual are trademarks or copyright of their respective owners.

Directory Opus makes use of several third-party libraries; acknowledgement is hereby given for these.

- brainchild: Copyright© 1993-2005 by Jan van den Baard, included with permission from the author as of August 2003.
- exiv2: Copyright© 2004-2014 Andreas Huggel, used under licence from the author.
- jpeg: This software is based in part on the work of the Independent JPEG Group.
- lcms: Copyright© Marti Maria Saguer. Little CMS engine is provided free of charge under the [MIT LICENSE](#)

- libpng: Copyright© 1998-2012 Glenn Randers-Pehrson, Copyright© 1996, 1997 Andreas Dilger, Copyright© 1995, 1996 Guy Eric Schalnat, Group 42, Inc.
- libtiff: Copyright© 1988-1997 Sam Leffler, Copyright© 1991-1997 Silicon Graphics, Inc. Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that (i) the above copyright notices and this permission notice appear in all copies of the software and related documentation, and (ii) the names of Sam Leffler and Silicon Graphics may not be used in any advertising or publicity relating to the software without the specific, prior written permission of Sam Leffler and Silicon Graphics.
- openssl: Copyright© 1998-2014 The Open SSL Project. This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (<http://www.openssl.org/>). All rights reserved.
- putty: Copyright© 1997-2016 Simon Tatham. Portions copyright Robert de Bath, Joris van Rantwijk, Delian Delchev, Andreas Schultz, Jeroen Massar, Wez Furlong, Nicolas Barry, Justin Bradford, Ben Harris, Malcolm Smith, Ahmad Khalifa, Markus Kuhn, and CORE SDI S.A.
- Scintilla: Copyright© 1998-2016 by Neil Hodgson <neilh@scintilla.org>, All Rights Reserved.
- taglib: Copyright© Scott Wheeler. TagLib is distributed under the [GNU Lesser General Public License](#) (LGPL) and [Mozilla Public License](#) (MPL).
- tinyxml: Copyright© Lee Thomason. TinyXML is distributed under the [zlib/libpng License](#).
- xmp: Copyright© Adobe Systems, Inc. XMP is a trademark of Adobe Systems, Inc.
- 7-Zip: Copyright© 1999-2016 Igor Pavlov. 7-Zip is distributed under the [GNU LGPL licence](#).
- zlib: Copyright© 1995-1998 Jean-loup Gailly and Mark Adler.

MIT licence (applies to lcms and putty):

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. *THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL SIMON TATHAM BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.*

Scintilla licence (applies to Scintilla):

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation. *NEIL HODGSON DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL NEIL HODGSON BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.*

libwebp:

Copyright (c) 2010, Google Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of Google nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# Introduction

Welcome to Directory Opus, *the* file manager for Windows!

Directory Opus is designed with four goals in mind:

- **Ease of use:** As far as possible, Opus (in its default configuration) works just like Explorer does. You don't have to learn any complicated scripting or non-standard mouse techniques to use Opus. If you've ever used Explorer to copy a file, you already know exactly how to do it in Opus as well.
- **Configurability:** We believe in the user's right to choose how their computer operates. You'll find that almost every aspect of Opus can be changed - from the buttons on the toolbar to the color used to draw the background of a compressed file. Of course you don't have to configure anything if you don't want to - "out of the box" Opus provides a comprehensive set of commands that will let you perform most file management tasks without ever going near the configuration.
- **Efficiency:** Opus is designed to be as efficient as possible. The entire program makes use of multi-threading to ensure that you should never have to wait for one operation to complete before beginning another.
- **Compatibility:** As an Explorer Replacement it's important that Opus appears (to the system) just like Explorer does. Within the limits set by Microsoft, Opus achieves this and most software written with only Explorer in mind should still work fine with Opus installed.

There's no denying that Opus is a complex product. From its roots on the Amiga computer in 1989, through two platforms and twelve major revisions, the program has grown as we have taken on board thousands of suggestions from our users. As you start to explore Opus you may feel daunted by the sheer number of options available - please don't be! Don't look on Opus as something that has to be learned, but as a tool that you can exploit.

We class Opus as "productivity software" and some people question this description, but really if you think about it - for anyone who uses their computer a significant amount, a large proportion of that time is spent in mundane tasks like moving files around, making folders, renaming files, cleaning out directories, etc. Using Opus you can significantly streamline your day-to-day tasks - and less time spent on the mundane means more time to be productive!

As you start to explore Opus here are a few tips to help you get around:

- All the toolbar buttons and menu commands have popup tooltips that describe what they do. If you ever want to know what a command is for, just hover over it with the mouse for a couple of seconds.
- The Preferences and Customize dialogs (the two dialogs where most configuration options can be found) both have filter fields at the bottom, that let you search for options matching your keywords.
- All the dialogs have context-sensitive help. Some dialogs (like Preferences) have a help button in the top-right corner, but in all other dialogs you can press the **F1** key to display the help.

- If you have time, have a quick read through the topics in the [Basic Concepts](#) section of this help file. Opus supports the same basic file management concepts as Explorer but this section describes some of the extra functionality that can really help you get the most out of Opus.
- The [Opus Resource Centre](#) is a wealth of hints, tips, FAQs, tutorials and knowledgeable Opus users.

Thanks for using Directory Opus! We hope you enjoy using it as much as we enjoy writing it!

# Installing and Registering

Like many programs, Directory Opus uses the industry-standard InstallShield installer to manage installation. Begin the installation by double-clicking the setup program (normally called *DOpusInstall.exe*).

The installer is also an updater - it can install Opus on a new machine or it can install a new version over an existing version. (When updating, do *not* uninstall the old version first, unless you wish to reset your configuration to the factory defaults.) The same installer works with both 32-bit (x86) and 64-bit (x64) Windows, automatically selecting the appropriate components for your computer. The installer also works for both Light and Pro editions of Opus and for both evaluation and purchased licences. If you get a new licence, you only need to install the licence itself (using the Licence Manager - see below) and do not need to re-install the whole program.

Once the installation process is complete, you will be given the option to start Opus immediately - the installer will also place a shortcut icon on your desktop that you can use to launch Opus. When you run Opus for the first time, the **Directory Opus Initialisation Wizard** will ask you a few questions that help define your initial configuration. We recommend that you choose the default settings but of course you don't have to, and all these settings can be modified later through the program's Preferences dialog.



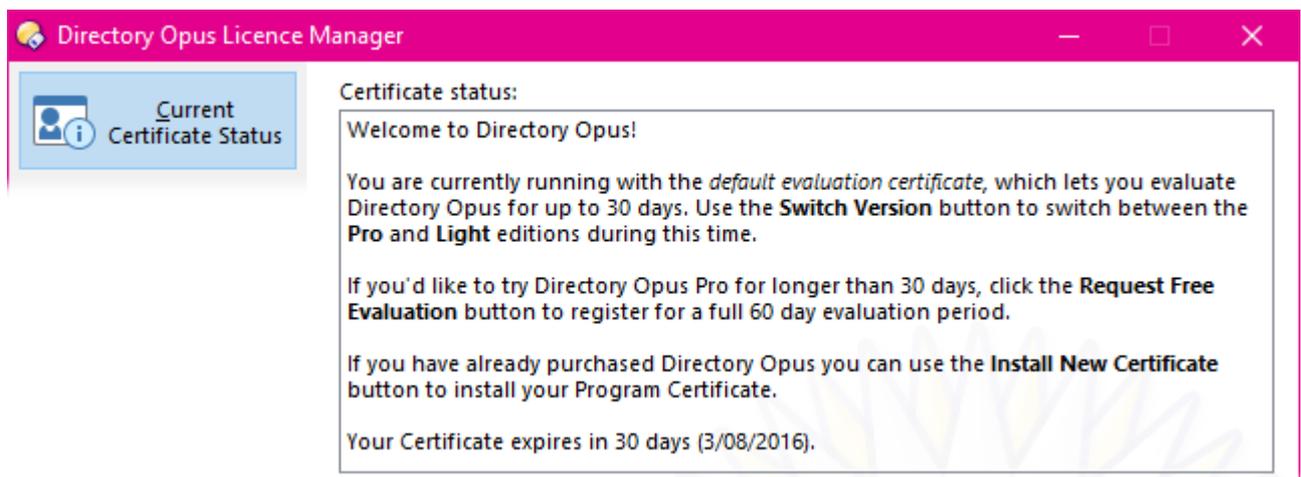
There are seven choices you must make in this wizard; one per page:

- **Language:** Directory Opus normally defaults to English but you can switch to another language here. You can also change language later on from the [Display / Language](#) page in Preferences.
- **Edition:** Directory Opus comes in two editions - Pro and Light. During your initial evaluation period you can choose to trial Opus in either "mode". You can switch edition later on using the **Licence Manager** (see below). See the [GP Software](#) website for more information on the differences between the editions.
- **Launch On Startup:** Select this option to have Directory Opus run automatically on Windows startup. This is recommended mainly because access to Opus is much quicker if it's already in memory - opening a new window via double-click on the desktop or a hotkey like **Windows+E** will be almost instantaneous if Opus is already running in the background. You can modify this option at any time from the [Launching Opus / Startup](#) page in Preferences.
- **Explorer Replacement:** Select this option if you want Directory Opus to replace Explorer. If you turn this on then double-clicking on a folder on the desktop, or performing any other action that normally opens an

Explorer window, will cause Opus to open instead. See the page on [Explorer Replacement](#) mode for more information. You can modify this option at any time from the [Launching Opus / Explorer Replacement](#) page in Preferences.

- **FTP Handling:** Select this option if you want Opus to register itself as the default FTP handler. If this is selected then clicking on an **ftp://** link will cause the FTP site to open in an Opus Lister rather than in your web browser or other FTP client. You can modify this option at any time using the [Miscellaneous / Windows Integration / Make Directory Opus the default handler for FTP sites](#) option in Preferences.
- **ZIP Handling:** If this option is selected then Opus will register itself as the default handler for ZIP files. Double-clicking on a **.zip** file will cause an Opus Lister to open showing the contents of the archive (and you can then view, extract, add, delete, etc. files within the archive using Opus). You can modify this option at any time using the [Zip & Other Archives / Zip Files / Make Opus the system default handler for Zip files](#) option in Preferences.
- **Picture & Sound Double-Click Handling:** If you enable this option, then double-clicking on an image or sound file (that Opus recognizes) in Opus will cause the internal viewer or player to be used rather than the normal default handler for that file type. This option doesn't modify any registry settings and doesn't actually change the default handlers for these file types in the system - instead, it enables an override that's only effective inside of Opus itself. You can modify these options at any time from the [File Operations / Double-click on Files](#) page in Preferences.

Directory Opus uses a certificate-based licencing system. When you purchase Opus, or register for a 60 day evaluation, you will receive a small text file that is your program certificate. This contains (in encoded form) data about your Opus registration, including enabled features, number of installs allowed and in the case of an evaluation licence, the expiry date. To activate your copy of Directory Opus you need to install the certificate using the program's **Licence Manager** function. The Licence Manager window will open by itself a short time after you run the program, but you can also invoke it at any time from the **Help** menu.



Directory Opus comes with a special program certificate called the **Default Evaluation Certificate**. This certificate lets you evaluate Directory Opus for up to 30 days from the date you first run the program. During this evaluation period you can switch back and forth between the Pro and Light editions using the **Switch Version** command in the Licence Manager.

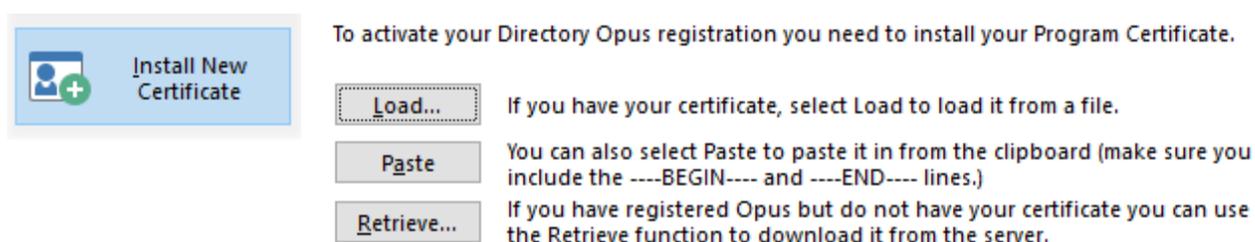
If you'd like to trial the Pro edition for longer than the initial 30 days, you can register for a free 60-day evaluation by clicking the **Request Free Evaluation** button in the Licence Manager.



The image shows a software interface. On the left, there is a button labeled "Request Free Evaluation" with a user icon and a plus sign. To the right, there is a blue header bar labeled "User Details". Below this header, there are three input fields: "Name:", "Email Address:", and "Confirm Email:". Each field is represented by a rectangular text box.

Enter your name and email address, and click the **Register** button and Opus will automatically contact our servers over the Internet and download an evaluation certificate for you. Alternatively, you can request an evaluation certificate through our [website](#).

If you have purchased Directory Opus, or registered through the website for an evaluation, and you have received your certificate via email, simply double-click on the attached certificate file to install it. Alternatively, use the **Install New Certificate** button in the Licence Manager.



The image shows a software interface. On the left, there is a button labeled "Install New Certificate" with a user icon and a plus sign. To the right, there is a text instruction: "To activate your Directory Opus registration you need to install your Program Certificate." Below this instruction, there are three buttons: "Load...", "Paste", and "Retrieve...". Each button is accompanied by a short instruction: "Load..." is for loading from a file, "Paste" is for pasting from the clipboard, and "Retrieve..." is for downloading from the server.

Use the **Load** button if you have your certificate stored on disk as a **.txt** or **.opuscert** file. You can also copy it to the clipboard, and then click the **Paste** button to paste it into the Licence Manager.

The **Retrieve** button can be used if you have previously purchased Opus and wish to install it on a new machine - by providing your registration code or product token and registered email address you can retrieve your certificate from our servers over the Internet. You can also retrieve a lost registration code and certificate through our [website](#).

The final page in the Licence Manager is the **Register Product Token** page. Product tokens are issued by our resellers, and you can use this page to register a product token to receive your full program certificate. You can also register a product token through our [website](#).



# Starting Opus

There are a two main ways you can run Directory Opus once it is installed. We recommend that you set Opus to start automatically when Windows starts, using the options in the [Launching Opus / Startup](#) page in Preferences. You can also specify whether Opus opens any Lister or not when it starts up. Opus is designed to stay running in the background, even when no Listers are open. There are several advantages to this:

- Opening a Lister is much quicker if Opus is already loaded in the background. Double-clicking on the desktop or (if [Explorer Replacement](#) is enabled) pressing **Windows+E** or double-clicking on a folder icon will cause a Lister to open almost instantly if Opus is already running - whereas there may be a delay of several seconds if Opus is not already in memory.
- Opus lets you create "[floating toolbars](#)" that you can use as command launchers - these can only be used when Opus is running.
- It is possible to configure [system-wide hotkeys](#) that run programs or initiate Opus functions - again, these only work when Opus is running.

You can also treat Opus like a traditional file manager, that you run when you need it and exit when not using it. The installer places shortcut icons on the desktop and in the start menu, and you can use these icons to run Opus - under Windows 7 and higher you can also pin the icon to the taskbar and launch it that way. To quit Opus when you're done using it, select the **Exit Directory Opus** command from the **File** menu in a Lister, or right-click on the taskbar notification icon (if enabled) and choose the exit command from that menu. You can also set Opus to exit automatically when the last Lister is closed, with the **Launching Opus / Startup / Shutdown Directory Opus when the last Lister closes Preferences** option. With this option on, and Opus not set to run on startup, it will behave like a traditional, monolithic file manager.

# Basic Concepts

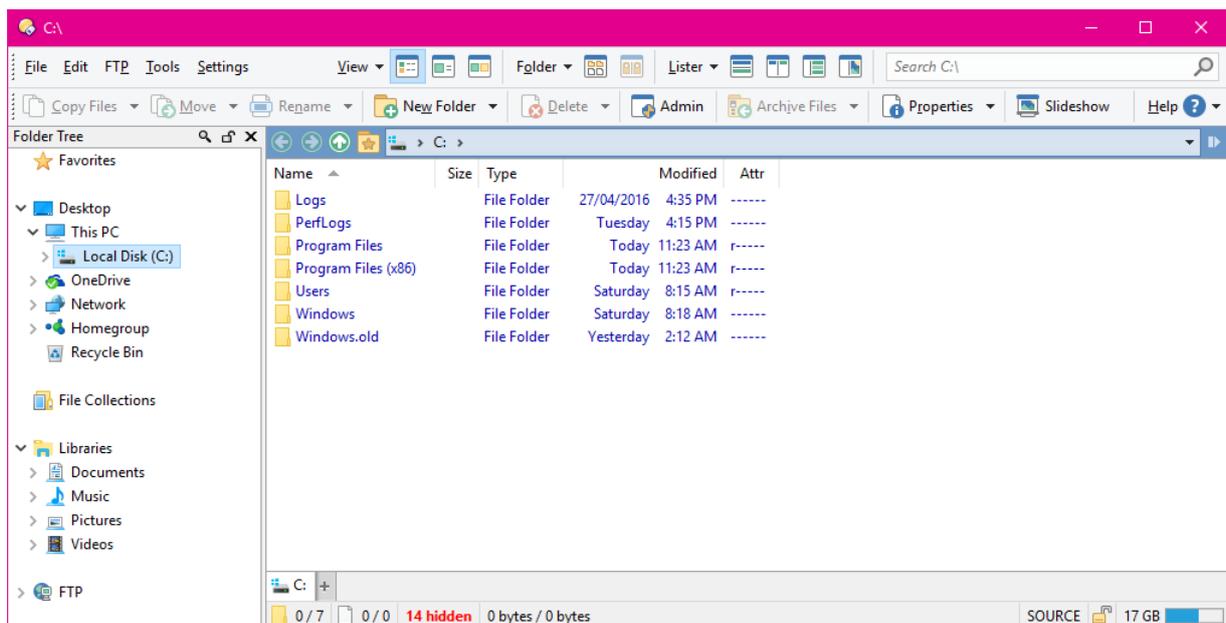
There are a number of concepts in Opus that may be different from other file managers you have used. While we try to avoid the use of jargon or geeky terminology there are some terms used in this help file that you may not be that familiar with; we suggest having a quick flip through the Basic Concepts section to familiarize yourself with some of these concepts.

- [The Lister](#): Listers are what you might call the Opus "main window" - in a traditional file manager, they would be the main window. Listers (the name is historic) contain file displays (where the contents of directories are shown), folder trees (presents your file system as a tree structure for easy navigation), toolbars, and various optional "panes" like the viewer pane, the metadata pane, etc. You can open as many Listers as you like.
- [Explorer Replacement](#): Opus has the option to operate in Explorer Replacement mode. In this mode, most actions that normally result in the opening of an Explorer window (pressing the Windows+E key, double-clicking on an icon on the desktop, etc.) will instead open an Opus Lister. This is the mode we recommend you run in.
- [Source and Destination](#): As well as the traditional copy/cut/paste method of file management you're probably used to, Opus provides an alternative method that uses the concept of "source" and "destination" folders. Instead of browsing to the files you want to copy, copying them to the clipboard, then browsing to the destination folder and pasting them in, this alternate method lets you display both the source and destination folders at the same time (either using two separate Listers, or with a single Lister in dual file display mode) and copy files from one to the other in a single action.
- [Selecting Files](#): There are a number of ways to select files for operations; other than the standard methods using the keyboard or mouse, you can select using a wildcard match on the filename, or use a filter to select files by attributes or metadata. Opus also supports a checkbox mode which lets you manipulate files (by double-clicking or drag-and-drop) without affecting their selection state.
- [Searching and Filtering](#): Searching is the process of locating files or folders that may not be in the currently viewed location - Opus supports Windows Search for indexed search, as well as its own powerful search engine that lets you build complex queries to find files based on attributes and metadata as well as filenames. Filtering is the process of hiding, or masking out, files and folders from the currently viewed location. Opus provides a number of ways to do this - the easiest to use is the Filter Bar, which lets you quickly show a sub-set of files by simply typing a wildcard pattern into the file display.
- [Sorting and Grouping](#): The file list can be sorted by a single field or by multiple fields, and you can also group the file list by any field with collapsible groups.
- [Folder Options](#): This is a powerful system that lets you control exactly how your folders will appear - you can define the view mode, sorting and grouping options, etc, and permanently assign them to a folder, a folder and its children, a "type" of folder (based either on the disk type or the contents of the folder), or multiple folders using wildcards.
- [Flat View](#): The Flat View system lets you "collapse" the contents of all the child folders of the current location, and make them appear as if they are all in the same physical location. Flat View can show a truly flat list of all sub-folders, or it can display the contents of sub-folders in a hierarchical structure.
- [Virtual File System](#): The real file-system is where you store your files and folders - generally contained on your hard drives, USB drives, etc. The virtual file-system is a concept that Opus uses to describe file-systems that aren't stored in this traditional manner. For example, File Collections are a virtual file-system because they are a collection of files stored on traditional media, rather than the underlying folders themselves. There are a number of concepts that are specific to particular virtual file systems - File Collections, Libraries, Archives and FTP are all described in this section.

# The Lister

Listers are what you might call the Opus "main window" - in a traditional file manager, they would be the main (or only) window. The name Lister derives from the original Opus on the Amiga computer (20 years ago!). Listers were much simpler then, doing nothing much more than displaying a list of files and folders - and what do you call something that *lists* things? A *lister*!

These days the Opus Lister contains multiple configurable user-interface elements that let you access various program functionality. At their core though they still display lists of files and folders, and indeed this is the only functionality of a Lister that can't be disabled.

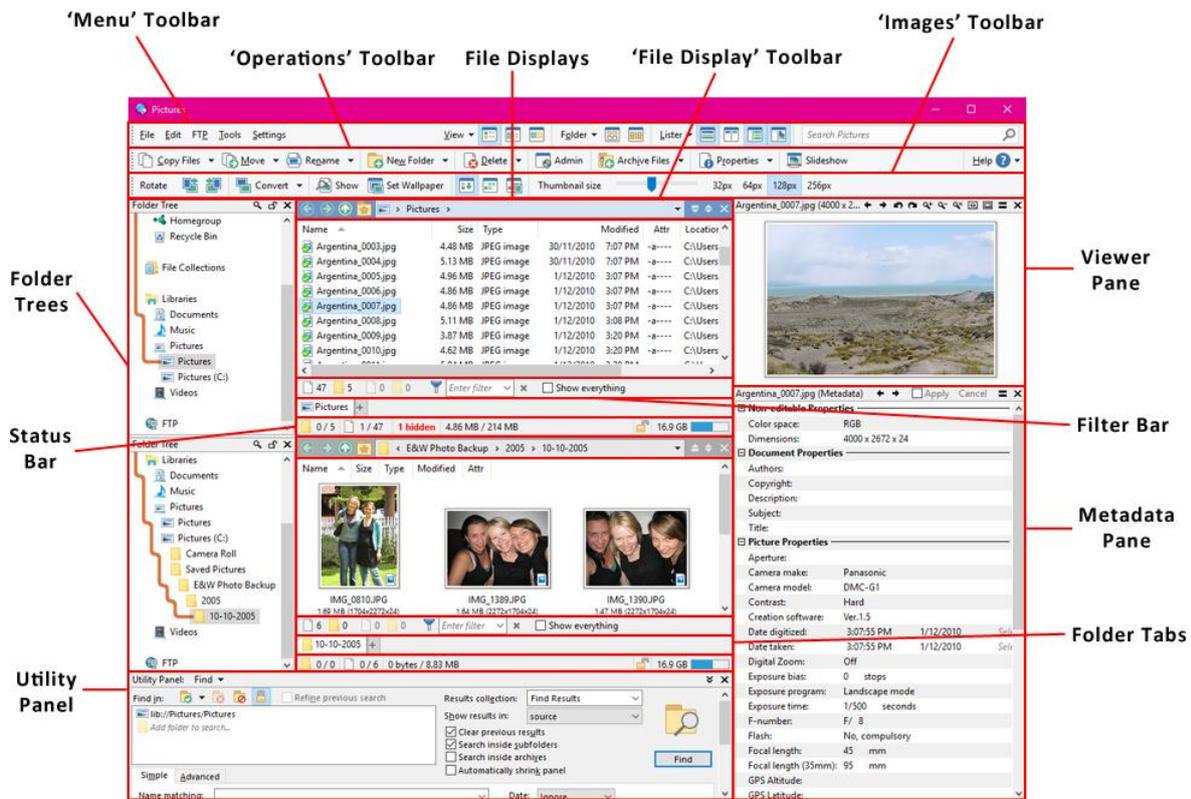


Above is a screenshot of a fairly simple Lister using the default Opus configuration. The two default toolbars are visible at the top of the Lister, then below this are the *folder tree* (on the left), and the *file display* on the right. The file display has a third toolbar at the top of it. The folder tree indicates that drive C: has been selected and indeed, you can see that the contents of drive C: are shown in the file display to the right of the tree. Note the *folder tab* displayed at the bottom of the file display - currently, only one tab is open (the one showing C:) but you can open as many folders as you like each in their own tab, and rapidly switch between them.

At the bottom of the file display is the *status bar*, which displays some summary information about the folder shown in the file display. The information displayed on the status bar can be configured through the Preferences system.

You might think from that image that a Lister is basically the same as Explorer - in which case, what's the point of Opus? In fact, Opus has been deliberately designed to resemble Explorer in its default configuration. This provides a sense of familiarity for new users first switching from

Explorer - if you've used Explorer before you can instantly start using Opus in the same way, and learn more about its additional functionality and configurability as you go.



This screenshot illustrates a much more complex Lister, and immediately you can see that the resemblance to Explorer is only superficial. The key user interface elements visible in this screenshot are:

- File Displays:** The core element of the Lister, the file display lets you see the contents of the current folder. You can choose how files and folders are arranged (both [sorting and grouping](#) can be configured), and you can use [filters](#) to hide or show certain files or types of files. Most actions (like copying / deleting files, creating folders, adding to archives, etc) take place within the file displays. Listers can display either one or two file displays simultaneously, and each file display can have one or more [tabs](#), each of which can display the contents of a different folder. Each file display has a configurable toolbar displayed above it that shows the current location and provides basic navigation buttons. There are a number of options in Preferences that affect the appearance and behaviour of file displays (for example, you can have grid lines shown, enable full-row selection mode, etc). See the [File Displays](#) and [File Display Modes](#) categories in Preferences for more information.
- Default Toolbars:** The Menu and Operations toolbars are the [default set of toolbars](#) built-in to Opus. These toolbars can be turned on or off, and edited, but they can't be deleted or renamed - and you can easily reset them to their defaults at any time. Also shown above is the default Images toolbar, which is displayed whenever a file display is set to thumbnails mode. Of course you can [create your own toolbars](#) with any combination of buttons that you like - many people turn off the default toolbars as soon as they install Opus and never look at them again!
- Folder Trees:** The [folder tree](#) displays the folder hierarchy that leads to the currently selected folder. You can turn the folder trees on or off, and in a dual display Lister, you can choose to have a separate folder tree for each file display, or a single folder tree that is shared between them. Folder Trees can be both a navigational tool (clicking on a new folder in the tree will display its contents in the file display), and as a

target for file operations (e.g. you can drag a file from the file display and drop it on another folder in the tree to copy or move it).

- **Viewer Pane:** The [Viewer pane](#) displays the contents of the selected file. Opus supports dozens of different image and movie file formats natively, and also has a viewer plugin system that lets third-party developers write viewers for unsupported file types. Additionally, Opus ships with a viewer plugin that leverages ActiveX technology to display files like Office and PDF documents.
- **Utility Panel:** This is a multi-facted panel that provides access to several [utility features](#), including [Find](#) (search for files and folders, based on definable criteria), [Synchronize](#) (synchronize the contents of one drive or folder, including remote FTP sites, with another) and [Duplicate Files](#) (search for any files that are duplicated, or for duplicates of specific files). Additionally, the Utility Panel is used to display several logs including the file operations log, the FTP activity log and script output.
- **Metadata Pane:** This pane lets you view and edit the [metadata](#) for the selected file or files. Many different image, music and document file types are supported.
- **Status Bar:** Displays statistics on the current folder, including total number of files, files hidden by filters, free space on the drive and more. You can [configure](#) what information is displayed through Preferences.
- **Filter Bar:** Provides quick access to filename filters, letting you [control which files are shown or hidden](#) in the folder.
- **Folder Tabs:** Using [Folder Tabs](#) you can open multiple folders at the same time and easily switch between them.

## Opening a Lister

There are many different ways to open a Lister (or Listers), including:

- When you start Opus from the desktop icon, or the icon on the Windows 7 taskbar, a Lister will open
- If [Explorer Replacement](#) mode is enabled, double-clicking a folder on the desktop, or pressing the **Windows+E** hotkey (even when Opus isn't running)
- If [enabled in Preferences](#), you can double-click on an empty spot on the desktop (even when Opus isn't running)
- If [enabled in Preferences](#), you can double-click on the system tray icon that Opus adds when it's running
- You can set Opus to [run on startup](#) and automatically show a Lister when it does
- You can load a saved [Lister Layout](#) from the desktop context menu or the **Settings** menu in an existing Lister
- You can use one of the [internal commands](#) (like **Go NEW**) to open a Lister from a button or a hotkey (**Windows+Shift+E** is assigned by default)

How you open your Listers really depends on you, and on how you want to work with Opus. You can configure Opus to always open a Lister when it starts up, keeping one open all the time - or to only open one when you need it. You can work with as many or as few Listers open at once as you like.

## Navigation

There are many ways you can navigate (that is, move from one folder to another) in a Lister, including:

- You can double-click a folder in the current location to enter it. You can also enter a folder by right-clicking on it and choosing **Open** from the context menu, or from the keyboard by using the cursor keys to select it and then pressing the **Enter** key.
- [The Up, Forward and Back buttons](#): The default **File Display** toolbar contains buttons that can move you up in the folder hierarchy, back to the previous folder or forward to the next folder.
- [The Folder Tree](#): You can change folder by simply clicking on a new folder in the tree (you can also navigate via the tree using the keyboard cursor keys).
- [File Display border](#): If the **File Display** toolbar has been disabled, a smaller border is displayed in dual-display mode which shows the current path and provides some default navigation buttons.
- [The Breadcrumbs location field](#): The default **File Display** toolbar contains a field known as the 'breadcrumbs location field'. This displays your current location as "breadcrumbs" that indicate the path from the desktop to your current folder. Each "crumb" in the path is active and can be clicked or expanded. You can also type into this bar in order to navigate to a folder using the keyboard.
- [Favorites](#): You can add folders to a list of your favorite locations - navigating to these folders is simply a matter of selecting them from the drop-down menu. Access the favorites from the drop-down on the default **File Display** toolbar or from the tree.
- [SmartFavorites](#): If enabled, the **SmartFavorites** system attempts to learn your most commonly used locations automatically. A list of these locations is then displayed in the drop-down Favorites menu.
- [Recent and History Lists](#): Opus maintains two lists of your recently used locations, that you can use to instantly go back to a folder you were recently in. One (the **Recent** list) is global to the program, and the other (the **History** list) is local to a file display.
- [Aliases](#): You can define aliases that reference a folder by a simple name - you can then navigate to an aliased folder by typing its name into the location field.

### ***Up, Forward, Back***

It's common to use a directional nomenclature when discussing navigation of the file system. As you would know, the file system is presented as a hierarchy of folders. For example, consider the following path:

```
C:\FolderOne\FolderTwo\FolderThree
```

In this path, *C:\* is the root folder, *FolderOne* is a child of *C:\*, *FolderTwo* is a child of *FolderOne*, and *FolderThree* is a child of *FolderTwo*. Another way of representing this hierarchy would be:

```
C:
```

FolderOne

FolderTwo

FolderThree

If you start in *C:* and then double-click on *FolderOne* to go into it, you can be said to have gone **down** in the folder hierarchy. It therefore follows that going from *FolderOne* to *C:* is moving **up** in the hierarchy. And in fact, that's what the **Up** function does - it navigates you from the current folder to the current folder's parent. From any folder you can go up repeatedly, until you reach the **Desktop**, which is the root of the Windows file system. There are no folders higher up in the hierarchy than the **Desktop** folder, and so the **Desktop** is the only\* place you can't go up from.

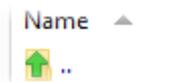
There are a number of ways in a Lister to go **Up**. The easiest is to click the **Up** button on the default File Display toolbar:



If the File Display toolbar has been disabled, there is also an **Up** button on the [file display border](#):



In the file displays, you can optionally choose to show a special **'..'** folder entry that corresponds to the parent folder. [If this is turned on](#), double-clicking it will navigate up to the parent folder.



You can also go **Up** using the keyboard - pressing the **Backspace** key in a Lister will navigate you to the parent folder.

*\* This is not strictly true in Opus; it is possible to have folder hierarchies that don't start under the Desktop. For example, File Collections and FTP can both be displayed in the tree with their own root items.*

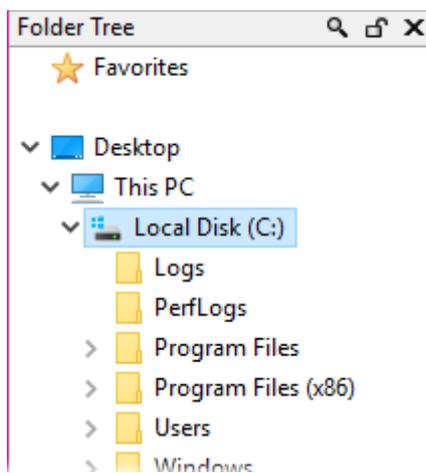
As you move from folder to folder in a file display, a list of the locations you have visited is remembered. This is called the **History** list. A **History** list is kept for each file display (and if multiple tabs are open, for each tab in the file display). You can use the **Back** and **Forward** buttons to move within the **History** List:



If you have a keyboard or mouse with a **Back** or **Forward** button on it, Opus will also recognise these and perform the appropriate action. Click the drop-down arrow to the right of the back and forward buttons displays the history list (the part of it that is either before or after your current location) in a drop-down menu.

## Folder Tree

The Folder Tree is a common user-interface both for navigation and representing the folder hierarchy of the file system.



The folder tree, while appearing at first as a simple list of folders, is actually quite a versatile control:

- **Navigation:** Navigating using the tree is as simple as clicking on a folder; the file display attached to that tree will instantly change location to display the contents of the folder you clicked on. If the selected folder is already open in another [folder tab](#), and the **Switch to existing tab if already open** option is enabled on the [Folder Tree / Selection Events](#) page in Preferences, Opus will switch to that tab rather than change the location in the current tab.

(A note on the relationship between folder trees and file displays: normally, there is one file display and one tree. In a dual-display Lister, there are two file displays, but you have the option of using one or two trees. If you have two trees then each file display has its own - but a single tree can be shared by both file displays. In this mode, the tree will change to display the location of the active, or source, file display, and clicking on a folder in the tree will only change the location of the source file display).

If the folder tree has input focus (i.e. if it is the active user-interface element in the Lister, and keystrokes are directed to it), then it's also possible to navigate with the keyboard using the cursor keys.

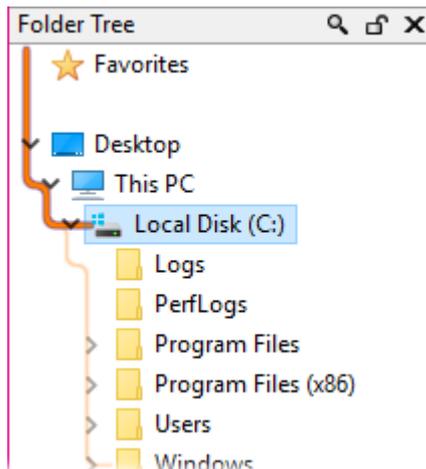
- **Displaying child folders:** The small glyph displayed to the left of each item in the folder tree is known as the *expansion button*. If you're using a different version of Windows or a third-party theme, this may look different to the screenshot above, but in general the glyph has two main states - **open** and **closed**. In the above screenshot, the **Libraries** and **Pictures** branches are **open** - the folders have been expanded, and their child folders in the hierarchy are visible. The **Documents**, **Music** and **Videos** branches are **closed** - their sub-folders are not displayed. The **My Pictures** branch doesn't show a glyph at all - this indicates that the folder has no sub-folders, and so can't be expanded.
- **Context menu:** As in other programs, you can access the context menu for an item in the tree by right-clicking on it. From the keyboard, you can press **Shift+F10** to display the context menu for the selected item. Some keyboards also have a dedicated key to the right of the space bar for this.

- **Rename:** You can use the folder tree to rename folders (those that support being renamed, anyway). This can be activated in several ways:
  - If you left-click an item (to select it), you can rename it by clicking it again after the double-click timeout has elapsed (this is referred to as a *slow double-click*).
  - If the tree has input focus (e.g. after you click in it) you can rename the selected item by pressing **F2**.
  - You can right-click on an item and choose **Rename** from the context menu.
  - When renaming a folder in the tree, you can use the up and down **cursor keys** to apply the current folder's new name and start renaming an adjacent folder. When done, push **Return** to apply the current folder's new name or **Esc** to cancel renaming the current folder.
- **Drag and drop:** You can copy or move files to folder in the tree using drag and drop. You can also drag folders out of the tree to copy or move (or create shortcuts to) them.
- **Keyboard shortcuts:** In addition to the well-known **cursor keys**, you can use the following shortcuts to control the folder tree:
  - **Numpad \*** will recursively expand everything below the selected item.
  - **Numpad +** expands the selected item the first time you push it; pushing it again expands the first level of child items (but not the children of those items).
  - **Numpad -** collapses the selected item (the same as **left-arrow**, except it won't go to the parent folder if you push it twice).

In the screenshot above you can see that the Folder Tree's title bar contains three buttons:

-  **Locate:** If the currently selected folder has been scrolled out of view, clicking this button will reposition it so that it is as close to the centre of the display as possible.
-  **Lock:** Normally the selection in the folder tree will change automatically as you navigate in the file display, to always reflect the current folder. If you turn the lock button on, the tree will never change selection by itself - the selected item will only change if you specifically select something else.
-  **Close:** This closes the folder tree. To re-open it, select the **Folder Tree** command from the **Lister Configuration** menu (or press **F8**).

The tree can be configured to highlight the path to the currently selected folder as a visual aid. It can make the current selection in the folder tree much more conspicuous. The highlight can be displayed all the time, or only when the tree itself is active.



Additionally, as in the screenshot above, the path to all currently open tabs can optionally be displayed as well. These options are all configured on the [Folder Tree / Appearance](#) page in Preferences.

### ***File Display border***

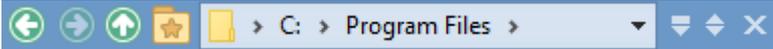
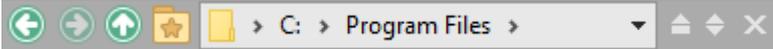
The File Display border is a title bar that appears at the top of the file display area in the Lister. By default the File Display border contains a toolbar, the [File Display](#) toolbar:



The first three buttons on the default File Display toolbar are, from left-to-right, **Back** (↶), **Forward** (↷) and **Up** (⬆). All of these buttons have an associated drop-down menu which you can access either by clicking-and-holding with the left mouse button, or single-clicking with the right mouse button. The Favorites button (★) provides access to your [favorite folders](#), and the [breadcrumbs location field](#) shows you the current folder.

This is a fully [configurable toolbar](#), and as such you can add your own buttons and modify or remove the default ones. See the page on the [File Display toolbar](#) for more information about these buttons.

The color of the file display border (and of the Opus icon, if enabled) indicates the state of the file display. A single display Lister can be one of three states - the default colors for these are shown below, but you can configure them through the [Display / Colors and Fonts](#) Preferences page. These colors apply whether the File Display toolbar is enabled or not.

-  **Source** (for actions involving two folders, like copy, this will be the source of the operation).
-  **Destination** (for actions involving two folders, this will be the destination of the operation).
-  **Off** (this Lister will not be involved in file operations with another window in this state).

In a dual display Lister, one display is always designated the source and the other the destination, so the file display border will never appear in the off state in a dual Lister. See the [Source and Destination](#) topic for more information about the source/destination concept.

In [Preferences / File Displays / Border](#) you can choose to turn the File Display toolbar off, in which case the file display border reverts to a simpler, static header:



In static header mode, in a single file display Lister the file display border is optional - you can choose whether it is shown or not using the **Show file display border in single display mode** option on the [File Displays / Border](#) page in Preferences. In a dual file display Lister the file display border is always shown.

This image is typical of the file display border in a single display Lister. The various elements of the single display border are:

- : The Opus icon can be used to create a shortcut to this folder - simply drag the icon out and drop it on the desktop or in a Lister to make a shortcut to the current folder.
- **C:\Program F**: The current location is displayed here. See below for more information about the location string.
- : Go [back](#) to the previous folder in the [history](#).
- : Go [forward](#) to the next folder in the [history](#).
- : Go [up](#) to the parent folder.
- : Display this folder in the other file display. In a single display Lister, this will automatically place the Lister in [dual display mode](#).

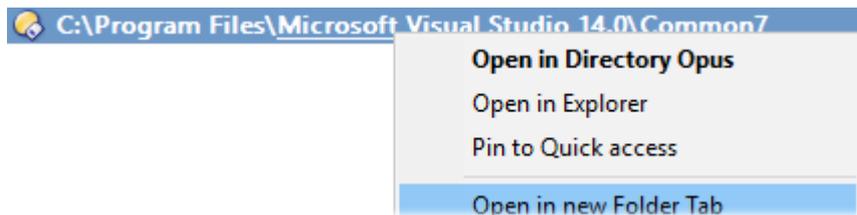
The file display borders in a dual display Lister have a few more buttons than a single display.



The additional buttons are:

- : Swap the two file displays (their positions are exchanged).
- : Change layout from side-by-side (vertical) to one-above-the-other (horizontal) and back again.
- : Close the file display - closing one will leave the remaining one behind as a single display Lister.

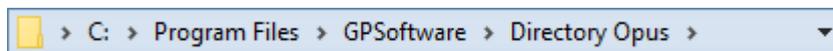
If you move your mouse over the location string displayed in the file display border you'll notice that the individual path components are actually links. The file display border operations like a simple version of the "breadcrumbs" location field - you can left-click on any component of the path to navigate to that folder, and you can right-click to access its folder context menu.



By default *hot paths* are only enabled when the file display is the source, and the Lister containing the file display is the active window. This is so you can click on a file display border to make it the source without worrying about accidentally clicking a link. You can change this behaviour with the options in the [File Displays / Border](#) page in Preferences.

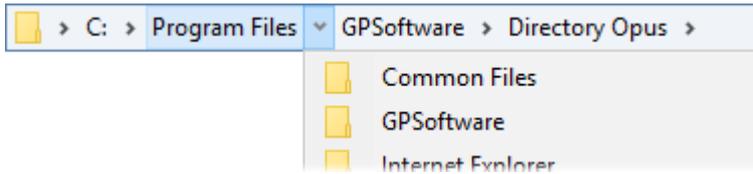
### **Breadcrumbs Location Field**

The "breadcrumbs" location field displays the current folder location as a "trail of crumbs".



Each component of the path is represented as a separate "crumb". Each crumb is actually a button and you can use them for navigation - taking the above screenshot as an example, to navigate to the **Program Files** folder you can simply click on it with the left mouse button. You can also right-click the crumb buttons to display the context menu for that level of the path, and drag and drop files to the buttons to copy or move files higher up the path.

The > glyphs following each crumb are also buttons; clicking these displays a pop-up menu showing the contents of that folder.



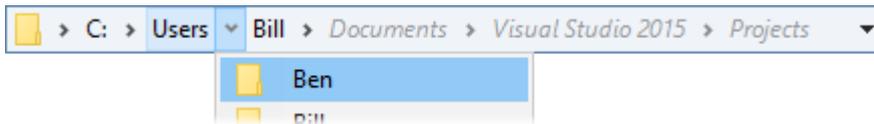
Again, you can use these pop-up menus for navigation, and you can drag and drop files to the menu items to copy or move files to that location.

When you navigate up the folder tree the branches of your previous deepest location are shown as "ghost" crumbs.



These can be very handy both as a visual reminder of the location you last visited as well as letting you jump back and forth between higher and lower folders in the same hierarchy. In the above screenshot we have navigated from the *Projects* folder back up the tree to the *Bill* folder.

Additionally, if you navigate to a sibling of an ancestor path, and the new folder has the same descendant hierarchy as the original ancestor, the ghost path is also preserved.

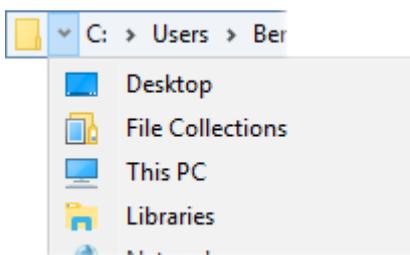


This lets you jump very quickly to the same relative spot in a folder tree - for example, moving from the sub-folders of one user's profile directory to another. In this screenshot we are about to navigate to the *Ben* folder, a sibling of the ancestor folder *Bill*.

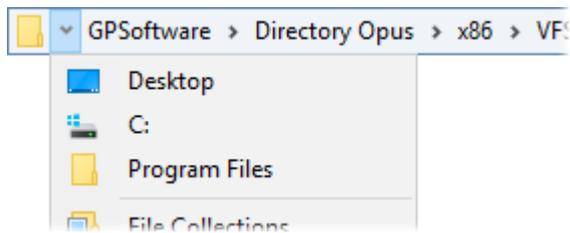


The result is that the previous deeper path remains as a ghost - clicking the *Projects* crumb will return you instantly to the same folder in the new hierarchy.

The icon at the far left of the field represents the current folder. You can drag and drop this icon to the desktop or another Lister to create a shortcut to the folder. The pop-up menu for this icon (accessed from its > button) is different to the others:

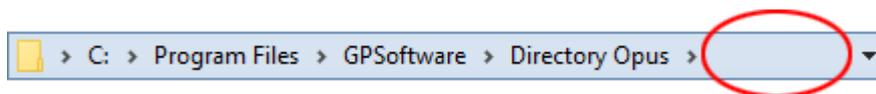


Rather than showing the contents of the current folder, it contains links to various special folders on the desktop as well as the disk drives in the system. This menu also acts as an "overflow" menu if the breadcrumbs field is not wide enough to display the full path.

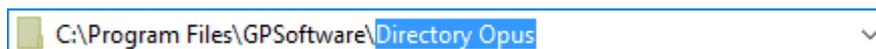


In these situations, the breadcrumbs field itself displays as many levels of the path as will fit, starting from the lowest level, and any higher levels that don't fit are displayed in the pop-up menu. Note that ghost crumbs are never displayed in the overflow menu.

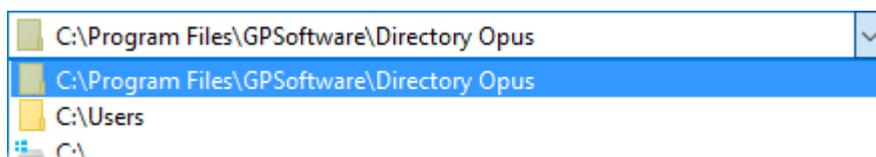
You can also use the breadcrumbs field to edit the path manually. If you click in an area of the field that doesn't contain a button:



The crumbs will disappear and be replaced by a traditional edit control, where you can manually type in a path to navigate to. In this mode, pressing the **F5** key will deactivate the edit control and instead open the *Desktop* branch drop-down menu. This is most useful when you have activated the breadcrumbs field by pressing its hotkey - for example, the default breadcrumbs field has **F4** assigned as the hotkey, meaning you can press **F4** followed by **F5** as a quick way to open the *Desktop* branch.



Finally, clicking the arrow button at the right-hand edge of the breadcrumbs control displays a drop-down list of [recently](#) visited folders:

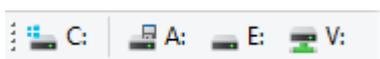


In a dual-display Lister, the breadcrumbs field displays the location of the current source file display. As you switch the two displays between [source and destination](#) the breadcrumbs field will update to show the current source location. Similarly, navigation actions you make using the breadcrumbs field only affect the source file display.

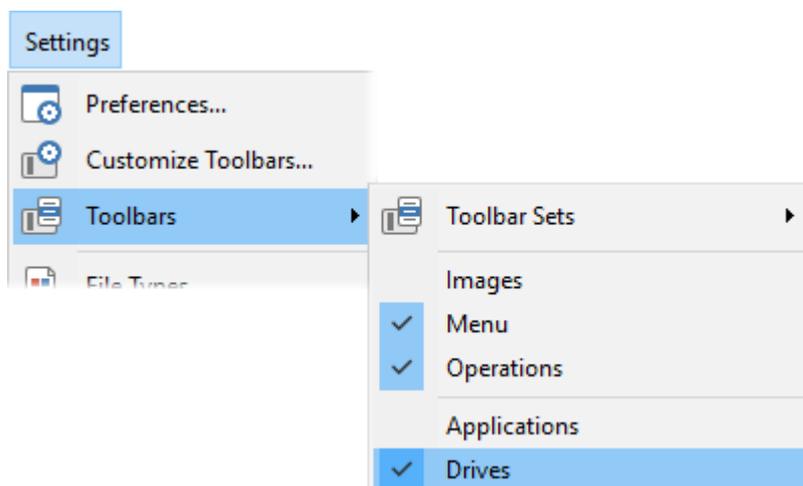
You can [configure the appearance and behavior](#) of the breadcrumbs field by editing the [field button](#) that generates it.

## Drive Buttons and Lists

The **Drives** toolbar that is supplied with Directory Opus displays a list of your disk drives. This provides an extremely simple way to navigate from the root of one drive to the next.



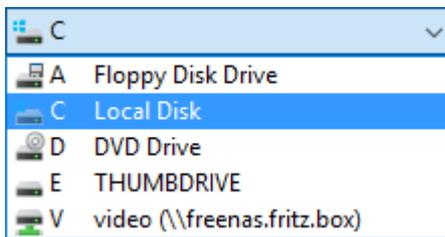
This toolbar is not actually one of the [default toolbars](#) (the ones built-in to the program) - instead, it is installed in your Opus toolbars folder by the program installer. To turn it on, select **Drives** from the *Settings / Toolbars* menu.



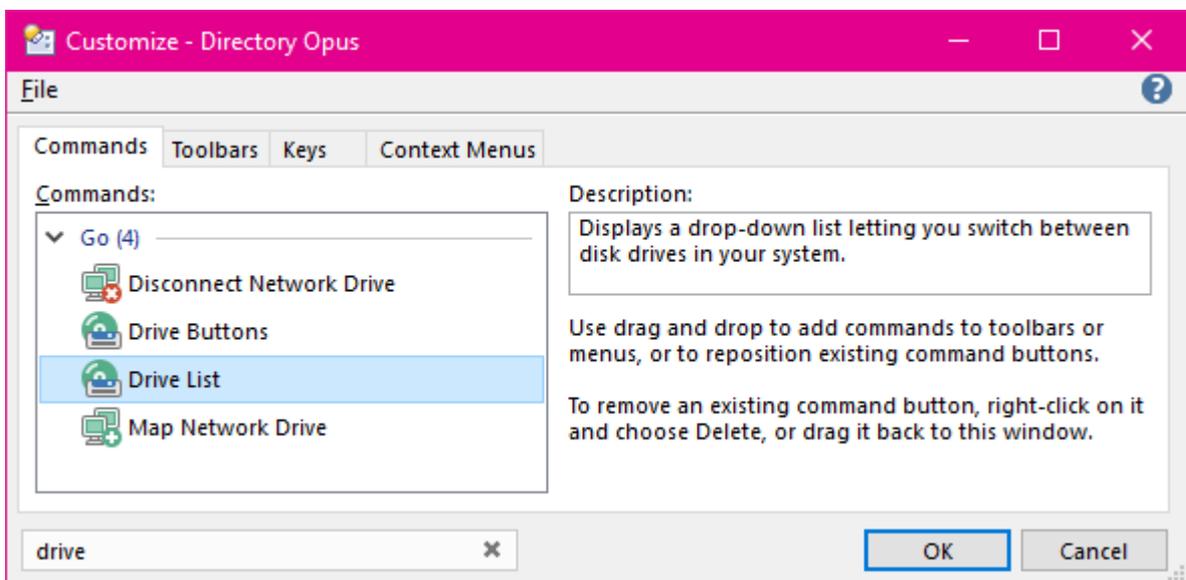
The toolbar displays one button for each of your installed disk drives. The buttons on the toolbar are actually grouped to display all "fixed drives" (mainly internal hard drives) first, followed by all other drives. To read the root folder of one of your disk drives, simply click the button. You can also right-click the buttons to display the system context menu for a drive.

The Drives toolbar is built from the internal [Go DRIVEBUTTONS](#) command, and you can edit the command that generates the drive buttons to configure their appearance and behaviour. See the [Editing the Toolbar](#) page for information on editing toolbars, and the [Drive Buttons Configuration](#) page for specific information about the changes you can make to the drive buttons.

As well as the option to display drive buttons, Opus also lets you add a drop-down **Drive List** to your toolbars. This is similar to the drive buttons in that it displays a list of your disk drives and lets you navigate by selecting a drive from the list.



The **Drive List** is available as a pre-defined command that you add from the [Commands](#) tab of the [Customize](#) dialog. The easiest way to locate the command is using the **Customize** dialog's filter.

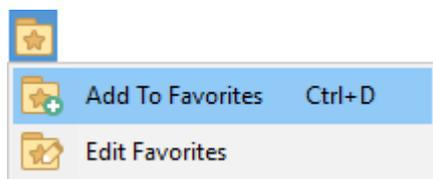


1. Select the **Customize** command from the **Settings** menu
2. Click on the **Commands** tab
3. Use the filter field at the bottom of the **Customize** dialog to enter the string *drive*
4. Select **Drive List** from the list of results and drag it to one of your toolbars

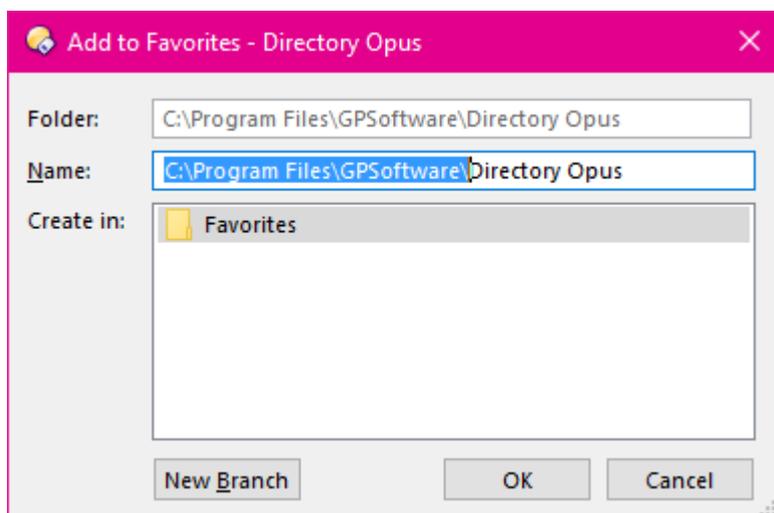
By default, navigation via the **Drive List** behaves slightly differently to navigation via the **Drive Buttons**. The **Drive List** remembers your current (or most recently used) folder on each drive. When you select, say, *C* from the drop-down list, Opus will take you to the folder on C: that you last used. If you want you can change this behaviour and make the **Drive List** always read the root folder. You can also configure which drives are shown in the list, and control which file display a list applies to - see the [Drive List Configuration](#) page for more information.

## Favorites

The Favorites system in Opus is very similar to that of a web browser. Important folders that you want to have quick access to can be added to the favorites list, and this list is then displayed in a drop-down menu at the top of each file display.



To add the current location to the favorites list, click the **Favorites** button (🌟) on the file display toolbar, and select the **Add To Favorites** command from the drop-down menu, or press **Ctrl+D**. This will display the **Add to Favorites** dialog:



The **Folder** field displays the current folder - this is picked up from the current source file display automatically and can't be edited. The **Name** field lets you assign a name to the favorite entry - by default, this is the full path of the folder but you can edit this to be just the name of the folder (the text preceding the folder's name is automatically selected for easy deletion) or indeed to any name at all.

The **Create in** field lets you create "sub-folders" (or branches) within your favorites list. Click the **New Branch** button to create a new folder - it will be created as a child of the currently selected branch. The new favorite will be stored in the selected branch. You can rearrange your favorites later using the [Favorites](#) page in Preferences.

To navigate to a favorite folder, simply select it from the drop-down favorites menu as shown above. The favorite folders shown in this menu can also be used to copy or move files to by dragging a file and dropping it onto the menu item. You can also right-click on the folders in the

favorites menu to display the context menu for that folder. Favorite folders can also be displayed in the [Folder Tree](#) - turn on the **Favorites** item on the [Folder Tree / Contents](#) Preferences page.

You can add, rename, delete and rearrange your favorite folders at any time using the [Favorites](#) page in Preferences.

## SmartFavorites

The SmartFavorites system in Opus is like an "automatic favorites list". It attempts to learn which folders you do most of your work in, and display them in a list as if you had manually added them as favorites. Folders that Opus notices you use a lot are displayed in the drop-down Favorites menu, in a separate section from your manually added favorites folder. See the page on the [Favorites](#) list for more information about using this drop-down menu.

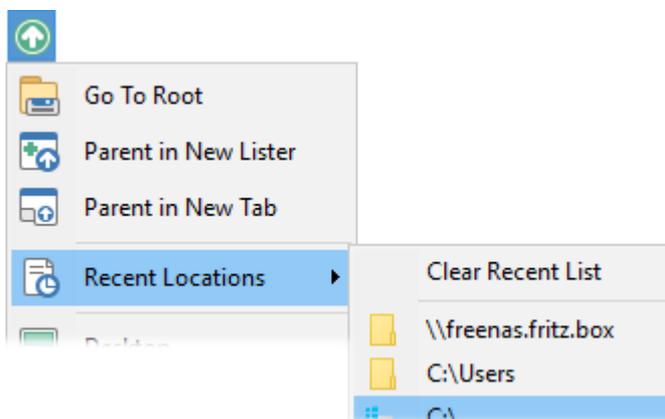
The SmartFavorites list can also be displayed in the [Folder Tree](#) - turn on the **SmartFavorites** item on the [Folder Tree / Contents](#) Preferences page.

You can enable or disable, as well as configure the behaviour of, the SmartFavorites system from the [Favorites and Recent / SmartFavorites](#) page in Preferences.

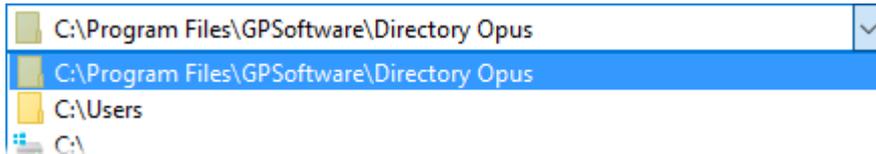
## Recent and History Lists

Opus maintains two separate lists of recently visited folders.

- The **Recent** list is a global list of your most recently visited folders. It is displayed in the **Recent Locations** sub-menu in the **Up** button's attached menu (right-click or click-and-hold the **Up** (↑) button on the [File Display toolbar](#) to access this menu).



It is also displayed in the drop-down list attached to the Breadcrumbs location field:



It can also be displayed in the [Folder Tree](#) - turn on the **Recent List** item on the [Folder Tree / Contents](#) Preferences page.

You can configure the recent list from the [Favorites and Recent / Recent List](#) page in Preferences.

- The **History** list is also a list of the most recently visited folders, but it is maintained on a per-file display (or per-tab) basis. The history list forms the basis of the [Back and Forward](#) navigation actions - when you click the back button, you are moving back through the history list. Right-click or click-and-hold the **Back** (  ) and **Forward** (  ) buttons on the File Display toolbar to access these menus.

You can configure the size of the history list (the maximum number of folders remembered) from the [Favorites and Recent / Recent List](#) page in Preferences.

## Aliases

The Directory Opus folder alias system lets you assign simple names (aliases) to folders. Say you had a long, complicated path to a folder that you use frequently - for example, **C:\Users\Jon\Documents\Company\Spreadsheets\Sales Projects\2011**. Using the alias system you could assign a name like **Sales11** to this folder. You can then use this alias anywhere in Opus that you would ordinarily use the full path. For example, you could click in the location field and enter **/Sales11** to navigate to that folder (all aliases are prefixed with a forward-slash when they are used).

There are also a large number of built-in aliases that are predefined to the location of common system folders. For example, **/dopusdata** is a built-in alias for the folder that stores your personal Directory Opus configuration files. The location of this folder changes depending on your user name and your operating system, but you don't need to know this or be aware of the details - you can simply use the alias to find it.

The alias system also allows you to refer to drives by volume label (or name) rather than drive letter. The format for this is **/\$<label>**. For example, you may have a USB thumb drive labelled "My\_Portable" that you want to refer to in Opus commands. One problem with removable drives is that they can be assigned different drive letters from one use to the next. Using the alias **/\$My\_Portable** would refer to this thumb drive no matter what drive letter had been assigned to it. Of course if you have two or more drives with the same label the behavior of this will be unpredictable - Opus simply uses the first drive with the specified label that it finds.

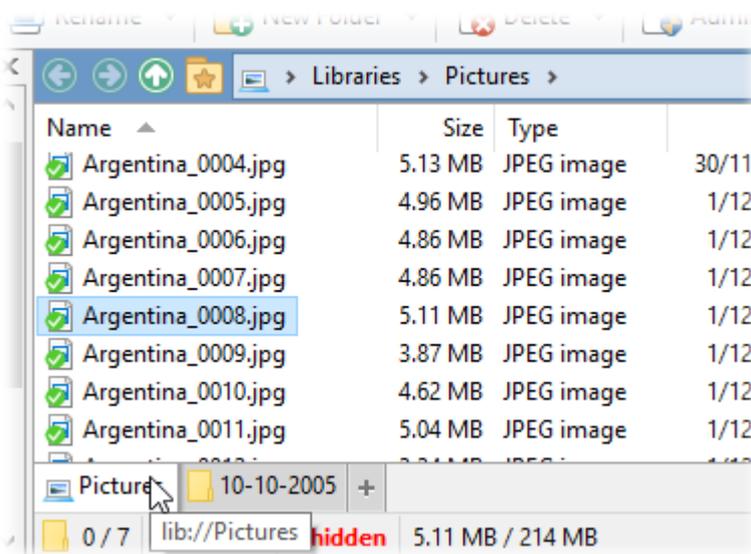
You can also use aliases in [buttons that you define yourself](#) (for example, a button with the command **Copy TO /Sales11** would automatically copy any selected files to the aliased folder).

As well as making it easy to remember and use complicated folder paths, another advantage of aliases is that you can change the folder an alias points to without having to manually update any references to that alias. In the above example, instead of using **/Sales11** as the alias, you could use something like **/CurrentYrSales** - then, at the start of the next financial year, simply change the folder that your alias points to to the **2012** directory and any buttons or references to it will automatically use the new location.

You can define your own aliases or see a list of the built-in ones from the [Favorites and Recent / Folder Aliases](#) page in Preferences.

## Tabs

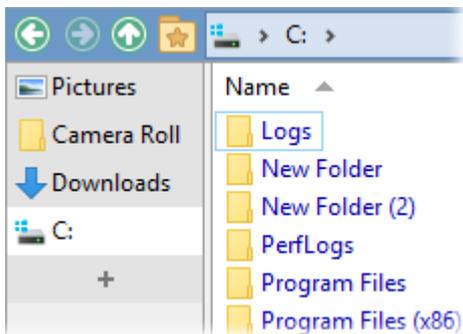
Directory Opus supports multiple tabs in each file display of a Lister, which means you can have multiple folders open at the same time, and rapidly switch between them to compare the contents, copy or move files from one to the other, or simply to provide quick access to multiple locations.



In the above screenshot, two tabs are open - the visible tab shows the **My Pictures** folder and the non-visible tab (if you were to switch to it) shows the **10-10-2005** folder. To switch between tabs, simply click on the tab you want to become active. Using the keyboard, you can press **Ctrl+Left** and **Ctrl+Right** to move from one tab to the next.

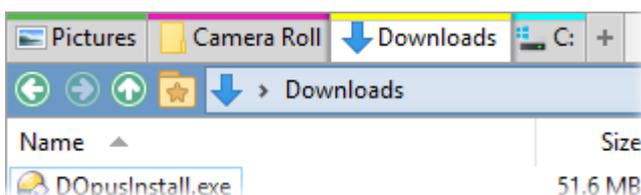
Folder tabs are enabled by default - even if only one tab is actually open, the *tab bar* will be displayed at the bottom of the file display (or in a dual-display Lister, at the bottom of each file display). There are a number of options in the [Folder Tabs / Options](#) page of Preferences that let you control the folder tabs system. For example, you can configure it so that the tab bar is only

displayed when there is more than one tab open, or display the tabs vertically on the left or right of the file display.

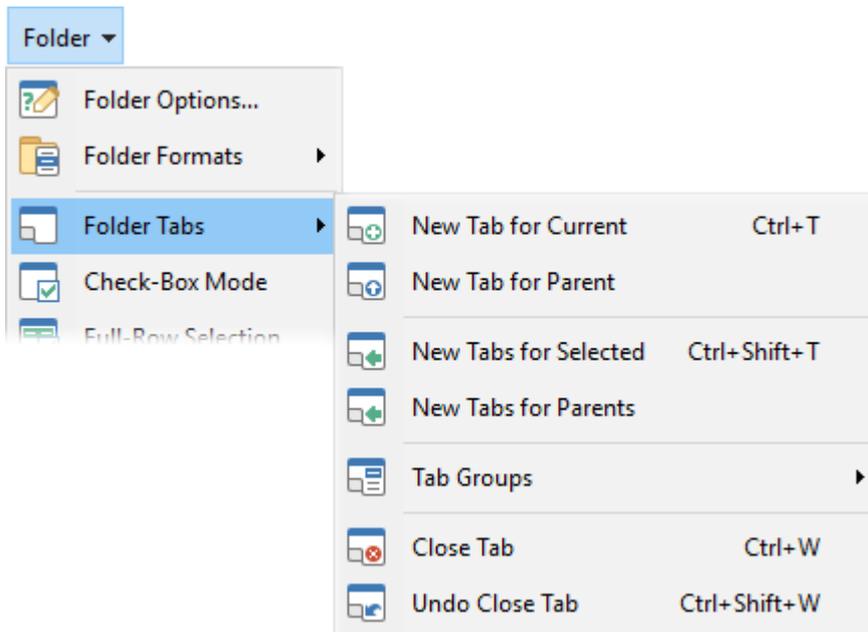


There are a number of ways to open a new tab. By default, you can double-click an empty part of the tab bar (in the above screenshot, double-click anywhere to the right of the **10-10-2005** tab) to open a new tab showing the current folder. You can also right-click on a folder and choose **Open in new folder tab** from the context menu, or hold the **Alt** key down when you double-click a folder to open it in a new tab. These behaviours can all be changed of course.

Tabs can be colored individually, to make them easier to identify. If you right-click on a folder tab you can select the **Set Tab Color** command from the context menu, which lets you change the color.



The **Folder Options / Folder Tabs** menu in the default toolbars contains a number of commands used to control tabs:



- **New Tab for Current:** Opens the current folder in a new tab.
- **New Tab for Parent:** Opens the parent of the current folder in a new tab.
- **New Tabs for Selected:** Opens any selected folders in new tabs (for example, selecting three sub-folders in the current folder and run this command, would result in three new tabs being opened showing those folders).
- **New Tabs for Parents:** Opens the parents of any selected items in new tabs. This is useful in a collection (e.g. Find Results) where the parent of the selected items is not necessarily the current folder.
- **Tab Groups:** This sub-menu lets you access any [tab groups](#) you have defined.
- **Close Tab:** Closes the current folder tab.
- **Undo Close Tab:** Re-opens the most recently closed folder tab.

You can manipulate a folder tab with the mouse in the following ways:

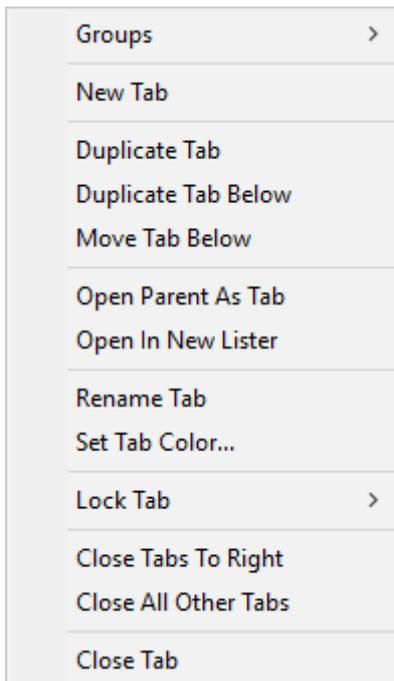
- You can switch between tabs by clicking on them with the left mouse button.
- Double-clicking a tab with the left button will close it (this can be disabled through Preferences).
- You can also click a tab with the middle-mouse button to close it.
- Dragging files from the current folder over another tab will copy or move them to that tab's folder. If you hover over the other tab without releasing the mouse button, that tab will be made active which lets you drop files into sub-folders within that tab.
- You can also drag and drop the folder tabs themselves - this lets you change the order they appear in the file display, and you can also move tabs from one file display to another (or one Lister to another) this way. You can create a duplicate of a tab by holding the **Ctrl** key down when you drag and drop the tab. You can also drag a tab out of a Lister and drop it on the desktop to open the folder in a new Lister.
- Normally tabs display the name of the folder they are showing, but if you click the left button on the label of the currently active tab, you are able to assign your own name to the tab. Once a tab has an assigned name it will not change even if you change folders in the tab. You can use several special "tokens" to insert information in the tab label:

**%P** - full path of the current folder

%N - name of the current folder  
%R - drive root of the current folder  
%% - insert a literal % character

- Right-clicking a tab displays the tab's context menu.

The tab context menu contains the following commands:



- **Groups:** This sub-menu lets you access any [tab groups](#) you have defined. You can also use this to save the current set of tabs as a new group. When selecting a group from this menu you can hold down either the **Shift** or the **Ctrl** keys to override the tab group's **Close existing Folder Tabs** setting; **Shift** means existing tabs will not be closed, **Ctrl** means they will.
- **New tab:** Opens a new tab using the settings in Preferences.
- **Duplicate Tab:** Opens a duplicate of the current tab.
- **Duplicate Tab Below:** Opens a duplicate of the current tab, in the other file display (depending on which file display you access this command from, 'below' may be 'above', 'left' or 'right').
- **Move Tab Below:** Moves the current tab to the other file display.
- **Open Parent As Tab:** Opens the parent of the current folder in a new tab.
- **Open In New Lister:** Opens the tab's folder in a new Lister.
- **Split To New Lister:** Splits this tab and any subsequent tabs to a new Lister (the existing tabs are closed, a new Lister is opened, and the tabs re-opened in the new Lister).
- **Rename Tab:** Assign your own name to the tab.
- **Set Tab Color:** Assign a color to the tab to make it easier to identify.
- **Lock Tab:** Lock or unlock the tab - see below for more information on locked tabs.

- **Link Tab:** Link or unlink the tab - see below for more information on linked tabs.
- **Close Tabs To Left:** Closes all tabs to the left of this tab.
- **Close Tabs To Right:** Closes all tabs to the right of this tab.
- **Close All Other Tabs:** Closes all open tabs except for this one.
- **Close Tab:** Closes this tab.

Note that most of these mouse actions can also be invoked using the Opus command set, which means you can also assign hotkeys to them. See the [Go](#) command for the full list of tab-related arguments you can use.

You can also hold various qualifier keys down to modify the linked state (see below for more details) of the clicked-on tab:

- **Control-click** a tab to link it with the active tab in the other file display (or, if already linked, unlink it).
- **Control+Shift-click** a tab to toggle navigation lock mode on or off (see below).
- **Shift-click** a tab to override the normal behavior when clicking linked tabs (e.g., its partner tab will not come to the front as normal).

### ***Linked tabs***

In a dual-display Lister, a folder tab on one side of the Lister can be linked with a tab on the other side. When two tabs are linked, selecting one in the Lister to make it active automatically activates the linked tab too. Linked tabs are displayed in different colors from unlinked tabs, and you can configure these colors from the [Display / Colors and Fonts](#) page in preferences.

### ***Navigation Lock tabs***

You can also place linked tabs into Navigation Lock mode, using the **Navigation Lock** option in the context menu. When linked tabs are in this mode the destination tab will always follow the source tab whenever the folder changes, in a similar manner to [Navigation Lock](#).

### ***Locked tabs***

Folder tabs can be locked in a number of different ways. Primarily, when a tab is locked, it always displays the same folder. To lock a tab, right-click on it and choose a command from the context menu's **Lock Tab** menu. The different ways a tab can be locked are:

-  **Unlocked:** The tab is not locked, meaning you can freely navigate away from the initial folder.

-  **Locked**: The tab is locked. Attempts to navigate away from the initial folder (that is, the folder that was displayed when you locked the tab) will cause a new tab to open, leaving the original tab unchanged.
-  **Locked (allow folder changes)**: The tab is locked, but you can navigate away from the initial folder. If you switch to another tab and then back again the original folder will be restored.
-  **Locked (reuse unlocked tab)**: The tab is locked. Attempts to navigate away from the initial folder will reuse an existing unlocked tab if one exists, otherwise a new tab will be opened.

You can also use the lock commands to lock or unlock multiple tabs at once - select the commands from the menu while holding down the following keys:

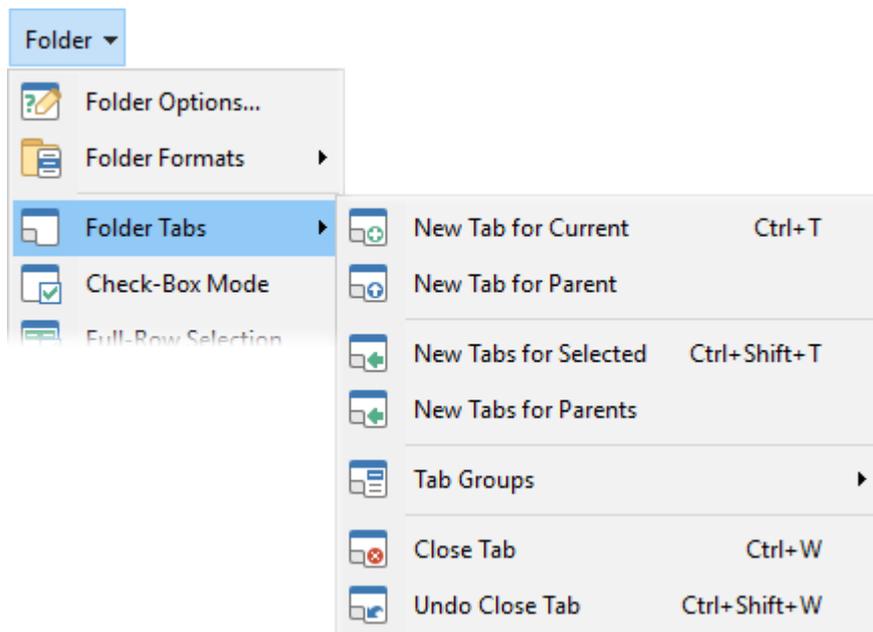
- **Shift**: Lock or unlock all tabs
- **Ctrl**: Lock or unlock the active tab and all other tabs to the right
- **Ctrl + Shift**: Lock or unlock the active tab and all tabs to the left

## ***Tab Groups***

Tab Groups are sets of predefined tabs (folders) that can be opened in one operation. For example, you might have a set of work folders for a particular project that you always need quick access to. You could define a tab group that opens all the folders in folder tabs, and then when you're ready to work on that project, you only need to select the group in order to open all those folders at once. Tab groups can also define tabs that open in both the left and right file displays simultaneously (these are called "specific sides" groups).

The easiest way to create a tab group is to save an existing set of tabs in a Lister. Simply right-click on any tab to display its context menu, and from the **Groups** sub-menu choose the **Save** command (or, in a dual-display Lister, the **Save Both Sides** command to save the tabs from both file displays to a single group). You can also create and edit tab groups from the [Folder Tabs / Tab Groups](#) page in Preferences.

Loading a tab group opens new tabs for all folders saved in that group. To load a tab group, right-click a tab and select the desired group from the **Groups** sub-menu. You can also access a list of tab groups from the **Folder Options / Folder Tabs** menu in the toolbar:

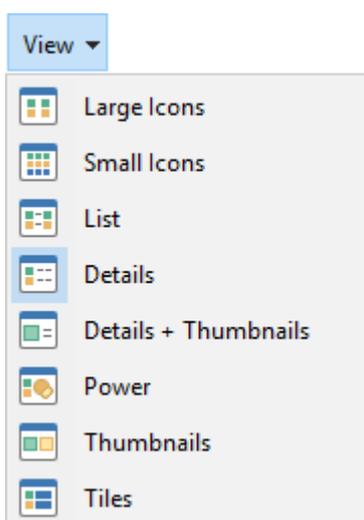


When you load a tab group, any existing tabs are automatically closed. You can override this behaviour on a per-group basis from the **Tab Groups** page in Preferences.

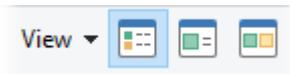
In a dual-display Lister, tabs on one side of the Lister can be linked with tabs on the other side. You can configure this in a tab group when the "specific sides" option is turned on.

## View Modes

Files and folders in the file displays can be shown in a number of different view modes. You can change view mode using the drop-down **View** menu in the default toolbar:



Next to the **View** menu are three buttons that provide quick access to the most commonly used view modes:

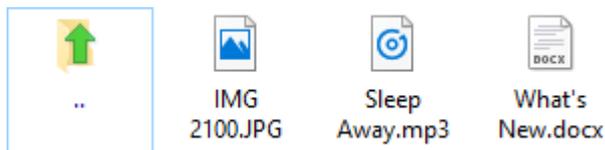


From left-to-right the buttons represent **Details** mode, **Details + Thumbnails** mode, and **Thumbnails** mode.

You can use the [Folder Options](#) system to automatically apply a view mode to specific folders. For example, you could have Opus automatically display your pictures library in thumbnails mode.

The available view modes are:

- **Large Icons:** Displays the folder contents using large icons.



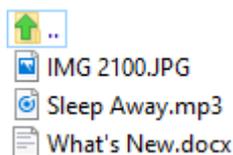
The "large" icon size is defined by Windows itself; on a standard system this is 32 x 32 pixels. You can change this size through the Windows control panel.

- **Small Icons:** Displays the folder contents using small icons.



The "small" icon size is defined by Windows; on a standard system this is 16 x 16 pixels.

- **List:** Similar to Small Icons, this mode displays the folder contents with a small icon and label.



This view mode is popular with many people as it displays the most files at once in each file display. The layout in this mode is different to almost all other modes - instead of icons running left-to-right and then top-to-bottom, the order is reversed - files are displayed in columns running down the file display, and the display scrolls horizontally rather than vertically.

- **Details:** Displays the folder contents as a table, with rows for each file and columns for information about each file.

Name ▲	Size	Type	Modified	Attr	Description
 ..		Parent Folder			
 IMG 2100.JPG	5.13 MB	JPEG image	30/11/2010 7:07 PM	-a----	4000 x 2672 x 24 JPEG
 Sleep Away.mp3	6.08 MB	MP3 Format Sound	19/02/2014 9:53 AM	-a----	MPEG-1 Audio Layer
 What's New.docx	259 KB	Office Open XML Document	Today 10:49 AM	-a----	What's new in Direct

This mode can display additional columns of information for each file besides the filename. You can choose which columns are displayed using the [Folder Options](#) system. See the [File Display Modes / Details Mode](#) Preferences page for a description of the settings that can be configured for Details mode. You can also configure things like grid lines to visibly separate items in the list.

When in Details mode you can hold the **Control** key down and turn the mouse wheel to increase or decrease the font size used to display the folder contents.

- **Details+Thumbnails:** This is not strictly a view mode in its own right - instead, this is standard **Details** mode with the **Thumbnail** column added.

	Name ▲	Size	Type	Modified	Attr
	Argentina_0020.jpg	5.02 MB	JPEG image	1/12/2010 7:00 PM	-a----
	Argentina_0021.jpg	5.22 MB	JPEG image	1/12/2010 7:01 PM	-a----

The size of the thumbnail is determined by the width of the column - you can set the default column width using the **Default width** option on the [Preferences / Display / Fields](#) page. You can position the *Thumbnail* column anywhere you like, but if it appears immediately adjacent to the *Name* column Opus will treat the two as a single column as far as selection and highlighting is concerned (as shown in the image above). The [Preferences / File Display Modes / Details](#) and [Power Mode](#) pages both have options that let you configure the *Thumbnail* column's aspect ratio (it defaults to 16:9), whether thumbnails in the column have a border displayed, and whether the file icon should be automatically hidden whenever the *Thumbnail* column is visible.

- **Power:** This is very similar to Details mode (it looks identical), except the behaviour of the list when [interacting with it using the mouse and keyboard](#) can be configured to a far greater extent.

 DeviceSetupManagerAPI.dll	115 KB	30/10/2015 6:19 PM
 dhcpcsvc6.dll	65.5 KB	28/05/2016 2:24 PM
 dhcpcsapi.dll	210 KB	30/10/2015 6:17 PM
 DiagCpl.dll	1 MB	30/10/2015 6:17 PM
 DiagnosticLogCSP.dll	281 KB	30/10/2015 6:17 PM
 diagperf.dll	1.37 MB	1/07/2016 1:38 PM
 diagtrack.dll	1.53 MB	1/07/2016 2:45 PM
 diagtrack_win.dll	360 KB	1/07/2016 1:46 PM
 discan.dll	247 KB	30/10/2015 6:17 PM

By default Power mode uses persistent selection - unlike the other view modes, files are not deselected automatically when you click an empty space in the file display. See the [File Display Modes / Power Mode](#)

and [Power Mode Buttons](#) Preferences pages for a full description of the Power Mode configuration options.

When in Power mode you can hold the **Control** key down and turn the mouse wheel to increase or decrease the font size used to display the folder contents.

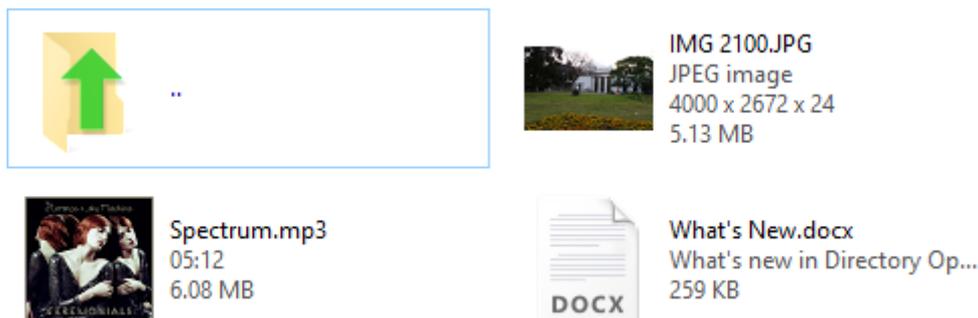
- **Thumbnails:** This mode displays thumbnails for files and folders, which is particularly useful for image or video files.



Opus can generate thumbnails for many different file formats; files that a thumbnail can not be generated for will be displayed with the usual icon for that file type. As well as the file name, this mode can optionally display the file size and (for image file formats) the dimensions of the image below each thumbnail. You can configure the size and appearance of thumbnails from the [File Display Modes / Thumbnails Mode](#) Preferences page.

By default Opus will display a special toolbar - the [Images Toolbar](#) - whenever the file display is set to thumbnails mode. This contains several commands for manipulating images, as well as a **Thumbnail Size** slider -  - that lets you adjust the size of thumbnails dynamically. You can also increase or decrease the thumbnail size by holding the **Control** key down and turning the mouse wheel.

- **Tiles:** This mode combines large icons (or optionally thumbnails) with the ability of Details mode to display information besides the filename.



You can choose what information is shown for each file type through the [File Types](#) system. Also see the [File Display Modes / Tiles Mode](#) Preferences page for settings that can be configured for Tiles mode.

Don't forget that in the modes that **don't** display any information besides the filename, you can hover over a file to view its info tip (tooltip). The info tip will often show you lots of pertinent information about a file, and you can configure exactly what's shown from the [File Types](#) dialog.

## Dual Display

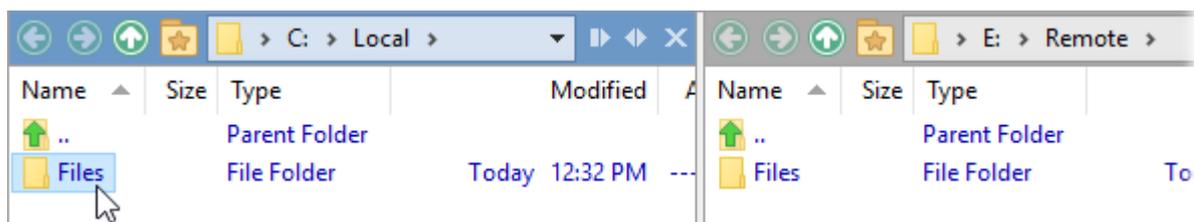
The term *dual-display mode* refers to the display of two independent file displays in the one Lister. Each file display can show the contents of a different folder. Dual-display mode makes it particularly easy to compare the contents of two folders, or copy or move files between two folders, without needing to open multiple windows. Another common use is to display the same folder in both displays, in two different view modes - for example, details mode in one and thumbnails in the other.

The easiest way to put a Lister into dual-display mode is with the buttons on the default toolbar -  for horizontal layout, and  for vertical layout. You can also press **F6**. When a Lister goes into dual-display mode, the currently displayed folder will be duplicated in the new display. You can change this to always show a specific initial folder in the second display with the *Specify initial folder when switching to dual file display* option on the [File Displays / Options](#) Preferences page.

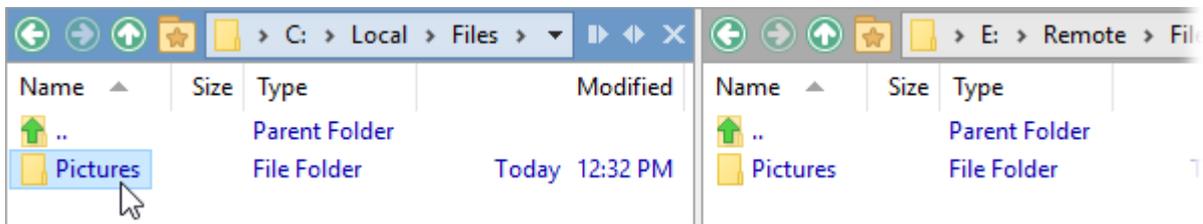
When a Lister is in dual-display mode you can copy or move files from one display to the other using drag-and-drop or copy-and-paste. Dual-display mode also ties in with the [Source and Destination](#) concept - whenever a Lister is in dual-display mode, one display is always designated the source folder and the other is the destination. This means that commands like **Copy Files** will always copy selected items from one display in the Lister to the other.

## Navigation Lock

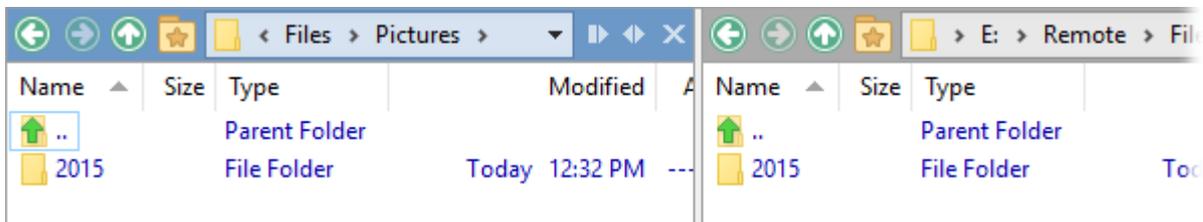
Navigation Lock (*NavLock*) is a mode you can turn on in a dual-display Lister. You can turn NavLock on using the  button on the default toolbar. When a Lister enters NavLock mode, Opus records the two folders that are currently displayed in both file displays. From this point on, any navigation you perform in one file display will be automatically replicated, if possible, on the other side. This mode is most useful when you have the same (or a similar) directory structure on two different drives (or on a local drive and an FTP site, for example).



NavLock has been turned on in the above Lister. Double-clicking on the **Files** folder in the left-hand file display results in...

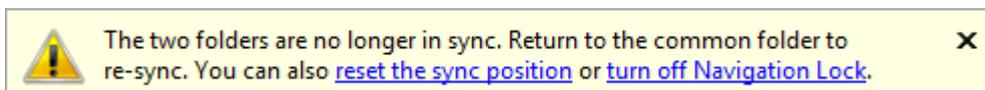


... the right-hand file display also navigating into the **Files** folder. We then double-click on the **Pictures** folder in the left-hand display and...



... the right-hand display replicates the action. The same applies to other navigation actions like Up, Back, Forward, etc. - as long as there is a matching folder on the other side of the Lister, the action will be replicated.

If, however, there is no matching folder on the other side, and the action **can't** be repeated, the file displays will get out of sync. In this event, the following message will be displayed:



At this point you have four options:

- You can ignore the message (and dismiss it by clicking the close button).
- You can manually navigate back to an in-sync location.
- You can **reset the sync position** - this will bring the folders back into sync, and NavLock mode will remain enabled, with the current locations set as the new base folders.
- You can **turn off Navigation Lock** - either using the link in the message or by clicking the  button again.

## **Navigation Lock with multiple folder tabs**

Navigation Lock is a mode which applies to the Lister as a whole and not specifically to the pair of folder tabs which were active when the mode was turned on.

If you switch between different folder tabs while Navigation Lock is turned on, the Navigation Lock base folders will be reset each time. Effectively, if Navigation Lock is on when you switch to another folder tab, it is momentarily switched off and then back on again.

If you are considering using Navigation Lock in a Lister with multiple tabs, you may wish to use **linked tabs** instead. Linked tabs are two tabs that have been linked together, so that when one tab is activated its partner also comes to the front. Linked tabs can also be put into a version of Navigation Lock that applies only to those tabs instead of whichever tabs are currently active. See the [Tabs](#) page for more information.

## **Toolbars**

Toolbars are one of the main ways of interacting with Opus - they let you perform operations on files and launch programs. Some important points to note about Opus toolbars are:

- All toolbars and menus in Opus are [configurable](#) via the [Customize](#) function.
- In Opus, toolbars and menus are the same thing.
- Toolbar buttons can run [internal commands](#), [scripts](#) and launch [external programs](#), and can pass along the names of selected files and other information automatically.
- Toolbar buttons can have associated *hotkeys* that let you launch the function from the keyboard rather than the mouse. From the Customize dialog you can elect to always enable a toolbar's hotkeys - even if the toolbar isn't current open, its hotkeys will still work.
- Toolbars can either be displayed in the Lister, or can be [floating](#) on the desktop, or both. You can even float multiple copies of the same toolbar if desired.
- Toolbar buttons can consist of a text label, a graphical image, or both.
- Toolbars can be set to [appear automatically](#) when changing to a certain display mode, or visiting a certain folder.
- Multiple toolbars can be saved to a [Toolbar set](#) which can then be reloaded later.
- Specific Listers can be saved with their own toolbars using the [Lister Layouts](#) system.
- The toolbars that a Lister displays by default are stored in what is known as the *Default Toolbar Set* - this is used whenever a Lister doesn't have its own set of toolbars defined in the Layout it came from. Use the **Settings / Toolbars / Set As Default Toolbar Set** command to update your default toolbar set if you want to change which toolbars are used by default.
- Opus ships with four [factory-default toolbars](#) - you can edit them but you can't delete them. You can create as many other toolbars as you want.

You can turn toolbars on and off using the list in [Customize](#), from the **Settings / Toolbars** menu, or by right-clicking an empty space on any toolbar to bring up its context menu.

## **The Default Toolbars**

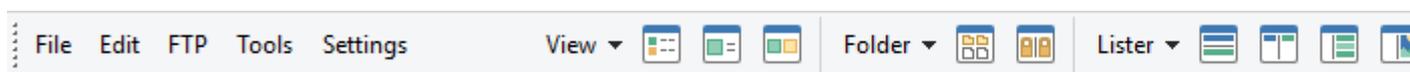
There are two types of "default" toolbars in Opus.

- The "real" default toolbars are a set of four toolbars that are hard-coded into the program. They are created on disk automatically when Opus first runs, and are automatically re-created if they are ever accidentally deleted. The default toolbars cannot be deleted or renamed, although they can be turned off and you are able to edit them freely. You can reset your configuration to the default toolbars at any time, by right-clicking on a toolbar and choosing **Factory Reset Toolbars** from the context menu's **Toolbars** sub-menu.
- The Opus installer creates several toolbars automatically when you install the program. These can more properly be described as "pre-supplied" toolbars, as they are not hard-coded into the program, and there is nothing special about them other than the fact that the installer creates them. They are provided mainly as examples. You are free to delete these as you wish.

Which toolbars a Lister opens with by default is determined by where the Lister came from. If the Lister came from a saved [layout](#), its toolbars may have been stored with it. If not, a Lister will usually open with the *Default Toolbar Set*. This is a special [toolbar set](#) that determines a Lister's initial set of toolbars. Use the **Settings / Toolbars / Set As Default Toolbar Set** command to update your default toolbar set if you want to change which toolbars are used by default.

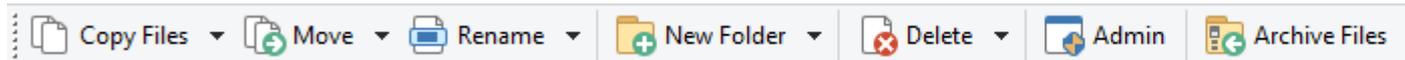
The default toolbars in Directory Opus are designed to expose as much as possible of the functionality of the program without appearing too busy or overly complicated. The four toolbars are as follows:

The [Menu toolbar](#):



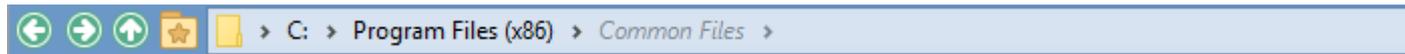
This toolbar contains the "traditional" drop-down menus, as well as controls for modifying the appearance of the Lister.

The [Operations toolbar](#):



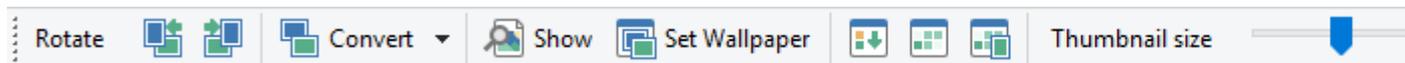
This toolbar contains commands relating to file and folder operations.

The [File Display](#) toolbar appears at the top of each file display (unless disabled in Preferences):



This toolbar mainly contains commands related to navigation.

The [Images](#) toolbar appears whenever a file display is set to [Thumbnails mode](#):



This toolbar contains basic image manipulation functions.

## Menu Toolbar

This toolbar contains the "traditional" drop-down menus, as well as controls for modifying the appearance of the Lister.



On the left-hand side of this toolbar are several drop-down menus:

- **File:** The File menu contains commands to open a new Lister, close the current one and exit the program. It also displays the context menu for the currently selected files.
- **Edit:** The Edit menu contains mainly clipboard related commands. As well as the traditional Cut / Copy / Paste commands, there are also commands for copying the names of selected files or the current path to the clipboard in various formats. The [Calculate Folder Sizes](#) command calculates and displays the total sizes of selected folders. There are also commands for selecting files and folders (select all, select by pattern, etc).
- **FTP:** The FTP menu displays your [FTP address book](#) entries, and lets you make an [ad-hoc connection](#) to an FTP site.
- **Tools:** This menu contains commands for various tool or utility functions, including:
  - [Find Files](#), [Synchronize](#) and [Find Duplicate Files](#)
  - [Join Files](#) lets you join two or more files together, and [Split File](#) lets you split a single file into multiple parts.

- [Print / Export Folder Listing](#) lets you print the current folder either to the printer, a file or the clipboard, in various formats (including .CSV for import into a spreadsheet).
- [Image conversion](#) tools
- **Settings:** This menu contains commands to access the Opus configuration including [Preferences](#), [Customize](#) (create your own toolbars or hotkeys) and [File Types](#) (change what happens when you double-click or right-click on specific types of files).

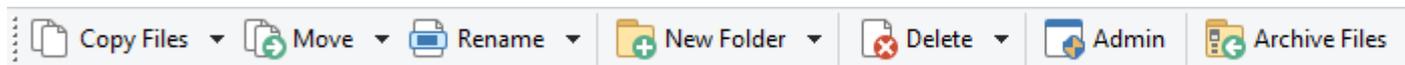
On the right-hand side are the following menus and commands:

- **View:** The View menu lets you switch between the various [view modes](#) in the current file display. It also contains sub-menus that let you modify the [sorting and grouping](#) of the current display.
-  **Details Mode:** This button provides a shortcut to [Details mode](#).
-  **Details + Thumbnails:** This button provides a shortcut to [Details + Thumbnails mode](#).
-  **Thumbnails Mode:** This button provides a shortcut to [Thumbnails mode](#).
- **Folder:** The Folder menu contains commands to access the [Folder Options](#) dialog, and the [Folder Formats](#) system. If you have created any [Favorite folder formats](#) you can select them from this menu. This menu also contains commands to manipulate [Folder Tabs](#), [Flat View](#), [Checkbox Mode](#), [Navigation Lock](#) and various [filtering](#) commands.
-  **Flat View:** This button lets you turn [Flat View mode](#) on or off. Clicking the button once puts the file display in *Flat View Grouped* mode, a second click puts it in *Flat View Mixed* mode, and a third click turns Flat View off. If you right-click the button a popup menu gives the additional option of *Mixed (No Folders)* mode.
-  **Navigation Lock:** This button lets you turn [Navigation Lock](#) on or off.
- **Lister:** The Lister menu contains commands to toggle the [folder tree](#), [metadata panel](#) and [status bar](#) on or off, as well as displaying a list of your configured [Lister Styles](#).
-  **Dual Horizontal:** This button toggles [dual display mode](#) on or off, with horizontal layout.
-  **Dual Vertical:** This button toggles [dual display mode](#) on or off, with vertical layout.
-  **Metadata Pane:** This button toggles the [metadata pane](#) on or off.
-  **Viewer Pane:** This button toggles the [viewer pane](#) on or off.
-  **Search:** A field that lets you perform a "quick search" of the current folder, using the indexed [Windows Search](#) system.

## Operations Toolbar

This toolbar contains commands relating to file and folder operations. The top-level buttons (**Copy Files**, etc) provide quick access to the most commonly used file functions, while the drop-down menus attached to those buttons contain more complicated or less-frequently used commands related to their parent button (so, for example, the drop-down menu for **Copy Files** contains **Copy As** and **Duplicate** commands as well).

Most of these buttons require one or more files or folders to be selected before they are available. They also generally use the [Source and Destination](#) concept - the **Copy Files** button will copy all selected items from the current source file display to the destination. In a [dual-display](#) Lister the destination is always the non-active file display; in a single-display Lister, another Lister (if there is one) will be the destination. If no destination file display or Lister is available, Opus will prompt you to select a destination folder.



-  **Copy Files:** The parent button will [copy](#) selected files and folders to the destination folder. The drop-down attached to this button contains the following additional commands:
  - **Copy As:** This command lets you [provide new filenames](#) for the copied files - either one at a time, or for all selected files as a batch using a wildcard.
  - **Duplicate:** This command makes copies of the selected files in the current directory (that is, the files are duplicated in the same location, they are not copied to the destination). You must [provide new filenames](#) for the duplicated files and folders.
  - **Update All:** This command copies selected files to the destination if they either a) don't exist in the destination, or b) exist but are newer than those in the destination (see [Copying Updated Files](#) for more information).
  - **Update Existing:** This command copies selected files to the destination only if they already exist but are newer than those in the destination (see [Copying Updated Files](#) for more information).
  - **Send To:** This sub-menu contains the system *Send To* menu that lets you send selected files and folders to various destinations. Using this drop-down menu is equivalent to right-clicking on the file and selecting from the *Send To* context menu.
  - **Copy Filter:** This command toggles the recursive [copy filter](#) on or off in the current Lister.
-  **Move:** The parent button will [move](#) selected files and folders to the destination. The drop-down menu contains the following additional commands:
  - **Move As:** This command lets you [provide new filenames](#) for the files as they are moved - either one at a time, or for all selected files as a batch using a wildcard.
-  **Rename:** The parent command lets you rename selected files and folders using the [Advanced Rename](#) dialog. The drop-down menu contains several examples of how the raw Rename command can be "scripted" to perform specialized renaming tasks:

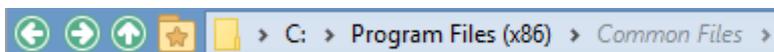
- **Use Simple Rename:** Turn this on to make the **Rename** button trigger the simple Rename dialog (which only supports basic wildcard renames), rather than the Advanced Rename dialog.
  - **Capitalize Words:** Capitalizes the first letter of each word in the filename, lower-cases all other letters.
  - **Make Web-safe:** Makes filenames "safe" for use on websites, by removing any characters that can cause problems in URLs (anything other than "-", "0"- "9", "a"- "z", "A"- "Z", ".", "\_", "]" and "+" is removed).
  - **Number Files:** Inserts an incrementing number before the names of selected files. This function is implemented as a VBScript [rename script](#), and so provides a good example of a more complicated rename function.
  - **Time-Stamp Names:** Appends a date- and time-stamp to the names of selected files.
  - **Underscores to Spaces:** Replaces any underscores in filenames with spaces.
-  **New Folder:** The parent command displays a dialog that lets you create a new folder in the active file display. If you are currently in the [File Collections](#) root folder, a new collection will be created, and if you are currently in the [Libraries](#) root folder, a new library will be created. The drop-down menu contains the following additional commands:
    - **New Archive:** This command [creates a new archive](#) (you will be prompted for the archive type and any options applicable to the archive format).
    - **New Text Document:** This command creates a new empty text document (.txt file) in the current folder. This is equivalent to right-clicking on the file display background and choosing **New -> Text Document** from the context menu.
    - **New:** This sub-menu displays the system **New** menu, that lets you create new files of various types.
    - **New Dated Folder:** Creates a new folder in the current file display with the current date and time as its name.
    - **Move into Dated Folder:** Creates a new folder using the current date and time, and automatically moves any selected files and folders into it.
-  **Delete:** The parent button will delete all selected files and folders. By default the recycle bin will be used to provide an undoable delete if possible, although this can be modified through [Preferences](#). The following additional commands can be found in the attached drop-down menu:
    - **Secure Wipe:** This command deletes all selected files using a [secure delete](#) algorithm, which makes it extremely difficult (if not impossible) for anyone to recover the deleted information.
    - **Remove From Collection:** When used within a [file collection](#), this command removes all selected items from that collection. If you use the parent **Delete** command on collection items, the real files and folders are deleted - use the **Remove** command to remove them from the collection without deleting the actual files.
    - **Delete Filter:** This command toggles the recursive [delete filter](#) on or off in the current Lister.
-  **Admin:** This button turns on [Administrator Mode](#) in the current Lister (Windows Vista and above only).
-  **Archive Files:** The parent command lets you add all selected files and folders to a [new archive](#) in the current folder (you will be prompted for the archive type and parameters). The drop-down menu contains the following additional commands:
    - **Files to Separate Archives:** Adds all selected items to separate archives (one archive per selected file or folder). This command will create ZIP files by default; you can change the archive format

used by editing the command (e.g. change **Copy ARCHIVE=single** to **Copy ARCHIVE=.7z,single** to use 7zip by default).

- **Files to Destination Archives:** This is the same as the parent **Archive Files** command except that the new archive will be created in the destination.
  - **ZIP And Email Files:** Adds selected items to a ZIP archive and sends it via e-mail to a recipient (you will be prompted for the e-mail address).
  - **Extract to Folders:** Extracts the contents of any selected archives to new folders named after the extracted archives (e.g. the contents of *Pictures.zip* would be extracted to a new folder called *Pictures*).
  - **Extract to Destination:** Extracts the contents of any selected archives to the destination folder.
  - **Extract to Destination Folders:** Extracts the contents of any selected archives to new folders in the destination.
-  **Properties:** The parent button displays the system Properties dialog for all selected items. The drop-down menu commands are:
    - **Attributes:** Displays the [Change Attributes & Times](#) dialog which lets you modify the attributes and date/time-stamps of selected items.
    - **Description:** Displays the [Set Description](#) dialog that lets you assign a description string to selected items.
    - **Edit Metadata:** Displays the [Set Metadata](#) dialog that lets you edit the metadata for any selected files.
    - **Set Label:** This sub-menu lets you assign a label to selected files and folders. Labelled files can be shown in different colors and/or font styles. Use the [Favorites and Recent / File and Folder Labels](#) page in Preferences to create and edit your labels.
  -  **Slideshow:** This button launches a [slideshow](#) of all selected image files. If no files are selected, all image files in the current folder will be displayed. The drop-down menu commands are:
    - **Show Pictures:** Displays selected files in the standalone viewer.
    - **Play Sounds:** Plays selected files in the internal sound player.
  -  **Help:** The Help menu contains links to various useful web sites (the main [GP Software](#) support site as well as the [Opus Resource Centre](#)), as well as a link to this help file itself. This menu also contains commands to access the [Licence Manager](#), information about the program (about) and the check for updates function. It also has a **Secure Screenshot** command which lets you take screenshots of Opus with the names of potentially sensitive files obscured.

## File Display Toolbar

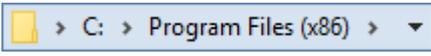
This toolbar is displayed at the top of each file display (assuming the option to use a toolbar for the [File Display border](#) is turned on in [Preferences](#)).



The elements on this toolbar are:

-  **Back:** Navigates back to the previously displayed folder. Right-click (or click and hold with the left button) the button to display a list of all prior folders.
-  **Forward:** Navigates forward in the history (once you have moved back, you can go forward again). Right-click (or click and hold with the left button) the button to display a list of any folders forward of the current position.
-  **Up:** Navigates up to the parent of the current folder. See the [Up, Forwards, Back](#) page for more information on these three concepts.

If you right-click (or click and hold with the left button) the button a drop-down menu is displayed that contains a number of links to special locations, for quick navigation. It also contains a menu of your [recent](#) locations. For users upgrading from Opus 9, this drop-down is the equivalent of the old "Go" menu (and you can still press **Alt+G** to open it).

-  **Favorites:** Displays a drop-down list of your [favorite folders](#), as well as your [SmartFavorites](#) (if enabled).
-  **Location:** A "[breadcrumbs](#)" location field that displays your current location.
-  **Compatibility Files:** In certain folders this button will appear. Clicking it lets you easily move to and from the [compatibility folder](#) for the current location.

The remaining elements are not actually part of the toolbar - they are part of the underlying [File Display border](#).

## Images Toolbar

By default Opus displays the **Images** toolbar whenever a file display is set to [Thumbnails mode](#) (you can change this in [Preferences](#)).



-  **Rotate Left:** Rotates selected image files left by 90 degrees.
-  **Rotate Right:** Rotates selected image files right by 90 degrees.

-  **Convert:** Clicking the main button opens the [Image Conversion](#) dialog for selected images. The drop-down menu contains command that let you quickly convert selected images to a particular format (JPG, PNG, BMP or GIF).
-  **Show:** Displays selected image files in the [standalone image viewer](#).
-  **Set Wallpaper:** Displays a drop-down menu that lets you set the selected image file as your desktop wallpaper.
-  **Sort by name:** Sorts the folder alphabetically (by name).
-  **Sort by date:** Sorts the folder by last modified date.
-  **Sort by date taken:** Sorts the folder by the "date taken" field. Any non-image files or those without a date taken field will be grouped at the bottom. These three sort buttons are provided to make it easy to quickly select commonly desired sorting options (because in Thumbnails mode the file display header isn't displayed, which makes it harder to change the sort order).
-  **Thumbnail size:** This slider lets you adjust the size of the thumbnails in the current file display. The maximum thumbnail size defaults to 256 pixels, but you can adjust this through the [Advanced Preferences](#) page with the `max_thumbnail_size` setting. Double-clicking the slider resets the value to the default size set on the [Thumbnails](#) page in Preferences. The four buttons to the right of the slider let you quickly adjust the size to **32**, **64**, **128** and **256** pixels.

## Dynamic Toolbars

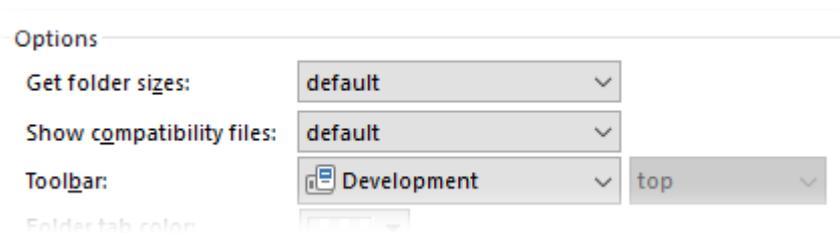
Besides being able to turn toolbars on and off manually, you can configure Opus to show toolbars (or [toolbar sets](#)) automatically based on certain conditions.

- Opus can automatically show a toolbar when the [display mode](#) changes. For example, the [Images](#) toolbar is configured by default to show whenever a file display is set to *Thumbnails* mode.



Display mode toolbars are configured from the [File Display Modes / Toolbars](#) page in Preferences.

- You can configure Opus to automatically show a toolbar whenever you navigate to a certain folder, or to a folder containing predominantly a certain type of file.



Folder-specific toolbars are configured using the [Folder Formats](#) system.

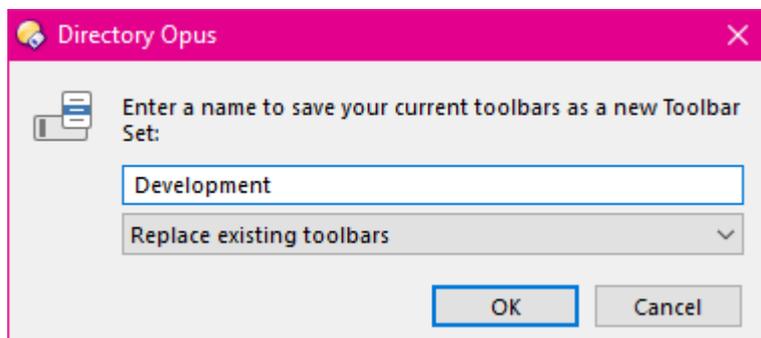
## Toolbar Sets

A toolbar set is one or more toolbars that has been saved as a group. They are useful in two main ways:

- You can load and switch between saved sets using the **Settings / Toolbars** menu (or the context menu displayed when you right-click an empty space on a toolbar). This makes it easy to switch quickly from one set of toolbars to another.
- You can configure Opus to [automatically load](#) a saved set whenever a particular view mode is in use, or a specific folder is visited.

Toolbar sets are created using the toolbars in an existing Lister as a template. To create a new toolbar set you can either:

- Use the **Settings / Toolbars / Toolbar Sets / Save Toolbar Sets** command to save the toolbars in a Lister as a set, or
- Use the controls on the [Toolbars / Toolbar Sets](#) page in Preferences.



When you create a toolbar set you can choose its default behavior - that is, how the set interacts with the existing toolbars in a Lister when it's loaded. The three choices are:

- **Add toolbars to existing toolbars** - any toolbars in the set that aren't already turned on are turned on when the set is loaded. Existing toolbars are unaffected.
- **Replace existing toolbars** - all currently open toolbars are closed and replaced with the toolbars in the set.
- **Toggle existing toolbars** - if all the toolbars in the set are turned on, they are removed and replaced with the toolbars from the default set. Otherwise, any toolbars in the set that aren't currently on are turned on. This option lets you quickly toggle between the default set and another toolbar set.

When the set is loaded the default behavior will be used unless overridden using the arguments to the internal **Toolbar** command. You can change the default behavior of an existing set on the [Toolbars / Toolbar Sets](#) page in Preferences.

Note that the *Default Toolbar Set*, which determines a Lister's initial set of toolbars when it opens (unless overridden by a saved Lister Layout), is not saved as a normal set. Instead, use the **Settings / Toolbars / Set As Default Toolbar Set** command to update your default toolbar set if you want to change which toolbars are used by default.

## Find-as-you-type Field

The Find-as-you-type field (*FAYT*) is a multi-purpose text field that pops up at the bottom of the file display when you type a key (letter or number) that hasn't been otherwise assigned to a hotkey.

The FAYT started out as a visible implementation of the Explorer feature that lets you jump to a specific file by simply typing the first letter or letters of its name. In its default mode that's exactly what it does, and the behaviour is exactly the same except that you can see what you've typed.

As well as typing to jump to a file, the FAYT has other modes, most of which can be triggered by first pressing a specific key.

These are the different FAYT modes:

- **Find:** Type a part of a filename to jump to the first matching file in the current directory.
- **Command:** Enter a command and run it immediately without having to set up a button. (Default key: greater-than >)
- **DOS Command:** Similar to command mode, but the command will run in a DOS window letting you view any text output. (Default key: question-mark ?)
- **WSL Command:** Assuming the Windows Subsystem for Linux has been installed from the Windows Store, this mode lets you run a Linux command in a shell window. (Default key: vertical bar |).
- **Error:** More a color than a real mode; if what you type is not recognised, the FAYT changes to a red background (by default) to indicate the problem.
- **Go:** Type a folder path (e.g. C:\Windows) or folder alias (e.g. /home) into the FAYT to navigate to that location without having to activate the path field.
- **Filter:** Filter out files that don't match the specified pattern. (Disabled by default in favor of the separate [Filter Bar](#).)
- **Range:** When the Index column is displayed in the file display, this lets you select a range of files. (Default key: hash/pound #)

- **Search:** Trigger a Windows Search indexed search. (Default key: equals =)
- **Select:** Select files by entering a wildcard pattern. (Default key: colon :)
- **Tabs:** Search and switch to open folder tabs by path or title. (Default key: at symbol @)

To initiate the FAYT field simply start typing into the file display (you may need to give it input focus first). If the first key you press is one of the assigned mode keys (these can be configured from the [File Displays / Find-As-You-Type](#) Preferences page), the FAYT will be set to the appropriate mode, which is indicated by its background color. If the first key you press is not one of the assigned keys, the FAYT defaults to **Find** mode - which provides the same functionality as typing into an Explorer window to jump to a specific file.

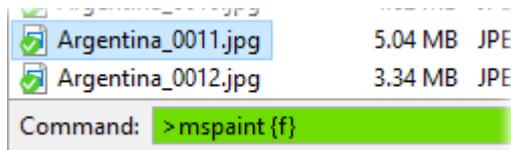


One small difference between the FAYT's **Find** mode and the traditional "type to jump to a file" feature in Explorer arises when you want to skip through multiple files beginning with the same letter. For example, in Explorer if you wanted to skip through all files beginning with N, you would simply press N N N N... to jump from one file to the next. In Opus, doing that will actually search for a file beginning with "nnnnn". Instead, to skip to the next match in Opus you press **F3** (and you can press **Shift-F3** to skip back to the previous match). If you'd like to change this behaviour to be the same as Explorer's, set the **fayt\_firstchar\_repeat** option to **True** on the [Miscellaneous / Advanced](#) page of Preferences. You can also press **Ctrl-T** in the FAYT itself to toggle this setting.

By default the FAYT will match the entered text anywhere in the filename, but you can change this in Preferences to only match at the start of the name.

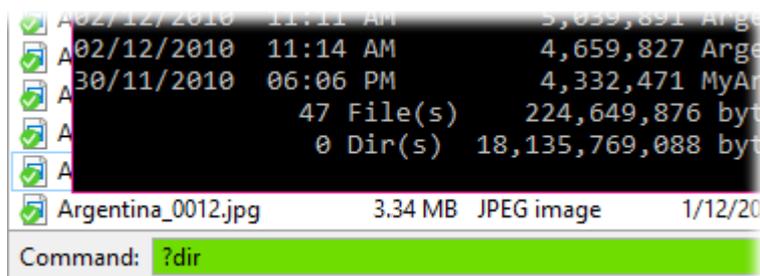
The FAYT will switch into **Go** mode if what you type into Find mode is recognised as the start of a folder path or alias. The FAYT will then act like a path field, offering automatic path-completion and navigating to the location you specify when you push return. The advantage of this over the standard path field on the toolbar is that you can start typing a path immediately, without first having to activate the path field. You can even remove the path field from your toolbar if you want to.

The **Command** mode lets you enter a command and execute it immediately. You can run any Opus internal command, or launch an external program. Any function you can run from a button or hotkey can be run as an ad-hoc command through the FAYT.



As you can see from this screenshot, you can use any of the [external control codes](#) in your command - the above example would open any selected files in Paint. As a shortcut when typing the command, you can also press **Ctrl-Enter** to automatically insert the names of any selected files in the command line. The command mode has a history of your most recently executed commands - press the **Up** or **Down** cursor keys to access it.

The **DOS Command** mode is very similar to the **Command** mode, except that it lets you run DOS commands and will display a DOS window showing you the output of any such commands.



If the Windows Subsystem for Linux has been installed from the Windows Store, the **WSL Command** mode lets you run Linux commands in a Linux shell in the same way DOS commands can be run in a DOS window.

The **Range** mode is only used when the file display is in details or power mode, and the **Index** column is displayed. It lets you select files by index or a range of indexes.

11	✓ Argentina_0005.jpg	4.96 MB	JPE
12	✓ Argentina_0006.jpg	4.86 MB	JPE
13	✓ Argentina_0007.jpg	4.86 MB	JPE
14	✓ Argentina_0008.jpg	5.11 MB	JPE
15	✓ Argentina_0009.jpg	3.87 MB	JPE
16	✓ Argentina_0010.jpg	4.62 MB	JPE
17	✓ Argentina_0011.jpg	5.04 MB	JPE
18	✓ Argentina_0012.jpg	3.34 MB	JPE
19	✓ Argentina_0013.jpg	3.32 MB	JPE
Range: #12,14,16-18			

In this example, the specified range selects file numbers 12 and 14, as well as the range from 16 through 18.

The **Search** mode lets you initiate an indexed [Windows Search](#) in the current folder. It is equivalent to entering a search string into the toolbar search field - but if you would rather remove the search field to save space on your toolbars, you can still access the search function through the FAYT.

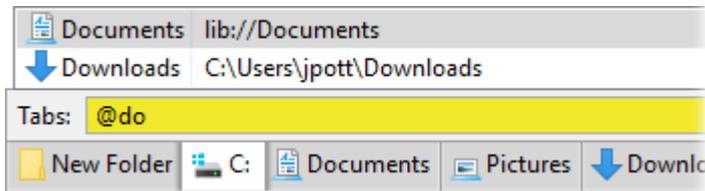
✓ Argentina_0011.jpg	5.04 MB	JPEG im
✓ Argentina_0012.jpg	3.34 MB	JPEG im
✓ Argentina_0013.jpg	3.32 MB	JPEG im
Search: =name:Argentina*		

The **Select** mode lets you select all items in the current folder that match the text you enter.

✓ Argentina_0027.jpg	4.96 MB	JPEG im
✓ Argentina_0028.jpg	3.83 MB	JPEG im
✓ Argentina_0029.jpg	5.10 MB	JPEG im
✓ Argentina_0030.jpg	5.17 MB	JPEG im
✓ Argentina_0031.jpg	4.56 MB	JPEG im
Select: :002		

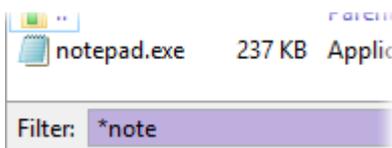
The match is automatically performed in *partial match* mode - meaning that you can either enter a sub-string to match on, or a full wildcard pattern. For example, to quickly select all JPEG files, you could either type **\*.jpg** or simply **.jpg** (although the latter would also match any files that had **.jpg** as part of their name as well as their extension). To invert the selection (i.e. to select all files that don't match the string) type a tilde character (~) at the start.

The **Tags** mode lets you search all open folder tabs, by folder path and tab title. The match is automatically performed in *partial match* mode - meaning that you can either enter a sub-string to match on, or a full wildcard pattern.



Matching tabs are displayed in a popup list and you can switch to one of the matching tabs by selecting it from the list (either using the mouse or keyboard).

The **Filter** mode lets you filter the display to exclude all items that don't match the text or wildcard pattern you enter.



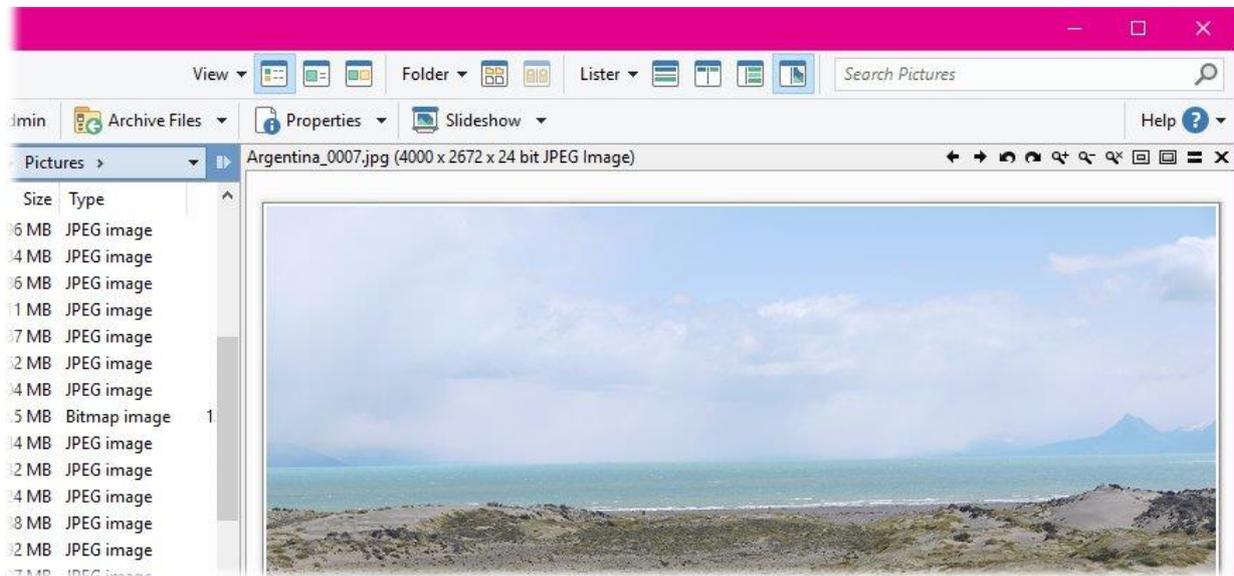
By default, **Filter** mode is not enabled in the FAYT - instead, the [Filter Bar](#) is used to perform a similar function (the dedicated Filter Bar has additional functionality that makes it more powerful than the FAYT for this task). You can enable Filter mode in the FAYT by configuring a key for it on the [File Displays / Find-As-You-Type](#) Preferences page. To invert the filter (i.e. to hide all files that don't match the string) type a tilde character (~) at the start.

If you have filtered the display with the FAYT (or with the Filter Bar) you can quickly clear the filter and redisplay the original contents by pressing the **Escape** key.

## Viewer Pane

The Viewer Pane is an integrated image and document viewer that can be displayed within the Lister. When it is enabled, it will automatically show (if possible) the contents of the most recently selected file. It provides a very quick way to preview files - simply navigate to the folder in question, and click on the file to show it. Using the cursor keys you can easily move up and down a list of files, viewing them as you go.

The easiest way to display the Viewer Pane in a Lister is with the button (  ) on the default toolbar. You can also press **F7**.



With the viewer pane displayed, viewing a file is as simple as selecting it in the file display. Opus can natively show many types of image files, and comes with [plugins](#) to display common document formats like Word and PDF files (providing suitable viewers are installed in the system). The title bar of the viewer pane displays the name of the currently viewed file, as well as some information about it - if it's an image format, it will generally display the resolution and type of image. For files that require the use of a plugin, the name of the plugin will generally be shown.

There are a number of buttons in the viewer pane's title bar:

- **Previous File** : Moves the selection in the file display to the previous file (and displays it in the viewer pane).
- **Next File** : Moves the selection to the next file.
- **Rotate Left** : Rotate the currently displayed image 90 degrees to the left. Note that this does not modify the image on disk - the rotation is not permanent, it simply affects the current display in the viewer pane. If you want to permanently rotate an image you can use the [Image Conversion](#) function.
- **Rotate Right** : Rotate the currently displayed image 90 degrees to the right.
- **Zoom In** : Zoom into (magnify) the currently displayed image. You can also zoom in by holding the **Control** key down and turning the mouse wheel.
- **Zoom Out** : Zoom out of the currently displayed image. You can also zoom out by holding the **Control** key down and turning the mouse wheel.
- **Original Size** : Display the current image at its original (full) size.
- **Fit To Page** : Scale the image to fit to the size of the window. This setting only scales large images down to fit in the window - it does not scale images up that are smaller than the window.
- **Grow To Page** : Scale the image to fit to the size of the window. By contrast with **Fit To Page**, this setting will scale a small image up to fill the window as well as scaling a large image down.

- **Toggle Layout** : This will toggle the layout of the viewer pane between vertical (displayed to the right of the Lister, as in the above screenshot) and horizontal (displayed at the bottom of the Lister).
- **Close** : Closes the viewer pane.

There are a number of options on the [Viewer / Viewer Pane](#) page in Preferences that let you control the appearance and behavior of the viewer pane. One of them, **Show control bar**, lets you enable an additional toolbar at the bottom of the viewer pane.

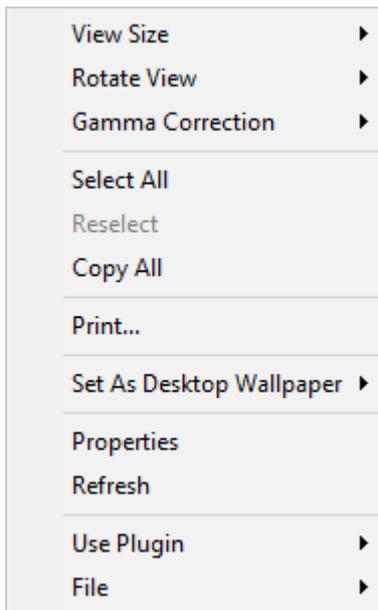


Many of the functions accessible from this toolbar are the same as those in the title bar (although the buttons are bigger, and thus easier to click!) - the buttons are **Previous File** () , **Next File** () , **Rotate Left** () , **Rotate Right** () , **Zoom In** () , **Zoom Out** () , **Original Size** () , **Fit To Page** () and **Grow To Page** () . The remaining buttons are:

- **Hex View** : This switches the display of the current file into hex view. Hex view uses the supplied text plugin (in hex mode) to display the binary contents of the file. Click the button again to switch back to the normal display of the image.
- **Slideshow** : This button enables slideshow mode. Opus will begin showing a slideshow (in the viewer pane) of files in the current folder, starting from the current selection. You can adjust the speed of the slideshow from the [Viewer / Viewer Pane](#) page in Preferences.
- **Full Screen** : This command clears the current file from the viewer pane, and re-opens it in the [standalone image viewer](#), in full-screen mode. You can leave full-screen mode (and remain in the standalone viewer) by pressing the **Enter** key, or press **Escape** to close the separate viewer and return to the Lister.
- **Print** : Lets you print the currently viewed image or document.
- **Settings** : This is a shortcut that takes you to the [Viewer / Viewer Pane](#) page in Preferences.

Depending on the plugin (if any) used to view a particular file, not all of the above functions may be supported. For example, some plugins may not support rotating the display, and in this case the rotate left and rotate right functions would be unavailable.

The viewer pane also has a context menu - although this too can vary depending on the plugin - accessed by right-clicking on the currently displayed image.



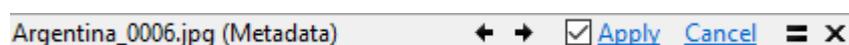
The commands available in the context menu are:

- **View Size:** This lets you change the magnification of the currently displayed image - various preset levels are provided, as well as the two 'fit' modes described above. The option to **Tile** the image is also available.
- **Rotate View:** Rotate the currently displayed image.
- **Gamma Correction:** Modify the gamma correction of the current image - useful if you need to compensate for a darker or brighter monitor.
- **Select All:** Select the entire image for copying to the clipboard. You can also select areas of the image using the mouse - by default you need to hold the **Shift** key down and click and drag to select a rectangular part of the image. If you disable the **Scroll with left mouse button** option on the [Viewer Pane](#) Preferences page then you don't need to hold the **Shift** key down.
- **Copy All:** Copy the entire image to the clipboard. When an area of the image has been selected (as with the **Select All** command, or with the mouse as described above), this command changes to **Copy**, and will only copy the selected area.
- **Print:** Print the currently displayed image.
- **Set As Desktop Wallpaper:** Set the image as your desktop wallpaper. You can choose a number of wallpaper styles from the sub-menu.
- **Properties:** Display the properties of the currently displayed image.
- **Refresh:** Refresh the image. This can be useful if the image or document is being continuously modified in the background (for example, a log file).
- **Use Plugin:** This lets you modify which plugin (if any) is displaying the image. Not all plugins can handle all types of file, of course, but it may be the case that you have multiple plugins installed that can handle the same file, and so this menu lets you experiment with switching between them. You can modify the default order that plugins are used (and also settings that control individual plugins) from the [Viewer / Viewer Plugins](#) page in Preferences.
- **File:** This displays the standard system context menu for the currently displayed file.

## Metadata Pane

Metadata is information that is stored in a file that describes the file itself. For example, photos you take with your digital camera contain metadata like the date and time you took the picture, and the exposure settings used to take it. The Metadata Pane is an integrated metadata viewer and editor that can be displayed within the Lister. When it is enabled, it will automatically show (if possible) metadata relating to the currently selected file or files. For supported fields, the metadata can be edited directly and saved back to the selected files.

The easiest way to display the Metadata Pane in a Lister is with the button () on the default toolbar. You can also press **F9**.



The title bar of the metadata pane displays the name of the currently selected file. The **Previous File** (←) and **Next File** (→) buttons let you move the selection to the previous or next file in the Lister. If you have edited the metadata for the current files, the **Apply** and **Cancel** links will become available - click **Apply** to save your changes or **Cancel** to abort and reload the original metadata. You can save changes automatically and move to the next or previous file by holding the **Shift** key when you click the appropriate button.

You can also have changes saved automatically by turning on the checkbox next to the **Apply** link. This corresponds to the **Apply changes automatically in the metadata panel option** on the Preferences [File Operations / Metadata](#) page.

For a full description of Metadata editing, please see the [Editing Metadata](#) section.

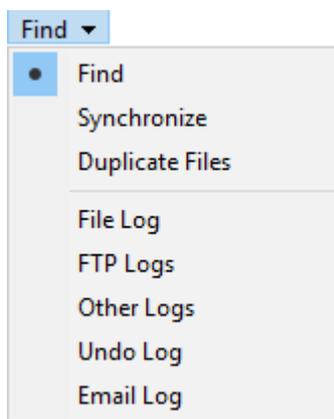
## Utility Panel

The Utility Panel is a multi-purpose panel that appears at the bottom of the Lister whenever certain functions are activated. It acts as a container for several different functions. You rarely turn on the Utility panel directly (although using the raw [Set](#) command you can); instead, it is displayed in a particular mode whenever one of the following commands is chosen:

- **Find Files** from the **Tools** menu: Displays the utility panel in [Find](#) mode, which lets you search your computer for files and folders.
- **Synchronize** from the **Tools** menu: Displays the panel in [Synchronize](#) mode, a function that lets you synchronize the contents of two different folders.
- **Find Duplicate Files** from the **Tools** menu: [Duplicate Files](#) mode lets you search your computer for any duplicated files (or for duplicates of a specific file).
- **FTP Log** from the **Help** menu (**Logs** sub-menu) or **Display FTP Logs** from the **FTP** menu: Displays [FTP](#) site activity.

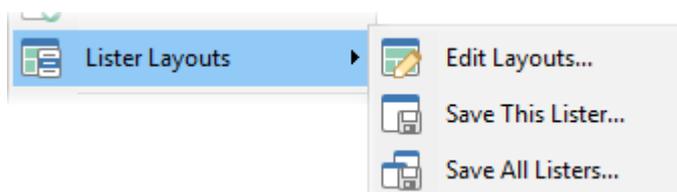
- **File Log** from the **Help** menu: Displays a [log of file actions](#) (files copied, deleted, renamed, etc). The log can be quite handy at times - anyone who's ever dropped files into the wrong folder by mistake and then spent 20 minutes looking for them will know what we're talking about! You can enable the file log (and configure it) from the [File Operations / Logging page](#) in Preferences.
- **Undo Log** from the **Help** menu: Displays the most recent file actions that can be [undone](#) (e.g. if a file has been deleted to the recycle bin, you can recover the file).
- **Email Log** from the **Help** menu: Displays a log of any [outgoing email](#) that Opus has sent.
- **Other Log** from the **Help** menu: Displays miscellaneous logs - currently only [scripts](#) make use of this.

When the utility panel is displayed, you can switch between its various modes using the dropdown in the title bar.



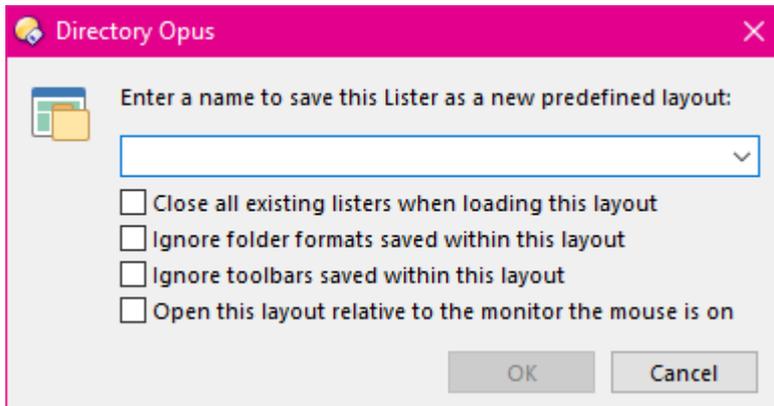
## Layouts

A Lister Layout is a saved collection of one or more Listers that you can re-open at any time. Layouts are created when you save your currently open Listers as a set using the commands in the *Settings / Lister Layouts* menu.



The **Save This Lister** command saves only the currently active Lister, whereas the **Save All Listers** command saves **all** open Listers to the layout. (If you are using the virtual desktops feature in Windows 10 and above, only the windows on the active virtual desktop will be saved by default. You can change this via the advanced [virtual desktop isolation](#) setting.) The **Edit Layouts** command will open the Preferences dialog showing the [Layouts and Styles / Layouts](#) page. From this page you can view a list of all your saved layouts, rename them, re-order them, delete them, etc.

When you save a Layout, you can choose various options that will apply whenever that layout is re-loaded:



Depending on the selected options, almost everything about the state of the saved Lister is remembered, including:

- Its size and position on-screen (can be affected by the **Open this layout relative to the monitor the mouse is on** option)
- Whether it is in single or dual-file display mode
- Whether the folder tree and other UI elements like the viewer panel and metadata pane are open
- The currently displayed folder or folders
- Any folder tabs currently open, and which tab is currently active
- The folder format of the current folder and any folder tabs (unless **Ignore folder formats saved within this layout** is turned on)
- The toolbars currently open in the Lister (unless **Ignore toolbars saved within this layout** is turned on)

The last two items are important to mention. Because the toolbars and folder formats for each folder and tab are stored in the layout, when the layout is re-opened the stored settings will be used, overriding any [saved folder formats](#) or the default toolbars if they have changed. If you want to make a change to a folder format or toolbars stored in a layout, you must either re-save the layout, or use the *Layouts & Folder Tabs* variants in the [Folder Options](#) dialog **Save** drop-down.

You can edit the options for existing Layouts on the [Layouts and Styles](#) Preferences page.

Saved layouts can be re-opened at any time using the layout list in the *Settings / Lister Layouts* menu or the context menu on the desktop (if the option in [Windows Integration](#) is on). You can also create shortcuts to layouts by dragging the layout from the list on the [Layouts and Styles](#) Preferences page, and dropping it on the desktop.

If the **Ignore toolbars** option is turned on for a layout, each Lister will open with the *Default Toolbar Set*. Use the **Settings / Toolbars / Set As Default Toolbar Set** command to update your default toolbar set if you want to change which toolbars are used by default.

## **The Default Lister**

The Default Lister is a special [saved layout](#) that consists of a single Lister. It is generally used whenever a "new" Lister is opened and a specific saved layout is not specified. The default Lister exists automatically - you do not need to create it yourself - and by default, it is updated automatically whenever you close a Lister.

There are many different ways that a new Lister can be opened, and whether the default Lister is used or not depends on the method you open the Lister and any settings that may affect that particular method.

- Opening a new Lister by double-clicking on the Directory Opus program shortcut, or clicking its icon on the Windows 7 taskbar, will generally use the default Lister if Opus was already running. If Opus was not currently in memory, the options on the [Launching Opus / Startup](#) Preferences page control whether the default Lister is used or not.
- If double-click on the desktop is enabled, the default Lister will be used if the **Open the Default Lister** option is enabled on the [Launching Opus / From the Desktop](#) Preferences page.
- If [Explorer Replacement](#) mode is enabled, the default Lister will be used whenever a new Lister is opened due to Explorer Replacement (e.g. double-clicking on a shortcut to a folder on the desktop, or pressing **Windows+E**).
- Double-clicking on the Opus taskbar notification icon will open the default Lister if the appropriate option is enabled on the [Launching Opus / From the Taskbar icon](#) Preferences page.
- A lister opened using the raw [Go NEW](#) command will generally use the default Lister.

The default Lister is **not** used whenever a saved Lister layout is opened - either by manually opening the layout, or by triggering an action (like double-click on the desktop) that is configured to open a layout.

The [Launching Opus / Default Lister](#) Preferences page contains options that control how the default Lister is used. Three options are worth mentioning here:

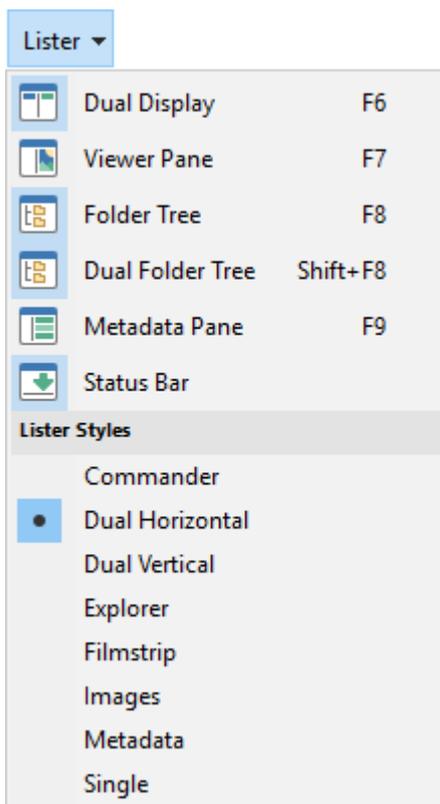
- Because the default Lister is actually a saved layout, it stores within it the folder format that was in use when the layout was saved or updated. If the **Ignore folder format of Default Lister** option is turned on, the folder format stored within the default Lister is not used - instead, the normal folder format for the path being displayed in the new Lister is used.
- Similarly, the toolbars stored in the Default Lister can be overridden using the **Ignore toolbars of Default Lister** option. If that option is turned on, the [Default Toolbar Set](#) will be used instead.

- If the **Update Default Lister automatically when closing a Lister option** is turned off, you can update the default Lister settings manually using the **Set As Default Lister** command in the *Settings* menu.

## Styles

The Lister Styles system lets you configure different configurations of [Lister elements](#) and quickly switch between them. Conceptually they are similar to [layouts](#), except that a style is applied to modify the appearance of an existing Lister - it does not cause a new Lister to be opened like a layout does. For example, you can define a style that opens the [viewer pane](#) and the [metadata pane](#), loads a [toolbar set](#), and closes the [folder tree](#) all in one operation. Styles can also cause a new folder to be read, multiple [folder tabs](#) to be created and can also modify the [format](#) of the currently displayed folder.

You can apply a style to the current lister using the [Prefs STYLE](#) raw command - this is useful if you want to create [hotkeys](#) to quickly switch between multiple Lister configurations. The default toolbars also contain a drop-down menu of styles in the **Lister** menu:



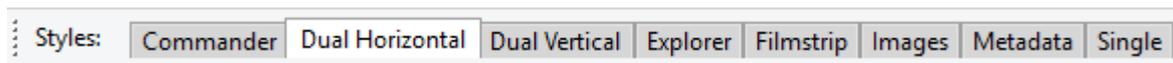
The styles that Opus pre-defines are:

- **Commander:** A dual file display with no tree, similar to the "classical" appearance of some older Windows file managers.
- **Dual Horizontal:** Dual file displays, dual trees. The layout of the file displays is horizontal (one above the other).

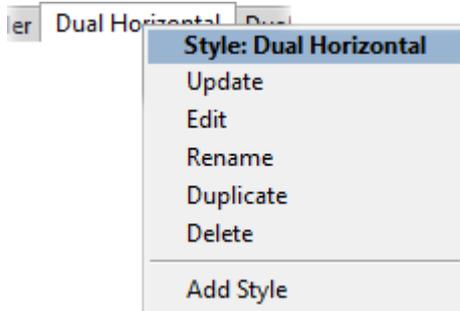
- **Dual Vertical:** Like Dual Horizontal, only laid out with the file displays next to each other.
- **Explorer:** Standard "Explorer" style, with a single tree and a file display.
- **Filmstrip:** Designed for viewing photos - a narrow file display in thumbnails mode, showing a single "strip" of thumbnails, and the viewer pane open to preview the selected image.
- **Images:** Like Filmstrip except with a wider file display.
- **Metadata:** Designed for editing metadata, this style opens both the viewer pane and the metadata pane.
- **Single:** A single file display with no tree.

These styles are just examples; of course you are free to modify or delete them, and you can also create your own. See the documentation for the [Layouts and Styles / Styles](#) Preferences page for a description of how to configure styles.

An alternative way to switch styles is with the style tabs. You can add these to a toolbar from the [Customize](#) dialog - look for the **Lister Styles - Tabs** command in the **View** category.



Both the style tabs and the style list in the drop-down menu support a right-click context menu that lets you quickly edit that style:

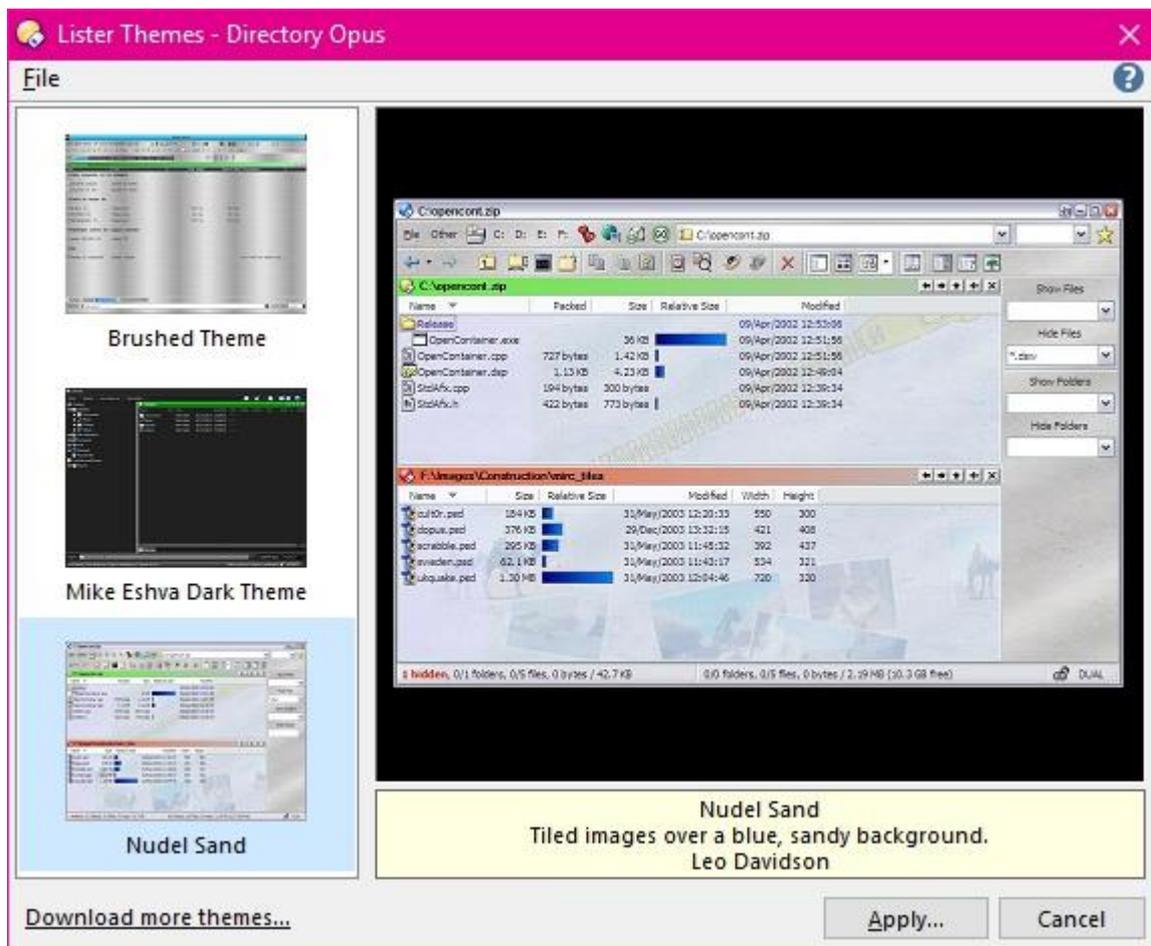


- **Update:** This command will update the selected style with the current settings in the Lister. This makes you quickly make changes to a style without having to go through Preferences.
- **Edit:** Opens the [Styles](#) page in Preferences, with the style selected.
- **Rename:** Change the name of the style.
- **Duplicate:** Make a copy of the style.
- **Delete:** Delete the style.
- **Add Style:** Add a new style using the current settings in the Lister.

## Themes

A theme is a collection of colors, fonts, images, icons and sound settings that can change the appearance of Lister. By importing a theme that someone has created you can radically alter the appearance of Directory Opus. You can also [create your own themes](#) by saving your current Opus configuration as a **.dlt** file and share this with other people.

To access the themes system, select the **Lister Themes** command from the **Settings** menu.



The Lister Themes dialog displays a list of all themes you have installed. Initially your themes list will be empty - you can find themes to download on the [Opus Resource Centre](#).

The **File** menu contains commands that let you manage your installed themes:

- **Import Theme:** Select this command if you have downloaded a new theme and you wish to import it to use in Directory Opus.
- **Export Theme:** Select this command to export one of your installed themes. This doesn't create a new theme - instead, the currently selected theme in the theme list will be exported so you can copy it to another machine or share with a friend.
- **Save New Theme:** Use this command if you want to [create a new theme](#).

- **Apply Theme Settings:** This command will apply the settings from the currently selected theme to your current Directory Opus configuration. You can also click the **Apply** button at the bottom of the dialog. As applying a theme will overwrite aspects of your configuration, you may like to make a [backup of your Preferences](#) first.
- **Delete Theme:** Use this command to delete the currently selected theme.

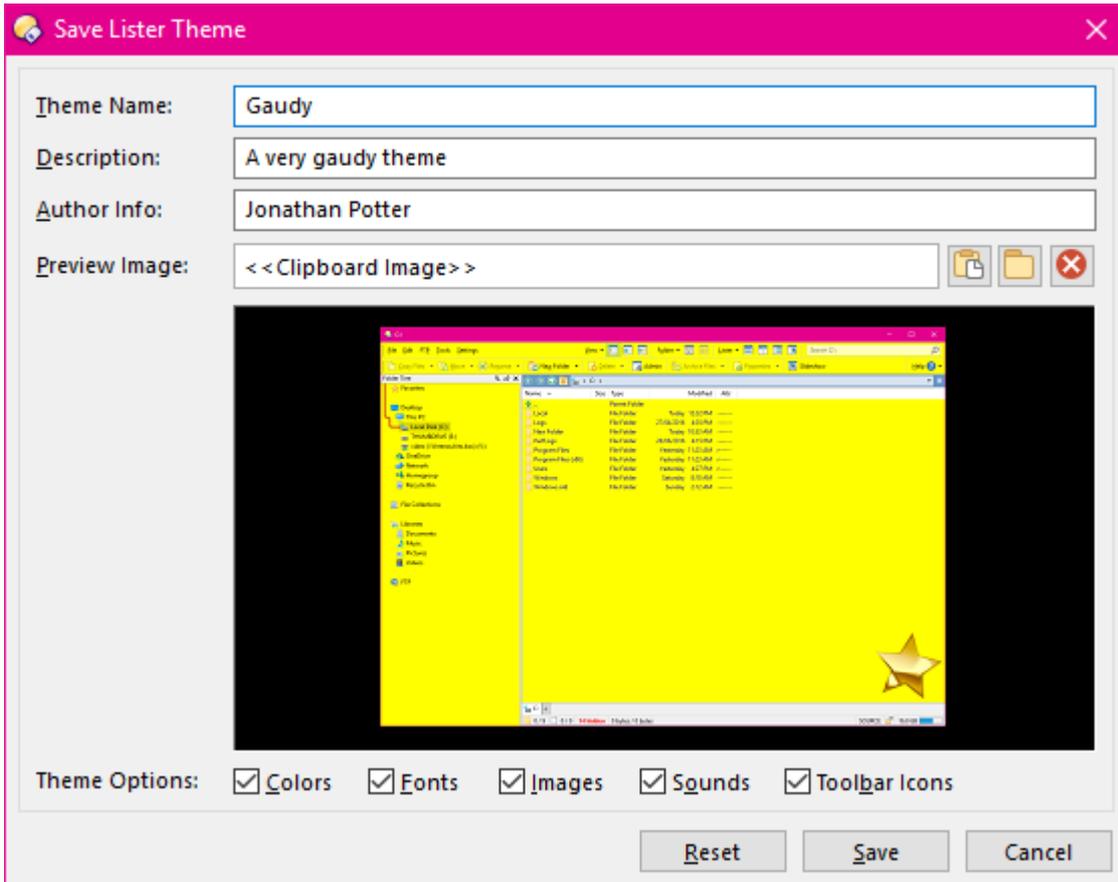
When you select a theme from the list and apply it to your configuration, you can choose which aspects of the theme are applied - you don't have to apply all elements the theme provides. The options when applying a theme are:

- **Colors:** This will apply the colors from the theme; these are mainly the colors specified on the [Colors and Fonts](#) page in Preferences.
- **Fonts:** This will apply the fonts specified in the theme; these are the fonts configured on the [Colors and Fonts](#) page in Preferences.
- **Images:** This applies any Lister background images contained in the theme. A theme can apply background images to all elements of the Lister (although only toolbars and menus that are set to use the *Standard Toolbar Image* will be changed by this). See the Preferences [Images](#) page for more information on background images.
- **Sounds:** If the theme specifies any sound effects these will be applied. These are the sounds configured on the [Miscellaneous / Sounds](#) page of Preferences.
- **Toolbar Icons:** If the theme provides any toolbar [icon sets](#) then these will be applied.

## Creating your own Themes

You can use the [Lister Themes](#) dialog to create a theme from your current Opus configuration. A theme is a collection of colors, fonts, sounds and images that all go towards making up the appearance of a Directory Opus Lister. When you create a theme file you can specify which elements to include in it, and you can then share the created **.dlt** file with your friends.

To create a new theme, select the **Lister Themes** command from the **Settings** menu, and then within the dialog choose the **Save New Theme** command from the **File** menu.



You must provide a number of parameters when saving a theme:

- **Theme Name:** The name of the new theme. This is the name that will be displayed in the themes list in the Lister Themes dialog.
- **Description:** This lets you provide a description of your new theme.
- **Author Info:** You can use this to provide some information about yourself (your name, or a copyright string if desired)
- **Preview Image:** This lets you embed a preview image in the theme file. This preview is displayed as a thumbnail in the themes list in the Lister Themes dialog, and as a larger preview when your theme is selected from the list. It's good to provide a preview so that people you share your theme with have some idea of what they're going to get when they apply it. Or, as in the above example, so they can choose not to apply it!

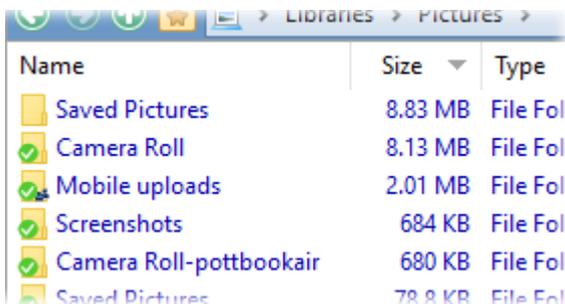
You can provide a preview image in two ways; either you can use the **Browse** (📁) button to select an image file from disk, or you can click the **Paste** (📄) button to paste an image from the clipboard into the theme. This lets you take a quick snapshot of a Lister by pressing **Alt+PrtScr** (which copies the active window to the clipboard) and paste it into your new theme without having to save it as a disk file first. Use the **Clear** (✖) button to clear the preview image.

- **Theme Options:** These options let you choose which aspects of your configuration are to be included in the theme. For example, you may only want your theme to specify colors and images. The options you check will be included in the theme and when the theme is applied, only those aspects of the configuration will be changed. If you choose to export toolbar icons then when you save the theme you will be prompted to select the installed icon sets that you want saved as part of the theme.

Once you have defined your theme click the **Save** button to save it. The theme will be saved to your Directory Opus themes folder, and will show up in the **Lister Themes** dialog immediately. You can then use the **Export Theme** command from the **File** menu of that dialog to export the **.dlt** file to share with others.

## Calculating Folder Sizes

Normally on Windows, only the size of files is easily visible - the (total) size of folders is not usually displayed. This is mainly because the file system does not keep track of the total size of a folder. Instead, displaying a size for a folder involves reading the contents of that folder (and all its sub-folders, recursively) and adding up the total size. There are two ways you can trigger this behaviour in Directory Opus - manually and automatically. Either way, whenever the size of a folder is calculated, Opus displays it in the normal size column along with file sizes.



The screenshot shows a window titled 'Libraries > Pictures' with a table of folders and their sizes. The table has three columns: Name, Size, and Type. The folders listed are Saved Pictures (8.83 MB), Camera Roll (8.13 MB), Mobile uploads (2.01 MB), Screenshots (684 KB), Camera Roll-pottbookair (680 KB), and Saved Pictures (78.8 KB).

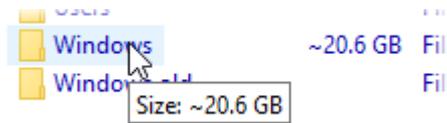
Name	Size	Type
Saved Pictures	8.83 MB	File Fol
Camera Roll	8.13 MB	File Fol
Mobile uploads	2.01 MB	File Fol
Screenshots	684 KB	File Fol
Camera Roll-pottbookair	680 KB	File Fol
Saved Pictures	78.8 KB	File Fol

To manually calculate the size of one or more folders, the simplest method is to select the folders in question, and then choose the **Calculate Folder Sizes** command from the **Edit** menu (or press **Ctrl+K**, or **Ctrl+L** if your toolbars date from an older version). If you execute this command with no folders selected, the size of all sub-folders in the current file display will be calculated.

Another way to calculate the size of a single folder is to hover over it with the mouse until its tooltip appears. The default tooltip for folders contains the **{foldersize}** code, and so displaying a folder's tooltip triggers the counting of its contents. Using this method you have to keep the mouse over the folder (and thus keep the tooltip visible) until the counting is complete. This method may not work, however, if the [info tip](#) for folders has been modified to remove this code.

Opus can also be configured to automatically calculate the size of sub-folders whenever a folder (or certain folders) is read. The [Folders / Folder Behaviour / Calculate folder sizes automatically](#) option controls this globally - through this Preferences option you can enable automatic folder size calculation for certain types of drives. You can also control automatic calculation on a per-folder basis using the [Folder Formats](#) system - the **Get folder sizes** switch on the **Options** tab of the [Folder Options](#) dialog lets you turn counting on or off for any folder.

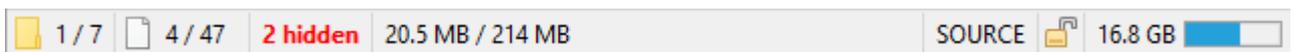
While a folder's size is being calculated, the size field is progressively updated - you will see a tilde character in front of the size to indicate that the displayed size is only approximate.



The tilde is also used to indicate that a folder contained one or more sub-folders that could not be read (for example, due to permissions set to deny read access).

## Status Bar

The status bar at the bottom of the each file display is primarily used to display basic statistics about the current folder.



This is an example of the default status bar. The information presented (from left to right) is:

- Number selected / Total number of folders
- Number selected / Total number of files
- Number of hidden items (if any items are currently [hidden by filters](#))
- Total size of all selected items / Total size of all items (this includes the size of any selected folders only if their [sizes have been calculated](#))
- [Source / destination](#) indicator for the current Lister (or file display)
- [Format lock](#) indicator (🔒)
- Space free on current drive
- Percentage of used space on current drive

You can use the [Display / Status Bar](#) page in Preferences to configure the information displayed in the status bar. Using the various available status bar codes it is possible to configure some quite complicated status bar displays:

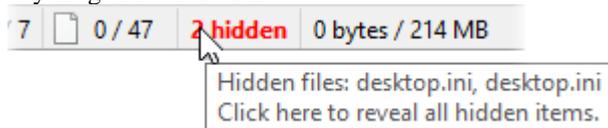


The options on that Preferences page also let you change the status bar to appear at the bottom of the Lister instead of the file display (so, for example, you can have one single file display at the bottom of the window that applies only to the active file display instead of each display having its own).

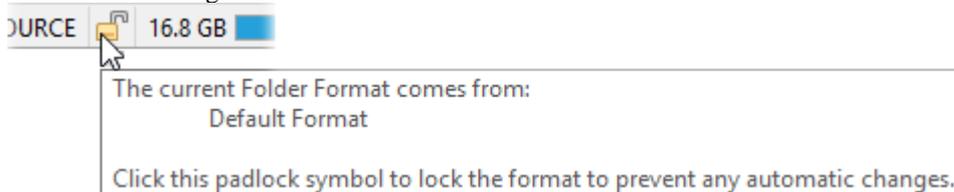
## Status Bar Tooltips

Some status bar elements will show you more information when you hover over them:

- Hidden item indicators show the names of the first few hidden files and folders, so you can quickly see if anything of interest is hidden.



- The format lock indicator shows you where the current folder format comes from, to help you determine how to affect changes to the format.



# Explorer Replacement

One of the key features of Directory Opus is its *Explorer Replacement* mode. To clarify exactly what we mean by Explorer Replacement:

*whenever an action is taken that would ordinarily result in an Explorer window opening, a Directory Opus window (Lister) will open instead.*

That's to say, Opus does **not** replace the desktop (which is actually implemented by explorer.exe), nor does it replace the standard File Open / Save dialogs in other applications.

Explorer Replacement mode is controlled by the options on the [Launching Opus / Explorer Replacement](#) Preferences page. Selecting any option other than **Don't replace Explorer** activates Explorer replacement mode.

When Explorer Replacement mode is enabled, it is still possible to get to Explorer if desired:

- From the Start Menu, select **Run** and enter *explorer.exe*
- Right-click on any folder and choose **Open in Explorer** from the context menu
- On Windows 7 and above, click the pinned Explorer icon on the taskbar

Please note that even when Explorer replacement mode is enabled, there are some cases when Explorer will continue to open:

- Subject to the setting on the [Launching Opus / Explorer Replacement](#) page, a Lister may not open for all types of folder.
- A program that [specifically invokes Explorer](#) may not be able to be intercepted by Opus (although many times it will).
- Except in Windows XP, Opus will never open the Control Panel (or its various sub-pages) - they will always open in Explorer.

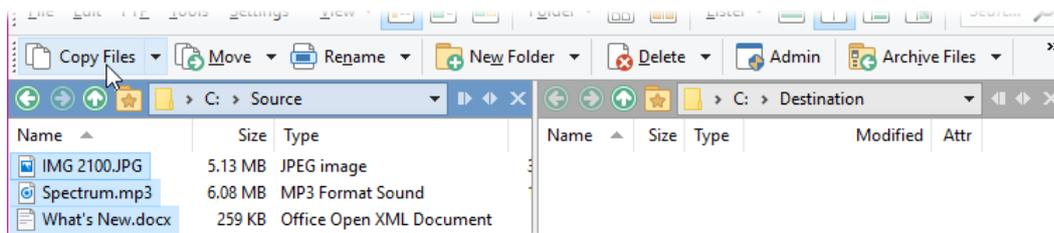
Explorer Replacement mode is generally unavailable when running a [USB exported version](#) of Directory Opus.

## Source and Destination

Many functions in Opus use the concept of a "source" folder and a "destination" folder. Even if you're new to Opus you are probably already familiar with this concept without realising it - whenever you copy a file and paste it somewhere else, you are copying it *from* the source folder and pasting it *to* the destination folder. This concept is made more obvious in Opus by the ability to have a dual-display Lister, or multiple single Listers at once.

- When a Lister is in [dual-display](#) mode, one side is always designated the *source*, and the other side is the *destination*.
- When a Lister is in single display mode, it can either be designated as the *source*, the *destination* or *off*. See below for more information about using single display Listers.

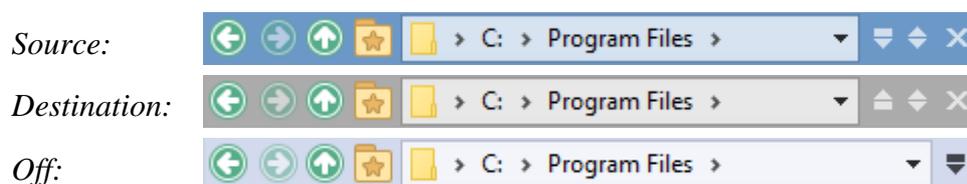
Functions like **Copy Files** or **Move** operate on selected items in the source folder, and use the destination folder as the target.



So in the simplest example, clicking the **Copy Files** button in the above Lister would cause the three selected files on the left to be copied to the folder on the right.

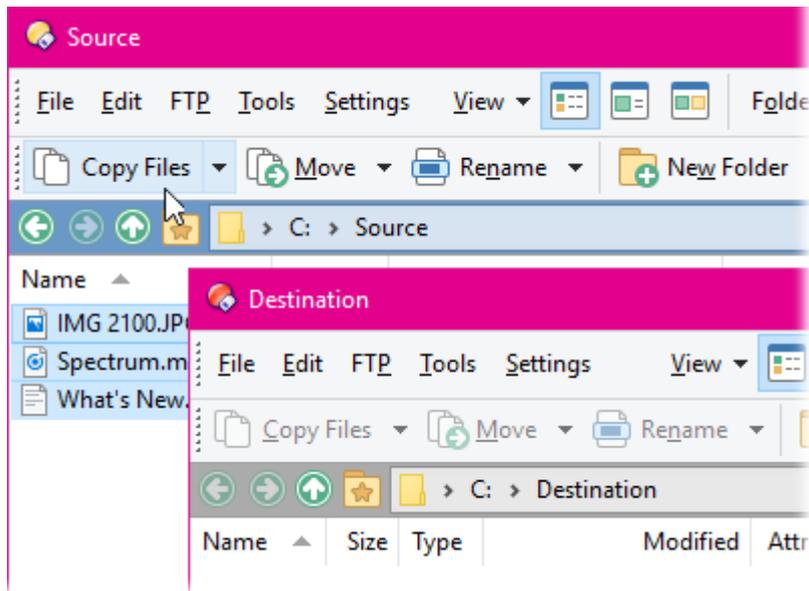
In a dual-display Lister, the source file display is always the active one - the one with input focus. You can make a file display active by clicking in it - therefore, it follows that the last file display you clicked in will generally be the source. You can also switch the source/destination states of the file displays by clicking on the status bar - this is useful to remember, as it allows you to change state without the risk of accidentally selecting a file (or losing the existing file selection) which could otherwise happen.

The source/destination concept operates slightly differently in a single display Listers. Because the Lister has only one file display, the source/destination state applies to the Lister as a whole. An individual Lister can be set as the source, the destination, or can be turned off. The state of a single-display Lister is indicated by the color of the icon in the top-right corner, as well as the inner file display border (if enabled).



These are only the default colors - you can modify them through Preferences.

Clicking the file display of a single-display Lister will set that Lister to source mode. The previous source Lister (if there was one) will automatically switch to destination mode - and the previous destination Lister (again, if there was one) will be set to "off". If a single-display Lister is set to off it will not participate in any source/destination operations.



Functions like **Copy Files** work with single-display Listers just as they do in a dual-display - the files will be copied from the source Lister to the destination. Note that when a Lister is in dual-display mode it is removed from the source/destination list - it won't be considered as a valid destination when copying from a single-display.

The source/destination model is an alternate way of working to the standard windows "copy and paste" or "drag and drop" methods of moving files around. Whichever way you want to work is up to you!

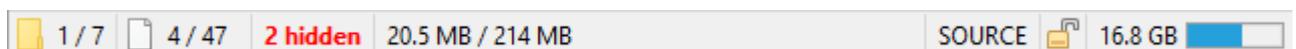
# Selecting Files

Whenever you want to do something with a file (copy it, delete it, or just view it) you must select it first. Most file operations in Directory Opus operate on all currently selected files (and folders), so it's important to know how to select files, and how to tell how many files you have selected.

The main ways files can be selected are:

- [Using the mouse or keyboard](#) to select files directly
- Using [simple wildcard patterns](#) to select files by name
- Using [advanced filters](#) to select files by multiple different criteria
- Using the raw [Select](#) command to programmatically select items. The default **Edit** menu has a number of pre-defined **Select** commands:
  - **Select All**: Select all items in the current folder.
  - **Select by Pattern**: Displays the Simple wildcard selection dialog.
  - **Invert Selection**: Inverts the selection state of all items in the folder.
  - **Select Other / Reselect Files**: Select all files that were used (and deselected) by the previously executed command.
  - **Select Same Extensions**: Selects all files with the same extensions as the currently selected files (e.g. select one **.txt** file and then use this command to automatically select all other **.txt** files).
  - **Select Source to Destination**: Select files in the destination with the same names as the files currently selected in the source.
  - **Select Source-not-in-Destination**: Select those files in the source which do not exist in the destination.
  - **Selection to Checkboxes**: Set the checkbox state to match the current selection. The file display will be set to [checkbox mode](#) if it isn't already.
  - **Checkboxes to Selection**: Set the selection to match the checkbox state.

The easiest way to tell how many files and folders are currently selected is to look in the [status bar](#).



The status bar shows you the number of files and folders selected (and the total number) in the current file display. Whenever you click a button like **Copy Files** or **Delete**, it is these selected files that will be used in the function.

## Selecting files with the mouse and keyboard

Selecting files with the mouse or keyboard, in all [view modes](#) except Power mode, works similarly to Explorer:

## Using the mouse,

- Left-clicking a single file will select that file and deselect all others.
- **Control** + left-clicking a single file will toggle the selection state of that file, leaving all others unchanged (i.e. **Control**+click to select, repeat to deselect).
- **Shift** + left-clicking lets you select a range of files. Click normally to select the first file in the range, then hold the **Shift** key and click again on the last file in the range. All files between and including those two will be selected.
- You can combine **Control** and **Shift** to add a range to already selected files - without the **Control** key held down, the first click in a range selection normally deselects all but the file you clicked on.
- Clicking on an empty part of the file display (i.e. not on a file) will deselect all files.
- Click on an empty part of the file display and (with the button still down) move to mouse to drag out a selection rectangle. All files you drag that rectangle over will be selected.
- If you hold the **Control** key down when you click to drag the selection rectangle, existing selections will be left unchanged. Additionally, files that you drag over will have their selection states inverted.
- You can middle-click on a file to toggle its selection state (equivalent to **Control** + left-clicking). You can also use the middle button to drag a selection rectangle that inverts the selections of files you drag over (equivalent to **Control** + left-drag). Note that some mouse drivers map the middle mouse button to another function by default - if you find middle-click isn't working as expected, see the [FAQ](#) for some hints on how to fix it. You can also configure Opus to treat a single middle-click as a double middle-click, which lets you obtain behavior similar to a web browser (e.g. single middle-click to open a folder in a new tab). See the [File Displays / Mouse](#) Preferences page for details on that.
- You can also use the right button to drag a selection rectangle; once the button is released, the context menu will be shown for all selected items.

## Using the keyboard,

- **Ctrl+A** is the default hotkey to select all items in the list (the same as selecting the **Select All** command from the **Edit** menu).
- **Ctrl+I** will invert the selection state of all items in the list (the same as the **Edit / Invert Selection** command).
- The cursor keys (up/down/left/right, plus Page Up/Down and Home/End) can be used to move the selection (and deselect all other items) in the file display when it has input focus - click on it or use the **Tab** key if it doesn't already have this.
- You can combine **Shift** with the cursor keys to perform range selection using the keyboard - move the selection to the first file in the range, and then hold **Shift** and use the cursor keys to expand the range to the last item.
- Holding the **Control** key lets you move the focus rectangle without changing the selection. You can use this to select multiple, non-contiguous files. For example, use the cursor keys to place the selection on a given file. Then hold the **Control** key down and use the cursor keys to move the focus rectangle to another file - note that the first file is not deselected.
- Pressing the **Space** bar when an item has input focus will select it. You can toggle the selection on the currently focused item by pressing **Control+Space**.
- Pressing the **Insert** key toggles the selection of the current item, and automatically moves the input focus to the next item in the list.

- Typing a partial filename will activate the [find-as-you-type](#) field and place the selection on the first file matching the entered string.

In power mode, mouse and keyboard selection works slightly differently by default (although the behaviour of the mouse can be modified quite a lot from the [File Display Modes / Power Mode Preferences](#) page).

#### Using the mouse,

- Left-clicking a single file will invert the selection state of that file, but will not disturb the selection state of any other items.
- You can select a range of files by simply clicking and dragging up or down (i.e. keep the mouse button down and move the mouse cursor up or down). If the file you click on first is already selected, dragging up or down will deselect all files dragged over.
- Clicking an empty part of the file display does not, by default, deselect any files. You can drag a selection rectangle by clicking on an empty part of the file display and dragging with the button held down - the selection rectangle will invert the selection state of any items you drag over.
- Note that because dragging up or down is used to select files, a drag and drop operation can only be initiated by dragging left or right.

#### Using the keyboard,

- Power mode is primarily a mouse-based interface, and by default Power mode file displays are not in "keyboard mode" - so pressing the cursor keys will simply scroll the list (if it can scroll) and will not select any files.
- Pressing the **Control** key toggles the file display in and out of keyboard mode. When in keyboard mode, a focus indicator is shown and the cursor keys will move the focus indicator from one file to the next (although they will not, by themselves, cause any files to be selected).
- Pressing the **Space** bar when in keyboard mode will invert the selection state of the file that currently displays the focus indicator.
- When in keyboard mode you can select a range of files by holding the **Shift** key and using the cursor keys to move the focus indicator. The state of the first file when the **Shift** key was pushed determines the selection state applied - if the first file was not already selected, all files in the range will be selected, and vice versa.
- When range-selecting with the **Shift** key, reversing direction will undo the selection you just applied. For example, if you press **Shift+Cursor Down** four times to select the first four files, and then (while still holding **Shift**), press **Cursor Up** twice, the last two selected files will be deselected.
- **Ctrl+A** and **Ctrl+I** work the same as with the other view modes (see above), as does the [find-as-you-type](#) field.

The default behaviour of power mode is for file selections to be persistent (that is, files do not deselect by themselves - you have to deselect them manually). In other file display modes you can simply click on an empty area to deselect all files, but in power mode:

- You can either press **Ctrl+A** followed by **Ctrl+I** (select all, followed by invert selection) to deselect all files, or
- Use the [Customize](#) dialog to add the [Select None](#) command to your toolbars (e.g. in the **Edit** menu). This is not included in the default toolbar set, but you may want to add it manually and assign a hotkey to it if you are going to be using power mode.

### ***Single-click mode***

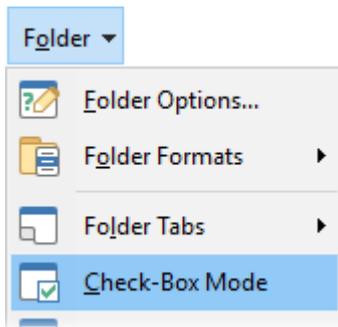
Single-click is an optional setting that changes the way files are selected with the mouse. It makes files behave kind of like links on a web page - files are opened with a single-click instead of a double-click, and you can select a file simply by hovering over it with the mouse. Some users prefer this way of working as it reduces the number of mouse clicks (especially double-clicks) that you need to do.

Single-click can be enabled with the **Single click to open an item** option on the [File Displays / Options](#) Preferences page. This option has no effect in Power mode, but in all other [view modes](#), the behaviour of the mouse is modified as follows:

- If you move the mouse over an item and remain there for more than half a second or so, the item will be automatically selected, and all other items are deselected.
- You can select multiple items by hovering with the **Control** key held down. You can also range select using the **Shift** key. Effectively this is the same behaviour as normal, except you don't actually click to perform the selection - simply hovering over a file is enough.
- If you click on an item with the left button the item will be opened (equivalent to double-clicking on it normally).

### ***Check-Box Mode***

Check-box mode is a temporary mode that can be turned on in a file display (it is not a global setting like [single-click mode](#)).



Check-box mode can be turned on with the command in the **Folder Options** menu. By default Check-box mode only remains turned on until you navigate to a new folder, or close that file display (or manually turn it off). If you turn off the **Cancel Checkbox mode when folder is changed** option on the [Folders / Folder Behaviour](#) page then the mode will remain enabled even after the folder changes.

Name ▲	Size	Type
<input checked="" type="checkbox"/> IMG 2100.JPG	5.13 MB	JPEG imag
<input checked="" type="checkbox"/> Spectrum.mp3	6.08 MB	MP3 Form
<input type="checkbox"/> What's New.docx	259 KB	Office Ope

As the name suggests, check-box mode causes check-boxes to be displayed next to each item. This changes which files are considered for use in file operations (for example, when you click the **Copy Files** button):

- Normally (when not in check-box mode), functions like **Copy Files** act on all *selected* files.
- When in check-box mode, functions like **Copy Files** act on all *checked* files.

So in the above screenshot, clicking the **Copy Files** button would copy the files **IMG2100.JPG** and **Spectrum.mp3** to the destination, but not **What's New.docx**.

The main advantage of check-box mode is that it provides selection persistence in the view modes where it is normally very easy to deselect files accidentally. Clicking an empty area of the file display (or indeed, clicking any file) will still cause all files to be deselected, but their *checked states* will not be disturbed. One practical outcome of this is that you can double-click on a file to open it, or select files and drag-and-drop them, without modifying the checked states of any other files in the list. Let's look at a real-world case: imagine that you want to browse through a folder of your holiday photos, to identify the ones you want to send to your friends.

1. Turn on Check-box mode in the file display, and open the [Viewer pane](#) by clicking the button in the toolbar (or press **F7**).
2. Click on the first file in the folder to select it. It will be displayed in the Viewer pane.
3. If you want to keep that image, click its check-box with the mouse (or, press the **Space** bar) to check it.
4. Click on the next image in the folder (or, press **Cursor Down**) to preview the next image, and so on.

At the end of this process, you will have viewed all the images in the folder, and the ones that you wanted to keep will still be checked. You can then use **Copy Files** to copy them to another folder (or alternatively, use the **Edit / Invert Selection** command to invert the check states, and then delete the files you don't want to keep). Without check-box mode you would have had to copy or delete each file as you viewed it - this way, you can concentrate on viewing the images, and then copy or delete them as a batch at the very end.

The main points to keep in mind about check-box mode are:

- When in check-box mode, functions that normally act on selected files instead act on checked files.
- Selection commands like **Edit / Select All** or **Edit / Invert Selection** will affect the checked state of files rather than the selected state.
- You can double-click on a file to open it without losing the check states of other files.
- Drag-and-drop still works on selected files, not checked ones.
- Pressing the **Space** bar toggles the check state of all currently selected files.

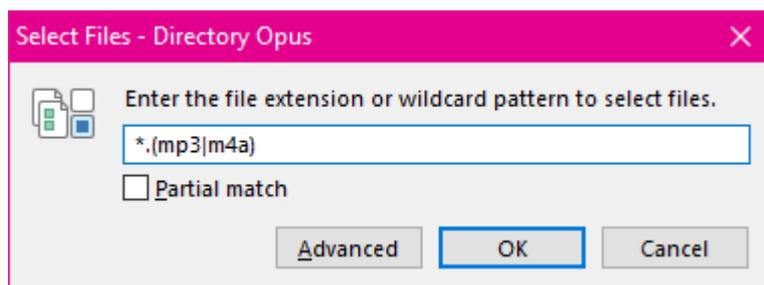
There are two commands in the **Edit / Select Other** menu that can be useful when in check-box mode:

- **Selection to Checkboxes:** This command copies the current selection state of each file to its checked state (so selected files will be checked, and unselected files will be unchecked). If the file display is not already in check-box mode when you run this command it will be turned on automatically. This could be useful if you have started selecting multiple files and decide that check-box mode would be a better way to accomplish your task.
- **Checkboxes to Selection:** The reverse of the above command - the current checked state of each file will be copied to its selection state. This is useful if you have checked some files and then want to select them for drag-and-drop.

The [image marking](#) function in the [standalone image viewer](#) can make use of check-box mode (if the default option to use a File Collection is turned off). When you tag a file through the viewer in this mode, the file display it came from is automatically placed in check-box mode, and the file in question is checked.

## Simple Wildcard Selection

The **Select Files** dialog has two modes; *simple* and *advanced*. It is accessed with the **Edit / Select by Pattern** command. The dialog remembers which mode it was last used in and will open in that mode the next time it is used; you can switch modes using the **Advanced** or **Simple** button at the bottom of the dialog.



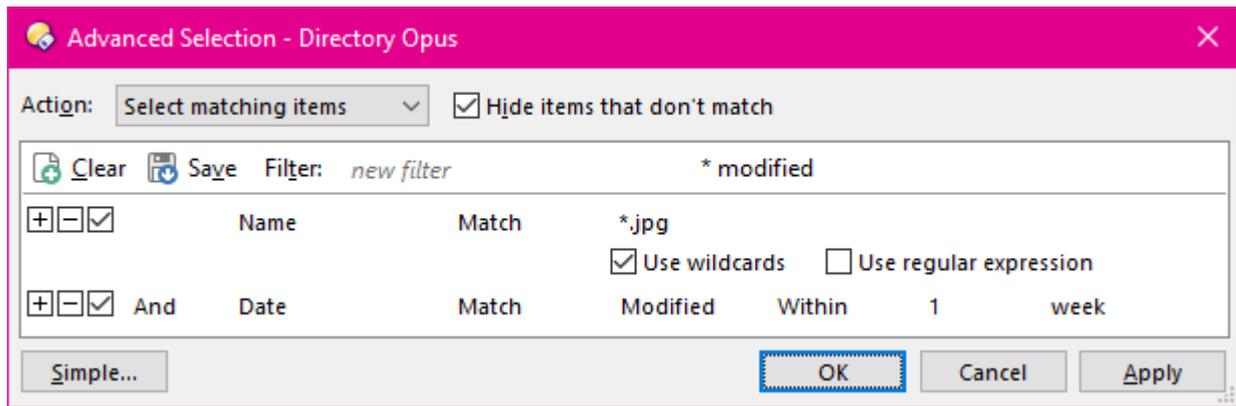
The simple-mode **Select Files** dialog lets you select files in the current file display by typing in a wildcard string using the [standard pattern matching syntax](#). In the example shown above, the string **\*.(mp3|m4a)** will select all files that end in either **.mp3** or **.m4a**, when you click the **OK** button.

The **Partial match** option enables partial matching of the string you enter - for example, **kit** would match **Cute Kitten.jpg** if that option was turned on.

You can click the **Advanced** button to go to [advanced mode](#), which lets you use more complex filters to select files.

## Advanced Selection

The **Select Files** dialog has two modes; *simple* and *advanced*. It is accessed with the **Edit / Select by Pattern** command. The dialog remembers which mode it was last used in and will open in that mode the next time it is used; you can switch modes using the **Advanced** or **Simple** button at the bottom of the dialog.



The **Advanced Selection** dialog is built upon the file filter control. It lets you select files by building up complex filters that can compare files and folders against many different attributes. Please see the section on [Filtered Operations](#) for a full description of the filter system. The above example shows a filter that would select all files that end in **.jpg** that were modified within the past week.

At the top of the dialog, the **Action** drop-down lets you choose the action to perform:

- **Select matching items:** Any files and folders that match the defined filter will be selected.
- **Deselect matching items:** Any files and folders that match the filter will be deselected. This lets you perform a simple "not" operation - you could first use the **Edit / Select All** command to select all files in the folder, and then use this action to deselect files based on the filter.
- **Hide matching items:** Any files and folders that match the filter will be removed from the display

The **Hide items that don't match** option can be used in conjunction with both the **Select** and **Deselect** actions (it doesn't make much sense to use it with the **Hide** action). If this option is enabled, items that don't match the filter will be removed from the display altogether.

When items are removed from the display (hidden) by the use of this function, they can be brought back in two ways:

- The easiest way is to simply refresh the folder listing by pressing **F5** (or select **Refresh** from the **Go** menu). This will re-read the folder and redisplay any files you had hidden using the selection function.
- They can also be redisplayed by running an internal command. This command is not present on the default menus or toolbars and so you would need to [create a button](#) for it or assign it to a [hotkey](#). The command you would use for this is **Select NOPATTERN SHOWHIDDEN**. Please see the documentation for the internal [Select](#) command for more information on its command line parameters.

Once you have built your filter, or selected one from the drop-down Filter list, click the **OK** button to make the selection. Alternatively, you can click the **Apply** button - this lets you apply

the selection without closing the dialog. You could then edit the filter and click **Apply** again if your selection filter didn't quite perform as expected.

Clicking the **Simple** button at the bottom of the dialog takes you back to [simple mode](#).

# Searching and Filtering

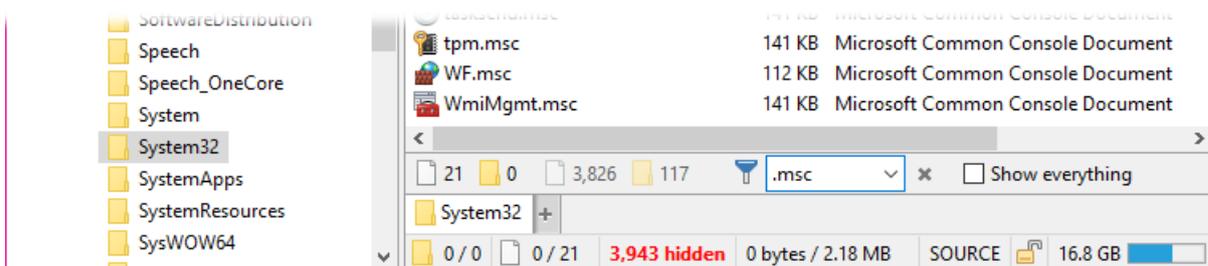
While working in folders you will often encounter the situation where there are many more files displayed than the ones you actually want to work with. There are two main systems for reducing the number of files displayed to those you specifically want to target:

- Filtering is used to hide files from the current display of a folder - for example, you might hide all files ending in **.bak**, or filter the display to only show image files. You can filter the list in two ways - with the quick filter using the [Filter Bar](#), or with the various filter fields in the [Folder Options](#) system. There are also some global filtering options on the [Folders / Folder Display](#) Preferences page that you can use to always hide particular files.
- Searching is used to locate files in and below the current folder - you might search for all MP3 files in the current folder or its sub-folders. The easiest way to search is with the indexed [Windows Search](#) system, although you can also use the [Find Files](#) function to perform more structured searches.

The main difference between filtering and searching is that filtering operates on the current list of files and removes files from the display, whereas searching starts from the current location and builds a new list of files that match the supplied criteria.

## Filter Bar

The filter bar is a small text field that pops up at the bottom of the file display, and makes it easy to quickly filter the current file list. By default you press the asterisk (the \* key) to display the filter bar, although this key can be modified through the [File Displays / Filter Bar](#) Preferences page.



The filter bar appears as soon as you press the activation key (\*) and you can continue typing immediately to begin filtering the list.

In the above screenshot, we have filtered the C:\Windows\System32 directory to show all files that end in **.msc**. You can see from the status bar that this resulted in 3,943 items being hidden from the display, leaving only 21 **.msc** files visible. The filter bar displays these statistics as well, in slightly more detail (it breaks the hidden item count into hidden files and hidden folders). The various icons on the filter bar are actually buttons that you can click to quickly enable or disable that element of the filter. From left to right, the filter bar elements are:

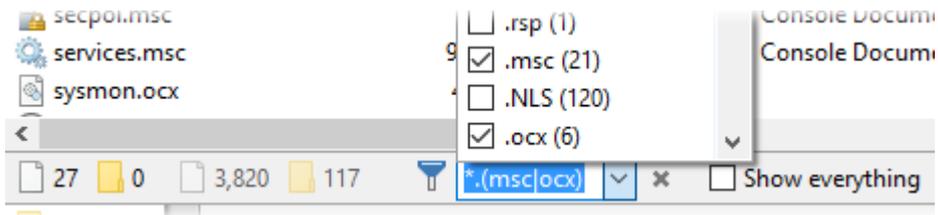
-  **21 Displayed file count:** Shows the number of files that are displayed by the filter (i.e. those that aren't hidden). Click this icon to temporarily *hide all* files.
-  **0 Displayed folder count:** Shows the number of folders that are displayed by the filter. Click this icon to temporarily *hide all* folders.
-  **3,826 Hidden file count:** Shows the number of files that are hidden by the filter. Click this icon to temporarily *show all* files.
-  **117 Hidden folder count:** Shows the number of folders that are hidden by the filter. Click this icon to temporarily *show all* folders.
-  **Filter:** Click this icon to temporarily disable the filter (all files and folders will be shown).
-  **Close:** Click this to close the filter and restore the display of all items in the list. You can also press **Escape** to do this. (If you are editing a previously-set filter, the first time you push **Escape** it will reset the filter to how it was. Push **Escape** a second time to clear the filter entirely.)
- **Filter folders in Flat view:** When the file display you are filtering is in [Flat View mode](#) this option appears and lets you control whether the filter applies to folders or only to files. When a folder is filtered out of the display in Flat View mode, all files within that folder (and its sub-folders) are also filtered out. This behavior may or may not be desirable and it can be confusing to have a whole lot of files suddenly disappear unexpectedly, which is which this option is available (and off by default). The default state of this option can be changed by modifying the `flatview_folder_filters` option on the [Miscellaneous / Advanced](#) page in Preferences.

The filter string uses the [standard pattern matching syntax](#).

By default, *partial matching* is used, which means the pattern you enter only has to match a substring of the filename. For example, the pattern **opus** will match **dopus.exe**. If you turn partial matching off through Preferences, the filter pattern must match the filename exactly (so, instead of just **opus** you would need to use **\*opus\*** to match **dopus.exe**). Note that if you explicitly add a **\*** at the start or end of the pattern then Opus will assume you do not wish to use partial matching even if it is switched on. This allows you the convenience of partial matching most of the time while still being able to filter by the start or end of things when you need to.

The **Show everything** option provides a quick toggle for the [Show Everything](#) mode - a way to temporarily disable filters and display all files and folders in the current folder.

Other options that you can control through the [Preferences page](#) include when the filter bar should stay visible, and whether the filter is cleared automatically whenever you change folders.



The drop-down attached to the filter field provides a quick way to filter on the various file types that are actually present in the current folder. As you turn extensions on in the drop-down list they are automatically added to the filter wildcard pattern. The drop-down will also list your [file type groups](#), which is a very quick way to quickly filter (for example) out everything that's not an image file.

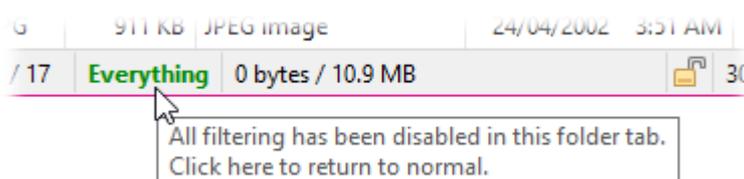
The filter bar affects what is called the "quick filter". This filter is applied on top of the filters set via the [Folder Options](#) system, and the global filtering options on the [Folders / Folder Display](#) Preferences page. It doesn't override these filters - so any files hidden via these other methods will stay hidden no matter what the quick filter is set to. You can clear the quick filter at any time by pressing the **Escape** key. (If you are editing a previously-set filter, the first time you push **Escape** it will reset the filter to how it was. Push **Escape** a second time to clear the filter entirely.)

You can also use the [find-as-you-type](#) field to edit the quick filter (although it doesn't have the additional functionality the filter bar does). To do this you need to assign an activation key for the **Filter** mode on the [File Displays / Find-As-You-Type](#) Preferences page. See the documentation for the find-as-you-type field for more information on this.

If desired you can configure the file display to display a different background color whenever the quick filter is active, using the File display background settings on the [Display / Colors and Fonts](#) page in Preferences.

## Show Everything

*Show Everything* is a convenient way to quickly see all files in the folder without actually clearing any filters that may be in place. It's local to the folder tab it is used in.



You can activate *Show Everything* mode by clicking the count of hidden items on the status bar. The default status bar displays **Everything** if *Show Everything* mode is active.

You can also turn it on using the **Folder > Show Everything** command on the default Menu toolbar. The Filter Bar has a **Show everything** checkbox which also lets you toggle the mode on and off.

*Show Everything* disables all types of filtering within the current folder tab, with one exception: It does not affect folders hidden while in [Flat View Mixed \(No Folders\)](#) mode.

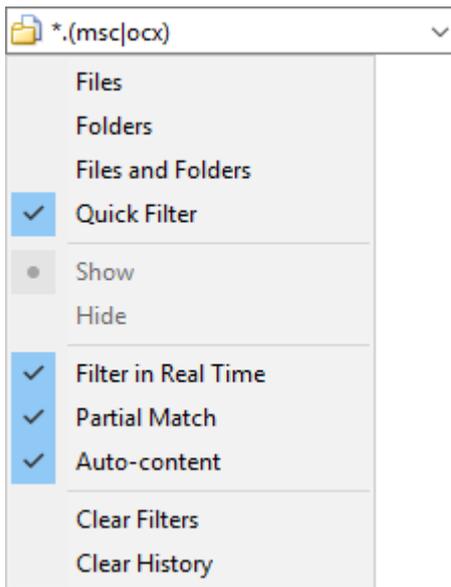
## Toolbar Filter Field

The **File Filter Field** is a field that you can add to your toolbars that lets you filter the current file display. Normally it's recommended that you use the [filter bar](#) rather than the toolbar field, simply to save space on your toolbars, but if you prefer to have a filter field always visible adding it to your toolbars is straightforward. Use the [Customize](#) dialog to add the field to a toolbar - you'll find the **File Filter Field** listed in the [View](#) category on the [Commands](#) tab.



When you leave Customize mode you'll see the filter field appear. The current filter string is displayed in the field - if you click in the field and edit this string, the displayed files in the list will update to reflect the new filter string. The drop-down list attached to the field contains (by default) an entry for every file type present in the current folder - this provides a quick way to filter on a specific file extension.

If you click the icon on the left of the field, a pop-up menu opens that lets you configure exactly how the filter field behaves.



By default, the filter field edits the quick filter (the same as the [filter bar](#)). This filter is applied on top of the filters set via the [Folder Options](#) system, and the global filtering options on the [Folders / Folder Display](#) Preferences page. It doesn't override these filters - so any files hidden via these other methods will stay hidden no matter what the quick filter is set to. Note that you can clear the quick filter at any time by pressing the **Escape** key.

The **Files**, **Folders** and **Files and Folders** options let you set the field to control the Folder Options filters rather than the quick field. When one of these options is chosen, the **Show** and **Hide** options also become available. These options work together to control exactly which of the Folder Options filters the field affects:



- **Files** and **Show** - affects the **Show Filter / Filename** field
- **Folders** and **Show** - affects the **Show Filter / Folders** field
- **Files and Folders** and **Show** - affects both the **Show Filter / Filename** and **Folders** fields
- **Files** and **Hide** - affects the **Hide Filter / Filename** field
- **Folders** and **Hide** - affects the **Hide Filter / Folders** field
- **Files and Folders** and **Hide** - affects both the **Hide Filter / Filename** and **Folders** fields

The other options in the filter field menu are:

- **Filter in Real Time:** If this is enabled, the list in the current file display is updated as you edit the filter. Normally this is preferable as you can see exactly how your filter is affecting the files that are displayed, but with a very large number of files you may find the real-time update a little slow - in this case, you can turn it off.
- **Partial Match:** With this option enabled, the pattern you enter must only match a sub-string of a filename, rather than the entire filename. It is functionally equivalent to always putting asterisks around the string you enter (e.g. `*cow*` instead of `cow`), and in fact this is how this feature is implemented internally.
- **Auto-content:** This option controls what is displayed in the drop-down list attached to the filter field. If auto-content is turned on the drop-down list contains a wildcard entry for every file type present in the current folder (as described above). If you turn this off the drop-down list instead displays a history of previously used filter patterns.

The **Clear Filters** command clears all filters currently in force in the current file display (except for global filters from the [Folders / Folder Display](#) Preferences page). The **Clear History** command clears the filter history that's displayed in the drop-down list (if auto-content is disabled).

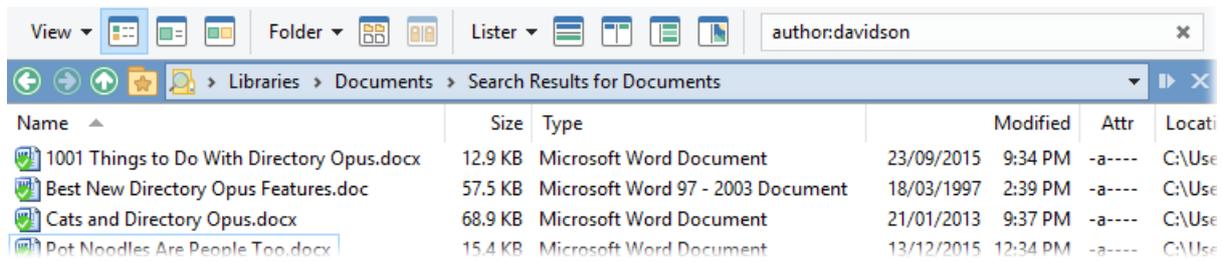
Changes you make to the behaviour of the filter field via the menu are not permanent - if you close the Lister and open a new one, the field will be reset to its default settings. You can change these defaults by [editing the definition of the filter field](#).

## Windows Search

The search field on the Menu toolbar lets you access the [Windows Search](#) system from within Opus. This provides an indexed search - the advantage of which is that results are normally returned almost immediately. Not all locations are necessarily indexed, however, and in these cases the search speed will be similar to that provided by the Opus [Find Files](#) function.



To search the current folder, click in the search field (or press **F3**) and enter your search term. You can also use the [find-as-you-type](#) field in search mode - by default this is activated by pressing the = key (equals sign). You can normally just enter one or two keywords to find the files you're looking for - the Windows Search system will return files that match your keywords both in contents and in filename. If you want greater control over the search results, you can enter a more complex query string using [Advanced Query Syntax](#).



In the above screenshot we have searched the **Documents** library for any documents with the name *davidson* in the author field. Matching files are displayed in a [file collection](#). This file collection is special in that it doesn't appear below the **File Collections** root folder like normal collections - instead, it appears as a child of the folder you searched in. The collection is also temporary - navigating away from the search results will cause it to be discarded.

To clear the search results and return to the folder you searched in, you can click the **Back** toolbar button, or use the **X** button in the search field to clear the search term.

If you want to save the results of an indexed search permanently, right-click on the background of the file display (not on a file) or on the status bar, and select the **Save as Stored Query** command from the context menu to save the search as a [stored query file collection](#).

Windows Search is included in Vista and above - for Windows XP you must [download](#) it from Microsoft and install it to use this feature in Opus.

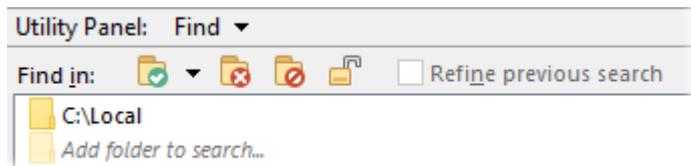
## Find Files

The Opus Find tool has two main modes of operation; [simple](#) and [advanced](#). The simple-mode Find tool lets you quickly search the current folder (or any location) for files or folders by *name*, *date*, *size*, *type* or by *text they contain*. The advanced-mode Find tool uses a [Filter control](#) to let you specify much more complex search queries. Using the advanced mode you can search based on attributes like image dimensions, audio bitrate, document author, etc.

Opus also supports an alternate search system - the integrated support for [Windows Search](#) lets you use the Windows search index to perform quick searches using a text field on the toolbar. You may find the indexed search is enough for your purposes - it will usually be quicker, although it doesn't provide as much control over the parameters of the search as the Opus Find tool.

To access the Opus Find tool, select the **Find Files** command from the **Tools** menu. The Find tool appears at the bottom of the Lister in the [Utility Panel](#).

The top section of the Find panel lets you specify the location or locations to search. You can search one or more folders (or entire drives) at once by adding entries to the **Find in** list.



Each entry in the list corresponds with a folder to search. To add a folder to the list, double-click the *Add folder to search* item. You can edit a folder path in the list by double-clicking it (or select it and press **F2** if you want to enter the path using the keyboard).

You can also use the **Select Folders to Search** (📁) button on the toolbar - this displays a folder selection dialog with checkboxes so you can select multiple folders simultaneously. This button also has a drop-down attached to it that displays a *most-recently used* list, which lets you select from your recent search locations.

To remove a folder from the **Find in** list, select it and click the **Remove Folder** (🗑️) button. The **Reset Folder List** (🔄) button clears the folder list, and the **Lock Folder** (🔒) button locks the **Find in** location to the folder displayed in the current file display. When the folder is locked, the **Find in** list will automatically reset to the current location whenever you navigate in the file display.

The **Refine** checkbox becomes available after using the Find tool once and allows you to narrow down the results of a previous find. **Refine** is usually left off, in which case using the Find tool again will cause it to start from scratch and return everything below the **Find In** folders which matches your new criteria. On the other hand, if you turn on **Refine** and use the Find tool again then it will apply your new criteria on top of the previous results leaving you with only the things which match both your old and new criteria.

As an example of what **Refine** does, consider a folder that contains these files:

- Directory Listing.txt
- Directory Opus.txt
- Magnum Opus.txt

Asking for files with "Directory" in their names will give these two results:

- Directory Listing.txt
- Directory Opus.txt

If you then turn on **Refine** and ask for names containing "Opus" then you'll get just the file which matches both your original request ("Directory") and your new one ("Opus"):

- Directory Opus.txt

(Of course, you could have specified "Directory Opus" in the first place but there are times when you are not sure exactly what to look for or how many results you'll get back.)

On the other hand, if you turn off **Refine** and ask the tool to find files with "Opus" in their names, you'll get everything containing "Opus", regardless of what was done before:

- Directory Opus.txt
- Magnum Opus.txt

To the right of the **Find in** list are a number of options that control how Opus searches for files.

- **Show results in:** This lets you specify the name of the [File Collection](#) that the results of the search will be displayed in. If the specified collection doesn't already exist it will be created automatically. Normally the current file display will automatically navigate to show this collection when the search begins, but you can use the drop-down below the **Show results in** field to specify that the collection should be shown in the destination (rather than the source) file display, and also opt to have it shown in a new [tab](#). As files are found by the Find tool they will appear in the collection.
- **Clear previous results:** If this option is on the contents of the **Show results in** file collection will be automatically cleared before the search begins. On the other hand, if the option is off then any files which you find will be added to the existing results collection, allowing you to accumulate the results of several otherwise unrelated find operations.
- **Search inside subfolders:** The function will search the contents of all sub-folders of the specified locations as well as the locations themselves.
- **Search inside archives:** The function will also search the contents of any archive files that are found in the specified locations.

The bottom part of the Find panel is where select the mode (simple or advanced), and where you define the parameters of the search.



The mode can be selected using the tab control. Please see the [Simple Find](#) and [Advanced Find](#) pages for more information about the two modes. When you have defined your search parameters, click the **Find** button to begin the search.

## ***Simple Find***

The simple mode of the [Find tool](#) lets you search for files using one or more criteria.

The screenshot shows a search configuration window with the following settings:

- Name matching:** \*.txt (dropdown),  Wildcards,  Any word,  Partial match
- Containing text:** individuality (text input),  Wildcards,  Case sensitive
- Type:** All files and folders (dropdown)
- Date:** Within (dropdown), 7 (spin box), days (dropdown)
- Time:** Ignore (dropdown)
- Size:** Ignore (dropdown)
- Reset** button

The labels of each search criteria appear in **bold** when something is defined for them, making it a easy to see exactly what you're searching for. In this example we are searching for all files that end with a **.txt** extension, that have been modified within the past seven days, and that contain the text string "individuality". Time, size and type are all being ignored.

- **Name matching:** Specify the name of the file to search for.

The **Wildcards** option lets you use [standard pattern matching](#) in this field to search for names using wildcards. You can also enter **regex:** followed by a pattern to use [regular expressions](#).

The **Any word** option treats every word you enter as a separate search term. For example, if this was turned on and you typed "horse donkey" into the field, Opus would match filenames containing "horse" or "donkey" (or both, in any order). This saves you having to construct complicated *OR* wildcard patterns.

The **Partial match** option means that the string you enter can match part of the filename, and does not have to match the whole filename.

- **Containing text:** If a string is specified here then each file (that matches the other criteria defined) will also be searched to see if it contains the specified text. Searching for containing text can make a search much slower.

The **Wildcards** option lets you use standard pattern matching when defining the text string to search for - if you leave this off, the string must match exactly. The **Case sensitive** option lets you specify that the case (upper/lowercase) of the supplied string must match exactly as well - if this is turned off, then upper and lower case letters are considered the same.

- **Type:** You can use this to search for files or folders of a certain type. For example, instead of specifying **\*.bmp** for the **Name matching** field, you can select **File Type, Bitmap Image** from the drop-downs to find bitmap files.
  - **All files and folders:** Search for all files or folders matching the other conditions.
  - **Files:** Search for only files.
  - **Folders:** Search for only folders.
  - **Junctions and Links:** Only search for junctions or soft-links.
  - **File Type Group:** Search files belonging to a specified file type group. You can use the second drop-down control to choose the group to search for.
  - **File Type:** Search files of a particular file type. Use the second drop-down to choose the file type to search for.
- **Date:** This lets you match on a file's last modification date. Only the date is considered, not the time. The following options are available for the **Date** drop-down:

- **Ignore:** The date is not considered.
  - **On:** Search for files whose datestamp falls exactly on the specified date.
  - **Before:** The datestamp must be before the specified date.
  - **After:** The datestamp must be after the specified date.
  - **Between:** The datestamp must be between the two specified dates.
  - **Not Between:** The datestamp must **not** be between the two dates (it must be either earlier than the first date or later than the second).
  - **Within:** Lets you specify a date relative to the current date. For example, **Within 7 days** to search for files modified within the past week.
- **Time:** This lets you match on a file's last modification time. Only the time is considered, not the date. Seconds are ignored by this - you only need to specify the hours and minutes of the time to search for. The following options are available:
    - **Ignore:** The time is not considered.
    - **At:** Search for files whose timestamp is exactly the specified time.
    - **Before:** The timestamp must be before the specified time.
    - **After:** The timestamp must be after the specified time.
    - **Between:** The timestamp must fall between the two specified times.
    - **Not Between:** The timestamp must **not** fall between the two times.
    - **+/- One Hour:** The timestamp must be within one hour (plus or minus) of the specified time.
    - **+/- Six Hours:** The timestamp must be within six hours of the specified time.
    - **Within:** Lets you specify a time relative to the current time. For example, **Within 30 minutes** to search for files modified in the past half-hour.
- **Size:** This lets you search for a file by size. The size value must be given in **KB** (kilobytes). When the size is compared, it is rounded to the nearest kilobyte, so a search for **Size Equals 1** (kb) would find a file that was 1400 bytes. The options available are:
    - **Ignore:** The size is not considered.
    - **Equals:** The size must match exactly (after rounding).
    - **Smaller Than:** The file must be smaller than the specified size.
    - **Greater Than:** The file must be larger than the specified size.
    - **Between:** The file size must be between the two specified values.
    - **Not Between:** The file size must not be between the two specified values.
    - **+/- 25%:** The file size must be within 25% (plus or minus) of the specified value.
    - **+/- 50%:** The file size must be within 50% (plus or minus) of the specified value.

At the bottom of the simple find dialog there's a **Reset** button which makes it easy to quickly reset the find criteria to the defaults.

## Advanced Find

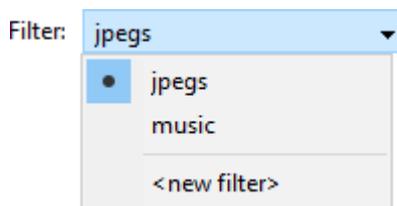
The advanced mode of the [Find tool](#) lets you use a Filter control to define the search.



The Filter control lets you build up complex search expressions that let you find files based on many different criteria. In the above example we are searching for all image files that are at least 1280x1024 pixels in size, and have been modified within the past two months.

The **Clear** (🗑️) button in the toolbar at the top of the filter control lets you quickly clear the existing filter. If you hold the **Shift** key down when you click the **Clear** button the filter will be reset to a simple template filter with some common conditions added.

The **Save** (💾) button lets you save the current filter definition as a pre-configured filter; it will then appear in the [Filters](#) page in Preferences.



The **Filter** drop-down button lets you quickly access a pre-configured filter. In the screenshot above, the *jpeg* filter has been chosen. Select the *<new filter>* item from the drop-down to create a new filter (this has the same effect as clicking the **Clear** button).

See the [Filtered Operations](#) section for a full description of the filtering system, including details on [how to define a filter](#) and what [attributes you can search for](#).

## Sorting and Grouping

You can control the order files and folders are displayed in several different ways. The primary way is by defining the *sort order*. A file list can be sorted by any column that is currently displayed. You can also use [manual sorting](#) to arbitrarily arrange the contents of your folders.

In details and power mode, you can change the sort order by clicking the desired sort field in the column header. You can also display the column header in the other view modes, by turning on the [File Displays / Options / Show sort header in icon modes](#) option in Preferences.

▲	Name #2	Size #1	Type	Modified	Attr
---	---------	---------	------	----------	------

If the field you want to sort by is not currently displayed in the list, use the [Folder Options](#) dialog to add the desired column. This applies even in the non-column based view modes - even if the column is not displayed, it must still be added to the folder format before you can sort by it.

Clicking on a field will sort the list by that field. If you click the same field again, the sort order will be reversed. You can go straight to the reverse order for the field by holding the **Shift** key down when you click it. You can sort by multiple fields (as shown in the image above) by holding the **Ctrl** key down. Select the first (primary) sort field by clicking normally, then hold the **Ctrl** key and click on the second field, third field, fourth field and so on. The direction for supplementary sort fields can be modified on an individual basis in the same way as for the primary field.

In the non-column based view modes, the easiest way to change the sort order is to right-click on the background of the file display (an empty area - or right-click on the status bar if you can't find an empty area) and select the desired sort field from the context menu. You can also use the **Folder Options** dialog to change the sort order. There are several other options in the [Folder Options](#) dialog that affect the order of items in the list - for example, you can choose whether files should come before or after folders, or if the two should be mixed together.

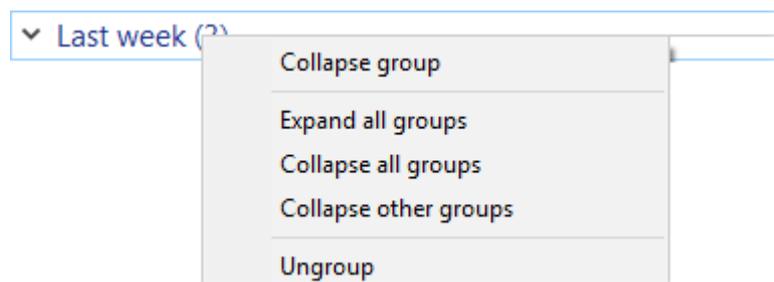
Grouping lets you split the list of files into multiple groups (fairly obviously!). You can group by the same column that you sort by, or by a completely different column. Unlike sorting, the column you group by doesn't have to be currently displayed - although the grouping context menus described below will only display those columns that are currently enabled. To group by a non-displayed column you need to use the internal [Set GROUPBY](#) command.



In this screenshot, the list has been grouped by last modified date. Each group has a heading that identifies the content of the group and displays the number of items in the group. You can collapse (hide) or expand (reveal) the contents of the group by clicking the little > glyph (or with the keyboard, place the focus on the group header and press the **Cursor Left** key to collapse it, or **Cursor Right** to expand it).

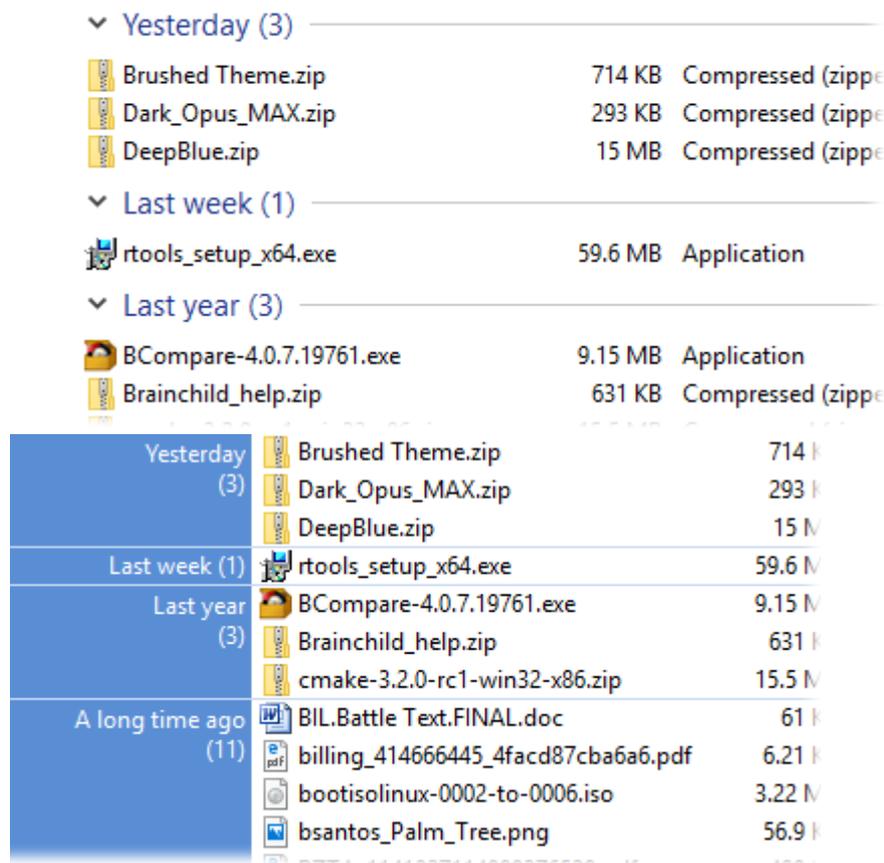
To group (or ungroup) the list, right click on the column header (or, in a non-column based view mode, right-click an empty area of the file display or the status bar) and select the desired field from the **Group By** menu. You can also use the column header to set the grouping field by holding the **Alt** key down and clicking the desired field. If you click it a second time (with the **Alt** key down) the grouping direction will be reversed. Grouping can be disabled altogether by holding **Shift+Alt** and clicking any field in the header.

Clicking a group header will automatically select all files in that group. If you right-click a group header, a number of grouping-related commands are displayed in the context menu.



**Collapse group** (or **Expand group**) lets you collapse (or expand) the group you clicked on. **Expand all groups** , **Collapse all groups** and **Collapse other groups** affect all groups at once.

As an alternative to the group appearance shown above (with headers separating each group vertically), the *Group* column can be added to the file display (via the [Folder Options](#) dialog). When this column is present in a grouped file display, the column itself displays the headings, making for a much more compact display.



*headings*

*Traditional group*

*Group column*

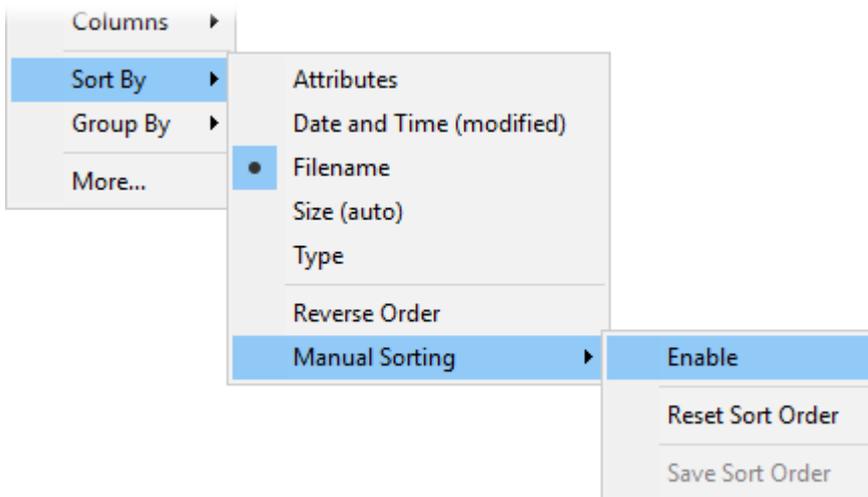
The colors used for the *Group* column can be configured on the [Display / Colors and Fonts](#) Preferences page. You can define different colors for source and destination file display if desired. You can also configure Opus to add the *Group* column automatically whenever the file display is grouped with the [Folders / Folder Display / Add 'Group' column automatically when file display is grouped](#) option.

## Manual Sorting

### An overview of manual sorting

Manual sorting refers to being able to control the display order of files and folders arbitrarily. Although for most purposes the automatic sorting methods (e.g. sorting files alphabetically by name) are all you need, many people have use cases where being able to control the sort order exactly would be useful.

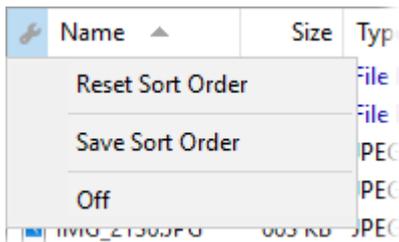
Manual sort mode needs to be activated in a Lister before files can be arbitrarily arranged. Right-click the column header and use the Manual Sorting options to activate manual sorting mode.



You can also turn on the *Sorting Options / Manual sorting* option on the *Display* tab in the Folder Options dialog. Obviously as a folder options setting, this mode can also be made permanent by saving the folder format.

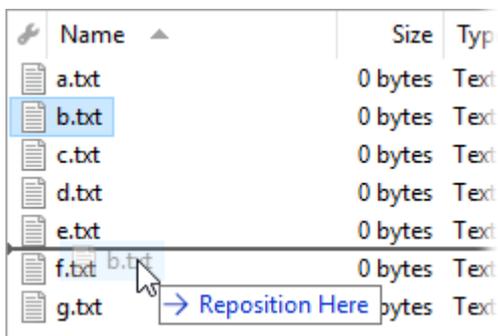
If a manual sort order had previously been defined (and saved) for the current folder, the file list will resort automatically when manual sorting is turned on. Otherwise, turning on manual sorting will have no immediate effect on the file list.

In details or power mode, or when the column header is enabled in the other modes, a new button (  ) appears on the left of the column header when in manual sort mode. Clicking this button displays a control menu with a few commands in it relating to manual sort mode:



## How to manually sort your folders

Once manual sort mode is active, you can reposition a file using drag-and-drop, or from the keyboard using **Shift + Alt** in conjunction with the cursor keys.



By default, your manual sort order will be persistent (saved automatically), whenever possible (see below for current limitations on manual sorting). If you want the freedom to change the sort order temporarily without making it permanent, you can turn off the *Folders / Folder Behaviour / Automatically save manual sort order where possible* option in Preferences. With that option turned off, you need to use the **Save Sort Order** command in the control menu.

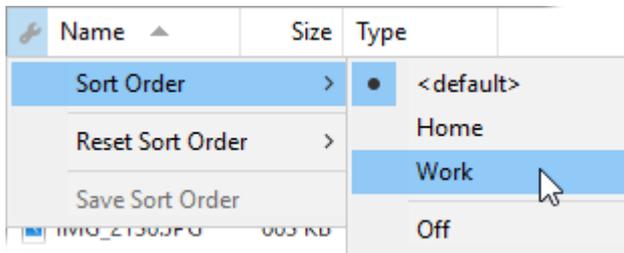
At any time, you can reset the sort order to the current automatic order (e.g. alphabetical by name) using the **Reset Sort Order** command in the control menu (shown above).

## Multiple custom sorts

By default you're limited to just a single manual sort order per folder but if you need to be able to define more than one, you can add "named orders" through the *Miscellaneous / Advanced / manual\_sort\_names* option in Preferences.

layout_string	
manual_sort_names	Home; Work
max folder thumb time	5000

When named orders are defined they appear in the column header's control menu, letting you switch between them at will.



The Folder Options dialog also displays a drop-down (on the *Display* tab) showing all the named sort orders, which lets you select the default for a folder if desired.

## Limitations

Currently manual sort orders can't be saved for some types of folders:

- Non-NTFS disk folders (e.g. FAT/FAT32)
- Archives
- FTP

By default, manual sorting is disabled completely in folders that the order can't be saved for. If you turn on the *Folders / Folder Behaviour / Allow manual sorting in all folders* Preferences option then manual sorting will be available everywhere, but changes to the sort order in those types of folders will only be temporary.

Manual sorting is currently not supported in Flat View or when the file display is grouped. It will also not work correctly in a folder when compatibility files are displayed and there are two files with the same name.

# Folder Options

The Folder Options dialog is used to change the appearance and presentation of the files currently displayed in the file display (this is known as the [folder format](#)). It includes things like the [view mode](#) (icons, thumbnails, details, etc), the columns shown (in details and power mode), the [sort and grouping options](#), filters that can hide or show files and folders based on wildcard patterns and other options that affect the display of the file list.

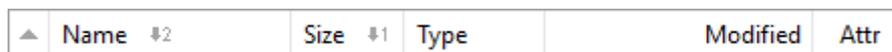
There are a number of other ways that the display can be modified that don't require using the [Folder Options dialog](#), which are worth mentioning at this point:

- In details and power mode (and optionally the other modes), you can change the sort order by clicking the desired sort field in the column header. Clicking on a field will sort the list in the default order for that field. The default order can be changed for individual fields on the [Fields](#) page in preferences. If you click the same field again, the sort order will be reversed.

You can go straight to the reverse order for the field by holding the **Shift** key down when you click it. For the **Name** column only, **Shift**-clicking the column puts it into a special mode that lets you sort by file extension as well as by name. When the **Name** column is sorted by extension rather than name, the sort arrow will change to point to the left or right:



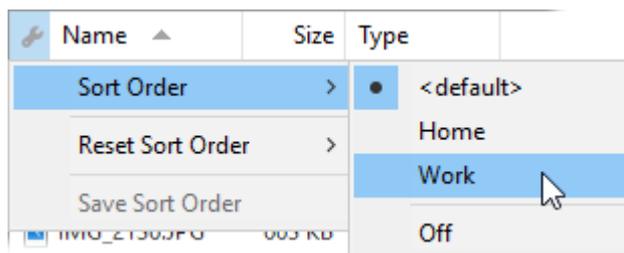
You can sort by multiple fields (as shown in the image below) by holding the **Ctrl** key down. Select the first (primary) sort field by clicking normally, then hold the **Ctrl** key and click on the second field, third field, fourth field and so on. The direction for supplementary sort fields can be modified on an individual basis in the same way as for the primary field.



You can also use the column header to set the [grouping](#) field by holding the **Alt** key down and clicking the desired field. If you click it a second time (with the **Alt** key down) the grouping direction will be reversed. Grouping can be disabled altogether by holding **Shift+Alt** and clicking any field in the header.

The column header also displays a [context menu](#) if you click it with the right button. By default this menu contains a list of columns that can be displayed in the lister, and you can turn individual columns on and off using this menu. You can also change the sorting and grouping options using the context menu, as well as access the Folder Options dialog.

If [manual sorting](#) is enabled, an additional button is shown in the column header letting you control the manual sort options.



The column header is always displayed in details and power modes. You can display the column header in

the other view modes, by turning on the [File Displays / Options / Show sort header in icon modes](#) option in Preferences.

- In all view modes, you can right-click on the background (any empty area that's not a file or folder) of the file display to display the Lister [context menu](#). By default this context menu contains sub-menus like **View** (change view mode, and access Folder Options), **Sort By** (change sort field) and **Group By** (change grouping field). If you can't find an empty area in the file display (because you have full-row selection enabled, for instance) you can also right-click on the status bar.
- There are a number of drop-down menus on the default toolbars that can affect the file display.

The **View** menu contains options to change the current view mode, sort field and grouping field.

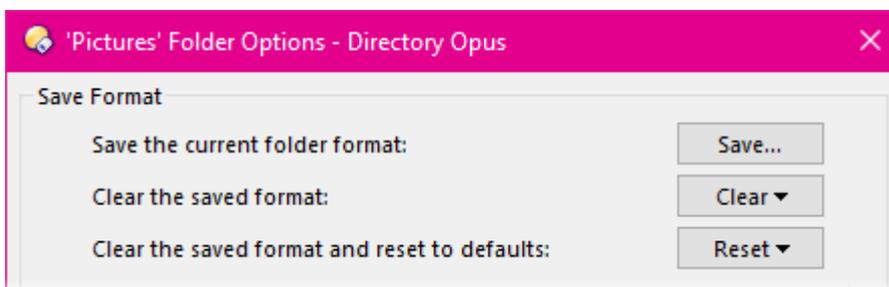
The **Folder** menu contains options to access your [favorite folder formats](#) (which can instantly change multiple file display parameters) and also gives you a way to access the Folder Options dialog.

The **Lister** menu displays a list of your configured [lister styles](#), which as well as doing things like opening the viewer or closing the tree are also able to change the folder format.

When you make a change to the file display with the Folder Options dialog, the changes you make will persist in that file display until something else changes them (either you making another manual change, or an automatic change triggered by the [Folder Formats system](#)).

## Folder Options Dialog

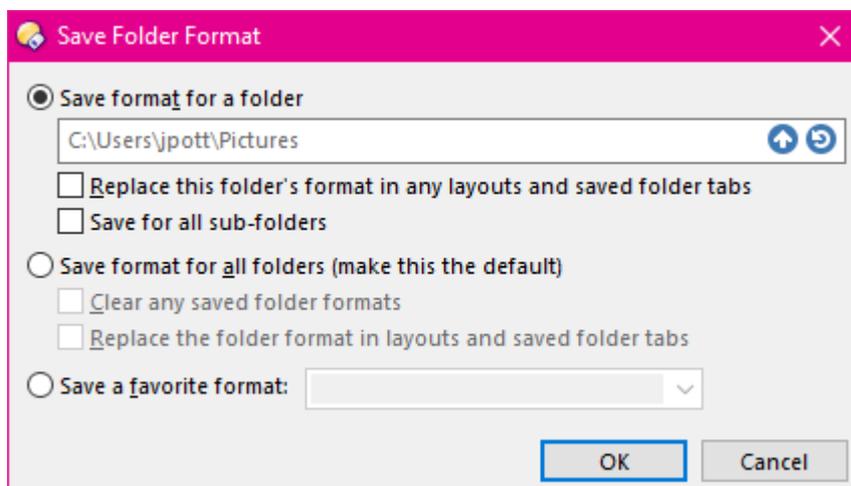
The Folder Options dialog can be accessed using the **Folder / Folder Options** command from the default **Menu** toolbar, or by right-clicking the file display column header and choosing **More**.



The **Save Format** section at the top of the Folder Options dialog is a quick way to manage the stored folder format for the current folder. If you want to change an aspect of the display for a folder permanently there are two ways to do it; you can go into Preferences, to the [Folder Formats](#) page, and define a folder format for that folder. Or, using the Folder Options dialog, you can make the changes in real time and then save them to create a folder format automatically.

- **Save:** Use this to save the current folder options as the format for the current folder. See below for a discussion of the save options.
- **Clear:** Use this to remove a stored format for the current folder. The options on the **Clear** drop-down are:
  - **For This Folder:** If the current folder has a folder format defined, this option will let you remove it.
  - **For All Folders:** This option will remove all stored folder formats.
- **Reset:** Use this to quickly reset the current folder options to the default values. You can choose from **User default** (the user-configurable default format) or **Factory default** (the built-in folder options that can't be changed).

Selecting the **Save** button brings up the Save Folder Format dialog:



There are three main options when saving a format:

- **Save format for a folder:** The format will be saved for one specific folder (by default, the one currently displayed in the file display - but you can use the up button (⬆) to save the format for a parent of the current folder if desired).

If you turn on the **Replace this folder's format...** option, Opus will search through all your saved Lister layouts, saved folder tabs (and styles), and any currently open folder tabs for this folder, and update their format as well.

The **Save for all sub-folders** option causes the **Use as the default format for all sub-folders** option to be turned on in the saved format.

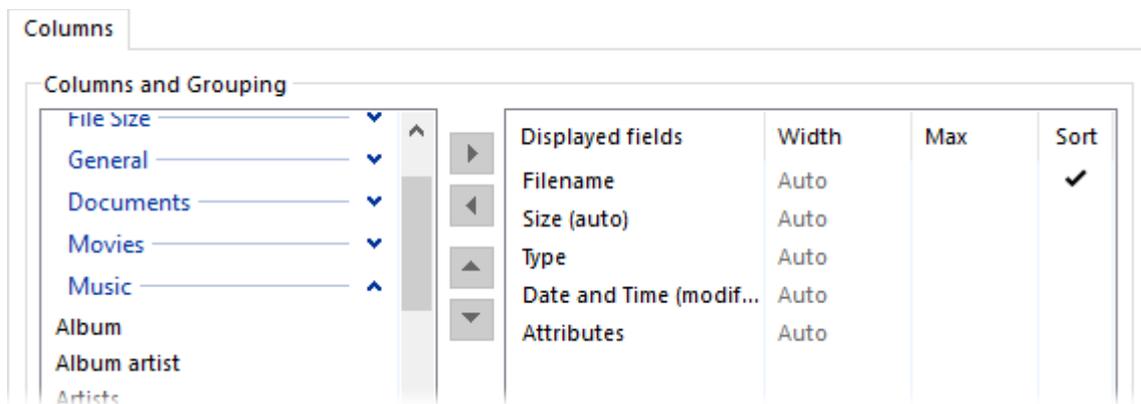
- **Save format for all folders:** The format will be saved as the new *User Default* format, which is used for a folder that doesn't have a specific folder format saved for it. If you turn on the **Clear any saved folder formats** option, and folders for which you've previously saved a specific format will be reset. And the **Replace the folder format...** option causes any saved layouts, tabs and styles to also have their formats reset.
- **Save a favorite format:** This lets you save the format as a favorite format (either a new one, or over the top of an existing favorite).

Below the **Save Format** section are six tabs which give you access to the various options that can be configured as part of the folder format.

- [Columns](#): Lets you choose which columns are displayed in the file display in details and power mode. The selection of columns also has an impact on other display modes, as you can only sort or group the list by a column that has been turned on.

- [Display](#): Contains various options that affect the display of the file list (view mode, etc) and options that affect how the list is sorted (note that the actual sort field is specified on the Columns tab, however).
- [Hide Filters / Show Filters](#): Let you filter the list by wildcard pattern or attributes.
- [Options](#): Contains miscellaneous options.
- [Labels](#): Lets you configure wildcard or filter [labels](#) that only apply in this folder.

## Columns



The **Columns** tab of the Folder Options dialog lets you configure which information fields (columns) are displayed for the folder when the file display is in Details or Power mode. The selection of columns also has an impact on other display modes like Thumbnails - even if the columns aren't displayed, you can only sort by columns that have been enabled. The available columns are grouped into a number of categories - you can use the filter field at the bottom of the column list to quickly find a column by name. The fields that aren't currently enabled appear in the list on the left, and the enabled fields are shown in the list on the right. To move a field from one list to another you can double-click it, drag and drop it, or use the left and right arrow buttons between the two lists.

In the list on the right, you can change the order the columns will appear in the file display by drag and drop or using the up and down arrow buttons. You can also re-order columns in the file display itself by dragging the column headers around.

The **Width** column in the fields list lets you specify the width for individual columns. If the *Auto-size all columns* option on the **Display** tab (see below) is turned on, all columns will be shown with their widths as *Auto* (and grayed out to indicate that option is in effect) - editing an individual column's width will automatically turn off the *Auto-size all columns* option.

Displayed fields	Width	Max	Sort
Filename	Auto		✓
Size (auto)	Auto		
Type	Auto		
Date and Time (modified)	Auto		
Attributes	Auto		

Click the *Width* field of one of the displayed columns to change its width. A drop-down menu appears with a number of width options available:

Displayed fields	Width	Max	Sort
Filename	Auto		✓
Size (auto)	Auto		
Type	Auto		
Date and Time (modified)	Auto		
Attributes	Auto		

The options are:

- **Auto**: Automatically size this column. This has the same effect as the *Auto-size all columns* option on the Display tab, but lets you apply it on a per-column basis.
- **Expand**: Automatically sizes the column the same as **Auto**. The difference is the widths of **Expand** columns are ignored in the calculation of **Fill** columns (see below), and columns set to **Expand** will go off the right hand side of the file display rather than making **Collapse** columns start to collapse.
- **Collapse**: Automatically sizes the column, but its width is able to collapse (down to zero width if necessary) to allow other fields to fit without horizontal scrollbars appearing. For example, you might want the *Description* column displayed, but not have it force other fields off the edge of the file display. Setting it to **Collapse** means it will only appear if there's space for it.
- **Fill**: Columns set to fill will be automatically sized to fill any available horizontal space in the file display. If there is more than one **Fill** column they divide the available space between them. Columns set to **Fill** can potentially end up wider than they need to be (contrast with **Auto** + **Fill** described below).

You can also enter a desired pixel width into the *Width* field.

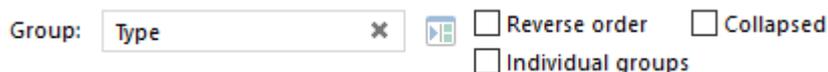
Columns that are set to automatically size (**Auto**, **Expand** or **Collapse**) can also have a maximum width set using the *Max* field.

Displayed fields	Width	Max	Sort
Filename	Auto		✓
Size (auto)	Auto	Fill	
Type	Expand	85	
Date and Time (modified)	Auto		

You can enter a maximum size in pixels. For a column set to **Auto** width you can also choose **Fill** for the maximum, which makes it auto-size up to but not beyond the width of the file display (to avoid horizontal scrolling), but unlike setting **Fill** for the width it won't auto-size larger than it needs to be.

The **Sort** column indicates which field the list will be sorted by. If a single checkmark is shown, there is only one sort field (in the screenshot above, we're sorting by *Date modified*). If a number with an arrow symbol is shown it indicates multiple sort fields. To change the sort field, single click in column on the desired field. To assign multiple sort fields, click on the first field as normal, then hold either the **Control** or **Shift** keys and click on the second field, then the third, and so on. If you **Control/Shift**-click a field that is already sorted, the direction of the sort for that field only is changed - so you can for example sort by size in one direction and date in the other.

Below the column lists are controls which let you choose which field (if any) the list is [grouped](#) by.

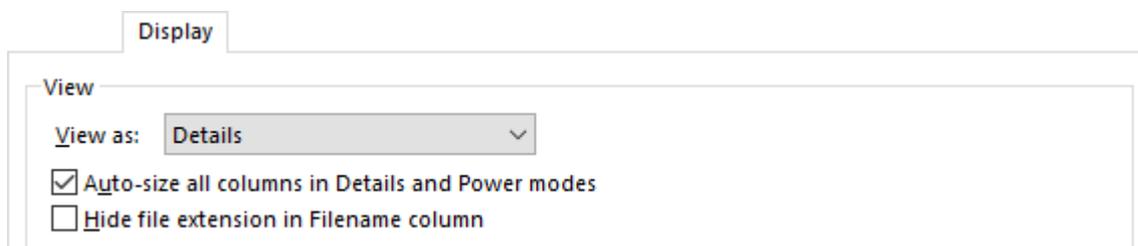


The current group column is displayed, if any, and to clear it click the **x** symbol. To group by a new column, select the desired column in *either* of the two column lists above, and click the arrow button. You'll notice that the arrow button changes between and depending on which list was most recently active – this is a visual cue to indicate which list the group column will be set from when the button is clicked.

The options available when the file display is grouped are:

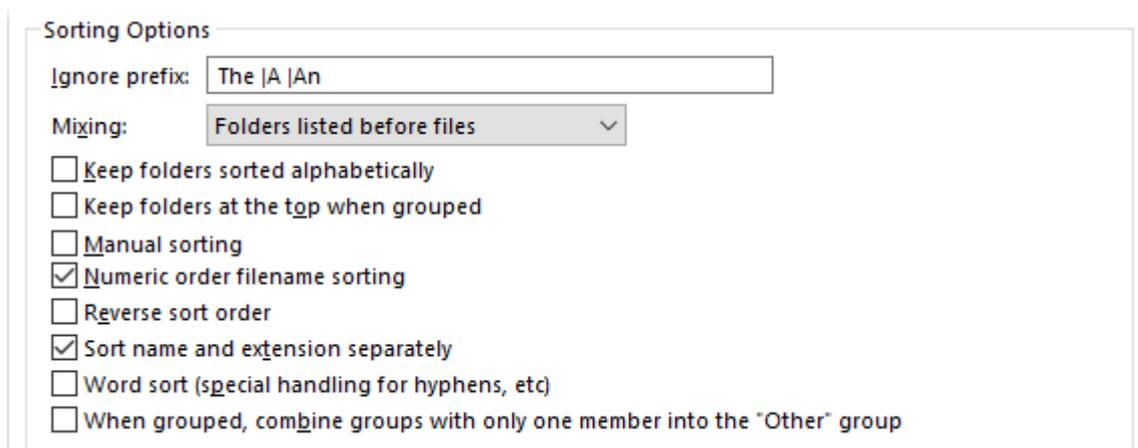
- **Reverse order:** This option lets you set the initial group order (how the individual groups are arranged) to the reverse of the default.
- **Collapsed:** This option lets you have the groups start out as collapsed.
- **Individual groups:** This option causes one group to be created for each distinct value rather than having a range of values fall into a single group (e.g. instead of A-H you would have A, B, C, D, E, F, G, H). This is only really useful for text fields like "User description".

## Display



On the **Display** tab of the Folder Options dialog, the **View** section contains options relating to the view mode of the file display.

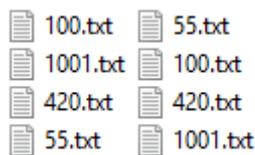
- **View as:** you can have the file display automatically switch to thumbnails, or details, for example.
- **Auto-size columns:** This option causes column widths in details and power mode to be automatically sized to fit their contents. Instead of using this option, which affects all columns, you can also configure this on a per-column basis on the **Columns** tab.
- **Hide file extension in Filename column:** This does just what it says - you might want to enable this if you have added the separate *Extension* column (or if you just hate file extensions!)



Also on the **Display** tab, the **Sorting Options** section contains options that affect the sorting of the file list (other than the actual field or fields the list is sorted by - this is specified on the **Columns** tab).

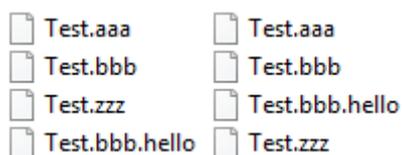
- **Ignore prefix:** This lets you specify one or more filename prefixes that Opus will ignore when sorting the file list. The screenshot above shows that *The |A |An* has been given, which means a leading *The*, *A* or *An* (all three are followed by a space) at the beginning of a filename will be ignored when sorting. If you specify more than one prefix you must separate them with a vertical bar character as shown above.

- **Mixing:** This option lets you choose how files and folders are to be sorted with respect to each other. You can choose to list all the files before the folders, all the folders before the files, or to have files and folders interspersed.
- **Keep folders sorted alphabetically:** With this option on, folders will always be sorted alphabetically even when the list is sorted by another field. This doesn't apply if you have chosen *Mix files and folders together* for the **Mixing** option.
- **Keep folders at the top when grouped:** When the file list is grouped, folders will always be kept in their own group which will be displayed before all other file groups.
- **Manual sorting:** Lets you enable [manual sorting](#) in a folder.
- **Numeric order filename sorting:** This option changes the way files and folders that begin or end with a number are sorted by name. When off, digits in names are treated like any other character when sorting. When turned on, digits at the beginning or end of a name are sorted based on their numerical value.



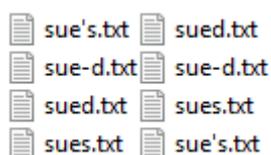
The image on the left shows an example of sort order when numeric order sorting is off; the image on the right is with it on.

- **Reverse sort order:** This option reverses the order of the sort - for example, if the sort is normally from A to Z, it would instead be from Z to A.
- **Sort name and extension separately:** When enabled, Opus sorts folders by name by splitting up the filename stem and the extension, and sorting them separately (i.e. it sorts by the stem first, and only if the stem is the same does it consider the file extension). This has the effect of keeping names with the same stem together, for example:



In above image, there are only three files with the stem *Test*, and a fourth with the stem *Test.bbb*. In the image on the left, with the option turned on, Opus has sorted two groups separately. In the image on the right, Opus has considered the whole filename including the extension as a single string.

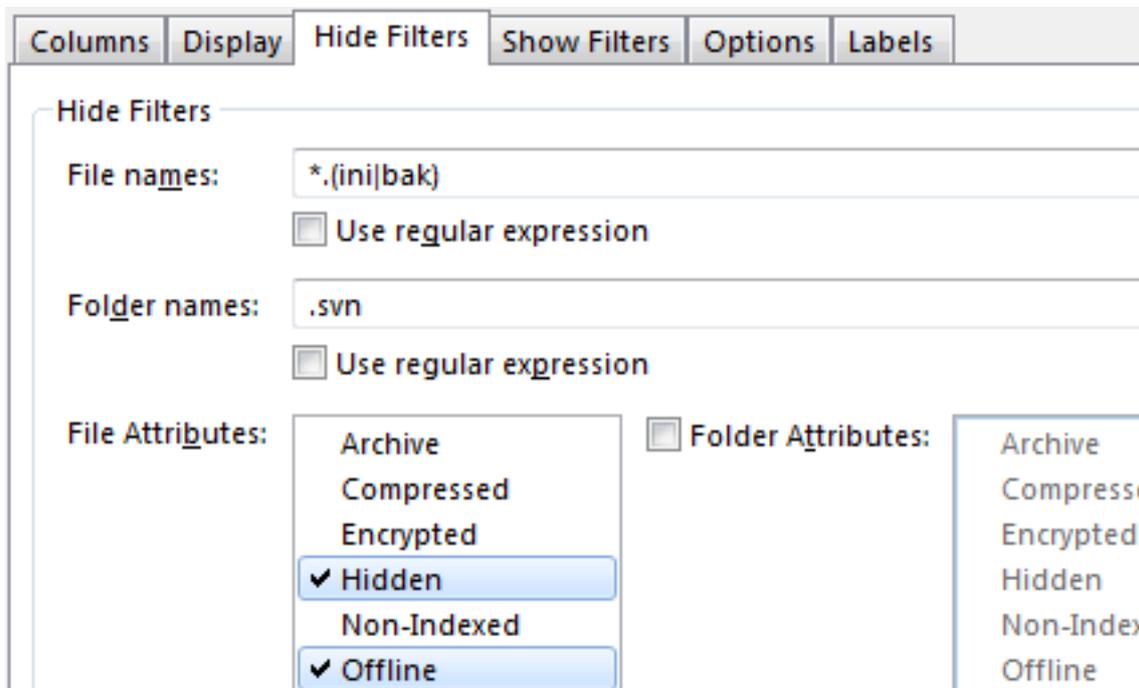
- **Word sort:** This option enables special handling for punctuation characters in filenames. In a word sort, all punctuation marks and other non-alphanumeric characters, except for the hyphen and the apostrophe, come before any alphanumeric character. The hyphen and the apostrophe are treated differently than the other non-alphanumeric symbols, in order to ensure that words such as "coop" and "co-op" stay together within a sorted list.



In the image on the left, word sort is turned off; it is turned on in the image on the right.

- **When grouped, combine groups with only one member into the “Other” group:** When enabled, and the file display is grouped, any items in a group by themselves will instead be shown in a group called *Other* (prevents cluttering up the folder with lots of groups containing only one file).

### Hide Filters / Show Filters



The **Show Filters** and **Hide Filters** tabs of the Folder Options dialog look identical. They can be used to control which files and folders are shown and which are hidden when the format is applied to a folder. The settings on the **Show Filters** tab causes files to be hidden if they **don't** match the filter, and the settings on the **Hide Filters** tab causes files to be hidden if they **do** match.

The **File names** and **Folder names** fields both take a wildcard string expressed using either [standard pattern matching](#) or [regular expression syntax](#) (if the **Use regular expression** option is turned on).

The **Attributes** field lets you filter files and folders based on their attributes - if any of the selected attributes are set on the item, it will match the filter and be shown or hidden as applicable.

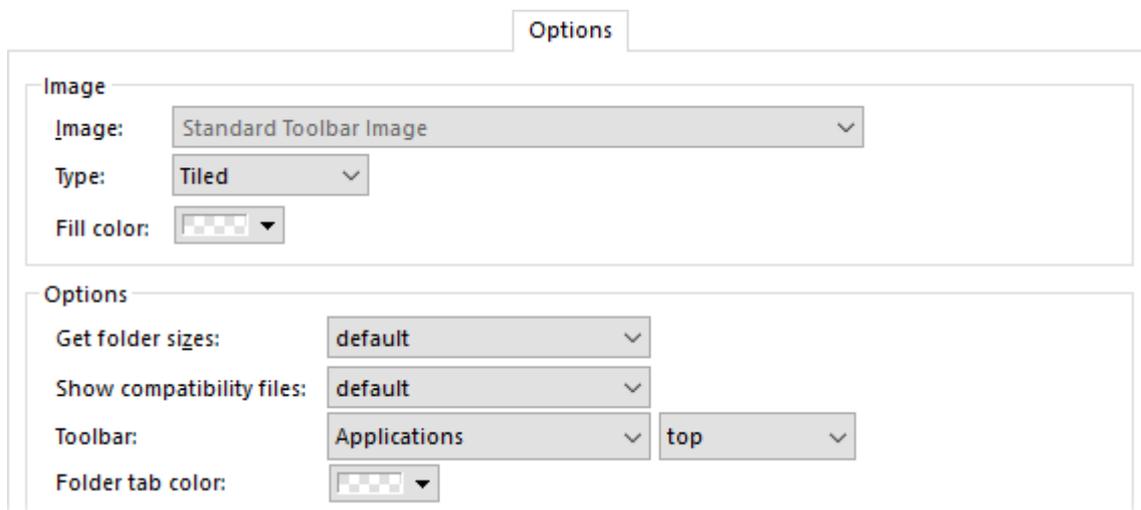
Both the Show and Hide Filters have the option of separate attribute filters for folders. If these separate filters are enabled, then the first **Attributes** field in each group only applies to files - otherwise, it applies to both files and folders.

In the screenshot above, the folder format is set to hide any files that end in **.ini** or **.bak**, and any folder called **.svn** (no wildcards are used for the folder name, so only that exact name will match). Any files marked as **Hidden** or **Offline** will also be hidden.

These filters stack on top of the [global filters](#).

All filtering can be quickly disabled in a folder tab by toggling [Show Everything](#) mode.

## Options



The first section on the Folder Options dialog's **Options** tab lets you define a background image that the format will apply whenever it is active. This can be nice to use with the [Content Types](#) system; for example, you could have a background image of a music note that appears whenever the Music Content Type format is in use, as a visual cue that you are in a music folder. The **Image** drop-down lets you define the image to use - the images in this list are added to Opus using the [Display / Images](#) Preferences page. You can control the position of the image in the file display with the **Type** drop-down. The **Fill Color** field lets you assign a background color for the file display that will override the color set in the [Display / Colors and Fonts](#) section. In the above screenshot, no color has been specified - this is indicated by the "transparent" checkerboard pattern.

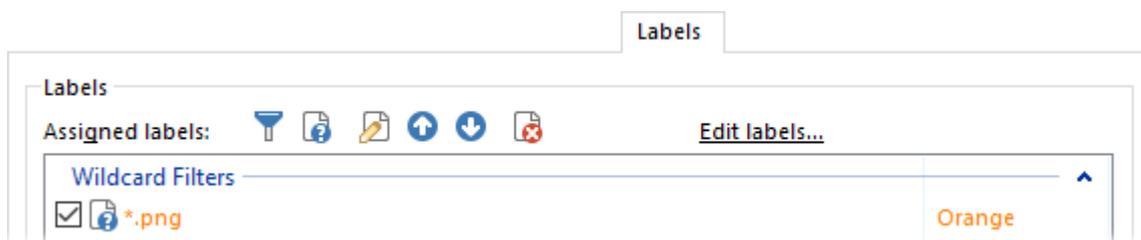
The final section contains general **Options** for the format.

- **Get folder sizes:** Use this to specify whether folder sizes are calculated automatically for sub-folders when this format is applied. If set to *Default* the setting in [Folders / Folder Behavior](#) will apply instead.
- **Show compatibility files:** If this is turned on, Opus will automatically display the contents of the [compatibility folder](#) along with the current folder, if applicable. If set to *Default*, the setting in [Folders / Folder Display](#) will apply.
- **Toolbar:** This lets you choose a toolbar or [toolbar set](#) that will be automatically displayed whenever this format is applied. (This option is only visible when [editing a saved Folder Format via Preferences](#). When editing the format currently in use by a folder tab, you can turn toolbars on and off in the lister itself.)
- **Folder tab color:** Lets you specify a custom color that's displayed on the folder tab for this folder.

Two additional checkboxes appear at the bottom of the **Options** tab when [editing a saved Folder Format via Preferences](#):

- **Include columns from other matching formats:** If turned on, when you change to a folder which matches the format being edited, Opus will continue looking for other folder formats which also match the folder. Any [columns](#) that those other formats specify will be added as well. For example, if you change to *C:\Photos* you might have a format which explicitly matches the *C:\Photos* path and will be the main format Opus uses to set up how the folder is displayed. If the *Include columns from...* checkbox is on in the main format, Opus will then look through the rest of the format list and might find that the **Images content type** format also matches the folder, and specifies some additional columns not in the main format, which will be added to those displayed in the lister.
- **Use as the default format for all sub-folders:** If turned on for a path format, the format will not only match the specified path but also any folders below it (unless another path format matches the child folders with more specificity). This also works with wildcard and regular expression path formats, to include folders below any path which matches their patterns.

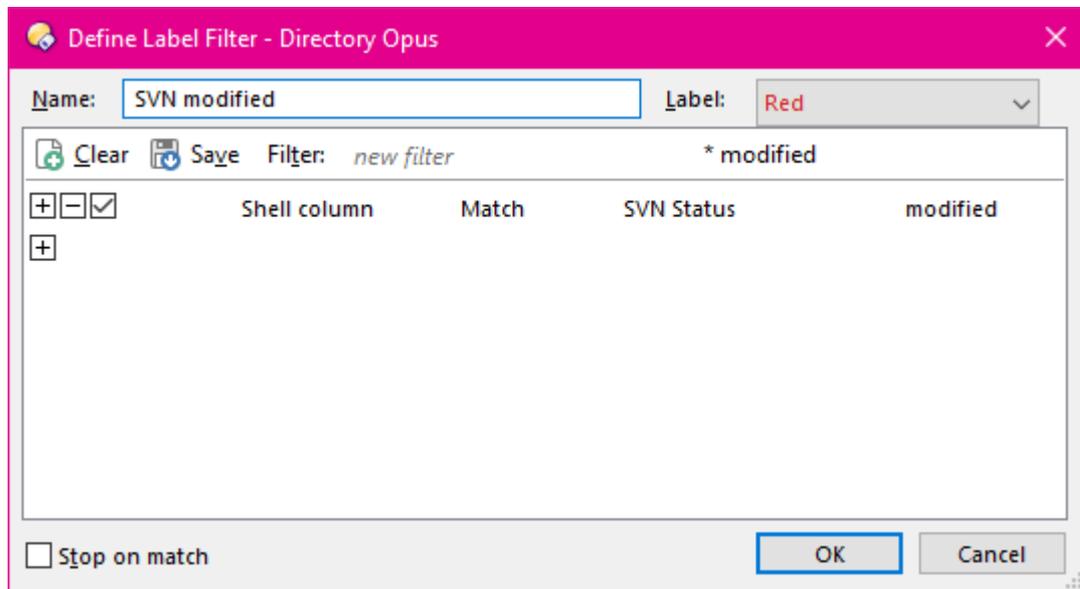
## Labels



The **Labels** tab on the Folder Options dialog lets you define wildcard labels and label filters that are only active when this folder format is used. It lets you apply labels to files and folders under specific folders without having to use them globally.

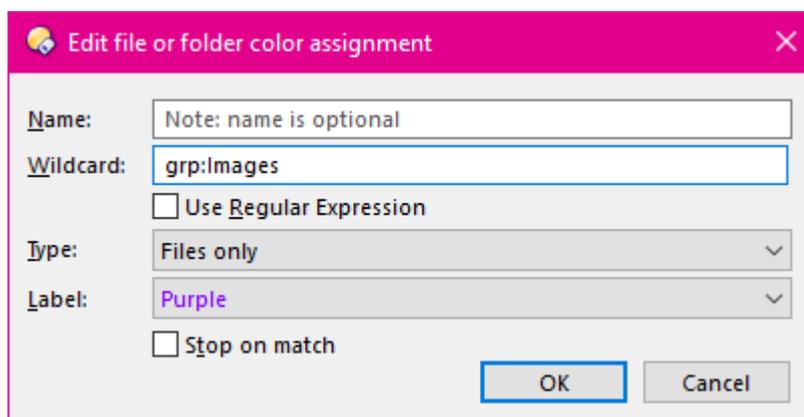
Use the toolbar buttons at the top to assign new labels:

-  **Create a new label filter:** You can use [filters](#) to assign labels based on attributes other than filename or location - they can be based on [any criteria supported by the filter system](#), although you should keep in mind that the filter will be compared against every file in every directory you visit - potentially slow filter clauses like *Contains* should be used with caution.



The above screenshot shows a label filter that will color red any files or folders that are under SVN control and have been modified. It uses a shell column provided by [TortoiseSVN](#) to match the SVN status.

-  **Create a new wildcard label:** Labels files and folders based on their names or locations. You can specify a pattern using standard [Opus wildcards](#) or [regular expressions](#), and choose whether to restrict the assignment to files, folders or both. The pattern can be used to match just the filename or a whole path; for example, you could have a wildcard pattern that matched **.doc** and **.docx** files in your Documents folder:



-  **Assign label to a specific folder path:** Lets you assign a label to a folder by its path. That is, the label will be attached to the path and name of the folder, not the folder itself.

-  **Assign label to a specific file:** Assign a label to a file by its path. That is, the label will be attached to the path and name of the file, not the file itself.

By default filter and wildcard labels assignments will "stack" on top of each other. One label could change the color of an icon, and another could change the icon itself. You can use the **Stop on match** option on both wildcard labels and label filters to prevent this from happening. Note that filter and wildcard label assignments can be rearranged to control their priority. The **up** (  ) and **down** (  ) buttons let you move label assignments above or below each other.

The labels themselves are configured on the [Favorites and Recent / Labels](#) page in Preferences (click the **Edit labels** link to be taken there automatically). See the [Labels](#) page for information on the labels system.

## Folder Formats

Opus lets you change many things to do with how it displays the contents of folders - view mode (details, thumbnails, etc), columns displayed in details mode, the sort order, grouping options, and more. Collectively these settings are referred to as folder options. You are able to save sets of folder options for specific folders so that those folders are always displayed in a particular way - saved folder options are referred to as *folder formats*.

The [Folders / Folder Formats](#) page in Preferences is the main interface for configuring folder formats. You can configure several different types of folder formats through that interface:

- **Path Formats:** Saved formats for specific paths. Path formats can affect a single folder or all sub-folders of that folder. Path formats can also be defined using wildcards rather than an absolute path.
- **System folders:** You can configure the format for certain system locations like *Computer*.
- **Content Type Formats:** You can configure a format for each [File Type group](#), which lets Opus [change the display format automatically](#) based on the contents of the folder.
- **Folder Type Formats:** You can configure default formats for many types of drive (local drives, network, file collection, etc.)
- **Favorite Formats:** You can define favorite formats, which you can access quickly using the **Folder** drop-down menu in the default toolbar.
- **Default Format:** This contains the *User Default* format, which is a format you can configure that will be used whenever another, more specific format isn't found for a folder.

Please see the [Folder Options](#) page for a description of the various options available, and the [Folder Formats](#) Preferences page for information on how to configure folder formats.

## Content Types

The Content Types system lets you configure Opus to automatically change the display format for a folder based on the contents of that folder. For example, the display can automatically change to thumbnails mode whenever you navigate to a folder containing mostly images.

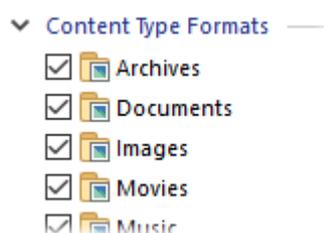
There are two distinct concepts that make up the Content Types system:

- **File Type Groups:** Content Types are built on the configured [file type groups](#). So you can have content type formats for Images, Music, etc. - any file type group that you create can have a content type format associated with it.
- **Folder Formats:** The [Folder Formats](#) system is used to define the display format that is applied for each file type group.

Because it can be confusing to have Opus change the folder format when you're not expecting it, you need to specifically enable the content types system in Preferences. The option is on the [Folders / Folder Behavior](#) page, called **Enable Folder Content Type detection**. You can enable it for all folders or selectively based on the type of drive (local, network, etc.)

Enable Folder Content Type detection for all local and network drives ▾

Once the Content Type system is enabled, you can define the folder formats (what aspects of the folder display are changed, and what they're changed to) through the [Folder Formats](#) page in Preferences.

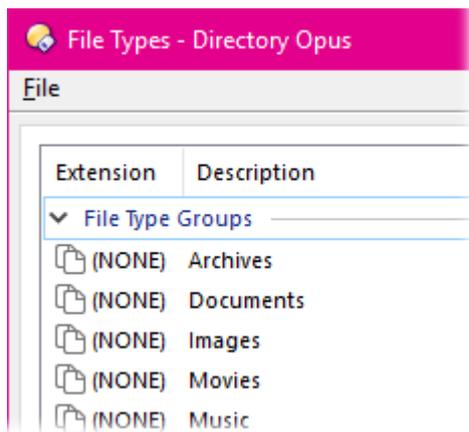


The content type format can be enabled or disabled, and the actual format can be defined, for each configured file type group. On the **Options** tab for each content type format, the **Content threshold** setting lets you configure the percentage of files in a folder that must match the file type group for the content type format to be used.

Content threshold:  %

In the above, rather artistic, screenshot, you can see that the **Content threshold** for the **Images** content type format has been set to 25%. This means that if you navigate to a folder where at least 25% of the files within it are images (or rather, files whose extensions have been added to the **Images** file type group), the display would change to use the format defined by the content type.

You can add new file type groups, or edit the existing ones, through the [File Types](#) editor.



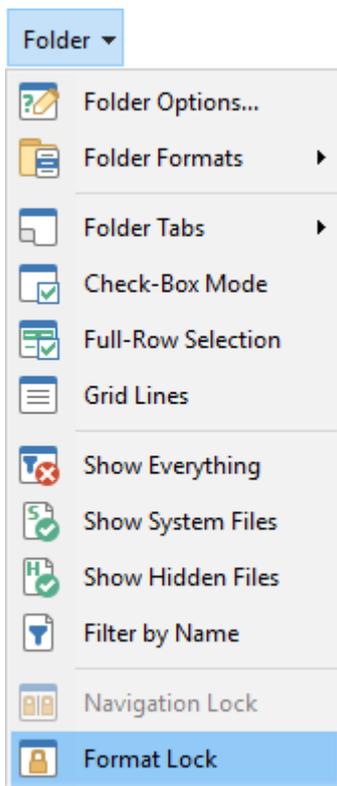
When you [add a new group](#) through the [File Types](#) editor an entry for it will automatically be created in the **Content Type Formats** section of the [Folder Formats](#) Preferences page.

## Locking the Format

Even if you have configured Opus to change format automatically as you navigate from one folder to the other (say, through the [Content Types](#) system, or by saving a [folder format](#) for a specific folder), there may be times when you don't want the format to change at all. You may have added a set of columns or changed the view mode specifically to accomplish a task, and having the format change automatically and to have to go back and reconfigure those things every time you change folder would certainly be a nuisance.

Opus therefore lets you **lock** the format in the file display. When the format is locked, automatic changes to the format are disabled - the display format won't change at all as you navigate from one folder to another, unless you manually change it using the [Folder Options](#) dialog.

There are two ways to toggle the format lock. The first is by selecting the **Format Lock** command from the **Folder Options** drop-down menu on the toolbar.



The other way is by clicking the padlock icon on the Lister's status bar.

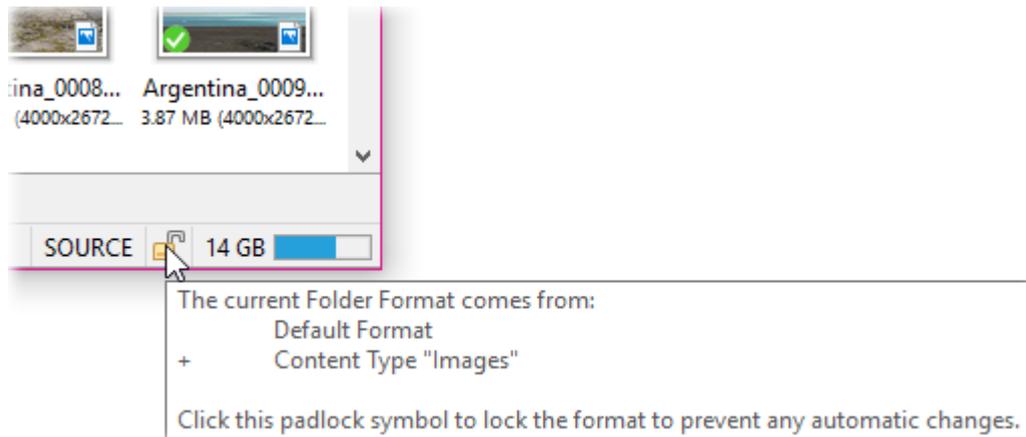


Either method will toggle the format lock on and off. When you turn the format lock on the current display format is preserved. You can still make manual changes using the [Folder Options](#) dialog (or by clicking the column headers in details mode, etc.), but it won't automatically change again until after you turn the lock off.

## Identifying the current format

As you may have gathered, the folder formats system is rather complex - we find it's one of the things new Opus users are often most confused by. The display format can change automatically, sometimes seemingly at random, and it can be quite possible to have no idea why a particular format is being used.

A feature is provided to try to help with this problem. If you hover the mouse over the padlock icon in the status bar, the tooltip that appears for it (when it's not locked) will give a brief description of how and why the current display format came about.



In the above example we can see that the current format in this file display started as the default format (no customisations), but was then modified by the **Images Content Type** format (which explains why it is displaying thumbnails).

# Flat View

Flat View is a mode you can turn on in a Lister that enables you to view all the contents of a folder's sub-folders at once. In effect it collapses, or flattens, the display of a nested hierarchy of folders and files into a single folder.

You can turn Flat View on using the  button on the default toolbar - this enables *Grouped* mode. The **Folder** menu has a *Flat View* sub-menu that gives you access to the other modes. You can also configure [Lister Styles](#) that automatically switch Flat View on when you switch to them.

The three different types of Flat View mode are:

- **Mixed:** Displays the contents of the current folder and all its sub-folders (and all their sub-folders, and so on recursively) as if they were all in the one location.
- **Mixed (No Folders):** Displays all the files in the current folder and all sub-folders as if they were all in the one location. Sub-folders themselves are hidden from the display.
- **Grouped:** Displays the contents of all sub-folders as a tree hierarchy.

The following screenshots illustrate the differences between the modes.

Name ▲	Size	Type	Modified	Attr
 ..		Parent Folder		
 one		File Folder	Today 4:49 PM	-----
 three		File Folder	Today 4:49 PM	-----
 two		File Folder	Today 4:49 PM	-----

In the starting folder, with Flat View off, all we can see is that there are three sub-folders. We have no idea what the contents of those folders might be.

Name ▲	Size	Type	Modified	Attr	Location
 ..		Parent Folder			
 five		File Folder	Today 4:49 PM	-----	three
 four		File Folder	Today 4:49 PM	-----	one
 one		File Folder	Today 4:49 PM	-----	
 six		File Folder	Today 4:49 PM	-----	two
 three		File Folder	Today 4:49 PM	-----	
 two		File Folder	Today 4:49 PM	-----	
 Argentina_0004.jpg	5.13 MB	JPEG image	30/11/2010 7:07 PM	-a----	one\four
 Argentina_0005.jpg	4.96 MB	JPEG image	1/12/2010 3:07 PM	-a----	three\five
 Argentina_0006.jpg	4.84 MB	JPEG image	1/12/2010 3:07 PM	-a----	two\six

Turning on **Flat View - Mixed** causes Opus to read the contents of all sub-folders (recursively) and add them to the list. Instantly we can see that below the top-level, there are three additional sub-folders, and that somewhere in the hierarchy there are also three image files. The **Location** column, which is displayed in Flat View mode by the default settings in the [Folder Formats](#) system, displays the location of each item, relative to the base folder.

Name ▲	Size	Type	Modified	Attr	Location
 ..		Parent Folder			
 Argentina_0004.jpg	5.13 MB	JPEG image	30/11/2010 7:07 PM	-a----	one\four
 Argentina_0005.jpg	4.96 MB	JPEG image	1/12/2010 3:07 PM	-a----	three\five
 Argentina_0006.jpg	4.84 MB	JPEG image	1/12/2010 3:07 PM	-a----	two\six

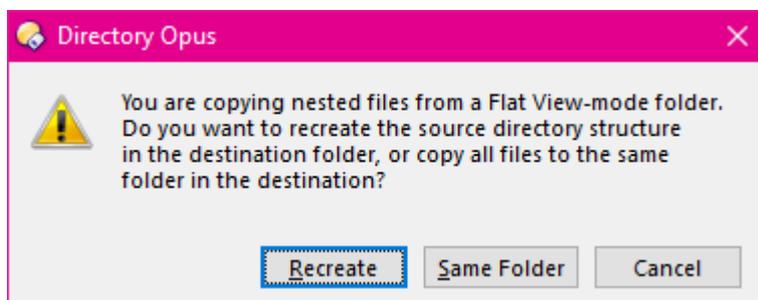
**Flat View - Mixed (No Folders)** removes all folders from the display, leaving only the files behind. This can be very useful if you want to quickly see what files are contained in a directory structure but don't care so much about the which folders the files are in (although the **Location** column does still display this information).

Name	Size	Type	Modified	Attr	Location
..		Parent Folder			
one		File Folder	Today 4:49 PM	-----	
four		File Folder	Today 4:49 PM	-----	one
Argentina_0004.jpg	5.13 MB	JPEG image	30/11/2010 7:07 PM	-a----	one\four
three		File Folder	Today 4:49 PM	-----	
five		File Folder	Today 4:49 PM	-----	three
Argentina_0005.jpg	4.96 MB	JPEG image	1/12/2010 3:07 PM	-a----	three\five
two		File Folder	Today 4:49 PM	-----	
six		File Folder	Today 4:49 PM	-----	two
Argentina_0006.jpg	4.84 MB	JPEG image	1/12/2010 3:07 PM	-a----	two\six

Finally, **Flat View - Grouped** mode displays a representation of the actual directory structure. The indenting makes the relative location of items immediately obvious - you can see instantly that the **IMGP2927.JPG** file is in the folder **five**, which is itself inside the folder **one**.

For the most part you don't have to treat a Lister that's in Flat View mode any differently to any other - you can double-click, drag-and-drop, view, copy, and delete files in the same way you normally do. You can use drag-and-drop to move or copy files into nested sub-folders - you can even move files from one sub-folder to another in this way. One thing to keep in mind is that if you drop a file into the file display - but **not** onto a sub-folder - then the base folder will be used as the target. Normally when you drag a file you can't drop it in the source folder - as this would be a no-op - but in Flat View you can do this, and one effect of this is that it can be easy to accidentally move a file from a nested sub-folder to the base folder without necessarily meaning to. You can always undo such operations, but it's worth remembering this is a possibility.

The only real difference in file operations in Flat View mode comes when you copy files from within sub-folders (via drag-and-drop, copy-and-paste or using the **Copy Files** command). For example, say you select the three image files in the above screenshot, and drag them to another Lister. Opus will display a dialog asking how you want to perform the copy:



If you choose the **Recreate** option, Opus will recreate the source folder structure (relative to the base folder) in the destination. In the above example, this means in the destination it would create a folder called **one**, a sub-folder inside that called **four**, and then copy the **IMGP2926.JPG** file into it. The **Same Folder** option discards the source folder structure - all files would be copied directly into the target folder. One complication that can arise with this is if you have selected

files with the same name from multiple sub-folders. In the source directory structure this would be no problem, as they would all be in separate folders, but if you try to copy them all to the same target folder the names would clash. In this case Opus will prompt you to rename or skip over the clashing files.

# Virtual File System

As you would know, the *file system* is the technology provided by the operating system that manages files and folders on your disks. It keeps track of where on the physical disk the data that represents a file is stored, maintains a list of filenames and information about files, and generally provides all services relating to creating and accessing disk files.

As well as managing standard files and folders stored on hard drives, flash disks, etc, Opus also provides several *virtual* file systems that are used to access and manage files that aren't stored (at least directly) on traditional media. For example, a Zip archive is a single file stored in the traditional file system, but Opus provides a virtual file system that represents all the files contained within that Zip file. So instead of the traditional way of accessing the contents of an archive (extracting the contents to a folder), you can use the virtual file system to access the contents directly. In general, virtual file systems in Opus behave the same as real ones, and you don't really need to be aware of the distinction between them - other than to know that the functionality exists.

The virtual file systems provided by Opus include:

- [System virtual folders](#): Folders like *Computer* and *Desktop* folders are handled natively in Opus.
- [File Collections](#): File Collections are virtual folders that let you create collections of files and folders that are not necessarily located in the one physical location.
- [Libraries](#): Libraries let you join multiple folders together (virtually) and presents a unified view of their contents.
- [Archives](#): Opus supports many different archive formats (Zip, RAR, 7-zip, etc) and lets you treat them as if they were normal folders.
- [FTP](#): Lets you access remote FTP sites as if they were local folders.
- [MTP](#): Lets you view the contents of MTP (*Media Transfer Protocol*) compatible devices like phones, tablets and cameras.

Opus has a [plugin system](#) that allows third-party developers to add their own virtual file systems. For example, a plugin might let you access the system registry as if it were a real folder.

## System virtual folders

In Windows there are a number of virtual folders that are presented as being part of the file system (also known as the *shell namespace*). The *Desktop* folder is an example of a virtual folder. It is a folder that can contain real files (e.g. if you save a document to the desktop, it is stored in a real disk folder) but it also contains virtual items that don't correspond to a real file on a disk (for example, the desktop contains a link to the *Recycle Bin*, and to the *Computer* folder).

When you navigate to a virtual folder in the shell namespace, the Lister acts as a container and hosts an instance of Explorer to provide the display. The advantage of doing this is that it lets you

navigate the entirety of the shell namespace from within Opus. Many third-party developers have written what are called *shell namespace extensions*. For example, if you connect your mobile phone or digital camera to your PC, you are often able to copy files to and from it in Explorer. This functionality is implemented by a small piece of software that extends the shell namespace and provides access to an otherwise proprietary device. If Opus did not support hosting Explorer for these third-party folders, you would be forced to leave Opus and actually use Explorer to access them - and we wouldn't want you to have to use Explorer any more than necessary!

The disadvantage of hosting Explorer is that to Opus, the folder appears as a "black box". Opus literally has no idea what the contents of the folder are - even the display is provided by Explorer, with Opus doing no more than providing a place for Explorer to put its window. Therefore much of the native Opus functionality is unavailable in these virtual folders.

For that reason, Opus provides a native implementation of several virtual folders. This means that you can use Opus features like [Folder Formats](#), [Power mode](#) or Preferences settings like [background images](#) in some commonly used virtual folders. The virtual folders currently supported by Opus are:

- **Desktop:** The Desktop is the root of the shell namespace. It displays the merged contents of two disk-based folders (your personal desktop folder, and the *All Users* desktop folder) as well as links to various virtual folders like Recycle Bin, Network, Libraries, etc. The native view provided by Opus can be configured from the [Folders / Virtual Folders](#) Preferences page - you can choose exactly which virtual folders are displayed on the Desktop.
- **Computer:** The Computer folder (or in Windows XP, *My Computer*) displays the disk drives and devices installed in your system. By default empty disk drives (e.g. a DVD drive without a disc in it) are not displayed - you can change this from the [Folders / Virtual Folders](#) Preferences page.
- **CD Burning:** Opus provides a native view of writeable CD and DVD disks when using the Windows CD burning system (with staged burning only - the LiveFS packet writing system is treated like a normal disk folder).

Some virtual folders, like *Computer*, are purely virtual - they don't correspond to a real folder on disk. Other folders, like the *C:\Windows\Fonts* directory, are real disk folders that have a virtual "overlay" when displayed in Explorer. The options on the [Folders / Virtual Folders](#) Preferences page let you control how Opus handles this sort of directory - if desired, you can choose to have Opus display the underlying "real" folder rather than the virtual presentation.

## File Collections

A file collection is a virtual folder that acts as a container for other files and folders. The items in a file collection do not need to be stored in the same physical folder - they can reside in other folders, on other disks, or even in other [virtual file systems](#) like FTP. When you add items to a collection they are not actually moved or modified at all - all Opus stores is a reference to the original file.

Name	Size	Type	
..		Parent Folder	
2009-07-30 22.33.10.jpg	875 KB	JPEG image	18/11/2009
2009-07-30 22.33.42.jpg	705 KB	JPEG image	18/11/2009
2009-07-30 22.33.52.jpg	1.20 MB	JPEG image	18/11/2009
Argentina_0002.jpg	4.11 MB	JPEG image	30/11/2009
Argentina_0003.jpg	4.48 MB	JPEG image	30/11/2009

File collections are used by the [Find Files](#) and [Duplicate File Finder](#) functions to display the results of the search. When you perform a search with these tools, any matching files are automatically added to the specified file collection. You can see an example of this in the above screenshot - a search for image files returned pictures stored in two different locations, but in the results collection they are presented as if they were all in the same place.

You can work with the files in a file collection (copy them, rename them, delete them, etc.) just like you would with the real files. Collections are also used by the [Flickr photo-sharing synchronization](#) system to store references to your photos no matter where on your system they are physically located.

The root **File Collections** folder contains all your file collections (you can't add items to the root itself). To access the file collections root you can:

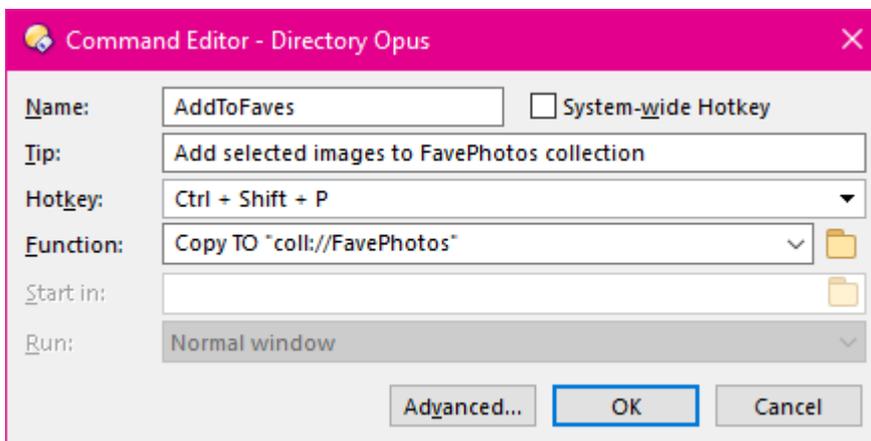
- Click on the **File Collections** item in the folder tree (unless you have disabled this from the [Folder Tree / Contents](#) page in Preferences)
- Double-click on the **File Collections** item on the Desktop (unless this has been disabled on the [Folders / Virtual Folders](#) Preferences page)
- Select the **File Collections** command from the drop-down menu attached to the Up button on the [File Display Toolbar](#) (the  button on the toolbar - right-click it or click-and-hold with the left button to show the menu)
- Click in the [location field](#) and enter the path **coll://** (see below for more information about this path format)

From the file collections root you can use the **New Folder** function on the toolbar to create a new collection. You can also create sub-collections in the same way - simply navigate to the parent collection and create a new folder like normal. You can assign your own icon to a file collection through its Properties dialog - you can also assign a description string if desired.

You can also define the default format for collections - this comes pre-configured to add the **Location** column whenever you navigate to a collection. This column is useful as it displays the real location of the items in the collection.

The [File Types](#) dialog can be used to add commands to the context menu for items in collections. Two commands are pre-defined; **Remove from Collection** (mentioned above), and **Open Containing Folder**. The latter command uses the [Go OPENCONTAINER](#) function to open the real folder that contains the selected collection item.

File collections are referenced internally using a URL-style path format, with the **coll://** prefix. For example, the path of the default find results collection is **coll://Find Results/**. You can type this sort of path into the location field, or use it in buttons and hotkeys with the internal command set to automate your use of collections. For example, say you had a collection called **FavePhotos** you use often to keep track of your favorite photos. Whenever you have an image that you want to add to it, you could expand the **File Collections** branch in the tree, locate your collection, then click and drag the image and drop it on the collection folder. If you're doing that often though, there's a much quicker way - you could [set up a button or hotkey](#) to automatically add the selected files to that collection, using the raw [Copy](#) command.



This screenshot illustrates an example hotkey that performs this function. Now all you would need to do is select the photos you want to keep, press **Ctrl+Shift+P**, and the files would be automatically added to the **FavePhotos** file collection.

Unfortunately the file collections system is only available within Directory Opus - you can't access your collections from other programs or from Explorer. However there are a couple of options on the [Miscellaneous / Windows Integration](#) page in Preferences that let you integrate collections more tightly into the system:

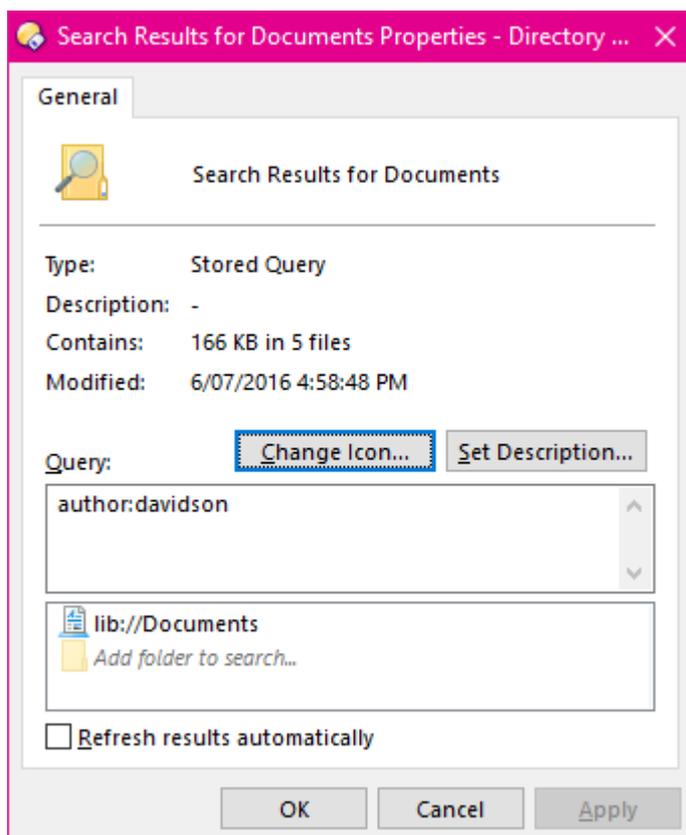
- **Add File Collections icon to the Desktop:** If this option is enabled Opus will place an icon on your desktop that represents the root file collections folder; double-clicking it will open a Lister showing your file collections.
- **Add File Collections list to the Send To menu:** This option causes Opus to add links for your file collections to the *Send To* menu (the menu that is displayed when you right-click a file and select *Send to* from its context menu). This lets you add items to file collections from outside of Opus.

The [DOpusRT](#) command can also be used from outside of Opus to access and manipulate file collections - see [External Manipulation of File Collections](#) for more information.

## Stored Queries

Stored Queries are special file collections that are used to create persistent searches with the [Windows Search](#) system. Each stored query remembers an [Advanced Query Syntax](#) search string and one or more folders to search within.

You can create a stored query from the **File Collections** root folder - right-click on the folder itself, or from within the folder right-click on the file display background, and choose the **New Stored Query** command. You can also create a stored query from an existing search - see the help on the [Windows Search](#) system for more details about this. Once you have created a stored query, right-click on it and choose the **Properties** command to display its Properties dialog - this is where you can configure the search parameters.



The **Change Icon** button lets you select your own icon for the stored query collection, and the **Set Description** button lets you assign your own description string if desired. These function the same as for normal collections. Below these buttons are options that only apply to stored queries.

The **Query** field is where you enter the AQS query term that defines the search. Below that is a list of folders to search in. The query can search in a single folder (as in the screenshot above - only the **Documents** library is being searched), or it can search multiple folders at once.

If the **Refresh results automatically** option is turned on, the query will be run and the results updated every time you navigate into the collection. You would generally only turn this option on for indexed locations, where the results will be returned almost immediately. If this option is left turned off, the collection will preserve the last set of search results until you manually run the query by refreshing the folder (by pressing the **F5** key, for example).

## Libraries

Libraries are a concept introduced by Microsoft in Windows 7. A library is a virtual folder that displays the contents of other folders. Unlike a file collection, which can store references to individual files and folders, a library displays the entire contents of one or more folders. You can't add individual files to a library - all you can do is change which folders make up that library. You can work with the files in a library (copy them, rename them, delete them, etc.) just like you would with the real files. Opus emulates the library system for Windows XP and Vista users - for these users, libraries won't be available in other programs like they are in Windows 7, but you can at least use them Opus.

By default four libraries are defined - **Documents**, **Music**, **Pictures** and **Videos**. Under Windows 7 the system sets these up; in the emulated libraries that Opus provides for earlier operating systems, Opus creates them automatically. In Microsoft's new model, the use of the old user profile folders like *My Documents* and *My Pictures* is discouraged - instead the corresponding libraries can be used instead. You can restore these default libraries at any time by right-clicking on the root **Libraries** folder and choosing the **Restore default libraries** command.

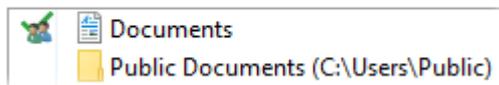
The root **Libraries** folder holds all your libraries. To access the libraries root you can:

- Click on the **Libraries** item in the folder tree (unless you have disabled this from the [Folder Tree / Contents](#) page in Preferences)
- Double-click on the **Libraries** item on the Desktop (unless this has been disabled on the [Folders / Virtual Folders](#) Preferences page)
- Select the **Libraries** command from the drop-down menu attached to the Up button on the [File Display Toolbar](#) (the  button on the toolbar - right-click it or click-and-hold with the left button to show the menu)
- Click in the [location field](#) and enter the path **lib://** (see below for more information about this path format)

From the libraries root you can use the **New Folder** function on the toolbar to create a new library. Newly created libraries initially have no member folders, and so will appear empty - you need to include one or more folders in the library in order to use them. This is accomplished with

the use of the **Properties** dialog - right-click on your newly created library and select the **Properties** command to display it.

Library locations:

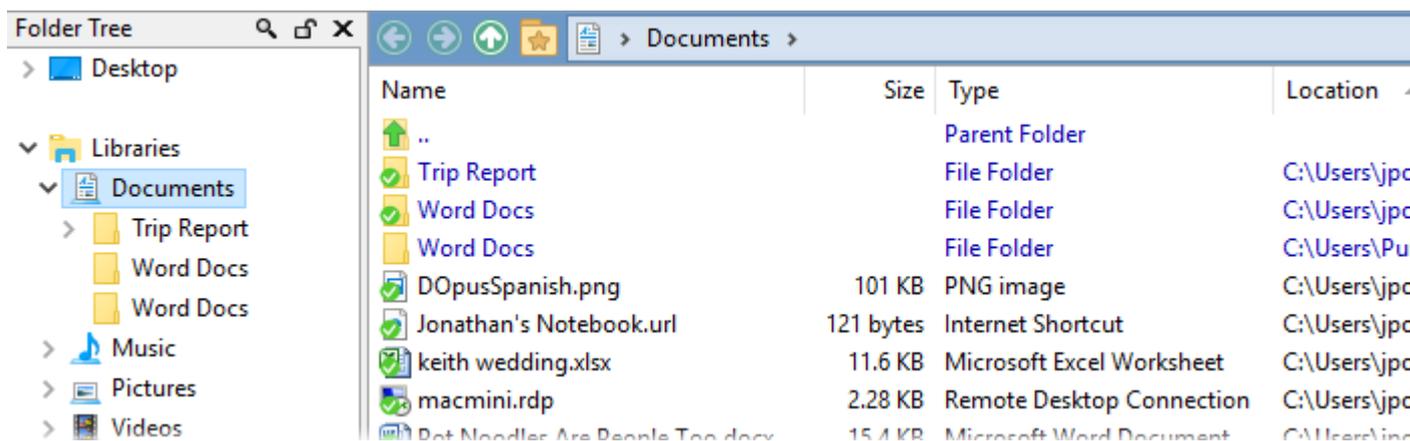


The list of member folders is displayed in the **Library locations** list. Click the **Include a folder** button to add a new member folder to the library. When you click **Apply** or **OK** the contents of the new member folder will instantly appear inside the library. To remove a member folder, select it in the list and click the **Remove** button.

The checked member folder (**My Pictures** in the above screenshot) is the default *save location* for the library. This folder is used whenever you copy or move a file into the library (or in Windows 7, when you save a file to the library from another program). To change which member is the default save location, select it in the list and click the **Set save location** button.

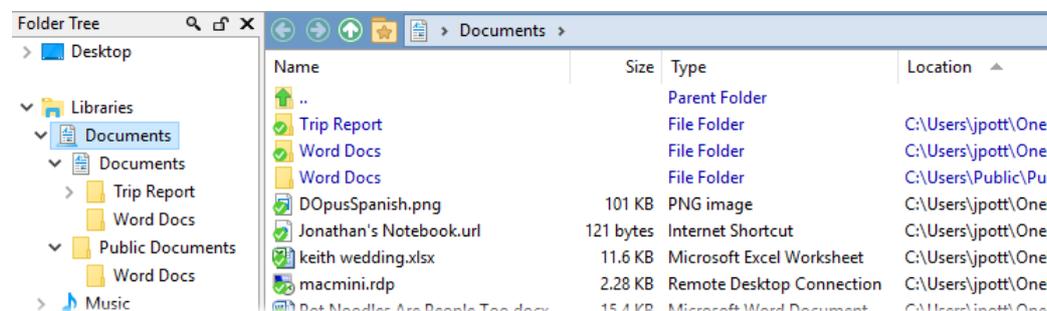
On Windows 7 you can also add a folder to a library from the right-click context menu (using the **Include in library** sub-menu). This is not available on earlier operating systems by default, but if you like you can add it yourself using the [File Types](#) system - all you need to do is add the command **Copy INCLUDEINLIBRARY** to the context menu for the *All Folders* file type.

When you navigate to a library, the file display shows the unified contents of all member folders. If your various member folders contain sub-folders or files with the same filename, it may not be obvious when looking at the library which item is which.



The **Location** column can help with this. As you can see in the above screenshot, two of the **Documents** library's member folders contain a sub-folder called **Word Docs**. The **Location** column displays the true location of these folders, making it much easier to tell them apart. You can add the **Location** column (and other columns) using the [Folder Options](#) dialog - or use the [Folder Formats](#) system to set the default format for a library.

Another option that can make it easier to keep track of library members is the **Show member folders individually** option on the [Folder Tree / Contents](#) page in Preferences.



With this option turned on the folder tree (this doesn't affect the file display) will display an additional branch level for each member folder. As you can see from the above screenshot, this makes it obvious there are two separate **Word Docs** folders. You can also take advantage of this to copy files into a library member other than the default save location, by dragging and dropping it to the specific member folder.

You can hide an individual library from the tree by right-clicking it and turning off the **Show in Folder Tree** option. To add it back to the tree, use the similarly-named option on the library's **Properties** dialog.

Libraries are referenced internally using a URL-style path format, with the **lib://** prefix. For example, the path of the default Pictures library is **lib://Pictures/**. You can type this sort of path into the location field, or use it in buttons and hotkeys with the internal command set to automate your use of collections. For example, you could [set up a button or hotkey](#) to automatically navigate to the Pictures library using the raw command **Go lib://Pictures**.

The **lib://** path system isn't recognised outside of Opus, even on Windows 7, so if you ever want to copy a library path to the clipboard and paste it into another program, you should use the **Edit / Copy Other / Folder Path** menu command rather than copying the contents of the location field.

## Archives

Directory Opus supports many different archive formats. Support for Zip files (the most common Windows archiving format) is built-in to the program, and many other formats (like RAR, 7-zip, ISO, etc) are supported via a plugin system. Whether support for a format is built-in or implemented via a plugin, Opus lets you treat all supported archive formats as if they were

normal folders. Double-clicking a supported archive format will navigate into it just like a folder, and you can copy files out of it, or view its contents directly, just like normal files. For the formats that support creating archives, you can add files to an archive using the normal copy functions like copy-and-paste or drag-and-drop.

There are a large number of options in Preferences that you can use to configure archive support in Opus. For Zip files, see the [Zip & Other Archives / Zip Files](#) page, and for all other formats see the [Zip & Other Archives / Archive and VFS Plugins](#) page. You can also choose whether archives are shown in the folder tree (making them even more like folders!) with the **Show in Folder Tree** setting on the [Folder Tree / Contents](#) page.

The **Archive Files** button on the default toolbar lets you add selected files to a new archive. The drop-down attached to that button contains a number of additional archive-related commands that let you create or extract archive files. Extensive control of archives is also provided for supported formats via the context menu - you can configure which context menu items are displayed for each format from the [Zip & Other Archives / Archive Context Menu](#) page in Preferences.

If you want to disable support in Opus for a certain format you can do that from the [Zip & Other Archives](#) Preferences section. However, if all you want to do is stop double-click from opening the archive in Opus (because you have another tool installed that you want to use for those archives), another way to do it is by removing the format's file extension from the **Archives file type group**. This is a special group that all supported archive formats are added to by default. You can also use this group to add context menu items for all supported archive formats in one go.

## FTP

You can access remote FTP sites as if they were local folders. Almost all Opus commands will function as they would with normal folders - copying, deleting, creating folders, etc, is all handled just like on a local disk. You can even use the viewer pane to view files on the FTP site without downloading them first.

See the [FTP](#) section of this help file for full information on using the FTP system in Opus.

## MTP

Using Opus you can access the contents of MTP (*Media Transfer Protocol*) compatible devices like cameras, tablets and phones. Almost all Opus commands will function as they would with

normal folders - copying, deleting, creating folders, etc, is all handled just like on a real disk drive. The viewer pane and thumbnails mode can be used as normal.

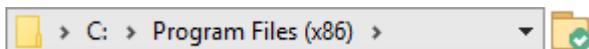
MTP devices are referenced internally using a URL-style path format, with the **mtp://** prefix. For example, the path of your Nexus 7 tablet would be **mtp://Nexus 7/**. You can type this sort of path into the location field, or use it in buttons and hotkeys with the internal command set. If you go to the *root* of the MTP namespace (**mtp://**) by typing the path into the location field, the file display will show you all currently-connected MTP devices. You will also see these devices listed underneath *Computer* (or *My Computer* in XP) in the *Portable Devices* category.

## Compatibility Files

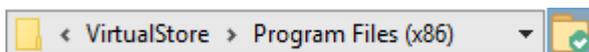
Windows Vista introduced several new security-related concepts - the most well known of these is [UAC](#). One of the features of the new security system is the ability for file operations in certain protected locations to be automatically redirected to a different, non-privileged location. This system kicks into action whenever an older program that hasn't been updated for UAC tries to create or write to files in locations like the *Windows* directory. The file-system silently diverts the application to a different location - this means the old application doesn't break, but security is not compromised by allowing non-privileged applications write permission in important system folders.

Files that have been written to disk as a result of this system are known as *compatibility files*. Opus gives you the ability to manage compatibility files in two ways:

- Whenever you are in a protect location that has a compatibility folder, the **Compatibility Files** button () will appear on the [File Display Toolbar](#). Clicking this button will instantly move you to the compatibility folder for the current location. Click the button again to return to the original folder.

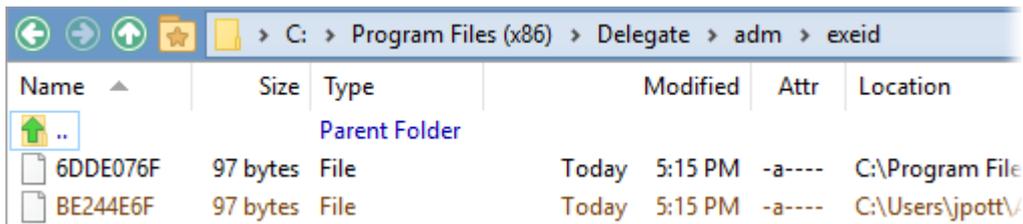


For example, say you are in the *C:\Program Files (x86)* directory. As this is a protected location, the **Compatibility Files** button may appear on the toolbar (if it doesn't appear it means there aren't any compatibility files for the current location).



Clicking the button would take you to your personal *VirtualStore* folder for the *Program Files (x86)* directory.

- You can turn on the **Show compatibility files** option on the [Folder Options](#) dialog's **Options** tab. If this option is on, any compatibility files for the current folder are automatically included in a merged view with the current folder. You can also set this option as a global default with the **Display Compatibility Files** where possible option on the [Folders / Folder Display](#) Preferences page.



Files from the alternate location are displayed in a different color to regular files and folders - you can configure this color from the [Display / Colors and Fonts](#) Preferences page. For example, the screenshot above shows that the program *Delegate* has attempted to write a file to the *C:\Program Files (x86)* folder, and this was silently diverted to the compatibility store. If you add the **Location** column to the display it also indicates which files are compatibility files.

# File Operations

While Opus has a lot of features, its file management functions are the most important (it is a file manager, after all!). Please see the following sections on the core file management functions:

- [Copying, Moving and Deleting Files](#): Probably the most fundamental file management functions, this includes information on using [copy-and-paste](#), [drag-and-drop](#) and the [source and destination](#) concept to move files around.
- [Renaming Files](#): Opus supports quick renaming of files by editing their names directly, and can also perform complicated wildcard and scripted renames.
- [Creating Folders](#) and [Creating Archives](#): Information on how to make new folders and archive files.
- [Changing Attributes](#): Opus makes it easy to modify file and folder attributes and timestamps.
- [Labels, Metadata](#) and [Descriptions](#): Describes the ways Opus can help you categorise and keep track of your files.
- [UAC and Administrator Mode](#): Opus provides full support for UAC in Windows Vista and later.

# Copying, Moving and Deleting Files

The most fundamental feature of a file manager is its ability to let you move files around. Opus provides several ways to copy and move files from one folder to another:

- [Copy-and-paste](#): This is the traditional method on Windows of moving files around using the clipboard.
- [Drag-and-drop](#): Again, fairly standard on Windows; using the mouse to drag files from one folder to another.
- [Toolbar buttons](#): Several default toolbar buttons let you copy or move selected files.

Other file copying-related features include:

- The [copy queue](#) function, which lets you queue multiple file copy operations to avoid disk thrashing or bandwidth contention.
- Files can be renamed as they are copied, either one at a time or using [wildcards](#).
- When filename collisions occur, the [replace dialog](#) lets you quickly choose what you want to do.
- Powerful features for [synchronizing](#) the contents of two folders (including remote FTP sites).
- A [filtering system](#) that lets you selectively copy the contents of folders

[Deleting files](#) is also a part of the basic file management tool set - Opus also provides a [secure delete](#) function to securely erase data.

## Copy and Paste

Copy-and-paste (or cut-and-paste) is possibly the simplest method of moving files around. Select the files you want to copy or move, choose the *Copy* (**Ctrl+C**) or *Cut* (**Ctrl+X**) command from the **Edit** menu, navigate to the target folder, and select *Paste* (**Ctrl+V**). Moving files using the clipboard is implemented in almost all file managers the same way, although Opus does extend this functionality in a few ways, which it's worth being aware of.

- Although these functions use the internal [Clipboard](#) command to initiate them, when you paste files into a Lister, the actual file copy or move is implemented "behind-the-scenes" using the internal [Copy](#) command. This means that features like the [copy queue](#) are available for clipboard operations just like they are when [copying via the toolbar buttons](#). You can control the queuing using the various arguments for the [Clipboard](#) command.
- If the clipboard contains text or graphics data, you can paste that in a Lister as if it were a real file. Opus will create a new file and write the clipboard data to it automatically. By default, text data will be written to a file called *Clipboard Text.txt*, and graphical data will be written to a file called *Clipboard Image.jpg*. You can modify the format that graphics data is saved in (Bitmap, GIF, PNG or JPG) using the

**clipboard\_image\_paste** option on the [Miscellaneous / Advanced](#) page in Preferences. The name of the new file can be controlled using the various arguments for the [Clipboard](#) command. You can also use the **clipboard\_image\_paste\_dpi** option to have the pasted image scaled automatically to compensate for the system DPI.

- The **ADD** argument for the internal [Clipboard](#) command lets you add selected files to any that are already on the clipboard. For example, you could copy a few files from one folder, a few from another folder, and a few from a third folder, and then paste them all into the destination in one go. To make use of this functionality, you would need to create a new toolbar button or hotkey that runs the **Clipboard COPY ADD** command (or assign this to a hotkey - for example, **Ctrl+C** runs **Clipboard COPY** by default, so you could assign this to **Ctrl+Shift+C**). See the section on [Customize](#) for information about creating your own buttons and hotkeys.

## Drag and drop

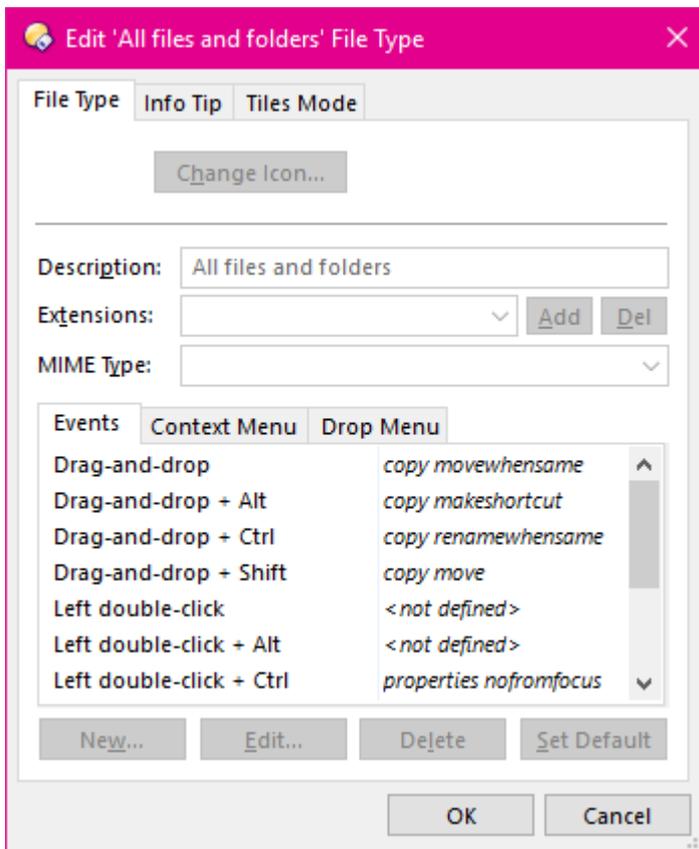
Drag-and-drop is a common way of moving files around in Windows. In Opus, drag-and-drop is controlled by the [File Types](#) system. The default behaviour is the same as in Explorer:

- Drag-and-drop of a file to another folder on the same drive will move it
- Drag-and-drop to a folder on a different drive will copy it
- Holding the **Shift** key when dropping forces the file to be moved
- Holding the **Ctrl** key when dropping forces it to be copied
- Holding the **Alt** key when dropping creates a shortcut to the file
- Drag-and-drop with the right button displays a menu that lets you select *Copy*, *Move* or *Create Shortcut* as the action.

Internally, these actions are defined by the items on the **Events** tab for the **All files and folders** file type. Whereas in Explorer these actions are fixed, in Opus they are configurable.

This screenshot shows the [File Type editor](#) for the **All files and folders** file type. As you can see, the events are all defined as some variant of the internal [Copy](#) command.

- *Drag-and-drop*: The **MOVEWHENSAME** argument specifies that the file is moved on the same drive, and copied otherwise.
- *Drag-and-drop + Alt*: The **MAKESHORTCUT** argument causes the Copy command to make a shortcut to the file.
- *Drag-and-drop + Ctrl*: The **RENAMEWHENSAME** argument will automatically rename the copied file if the name clashes with an existing one.
- *Drag-and-drop + Shift*: The **MOVE** argument causes the file to be always moved.



Although not shown here, the **Drop Menu** tab also defines the contents of the menu shown when dragging with the right mouse button. This can also be configured.

Editing these functions modifies the drag-and-drop behaviour in Opus. For example, if you wanted drag-and-drop to always copy (rather than moving on the same drive), you could edit the command definition to **Copy RENAMEWHENSAME** (the same as the *Drag-and-drop + Ctrl* event).

The **All files and folders** file type is a special file type that by definition matches everything - both files and folders. You can use the File Types system to define overriding events

for specific file types. For example, you could configure drag-and-drop of a Zip file to extract the contents of the archive when dropped with the **Shift** key held down.

See the documentation on the [File Types](#) system for more information.

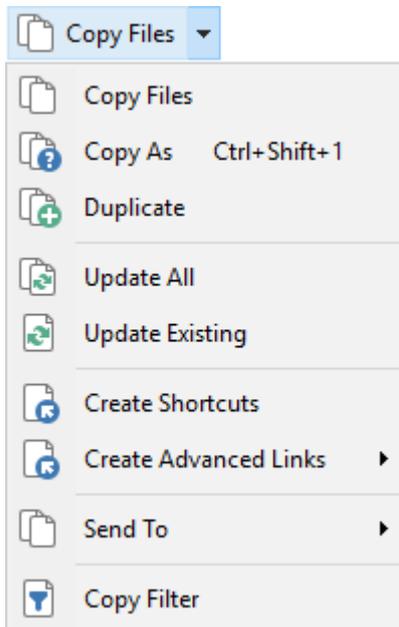
## Copying using the toolbar buttons

The third way of copying files, besides copy-and-paste and drag-and-drop, is to use the **Copy Files** and **Move** buttons on the toolbar.



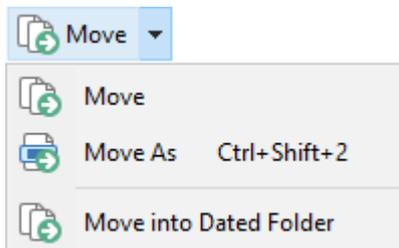
These buttons make use of the [source and destination](#) concept. Selected files will be copied or moved from the current source folder to the current destination - or, if there is no current destination folder, Opus will prompt you to select the destination folder.

The drop-down menu attached to the **Copy Files** button contains the following file copy commands:



- **Copy Files:** Copy selected files and folders to the destination folder.
- **Copy As:** Lets you enter new filenames or provide a wildcard pattern to [rename files as they are copied](#)
- **Duplicate:** Makes copies (duplicates) of selected files in the same folder.
- **Update All:** Updates all files. Only files that don't exist in the destination, or exist but are newer, will be copied. See [Copying Updated Files](#) for more information.
- **Update Existing:** Updates existing files. Only files that already exist in the destination and are newer will be copied. See [Copying Updated Files](#) for more information.
- **Create Shortcuts:** Creates shortcuts in the destination folder (.lnk) that point to the selected files. See [Making Links and Junctions](#) for more information.
- **Create Advanced Links:** Creates junctions and softlinks in the destination folder that point to the selected files. See [Making Links and Junctions](#) for more information.
- **Send To:** This displays the standard system *Send To* menu that lets you send selected files to various locations.
- **Copy Filter:** Turns on the [recursive filter](#), which lets you selectively copy the contents of sub-folders.

The drop-down menu attached to the **Move** button contains the following commands:

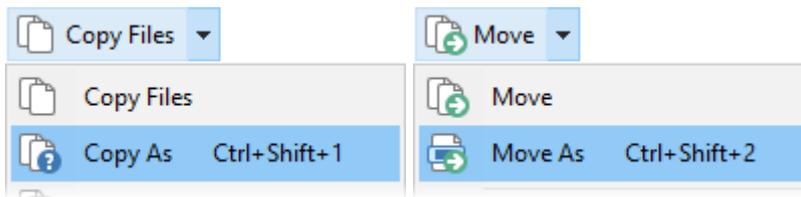


- **Move:** Move selected files and folders to the destination.
- **Move As:** Enter new filenames or a wildcard pattern to [rename files as they are moved](#).
- **Move into Dated Folder:** Creates a folder with the current date as its name, and moves selected files and folders into it.

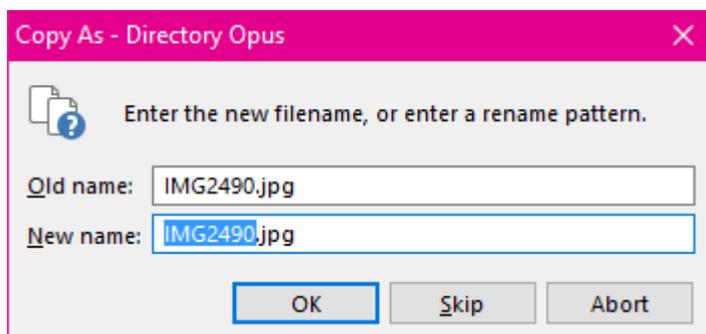
## Using Wildcards when Copying

The **Copy File** and **Move** commands have the ability to rename files as they are copied (or moved). This is known as **Copy As** or **Move As** (as in, *copy file XXX as YYY*). To use this functionality, select the files you want to copy or move and then select the **Copy As** or **Move As**

commands from the **Copy Files** or **Move** drop-down menus.



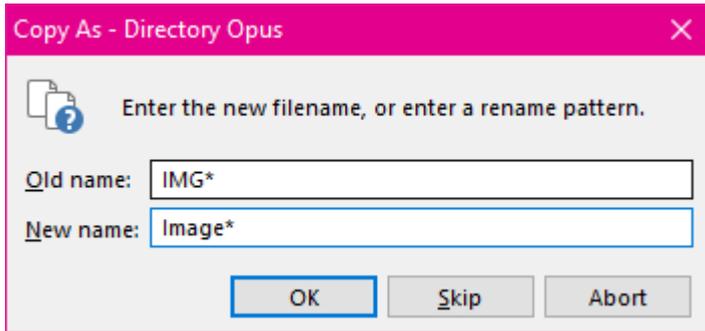
Selecting either of these commands will display a dialog box that lets you enter the new name for the copied or moved file.



The **Old name** field identifies the original file name, and the **New name** field lets you enter a new name for the file. If you simply enter a new name and click **OK**, the file will be copied or moved using the name you supplied. If you had multiple files selected then the dialog will reappear after the first file has been copied, to prompt you for a new name for the second file (and so on).

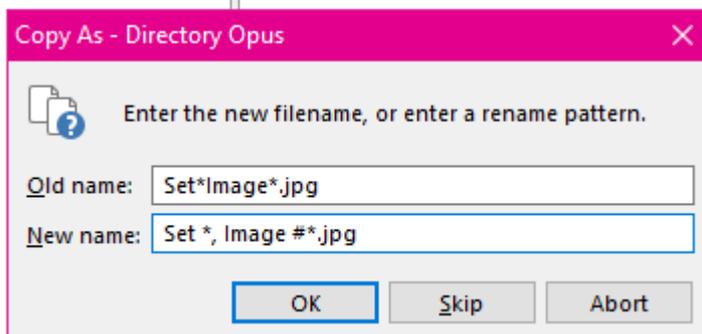
As an alternative to entering a new name for each file, you can use a simple wildcard system to rename all copied or moved files at once. This is not a full pattern matching (or regular expressions) system; instead, the only wildcard character that's recognised is \* (the asterisk). This is the same system used in the [Simple Wildcard Rename](#) function.

To rename using wildcards, you need to enter the "from" pattern in the **Old name** field, and the "to" pattern in the **New name** field. The asterisk is used to mark areas of text that are to be matched and retained from the old to new names. For example, to copy all files beginning with **IMG** to new files that begin with **Image** instead, you might do the following:



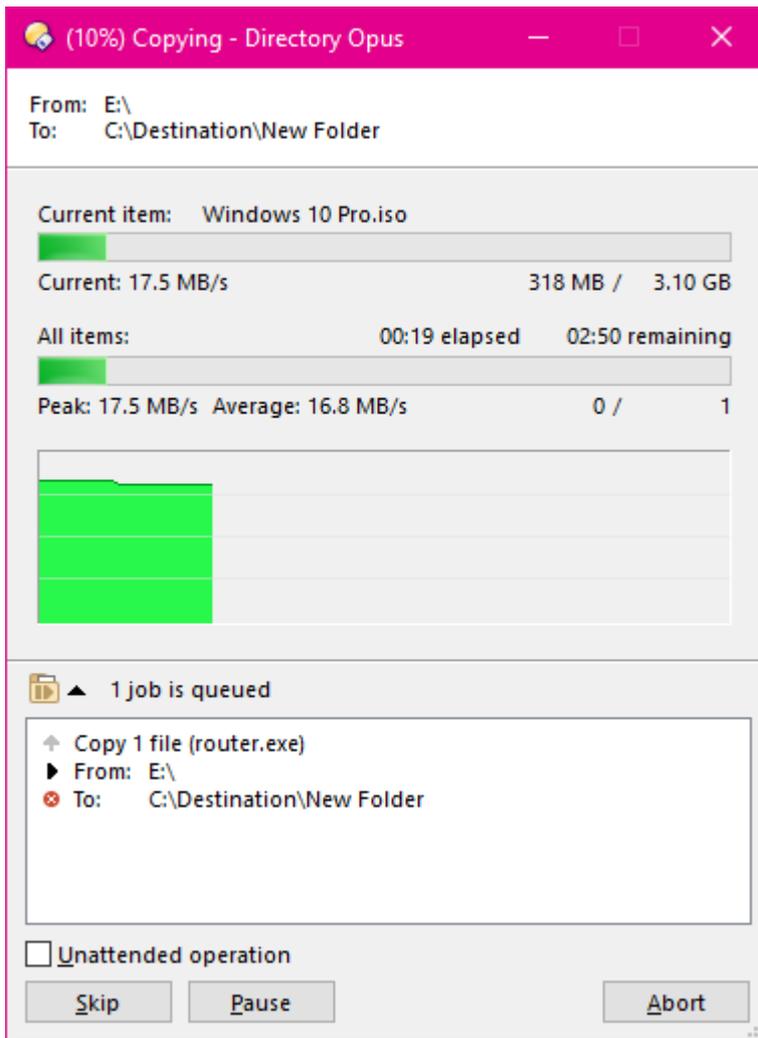
You can use as many asterisks as you like in the old and new patterns. The following image shows an example of this; the **Old** and **New name** fields each used two asterisks to preserve two separate parts of the name, while replacing the rest.

SetAlmage1.jpg	Set A, Image #1.jpg	4.13 MB	JPEG im
SetAlmage2.jpg	Set A, Image #2.jpg	4.13 MB	JPEG im
SetBlmage1.jpg	Set B, Image #1.jpg	4.13 MB	JPEG im
SetBlmage2.jpg	Set B, Image #2.jpg	4.13 MB	JPEG im
SetBlmage3.jpg	Set B, Image #3.jpg	4.13 MB	JPEG im



## Copy Queues

In Explorer, if you start a copy operation, and then while it is still running, start another one, the two operations will run in parallel. This can be undesirable - if the two copies involve the same physical hard drives, the drive heads will need to continuously seek back and forth ("thrashing"), reducing overall transfer speed. Similarly, when transferring files over an FTP connection, multiple simultaneous transfers can often reduce the overall throughput. To avoid this situation in Opus, multiple file copy operations can be queued to execute sequentially rather than in parallel.



The copy queue is used automatically whenever an attempt to make certain simultaneous copy operations occurs:

- If the copy originates **from** a removable disc, a CD/DVD or an FTP site, the copy queue is used if an existing copy from that source folder is in progress
- If the copy is **to** a removable disc, a CD/DVD or an FTP site, the queue is used if an existing copy to that folder is in progress
- If the copy is **to** a fixed (local) hard disk, the queue is used if an existing copy to that physical hard disk is in progress (multiple partitions on the same physical drive are considered the same)
- For all other copy operations, the queue is used if a copy is in progress **from** the same source **to** the destination

When a copy job is queued, the existing progress dialog expands to display a summary of the queued jobs (as shown in the screenshot above). You can use the controls next to each queued job to manage the queue:

- **⬆ Move up**: The queued jobs are executed in order from top to bottom so this button lets you move a job up the list to change the order it is run in.
- **▶ Run now**: Starts the job immediately, without waiting for the current job to finish.
- **✖ Abort**: Aborts the job - all other jobs will be unaffected.

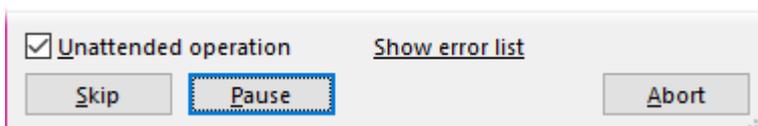
By default Opus will notify you when a job is queued, but you can disable this message with the **Display confirmation when a job is queued** option on the [File Operations / Copy Options](#) page in Preferences. Note that when jobs are queued, clicking the **Abort** button at the bottom of the progress dialog will give you the option to abort only the currently executing job, or all jobs.

Normally Opus manages the copy queue automatically (as described above), but you can use the internal [Copy](#) command to manage queuing manually if desired. Using the **QUEUE** argument you can create a queue with a specified name (this name will then be shown in the title bar of the progress dialog). For example, the command **Copy QUEUE=MyQueue** will copy selected files using the queue *MyQueue*. You could then configure a button or hotkey to copy using your named queue when a particular key was held down (e.g. **Shift**) - that way you could force the queue to be used when desired, even if Opus would ordinarily not queue the operation. See the description of the internal [Copy](#) command for more details about using the **QUEUE** argument, and the [Customize](#) section for information on how to create your own buttons or hotkeys.

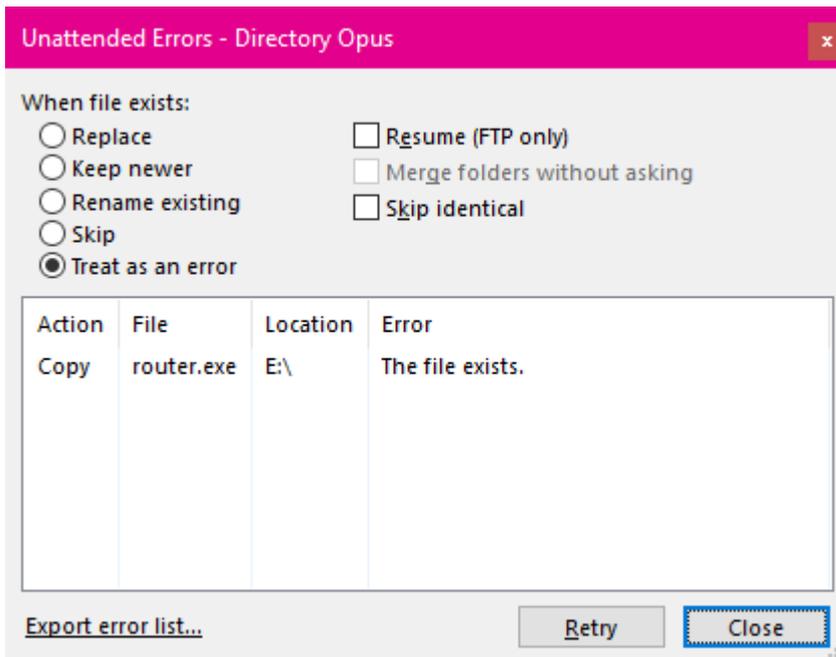
You can completely disable automatic queuing with the **Automatically manage file copy queues** option on the [File Operations / Copy Options](#) page in Preferences.

### ***Unattended operation***

The **Unattended operation** checkbox at the bottom of the Copy progress dialog is used to make a copy operation (including any copies in that queue) run non-interactively.



When a copy operation is set to unattended mode, no error or confirmation dialogs are displayed during the copy process. Instead, any errors will be silently logged and you can review these, and retry the failed operations, at the very end of the operation. This lets you start a very long copy job (or queue a whole lot of very long copy jobs), and walk away (or go to bed!) without the risk that you'll come back after several hours to find the queue stalled because of an error or confirmation prompt.



Turning on unattended mode displays the **Unattended Errors** dialog, which collects and displays all errors resulting from the operation. You can close this window if you like - the **Show error list** link at the bottom of the progress dialog lets you redisplay it at any time, and it will be shown automatically at the end of the operation if any errors did occur.

The options at the top of the dialog under *When file exists* let you configure how Opus is to handle a filename collision (that is, when a file in the destination already exists with the same name as one being copied). Normally the [Replace Dialog](#) would be shown, but this option lets you instruct Opus to handle the situation in several different ways:

- **Replace:** Existing files will be overwritten.
- **Keep newer:** Existing files will be overwritten if the file being copied has a more recent timestamp (is newer).
- **Rename existing:** Existing files will be renamed so that the filename no longer clashes.
- **Skip:** The file will be skipped.
- **Treat as an error:** The file will not be copied, and an error will be logged. Logging an error means you can then retry the copy at a later time, and choose how to handle the collision yourself.
- **Resume:** When copying to or from an FTP site, this will attempt to resume the file copy if possible - if the existing file is smaller than the one being copied, and the remote server supports resume. If resume is not possible, the other selected option is used.
- **Merge folders without asking:** Overrides the **Ask for confirmation before merging existing folders** option on the [File Operations / Copy Options](#) page in Preferences.
- **Skip identical:** Any conflicting files will be automatically skipped if they are identical (same size and date).

The **Retry** button at the bottom of the **Unattended Errors** dialog lets you retry any failed copies. The **Export error list** option will appear if any errors occur and this lets you save the list of failed operations to a file (in either text or CSV format).

You can use the **UNATTENDED** argument of the internal [Copy](#) command to automatically start a copy operation in unattended mode, and the **WHENEXISTS** argument can be used to set the default *When file exists* behaviour. See the description of the [Copy](#) command for more information about these arguments, and the [Customize](#) section for information on how to configure a button or hotkey.

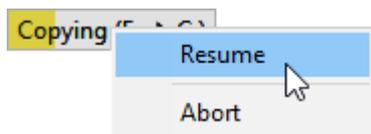
## The Jobs Bar

If you tend to have multiple file operations happening at once you have probably found it hard to manage the individual progress dialogs that Opus creates. The *Jobs Bar* is designed to make it easier to keep track of multiple simultaneous file operations. It is a small bar that appears at the bottom of the Lister, and displays a button for each currently running job.



You can configure the *Jobs Bar* to appear automatically whenever a job is started (with the [Preferences / File Operations / Progress Indicators / Display the jobs bar automatically when starting a new job](#) option), or you can open and close it manually (using the [Set JOBSBAR](#) command). The buttons show basic information about the job (the *action*, *source* and *destination*) and the current progress in the form of a bar graph. The color of the bar graph indicates the current state of the job (*running*, *paused* or *error*).

The *Jobs Bar* buttons also let you manage the jobs directly. Click a button to display its progress dialog - click it again to hide it.



If you right-click the button a context menu is displayed that lets you pause, resume or abort the job.

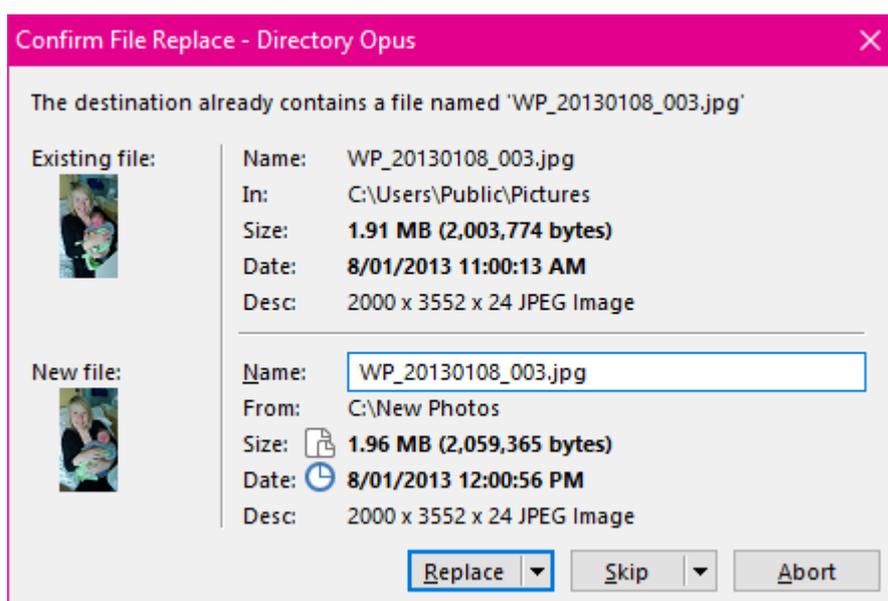
If you're using the *Jobs Bar* you may also like to configure Opus to minimize all progress indicators automatically. This can dramatically streamline your workflow - if you are launching multiple simultaneous file operations it can be incredibly distracting to have progress indicators popping up in front of the Lister every time you launch an operation. This can be enabled by turning on the **Minimize progress indicators** option on the [Preferences / File Operations / Progress Indicators](#) page. You can also set progress indicators to only minimize when the *Jobs Bar* is visible - that way, you can manually open the *Jobs Bar* when you want to start a lot of operations simultaneously and have the normal behavior when the *Jobs Bar* is closed.

Note that when the *Jobs Bar* is visible and a new operation is started with its progress indicator minimized, a subtle animation is used to provide visual feedback of the newly started job.

If the *Jobs Bar* is set to display automatically, you can also choose to prevent the individual progress indicator windows from appearing on the taskbar at all. If you turn on the **Prevent progress indicators from showing on the taskbar** option progress indicators won't appear on the taskbar or the *Alt+Tab* program list - when minimized (either manually, or automatically using the above option) they will instead be completely hidden. You can keep track of their progress and redisplay the progress indicator using the *Jobs Bar* buttons. If no *Jobs Bars* are visible (say because you have closed the Lister) the progress indicators will reappear on the taskbar automatically.

## The Confirm File Replace Dialog

When copying files around, you'll often encounter the case where the file you're copying already exists in the target folder. In this case Opus normally displays the **Confirm File Replace** dialog, asking you what to do.



The goal of this dialog is to show you enough information about the existing file (the one that already exists in the target folder) and the new file (the one you are copying) to help you decide what action to take.

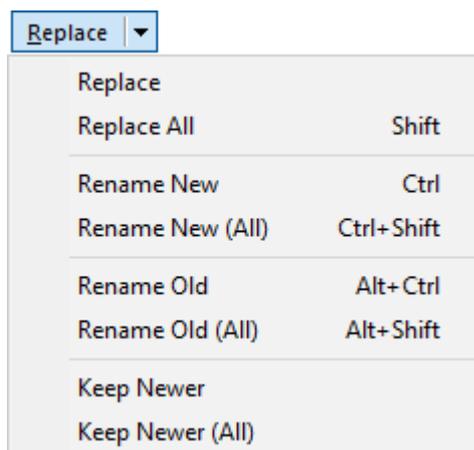
The basic information that's presented for the two files is **Name**, **Location**, **Size**, **Date** (the last modified date) and **Desc** (the file's description). When this information is different between the two files it is displayed in **bold**, making it easier to see at a glance how the two files differ (except for the locations, since they are almost always different and highlighting them would make the more important differences harder to notice). The dialog also uses two little icons to indicate which file is larger (📄) and which is newer (🕒).

If possible, thumbnail images are also displayed for the two files - if not, the file's icon will be displayed. The thumbnails (or icons) have some additional functionality:

- If a thumbnail is displayed for the file, hovering the mouse over the top of it will display a larger preview image.
- You can right-click the thumbnail or icon to display the context menu for the file.
- You can double-click the thumbnail or icon to open the file.

When the **Confirm File Replace** dialog has been displayed, there are several actions you can take. Fundamentally, you need to decide if you still want to copy the file or not.

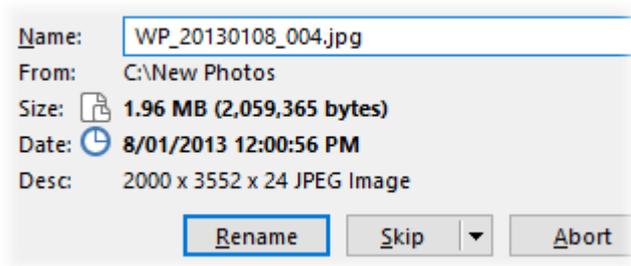
If you **do** still want to copy the file, your available choices are:



- **Replace:** Copy the incoming file over the top of the existing one. You can click the main part of the button to do this. Alternatively, click the arrow on the right of the button to open the menu, then choose the first option.
- **Replace All:** Replace the current file, and automatically replace any subsequent clashing files for the remainder of the operation. You can use the menu to choose this or, as a shortcut, hold the **Shift** key while clicking the **Replace** button.

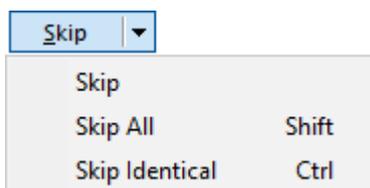
- **Rename New:** Automatically choose an alternative name for the incoming file, so both it and the existing file will exist side-by-side. (To manually specify the new name, see below.) As a shortcut, the menu item can be accessed by holding **Ctrl** while clicking the button.
- **Rename New (All):** Automatically choose an alternative name for the incoming file, and all subsequent clashing files. As a shortcut, hold **Ctrl+Shift**.
- **Rename Old:** Automatically choose an alternative name for the existing file, so both it and the incoming file will exist side-by-side. As a shortcut, hold **Alt+Ctrl**.
- **Rename Old (All):** Automatically choose an alternative name for the existing file, and all subsequent clashing files. As a shortcut, hold **Alt+Shift**.
- **Keep Newer:** Copy the incoming file over the top of the existing file, but only if the incoming file was modified more recently than the existing one. Otherwise, leaving the existing file alone and do not copy the incoming file at all. (This is done by comparing the **modified** timestamps of the two files. In the case of a tie, where both timestamps are identical, the incoming file is skipped.)
- **Keep Newer (All):** For this and all subsequent clashes, copy the incoming file over the top of the existing file if the incoming file was modified more recently, and skip it otherwise.

In addition to the automatic rename options discussed above, you can also manually edit the incoming file's name. The **Replace** button changes to a **Rename** button if the name has been edited, and clicking the button will copy the incoming file with the specified name and leave the existing file as it was.



The new file's **Name** is an edit field and, similar to the [Inline Rename](#) function, this edit field responds to a few control keys that let you quickly select parts of the filename: **Ctrl+A** (select all), **Ctrl+F** or **Ctrl+N** (select file stem) and **Ctrl+E** (select file extension).

If you decide you **do not** want to copy the file after all, the available choices are:



- **Skip:** Skip over this file. The incoming file is not copied and the existing file is left alone. You can click the main part of the button to do this. Alternatively, click the arrow on the right of the button to open the menu, then choose the first option.
- **Skip All:** Skip over this file, and skip all subsequent clashing files for the remainder of the operation. You can use the menu to choose this or, as a shortcut, hold the **Shift** key while clicking the **Skip** button.

- **Skip Identical:** Skip over this file, and any subsequent files, if the incoming and existing files have identical **size** and **modified** timestamp. *Note that the actual file contents are not compared!* You will still be prompted for subsequent clashes where the two files have different sizes or Modified timestamps. You can hold **Ctrl** and click the button as a shortcut.

You can also abort the whole operation by clicking the **Abort** button or closing the dialog.

The context menu displayed when right-clicking a file's thumbnail (or icon) can be customized using the [File Types](#) system. For example, you could add a command to compare the two files using an external comparison tool.

If you are copying a file to or from an [FTP site](#), and the remote FTP server supports *resume*, you will also have the choice to **Resume** the copy (as well as a drop-down for **Resume All**). If you select the **Resume** option Opus will attempt to resume the file transfer where it was (presumably) interrupted previously. For example, if the file you are copying is 100 KB and the existing file is only 50 KB, Opus would start transferring data from the 50 KB mark. This option will only be available if the existing file is smaller than the new one. You need to be sure of what you're doing when you select this option, as it would be very easy to wind up with a corrupted file if you accidentally resume copying the wrong file.

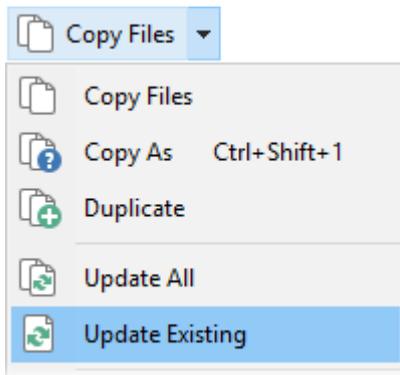
## Copying Updated Files

There are two different ways to copy updated files (that is, copy newer files over the top of older ones) automatically:

- The **Update All** and **Update Existing** commands offer a simple one-way form of file synchronization
- The **Synchronize** tool offers a full, two-way synchronization system, that lets you see and control which files are going to be copied or deleted

See the [Synchronize](#) page for a full description of using the **Synchronize** tool.

The **Update All** and **Update Existing** commands are found in the drop-down menu attached to the **Copy Files** button on the default toolbar.



These two commands are variants of the standard **Copy Files** command, and rely on the [source and destination](#) concept. Selected files will be copied from the current source file display to the destination folder, under the following conditions:

- **Update All:** Files will be copied if they
  - Don't already exist in the destination, **or**
  - Do exist in the destination, but are a different size or older than the files being copied
- **Update Existing:** Files will be copied if they
  - Do exist in the destination, **and**
  - Are a different size or older than the files being copied (this rule can be modified, see below).

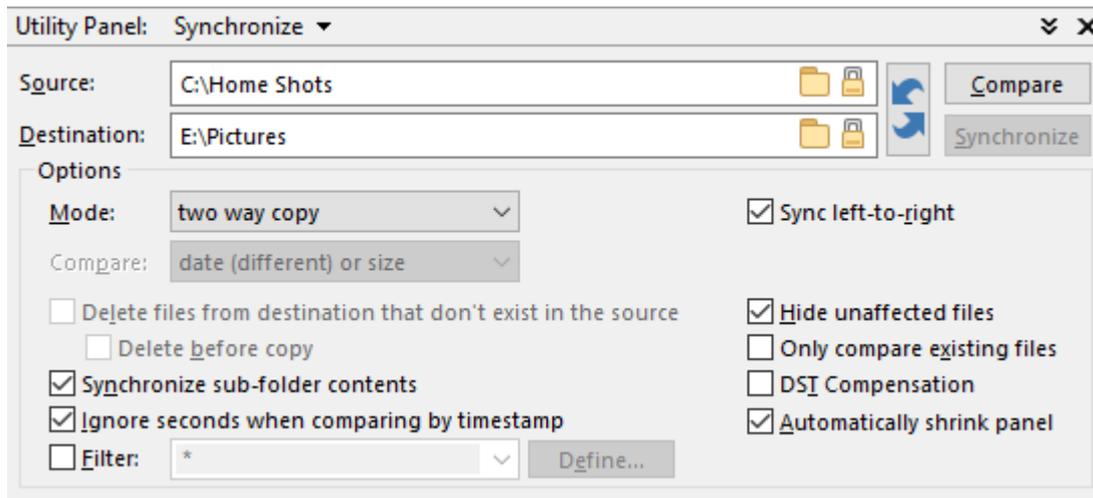
Files that don't match those conditions are skipped. You can modify the test conditions using the arguments for the internal [Copy](#) command:

- Update All: The **UPDATETOLERANCE** argument can be used to control how different a timestamp can be before it is considered as "older".
- Update Existing: As well as **UPDATETOLERANCE**, the **UPDATEEXISTING** argument can be used to ignore file size or date for the comparison

See the description of the internal [Copy](#) command for details on these arguments, and the [Customize](#) section for information on configuring toolbar buttons.

## **Synchronize**

The **Synchronize** tool lets you synchronize the contents of two folders. The folders involved can be any location that Opus can access - local drives, archives, network shares or even on an FTP site. You can specify the criteria used to determine whether a file should be copied or not, and you can choose whether synchronization is one-way or two-way. To access the Synchronize tool, select the **Synchronize** command from the **Tools** menu.



The **Synchronize** tool is displayed in the [Utility Panel](#) at the bottom of the Lister. The fields at the top of the panel control the folders that are being synchronized. When the panel opens the **Source** and **Destination** folders will default to the locations in the current source and destination file displays. You can use the **Browse** (📁) buttons for each field to select a different location, or type the path in manually.

The **Lock** (🔒) buttons for each field let you lock the folder to the current location in the Lister. When the field is locked it will update automatically whenever you navigate to a new location - this lets you use the Lister for navigation without having to update the **Source** and **Destination** fields manually.

The **Swap** (↔) button swaps the values of the **Source** and **Destination** fields around.

The **Options** section of the panel defines the type of comparison and other options affecting the synchronize operation.

- **Mode:** This lets you choose whether the synchronize operation is one-way or two-way.
  - **One-way copy:** Files will be copied from the **Source** to the **Destination** folder only. In this mode you can choose the comparison method used to determine whether a file needs to be updated or not.
  - **Two-way copy:** Files will be copied in both directions. Where the same file exists in both locations, the newer one will be copied over the top of the older one.
- **Compare:** In one-way copy mode, this drop-down lets you choose the comparison method used to determine whether an existing file needs to be updated.
  - **Byte comparison:** The file in the destination folder is compared byte-for-byte against the source file and replaced if it differs. Even though the actual comparison is only done if the two files have the same size, this mode can still be very slow compared to the other options.
  - **Date (different):** The file in the destination folder will be replaced if its last modified timestamp is different to that of the source file.
  - **Date (different) or size:** The file will be replaced if either its last modified timestamp or its size are different to that of the source file.

- **Date (newer):** The file will be replaced if the last modified timestamp of the source file is newer than that of the existing one.
- **Date (newer) or size:** The file will be replaced if the timestamp of the source is newer or the size is different.
- **Size:** The file will be replaced if its size is different to the size of the source file.

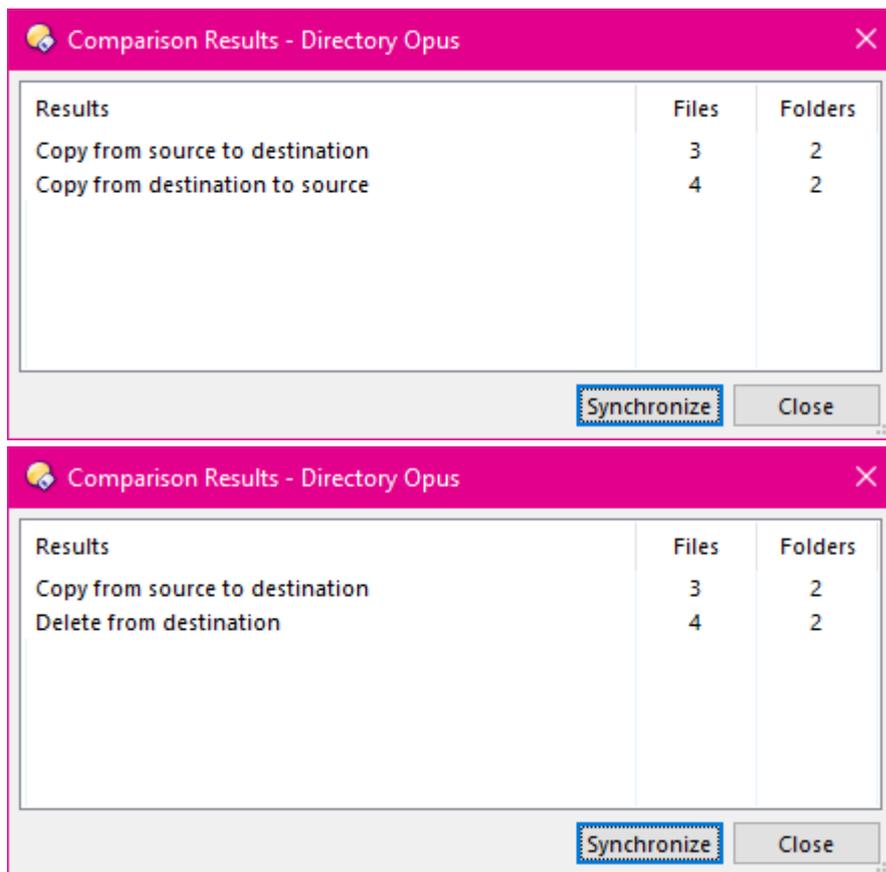
In two-way copy mode you can't choose the comparison method as the only mode that makes sense is **Date (newer)**.

- **Sync left-to-right:** When turned on, the synchronize panel gets its Source path from the left (or top) file display and its Destination path from the right (or bottom) file display, even if the actual source/destination status of the two file displays is reversed. Turn this on to prevent accidentally performing a synch in the wrong direction. When this is off, clicking within the right (or bottom) file display may cause the synchronization to be performed in the opposite direction to what you expect unless you remember to click again on the left (or top) file display first. Of course, the difference matters most when doing a one-way copy.
- **Delete files from destination that don't exist in the source:** In one-way copy mode you can turn this option on to delete files from the destination directory if they don't exist in the source folder. This option is not available in two-way copy mode since in that mode files that exist in one folder but not the other will be copied.
- **Delete before copy:** When the **Delete files from destination** option is enabled, this option makes Opus perform the delete operation before the copy operation, instead of after. You may want to use this option if you are worried that the destination drive won't have enough space.
- **Synchronize sub-folder contents:** Turn this option on if you want to synchronize the contents of sub-folders in the source and destination folder. If you leave this off then only files in the specified folders themselves will be synchronized.
- **Ignore seconds when comparing by timestamp:** When **Mode** is set to two-way copy, or one of the **Date** comparison methods is selected in one-way copy mode, this option causes Opus to ignore the seconds field of timestamps when comparing the dates of two files. For example, the dates **21-Feb-2010 10:35:12** and **21-Feb-2010 10:35:38** would be considered the same with this option turned on. This can be useful when synchronizing between locations that don't store timestamps to the same resolution (e.g. some FTP servers don't store seconds at all, so without this option your files would almost always look like they were out-of-date even if they weren't).
- **Filter:** This option lets you specify a filter to control which files are synchronized. You can either enter a [wildcard pattern](#) directly, select a [pre-configured filter](#) from the drop-down, or click the **Define** button to [define a new filter](#).
- **Hide unaffected files:** If this option is on then any files that don't need to be synchronized (in either direction) will be hidden when the comparison is performed. This can be handy when you have a very large directory tree that might only have a few files that need to be synchronized - it makes it much easier to see what's going to happen. You can also hide unaffected files manually (or show them again) by right-clicking the background of the file display and choosing the **Hide unaffected items** or **Reveal hidden items** context menu commands.
- **Only compare existing files:** Normally the synchronize function will copy files that don't exist as well as those that already exist but are different (based on the comparison method chosen). If this option is turned on then only files that already exist but are different will be copied - files that don't already exist on both sides will be ignored.
- **DST Compensation:** This option enables daylight savings compensation. Some file systems (FAT/FAT32, and some FTP servers) store timestamps as local time rather than as an absolute UTC time. What this means in practical terms is that when your system clock changes forward or back one hour for the beginning or ending of daylight savings (summer time), all your files can suddenly look like their timestamps have changed. If you turn this option on then Opus will ignore timestamp changes of exactly one hour.

- **Automatically shrink panel:** If this option is on then when the comparison begins, the synchronize panel will be automatically shrunk down to a narrow bar at the bottom of the Lister. This results in the maximum amount of available space in the Lister for viewing the results of the comparison. To restore the panel after it has been shrunk, click the  button on the right-hand side of the panel (or click on the header and drag it back up to the desired size).

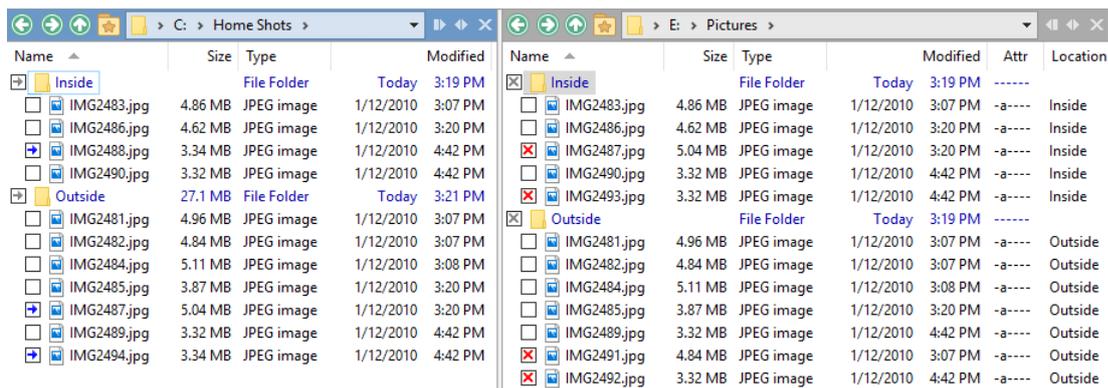
Synchronization is a two-stage process - *comparison* followed by *synchronization*. The comparison stage is when Opus compares the source and destination folders and works out what needs to be copied and/or deleted. After setting your desired options, click the **Compare** button in the top-right corner of the panel to begin the comparison. The procedure Opus follows when you do this is:

1. If the Lister is not currently in dual-display mode, a second file display will be opened.
2. The source and destination folders will be read into the source and destination file displays (if they're not already showing them).
3. Both file displays will be put into [checkbox mode](#) (actually a special variant of checkbox mode that's only used for Synchronize and the [Duplicate File Finder](#)).
4. If the **Synchronize sub-folder contents** option is on, both file displays will be put into [Flat View \(Grouped\)](#) mode, and Opus will read the contents of all sub-folders.
5. The source and destination folders will be compared based on the options chosen. This step can take some time depending on the size of the folders involved.
6. The files and folders selected for synchronization will be indicated as such with a series of glyphs in their checkboxes to indicate what action will be taken.
7. If the **Hide unaffected files** option is on, any files not selected for synchronization will be removed from the display.
8. The results of the comparison will be summarised in a dialog box.



These images show two examples of the comparison results summary dialog. The one on the left was a comparison run on two folders in two-way copy mode - you can see that several files have been selected to be copied in each direction. The image on the right was the result of a one-way copy mode comparison on the same two folders. The same two files were selected to be copied from source to destination, but instead of three files being copied the other way, they are going to be deleted instead. This is because the copy is one-way only and the **Delete files from destination that don't exist in the source** option was enabled.

At this point, you can click the **Synchronize** button to begin the synchronization. If however you want to check the file list first to see exactly what's going to happen, or you want to make changes to the files selected for synchronization or to the comparison options, you can click the **Close** button to return to the Lister. You'll then see the following kind of display in the file list.

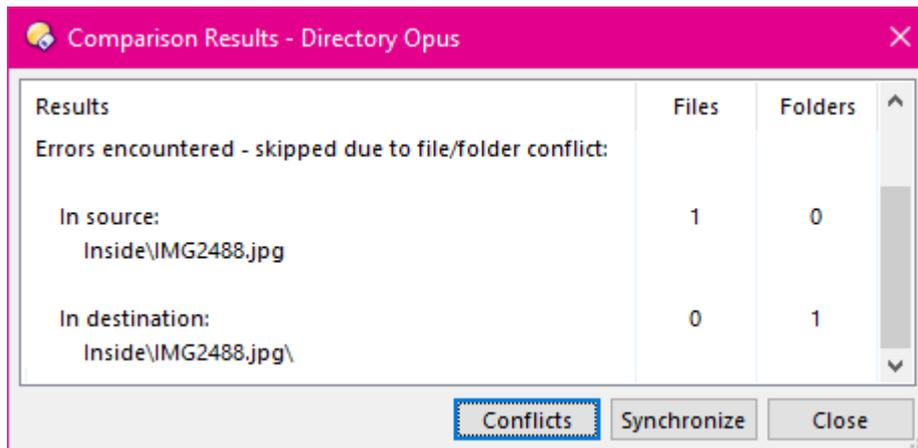


This screenshot corresponds to a one-way copy comparison from **B:\Home Shots** to **C:\Test\Pictures**. As you can see, the files that have been selected for comparison have been marked with a series of glyphs that indicate the recommended action: → (copy to the folder on the right), ← (copy to the folder on the left) and ✖ (delete). For folders, the glyphs in grey (→ ← and ✖) indicate that some but not all of the items within the folder have that state.

From the above screenshot you can see that in the destination, **IMG2488.jpg** and **IMG2494.jpg** did not exist, so these have been selected in the source to be copied. In the source, **IMG2487.jpg**, **IMG2493.jpg** and **IMG2489.jpg** did not exist, and so because the **Delete files from destination that don't exist in the source** option was enabled, these have been selected to be deleted. If we had used a two-way copy, they would have been selected to be copied to the source instead.

You can make changes to the synchronize action for a file or folder by simply clicking on its checkbox - Opus will cycle through the various actions available every time you click. If you are dealing with a large number of files, you may wish to remove from the display those that you have marked to not be synchronized. To do this, right-click the background of the file display and choose the **Hide unaffected items** command from the context menu. Any files that are not marked as either copy or delete will be removed from the display. You can redisplay the hidden files with the **Reveal hidden items** command.

The results summary dialog will also inform you of any errors encountered during the comparison stage. An error can arise if a file on one side of the comparison has the same name as a folder on the other side.



You can see that in this instance, the file **IMG2488.jpg** in the source folder clashed with a folder of the same name in the destination folder. If conflicts arise, the files and folders concerned will be skipped by the synchronization process - but if you want to investigate further and rectify the issue, you can click the **Conflicts** button at this point. This will return you to the Lister and highlight the conflicting items for you.



Once you've verified (and possibly modified) the synchronize actions, you can begin the synchronization by clicking the **Synchronize** button in the top-right corner of the panel. If you change any of the options at this point the **Synchronize** button will be disabled and you will need to re-run the comparison by clicking **Compare** again before you can proceed with the synchronization. Once the synchronization stage begins, it proceeds as more or less an automated operation. The progress dialog will step through the various parts of the synchronization (copy from source to destination, copy from destination to source, delete from destination) as the operation progresses.

As an alternative to the **Synchronize** tool, Opus provides another, simpler method for updating the contents of one folder from another. See the [Copying Updated Files](#) page for more information.

## Filtered Operations

Normally when a command acts on a folder, the entire contents of that folder are processed as well. For example, if you select a folder and click the **Copy File** button, that folder including all of its contents are copied.

File filters let you control which items within folders are processed. You might want to only copy the image files from sub-folders; or you might want to delete any **.tmp** files but leave all others unaffected. You can use a file filter to accomplish this.

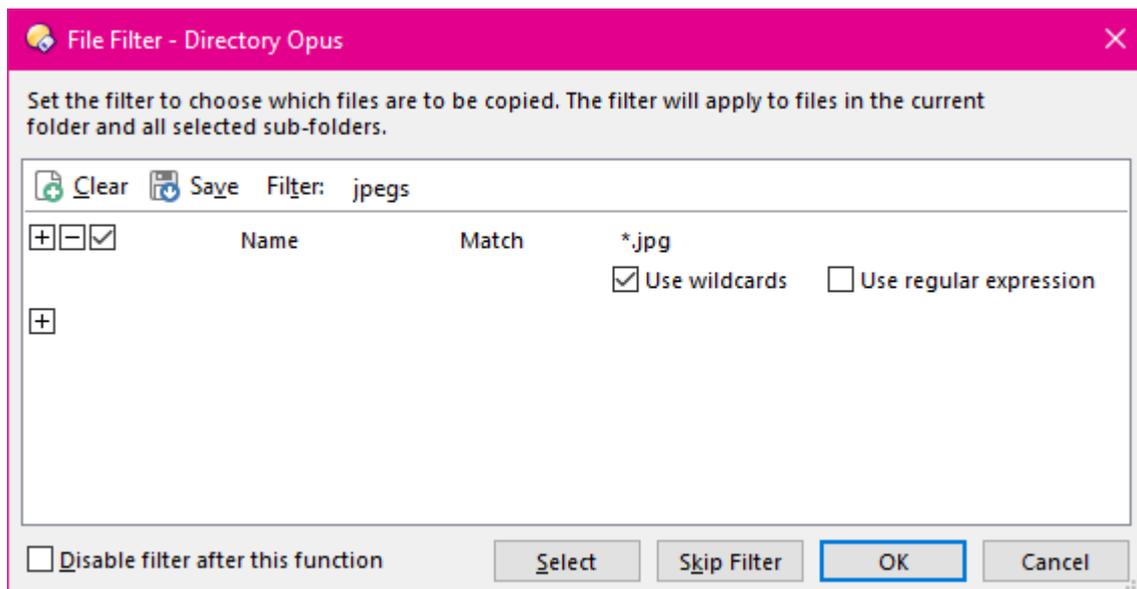
There are a number of functions that support the use of file filters:

- **Copy Files:** When [copying or moving files](#), filtering is enabled by selecting the **Copy Filter** option in the drop-down attached to the **Copy Files** button. The copy filter is local to the Lister, and it remains enabled in that Lister until it is turned off or the Lister is closed.
- **Delete:** When [deleting files](#), filtering is enabled by selecting the **Delete Filter** option in the drop-down attached to the **Delete** button. The delete filter is local to the Lister, and it remains enabled in that Lister until it is turned off or the Lister is closed.
- **Find Files:** The [advanced mode](#) of the [Find Files](#) tool uses a filter to define the search parameters.
- **File Selection:** The [Advanced Selection](#) dialog lets you select, deselect and hide files in the current file display using a filter.
- **Synchronize:** The **Filter** option in the [Synchronize](#) tool lets you use a filter to control how synchronize handles sub-folders.
- **Duplicate File Finder:** The **Filter** option lets you control which files are compared when [searching for duplicate files](#).
- **Attributes:** The [Attributes](#) command in the drop-down attached to the **Properties** button lets you use a filter when applying attributes to the contents of sub-folders.
- **Print Folder:** The [Print / Export Folder Listing](#) command in the **Tools** menu lets you use a filter to control which files are printed to the list.
- **Labels:** You can use the [Labels](#) system to automatically color or highlight files and folders that match complex filters.

When using a filter in a function, there are two ways to define it:

- You can define the filter at the time you actually use it. Functions that let you specify a filter through their user-interface also let you edit it. You can start with a pre-saved one, or build one up from scratch.
- You can select a pre-configured filter from a drop-down list. Filters can be pre-configured and stored using the [Filters](#) Preferences page. This lets you build up a repository of useful filters that can be accessed easily throughout the program.

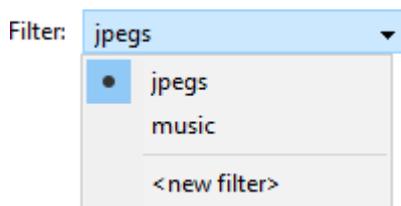
Several of the [internal commands](#) let you specify a filter on the command line (for example, the **FILTER** argument for the [Copy](#) command lets you invoke it with the name of a filter). In this case, the filter must have been pre-configured and stored.



This is an example of a dialog that uses the filter control. This particular dialog is displayed when you run the **Copy File** function and the **Copy Filter** option is turned on. Opus displays the filter dialog at the beginning of the copy process, and you can define a new filter in place, or select a pre-configured one from the **Filter** drop-down.

The **Clear** (🗑️) button in the toolbar at the top of the filter control lets you quickly clear the existing filter. If you hold the **Shift** key down when you click the **Clear** button the filter will be reset to a simple template filter with some common conditions added.

The **Save** (💾) button lets you save the current filter definition as a pre-configured filter; it will then appear in the [Filters](#) page in Preferences.



The **Filter** drop-down button lets you quickly access a pre-configured filter. In the screenshot above, the *jpegs* filter has been chosen. Select the *<new filter>* item from the drop-down to create a new filter (this has the same effect as clicking the **Clear** button).

Below the toolbar in the above screenshot is the area where the filter itself is defined, and below the filter control are some controls that are specific to the **Copy** or **Delete** filters:

- **Disable filter after this function:** This lets you have the filter (copy or delete) disabled automatically at the completion of the current function (otherwise the filter will remain in force until you turn it off manually).

- **Select:** This button takes the current filter definition and uses it to select all matching files and folders in the current file display. This lets you see what the results of the filter would be on the current folder before running the command.
- **Skip Filter:** Skips the use of the filter for the current operation, but the filter will remain turned on (unless you also turned on the **Disable filter after this function** option).

Again, these options only apply to the copy and delete filters. When you use filters in other functions (for example, the [Find](#) or [Synchronize](#) tools), these options will not be displayed.

## Defining a Filter

A filter is constructed from one or more *clauses*: instructions to compare a particular attribute of a file with the provided values. When Opus applies a filter to an operation, every potential file that the operation could affect is compared against the clauses in the filter, and the file is only processed if it matches the filter.

Filters are effectively Boolean operations expressed in human-readable form.

- Individual clauses can be set to *Match* (true) or *No Match* (false), which give the effect of a **not** operator.
- Clauses are linked with the Boolean operators **and** or **or**.
- A filter can also have sub-clauses, which give the effect of parentheses in Boolean operations.

For example, consider the following Boolean expression. The effect of this is to match all **.jpg** files that are larger than 100KB or all **.gif** files that are smaller than 50KB (quite why you'd want to do that is another question):

```
( Name matches *.jpg and size > 100KB ) or
( Name matches *.gif and size < 50KB )
```

This would be represented in a filter control by the following clauses:

	Subclause	Match			
	Name	Match	*.jpg	<input checked="" type="checkbox"/> Use wildcards	<input type="checkbox"/> Use regular expression
And	Size	Match	Is greater than	100	KB
Or	Subclause	Match			
	Name	Match	*.gif	<input checked="" type="checkbox"/> Use wildcards	<input type="checkbox"/> Use regular expression
And	Size	Match	Is less than	50	KB

The two expressions enclosed in parentheses are represented by sub-clauses. Each sub-clause contains two clauses linked by **and** operators; a **Name** match which compares the filename against a wildcard pattern, and a **Size** match which compares the size of the file against a given value. Finally the two sub-clauses are linked by an **or** operator. You can see that in the filter control the contents of sub-clauses are indented from their parent clause.

In the above example, all the clauses are set to *Match* which means the condition in the clause must be **true** for it to match. If a clause is set to *No Match* then it must be **false** for the clause to match. Imagine if we wanted to match all **.jpg** files that are larger than 100KB, but **not** those that are larger than 1MB or have the word "horse" in their name. The human-readable form of this would be:

```
Name matches *.jpg and size > 100KB and
not ( size > 1MB or name contains "horse" )
```

In the filter control you could represent this in the following way:

	Name	Match	*.jpg	<input checked="" type="checkbox"/> Use wildcards	<input type="checkbox"/> Use regular expression
And	Size	Match	Is greater than	100	KB
And	Subclause	No Match			
	Size	Match	Is greater than	1	MB
Or	Name	Match	*horse*	<input checked="" type="checkbox"/> Use wildcards	<input type="checkbox"/> Use regular expression

The sub-clause in the above example is set to *No Match* meaning the filter won't match a file unless sub-clause **fails** to match. The sub-clause contains two child clauses, both of which are set to *Match* and linked by the **or** operator. So for the file in question, if its **Size** is greater than 1 MB or its **Name** contains "horse", one or other (or both) of those clauses will match, which means its parent sub-clause will match, which means (because the sub-clause was set to *No Match*) the filter will fail. Confused yet? :)

## Adding, Removing and Editing Clauses

Each clause in the filter control has a number of controls that are used to edit the filter.



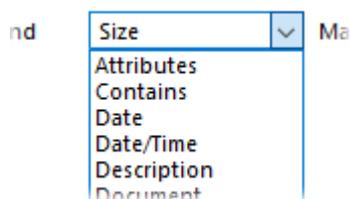
From left to right:

- The  button adds a new clause to the filter *above* the line clicked on (to add a clause to the end of the filter, click the  button displayed at the bottom on a line by itself).
- The  button deletes a clause.
- The  button lets you disable a clause while preserving its information (as opposed to deleting it). When a clause is disabled it has no effect on the filter at all.
- The  dropdown lets you choose the Boolean operator (**and** or **or**) linking the selected clause with the one *above* it. The very first clause in the filter, or the first clause in a sub-clause, will not have this drop-down because there is nothing to link it with.
- The  dropdown defines the [type of the clause](#) - which attribute of the file is actually compared (name, size, date, etc). This is also used to begin a new sub-clause by selecting *Subclause* from the drop-down list.
- The  dropdown lets you choose whether the clause must be **true** (*Match*) or **false** (No Match) for the filter (or parent clause) to match.

You can also edit the filter using the keyboard. Use the **Tab** key to move between controls in the current clause, and the **Up** and **Down Cursor** keys to move between clauses. You can press the **+** or **Insert** keys to add a clause, the **-** or **Delete** keys to delete a clause, and the **Space** bar to turn a clause on or off.

## Filter Clause Types

Each clause has a type which specifies exactly which attribute of a candidate file that clause will match. The clause type is chosen from a drop-down list. A sub-clause is created by selecting *Subclause* from the same drop-down list, but sub-clauses are described on the [Defining a Filter](#) page rather than here.





- **Description:** Compares the file's description (if it has one assigned to it). The search options are the same as for a **Contains** clause.
- **Label:** Matches a file on the basis of its assigned [label](#) (if it has one).

Label                      Match                      Green

The drop-down control displays a list of the [configured labels](#) for you to choose from, or you can type in a wildcard pattern.

- **Name:** Compares the file's name against the specified string.

Name                      Match                      \*.jpg  
 Use wildcards       Use regular expression

The options for a **Name** clause are:

- **Use wildcards:** Lets you specify a search pattern using [standard pattern matching](#).
  - **Use regular expression:** Lets you specify the search pattern using [regular expressions](#).
  - **Case sensitive:** Makes the search case-sensitive (upper and lower-case letters will not be treated as the same).
- **Owner:** Match against the file's owner. [Standard wildcards](#) are enabled for this search; if you don't specify any wildcards then the search will automatically perform a partial match (so "Fred" will match "Frederick" automatically).
  - **Path length:** Lets you search for files or folders based on the total path length (that is, the number of characters in the full file path, including its name). This is useful for identifying files whose paths exceed the normal Windows 260 character path length limit.
  - **Rating:** Match the rating assigned to the file or folder (if any). You can rate your files using the [Metadata Pane](#).
  - **Script column:** Test the file or folder against a [column generated by a script add-in](#).

Details depend on the type of column the script adds, but will generally be similar to one of the other clause types.

Filtering using script columns allows you to expand the capabilities of Opus's filtering in arbitrary ways, based on anything you can express as a script. Some script add-ins provide columns intended more for filtering than display, although there is no technical difference between the two. A common use for this is to filter folders based on the files they contain, by writing a script column which looks inside each folder and populates a column with "Yes" and "No" values (or similar) to signal whether it should match or not.

- **Shell column:** Match the value of any column added by third-party shell extension handlers.

Shell column                      Match                      SVN Revision                      >5000 && <6000

The drop-down gives you a list of available columns, and you can use [standard pattern matching](#) to match against the value of the selected column. For numeric columns, limited numeric comparison is available using the operators != (not equals), == (equals), < (less than), > (greater than), <= (less than or equal), >=

(greater than or equal), **&&** (and) and **||** (or).

- **Size:** Compare the file's size.

**Size**                      **Match**                      **Is greater or equal to**                      **1**                      **MB**

The parameters that affect a **Size** comparison are:

- Comparison type: Select from *equal to*, *greater or equal to*, *greater than*, *less than or equal to* and *less than*.
- Comparison size: Enter the actual size value to compare against.
- Comparison units: Specify the units that the comparison size is specified in. Select from *bytes*, *KB* (kilobytes), *MB* (megabytes) and *GB* (gigabytes). The search engine automatically rounds file sizes when comparing against units other than bytes; for example, a 1158-byte file would match a clause that specified **equal to 1 KB**.

A **Size** clause can also be used to compare the size of folders. Because calculating a folder's size can be time consuming, you must specifically enable this behaviour by also including a **Type** clause that specifies *Folders*. For example, the following two clauses would match all empty folders:

**Type**                      **Match**                      **Folders**  
**And**                      **Size**                      **Match**                      **Is equal to**                      **0**                      **bytes**

- **Target:** If a file is a junction, shortcut or link, the **Target** clause lets you match on what it points to (the path of the target of the shortcut).

**Target**                      **Match**                      **C:\Program Files\\*\\*.exe**  
 **Use wildcards**                       **Use regular expression**

The above example would match shortcuts that pointed to any .exe (application files) below the *C:\Program Files* folder. You can choose from the following options:

- **Use wildcards:** Lets you specify a search pattern using [standard pattern matching](#).
  - **Use regular expression:** Lets you specify the search pattern using [regular expressions](#).
  - **Case sensitive:** Makes the search case-sensitive (upper and lower-case letters will not be treated as the same).
  - **File:** Only match shortcuts that point to files.
  - **Folder:** Only match shortcuts that point to folders.
- **Tags:** Compares against any tags assigned to the file. [Standard wildcards](#) are enabled for this search; if you don't specify any wildcards then the search will automatically perform a partial match (so "account" will match "accounting" automatically). The string you enter is compared against all tags assigned to the file separately, so if you want to match two specific tags, you need to use two separate **Tags** clauses.
  - **Time:** Similar to a **Date** clause, this compares the file's timestamp, but only looks at the time portion and not the date.

**Time**                      **Match**                      **Modified**                      **Within**                      **1**                      **hour**

The parameters that affect a **Time** comparison are:

- Which timestamp: Select from *Modified* (last modified date), *Created* (creation date) and *Accessed* (last access date).

- Comparison type: Select from *After*, *Before*, *Between* and *Within*.
- Comparison time: Enter the actual time to compare against. For *After* and *Before* this is a single time. For *Between* this is two times (making a range the time must fall within), and for *Within* this is a relative time from the current time within hours, minutes or seconds. The time for a *Within* comparison is relative to *now* - that is, if the current time when the filter is used is 15:35 and the clause specified **within 1 hour**, anything dated the current date between 14:35 and 15:35 would match, but a file from yesterday at 15:10 would not.
- **Type:** This lets you match a file by file type.

Type	Match	File Type	Bitmap Image
------	-------	-----------	--------------

The drop-down list lets you pick from a list of all registered file types. So instead of using a **Name Match \*.bmp** clause, you can specify **Type Match Bitmap Image**. The **Type** clause also supports the following special types:

- **Files only:** Matches all files, but not folders
- **Folders only:** Matches all folders, but not files.
- **Junctions/Links/Shortcuts:** Matches shortcuts, folder junctions and hard or soft links. You can use this in conjunction with the **Target** clause to search for, for example, shortcuts pointing to a specific folder or file.

The following clauses only apply to certain types of file:

- **Document:** Lets you filter on various document fields, applicable to files like Office or PDF documents.

Document	Match	Document Created	Before	5/04/2011
----------	-------	------------------	--------	-----------

Use the drop-down (in the above screenshot, **Document Created** is selected) to select from the various document fields to compare against:

- **Any field:** Searches for the specified string in any of the text-based document fields.
- **Author:** The document's author.
- **Category:** The document's category (if it has been assigned one).
- **Comment:** Any comment stored in the document.
- **Company:** The company field from the document.
- **Copyright:** The copyright field.
- **Document Created:** The date the document was created on. You can specify a date and select a *before*, *after* or *on* comparison.
- **Last Edit Time:** The date the document was last edited.
- **Last Saved By:** The person who last saved the document.
- **Last Saved Date:** The date the document was last saved.
- **Pages:** The number of pages in the document. You can specify the number of pages and select a *greater than*, *less than* or *equals* comparison.
- **Subject:** The document's subject.
- **Title:** The document's title.

The document fields that take a string parameter to compare against (which are most of them) support [standard wildcards](#) for the search.

- **Image:** Lets you filter on various image-related fields. All recognized image formats are supported.

Image	Match	Date taken	On	16/03/2011
-------	-------	------------	----	------------

Use the drop-down (in the above screenshot, **Date taken** is selected) to select from the various fields you can compare on:

- **Camera make:** The make (manufacturer) of the camera that took the image.
- **Camera model:** The model of the camera that took the image.
- **Color space:** The color space of the image (*grayscale, RGB, CMYK, YCKK or YUV*).
- **Contrast:** The contrast setting when the image was recorded (*hard, normal or soft*).
- **Copyright:** The copyright information from the image (if any).
- **Creation software:** The software that was used to create the image.
- **Date digitized:** The date the image was digitized. This is sometimes the same as **Date taken** but might be different if the image was originally taken as a non-digital image.
- **Date taken:** The date the image was recorded (a.k.a. shooting time). You can specify a date and select *on, after* or *before* as a comparison.
- **Digital zoom:** The digital zoom ratio used to record the image (if any). You can specify a multiplier and select *equal to, greater than* or *less than* as a comparison.
- **Dimensions:** The image dimensions. This lets you pick from a drop-down of several common image sizes. If you want more control you can use the **Width** and **Height** fields.
- **Exposure bias:** The camera's exposure bias setting when the image was recorded, specified in *stops*.
- **Exposure program:** The exposure program used to record the image (aperture priority, shutter priority, action, landscape, etc.)
- **Exposure time:** The exposure time used to record the image, specified in seconds or fractions of a second. You can enter the time as a whole number, a fraction (e.g. **1/5**) or a decimal (e.g. **1.8**).
- **F-number:** The f-number (aperture setting) used to record the image, specified in *stops*.
- **Flash:** Whether the flash fired or not when the image was taken, and if so what mode was used.
- **Focal length:** The focal length of the lens when the image was recorded, specified in mm. This is the absolute focal length of the lens.
- **Focal length (35mm):** The focal length of the lens as a 35mm equivalent value (e.g. a digital camera with a lens focal length of 6mm may be equivalent to a 28mm focal length lens on a 35mm film camera).
- **Height:** The height of the image, in pixels. You can select *equal to, greater than* or *less than* as a comparison.
- **ISO speed:** The (equivalent) film speed when the image was taken, expressed using the ISO scale.
- **Metering mode:** The exposure metering mode used when the image was taken (average, spot, multi-spot, etc).
- **Resolution (X):** The horizontal (X) resolution of the image, expressed in dots per inch (dpi).
- **Resolution (Y):** The vertical (Y) resolution of the image in dpi.

- **Rotation:** The rotation information stored in the image, usually from the camera's orientation sensor. This reflects the orientation of the camera when the image was recorded. Select from 0, 90, 180 and 270 degrees.
- **Saturation:** The saturation of the image (*normal, high or low*).
- **Scene capture type:** The type of scene capture program used by the camera (standard, night, portrait, etc).
- **Sharpness:** The sharpness of the image (*normal, hard or soft*).
- **Shutter speed:** The speed of the shutter when the image was taken, expressed in seconds or fractions of a second. You can enter the time as a whole number, a fraction (e.g. **1/5**) or a decimal (e.g. **1.8**).
- **Subject distance:** The distance from the focal point to the subject, expressed in metres or fractions of a metre. You can enter the distance as a whole number, a fraction or a decimal. For example, **0.2** would correspond to 20 centimeters.
- **White balance:** The cameras white balance setting.
- **Width:** The width of the image, in pixels. You can select *equal to, greater than or less than* as a comparison.

The image fields that take a string parameter to compare against (for example, **Camera make or model**) support [standard wildcards](#) for the search.

- **Music:** Lets you filter on various music-related fields. All recognized music formats are supported.

<b>Music</b>	<b>Match</b>	<b>Bit rate</b>	<b>Is at least</b>	<b>256</b>
--------------	--------------	-----------------	--------------------	------------

Use the drop-down to select from the various fields you can compare on:

- **Album:** The title of the album.
- **Any field:** Compares the specified text against all text-based music fields.
- **Artists:** The artist or artists of the album (or track).
- **Bit rate:** The bit rate the music is digitized at, specified in bits-per-second (bps).
- **BPM:** The beats-per-minute setting of the track (if any).
- **Codec:** The codec used to encode the music file.
- **Comment:** Any comment assigned to the music file.
- **Disc number:** The number of the disc (for multi-disc sets).
- **Encoded by:** This field is either used for the name of the person who encoded the track, or the name of the software application they used to do it.
- **Genre:** The genre of the music.
- **Protected:** Whether the music is DRM-protected or not (*yes or no*).
- **Release date:** The date the music (normally the actual recording) was released.
- **Sample rate:** The sample rate the music is digitized at, specified in hertz (Hz).
- **Song length:** The duration of the music, specified as *hours:minutes:seconds*.
- **Title:** The title of the track.
- **Track number:** The track number within an album (if any).

The music fields that take a string parameter to compare against (for example, **Album** or **Genre**) support [standard wildcards](#) for the search.

There are three clause types that relate to the location of a file or folder rather than the file or folder itself:

- **Subfolder:** This is a special type of sub-clause that controls which folders a recursive operation will descend into. When Opus encounters a sub-folder that it would normally enter as part of an operation (for example, when copying a folder containing multiply-nested sub-folders), it tests each sub-folder against the **Subfolder** clause. If it fails, the operation skips that sub-folder and all its contents completely.

**Subfolder**                      **No Match**  
**Name**                              **Match**                      **.svn**  
 Use wildcards       Use regular expression

In the above example, a **Copy File** operation using that filter would skip over any directories called **.svn** (and all of their contents) no matter where in the folder tree they were encountered. **Subfolder** clauses have no effect on files matched by the filter; they only control which sub-folders are entered.

*Note: Subfolder clauses do not work with the [Synchronize](#) tool.*

- **Full path:** This lets you match a file or folder on its full pathname (which includes its own filename). For example,

**Full path**                      **Match**                      **C:\Data\Pictures\\*\\*.jpg**  
 Use wildcards       Use regular expression

would match any file ending in **.jpg** that was located in any sub-folder of the *C:\Data\Pictures* directory. Wildcards in Full path clauses are applied to the string as a whole, rather than on individual path components. So the above example, as well as matching **C:\Data\Pictures\Snapshots\IMG1003.JPG** would also match **C:\Data\Pictures\Weddings\Emily\IMG2138.JPG**. When treated as normal strings (instead of paths) they both match the wildcard pattern provided.

- **Location:** This is similar to **Full Path** except it only considers the location (path) of the file, but not the filename. This can be particularly useful when searching [File Collections](#), as it lets you filter for files based on their real location on disk.

The **Subfolder**, **Full Path** and **Location** clauses all let you choose from standard wildcards or regular expressions.

Only the **Subfolder** clause stops Opus from recursing into directories. If a **Subfolder** clause does not match a folder then Opus will skip that folder and all of its children, even if some of the children may have matched the filter. On the other hand, with the **Full path** and **Location** clauses, Opus will still examine a folder's children even if the folder itself does not match the filter. Both of these methods can be useful, depending on what you actually want to do. If your aim is to ignore everything below a certain folder then a **Subfolder** clause is usually best because it avoids inspecting the folder's children unnecessarily. For some examples, see the Opus Forum/FAQ post, [How to filter items by location or sub-folder](#).

The final clause type is **Filter**. This lets you refer to another pre-configured filter (one listed on the [Filters](#) page in Preferences). For example, you could have a standard filter that excludes certain types of files that you never want to process it, and then refer to that in other, more specific filters that you create later on. Referencing another filter saves you from having to duplicate all those filters' parameters, and also any changes you make later to the referenced filter will automatically have effect on any filters that use it.

	<b>Filter</b>	<b>Match</b>	<b>jpegs</b>		
<b>And</b>	<b>Image</b>	<b>Match</b>	<b>Dimensions</b>	<b>Are at least</b>	<b>1280 x 1024</b>

In this example we use the pre-configured **jpegs** filter to first match **.jpg** files, and then use an **Image Dimensions** clause to only match jpegs that are 1280x1024 or larger.

## Deleting Files

To delete a file or folder, you can either:

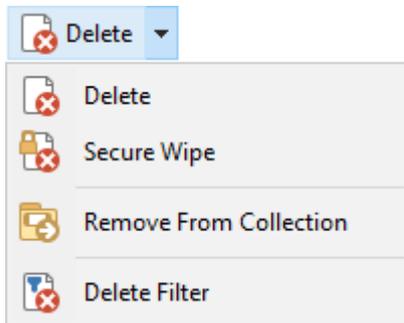
- Select the file and press the **Delete** key on the keyboard
- Right-click the file and choose the **Delete** command from the context menu
- Select the file and click the **Delete** button on the toolbar.

By default, Opus deletes files to the recycle bin when possible. Files that are deleted to the recycle bin can normally be recovered (via the [Undo](#) command), until the recycle bin is emptied. The recycle bin is not supported on network drives or removable media, and files deleted from inside archives or on FTP sites also can not be recovered.

If you hold the **Shift** key down when pressing **Delete** or clicking the toolbar button, you can bypass the recycle bin and permanently delete the file. You can also disable the recycle bin altogether through Preferences.

The [File Operations / Deleting Files](#) page contains several options that let you configure the Delete function in Opus. You can choose whether Opus asks you for confirmation before the delete (you can even have it ask you before each file if you like). As mentioned above, the use of the recycle bin can be configured. You can also choose whether files marked as read-only can be deleted without a special confirmation prompt.

The drop-down menu attached to the **Delete** button on the default toolbar contains several other commands relating to deleting files:



- **Delete:** Delete selected files or folders, to the recycle bin by default
- **Secure Wipe:** [Securely deletes](#) selected files or folders so they can't be recovered
- **Remove From Collection:** When viewing the contents of a [file collection](#), this lets you remove an item from the collection without actually deleting it. If you use the regular Delete command in a file collection, the real file will be deleted.
- **Delete Filter:** Activates the [recursive filter](#) for the Delete function, which lets you selectively delete the contents of selected sub-folders.

## Secure Delete

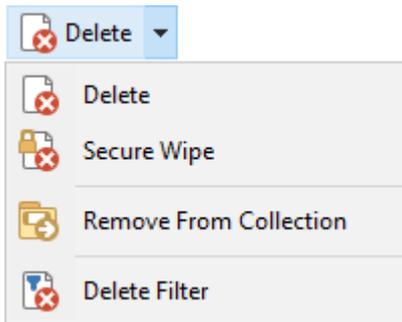
Normally when files are deleted from a computer hard drive, the data that those files contained is not actually removed. Instead, the file's directory entry is removed from the file allocation table and the space it occupies on disk is marked as available for use. So this has the effect of deleting the file - it's not displayed in the directory any more, and the space it took up is available for re-use, but until that space actually is re-used, the file data is still there and is relatively easy to recover.

(Note that this is unrelated to the use of the recycle bin for file deletes - when a file is deleted to the recycle bin, it isn't even removed from the file allocation table).

For confidential or sensitive documents, the secure delete function can be used to overwrite the file data before the file is deleted. The file data is overwritten using algorithms similar to those recommended by the US NSA and other security agencies. You can select the number of overwrite passes (from one to 31), although three passes is required as generally quite secure. Deleting a file in this way is much slower than normal, as every byte of the file must be overwritten one or more times, but it makes the original file data practically impossible to recover.

There are two ways to use secure delete:

- On a per-file basis, using the **Secure Wipe** command in the drop-down attached to the Delete button on the toolbar.



When you run this command, selected files and folders will be overwritten the number of times specified on the [File Operations / Deleting Files](#) page in Preferences.

- If you always want to use secure delete, you can turn on the **Use Secure Wipe** option on the [Deleting Files](#) page in Preferences. With this option on, Opus will use secure delete whenever it's not deleting files to the recycle bin - so if you for example, press **Shift+Delete** to delete a file bypassing the recycle bin, the file will be securely deleted. You can even turn off the recycle bin altogether if desired.

You should use this function with some caution, as files deleted in this manner can not be recovered!

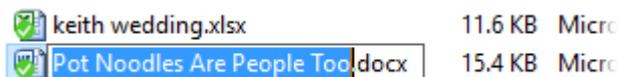
# Renaming Files

There are three main ways to rename files in Opus:

- [Inline Rename](#) refers to editing the names of files directly in the Lister (inline). This is the standard method Explorer uses and may be the one you're most familiar with. You can enter inline rename mode by slow double-clicking a filename (left-click, wait *longer* than the system double-click speed, then left-click again), by selecting a file and pressing F2, or by right-clicking a file and choosing Rename from the context menu.
- [Simple Wildcard Rename](#) is a simple way to perform batch renaming. It lets you use a wildcard character to specify one or more parts of the existing filename that are to be retained in the new name. This mode isn't accessible by default, because the Advanced Rename function supports everything this can do and much more.
- [Advanced Rename](#) is the most powerful way of performing batch renames in Opus. Using this tool you can use complicated wildcard expressions, rename using file metadata, create rename macros, automatically number files, write complex scripts and much more.

## Inline Rename

*Inline Rename* is the method of renaming files you're probably most familiar with, as it is the method that Explorer uses. You edit the name of the file directly in the Lister, and pressing the **Enter** key applies the change immediately. It's straightforward, but only lets you rename one file at a time.



To enter *Inline Rename* mode in Opus is basically the same as in Explorer - select the file and press **F2**, click it twice (with a long delay between clicks), or right-click and choose Rename from the context menu. However you get there, simply edit the filename as desired and press **Enter** to accept the change.

Opus provides some additional functionality in this mode that may not be immediately apparent:

1. Initially only the stem of the filename is selected, not the entire name. Actually Explorer behaves like this too these days, but in XP it still selects the whole filename by default. Normally you don't want to change a file's extension, so auto-selecting only the stem saves you having to deselect the extension manually. You can change what's initially selected with the [Default selection mode](#) option in Preferences.
2. You can press **F2** to cycle between selecting the stem, selecting the extension, and selecting the whole name. You can also select these parts of the name by pressing **Ctrl+A** (select all), **Ctrl+N** or **Ctrl+F** (select stem) and **Ctrl+E** (select extension).
3. You can change the case of the filename by pressing **Ctrl+L** (lower-case all), **Ctrl+U** (upper-case all), **Ctrl+W** (capitalize all words) or **Ctrl+P** (capitalize first word).
4. You can move to the next or previous file in the list by pressing **Cursor Down** (or **Tab**) or **Cursor Up** (or **Shift+Tab**). Any changes you have made to the current file will be applied and the rename field will be moved to the next or previous file automatically. If you use this you can turn on the [Retain cursor position when moving to next/previous file](#) option in Preferences to have the cursor position retained from one file to the next.
5. You can access a history of previously used filenames by holding either the **Ctrl** or **Shift** keys and pressing **Cursor Up / Down**.

6. You can copy the name from the previous file by pressing **Ctrl+Shift+Up** (or **Ctrl+'**), or from the next file by pressing **Ctrl+Shift+Down** (hold the **Alt** key down as well to also copy the extension).

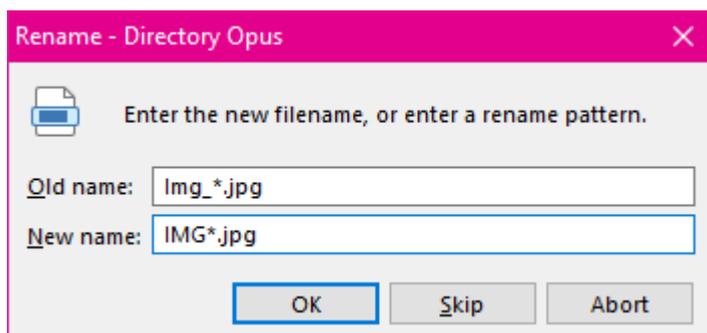
To exit *Inline Rename* mode without changing the name of the file, press the **Escape** key.

As well as choosing the initial selection in Preferences (as described above), you can also control it using the [Rename INLINE](#) raw command. For example, this would let you configure one [hotkey](#) to enter inline rename with the filename stem selected, and another hotkey with the file extension selected instead.

## Simple Wildcard Rename

The *Simple Rename* dialog lets you batch rename files using a simple wildcard system. The **\*** character (asterisk) is used to specify one or more parts of the existing filename that are to be retained in the new name. It is basically equivalent to the [Standard Rename](#) mode in the [Advanced Rename](#) dialog. This wildcard system is also used in the **Copy As** and **Move As** functions - see the documentation on [Using Wildcards when Copying](#) for more examples.

The *Simple Rename* function isn't exposed in the default toolbars - instead, we recommend the use of the *Advanced Rename* dialog, as it can do everything the simple one can do plus much more. If you want to use *Simple Rename*, however, you'll need to [configure a button or hotkey](#) to access it. The command you need is **Rename SIMPLE**.



In this example, we want to rename any file that begins with **Img\_** and ends with **.jpg**. The match is not case sensitive, so this will match **Img\_2481.jpg**, **img\_2481.jpg**, etc. The **\*** indicates that any text between the prefix and suffix is to be preserved, and the **\*** in the **New name** field indicates where that text would be placed. Given the following input filenames, the resultant new names would be:

Img_2481.jpg	IMG2481.jpg
IMG_2483.JPG	IMG2483.jpg
Img_2486.jpg	IMG2486.jpg

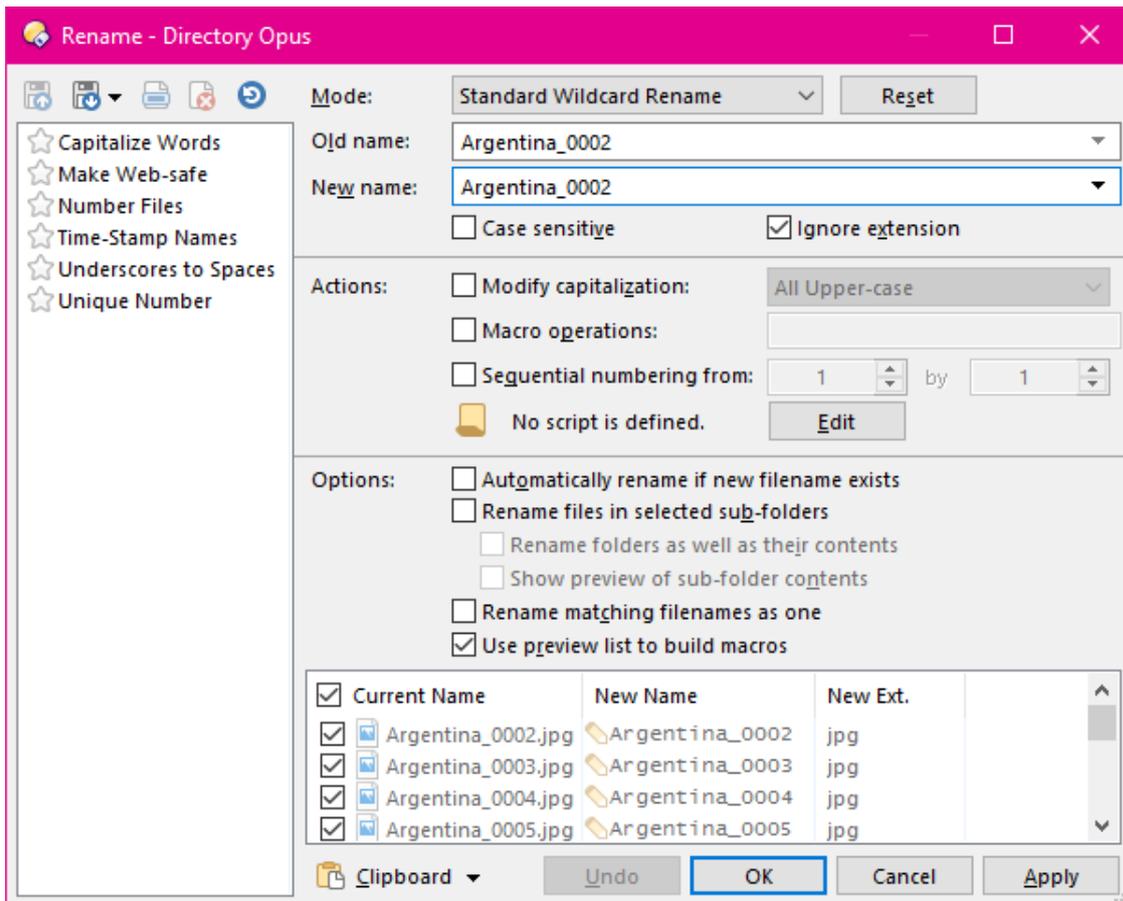
The rename operation will operate on all files and folders that were selected when the command was invoked, however files within sub-folders will not be renamed (you need to use *Advanced Rename* for that).

If you don't enter a wildcard pattern, but instead just supply a literal new name, the function will rename each selected file in turn - the first file will be renamed, and the dialog will re-open showing the name for the second selected file, and so on.

## Advanced Rename

The Advanced Rename dialog lets you perform complicated batch renaming operations. You can use wildcards or regular expressions, find and replace, create macros, modify capitalization, number or renumber files, rename using file metadata and even write complicated rename scripts using this dialog.

To access the Advanced Rename function, select the files you wish to rename and click the **Rename** button on the toolbar.



The Advanced Rename dialog has five distinct sections:

1. The left panel is the [presets list](#), where you can load and manage your rename presets.
2. The top section controls the [rename mode](#) (described below) and lets you edit the **old name** and **new name** strings. These are used, among other things, to provide wildcard patterns or find and replace strings.
3. The [actions](#) section controls the transformations that will be applied to each filename. The script **edit** button is used to expand the dialog to reveal the rename script editor.
4. The [options](#) section contains several checkboxes that control the rename operation.
5. Finally, the **preview** section displays a preview of the transformed filenames. Each file listed in the preview list (except for those from sub-folders) has a checkbox that you can use to remove the file from the rename operation.

When the **Use preview list to build macros** option is turned on, the preview list also doubles as the [macro builder](#), a powerful mass-editing feature for making the same changes to every filename at once. When that option is turned off, you can edit the new filenames individually. Editing individual names is great for making small changes which the macro (or other batch operations) may not have got exactly right. Once the name for a file has been edited directly it

will be displayed in red and the name is locked in until the rename operation occurs (or until you clear the custom name).

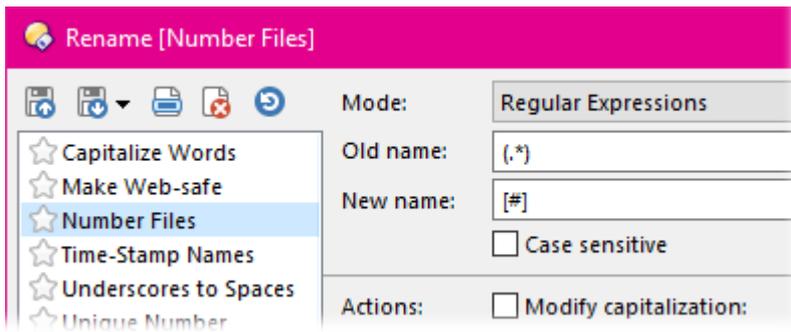
The Clipboard button (  **Clipboard** ▾ ) contains commands that let you copy the full list of original filenames to the clipboard, so that you can paste them over another set of files, or edit them in an external text editor or spreadsheet. With filenames in the clipboard (one per line), use the **Paste new names from clipboard** command to paste them into the *Rename* dialog. The menu also contains **Prefix**, **Append** and **Reset** commands. **Prefix** and **Append** let you add the clipboard content to the start or end of the existing names instead of replacing the names. If one line is in the clipboard, it will be added to every name. If multiple lines are in the clipboard, one will be added to each name, and blank lines can be used to skip names. (If the number of files is larger than the number of lines, the clipboard content will loop around.) Finally, the **Reset** option clears any new names set via the same menu or via manually typing over individual names.

The macro builder cannot be used at the same time as the Clipboard **Paste**, **Prefix** and **Append** commands; it will be turned off if you use them. If you need to apply macros to names from the clipboard, paste the new names, then click **Apply**, then use the macro builder. (Note that the macro builder can also use the clipboard, copy and paste in its own way. See the [macro builder](#) page for details.)

Once you've configured your Rename parameters, you can click **OK** to close the dialog and perform the rename. Alternatively, the **Apply** button allows changes to be applied to the selected files while leaving the Rename dialog open. Once you use the **Apply** button the **Undo** button next to it becomes available, allowing you to undo the rename you just applied.

## ***Rename Presets***

The **presets** section of the Rename dialog lets you store and manage rename presets: predefined rename operations that can be invoked repeatedly without having to configure the parameters of the rename each time.

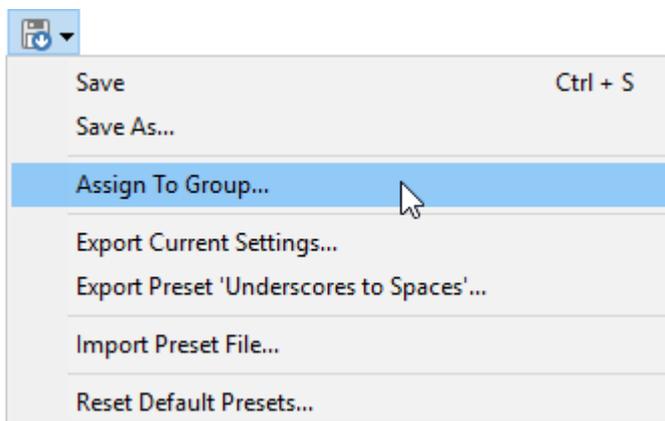


In the above image, *Number Files* is a rename preset that comes supplied with Directory Opus as an example. To load a preset, simply select it from the list and click the **Load** button (  ), or double-click its name in the list. The Rename Dialog will be updated immediately with the settings from the preset. The title bar of the Rename dialog will display the name of the currently loaded preset. Opus will also notice if you've made changes to a loaded preset and ask if you want to save the changes before closing the Rename dialog.

The buttons at the top of the presets section are:

-  **Load**: Select a preset and click Load to load its settings.
-  **Preset Management**: Displays a drop-down menu letting you manage your presets.
-  **Rename**: Rename the selected preset.
-  **Delete**: Delete the selected preset.
-  **Last Rename**: Load the settings for the last-performed rename operation.

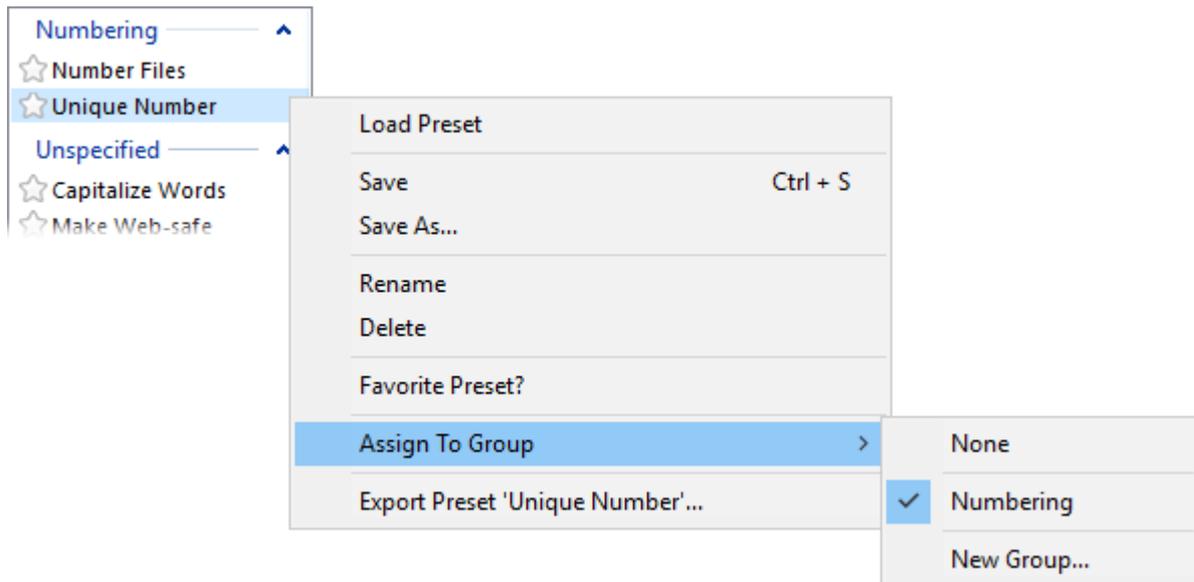
The drop-down **Preset Management** menu provides various commands for managing your presets (you can also right-click a preset to display its context menu).



Use the **Save** or **Save As** commands to save the current Rename Dialog settings as a new preset. The two export commands let you export a preset file that you can share with others - **Export**

**Current Settings** exports the current configuration of the Rename dialog, and if you have an existing preset selected in the preset list, the second command lets you export that preset. **Import Preset File** lets you import a preset that someone else has shared with you.

The **Assign To Group** command lets you put your presets into groups, which can be very useful if you've got a lot of presets to keep track of.



To move a preset to a group, select the **Assign To Group** command either from the **Preset Management** drop-down, or by right-clicking the preset. If any groups have already been defined they'll be shown in the menu as seen above - you can either pick one or those, or create a new group (Opus will prompt you for its name). Any presets that aren't assigned to a specific group are displayed in the *Unspecified* group.

Each preset in the list has a star icon displayed next to its name, which you can click to designate that preset as a favorite. Favorite presets are displayed at the top of the preset list - if the list is grouped, a separate *Favorites* group will be created automatically.

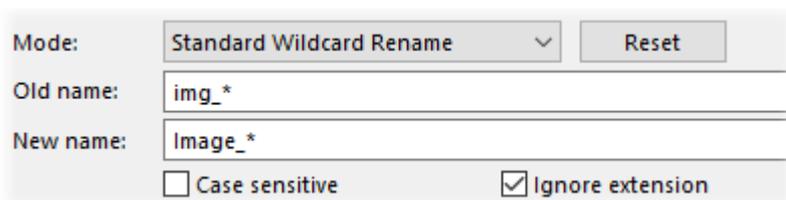
You can rename a preset using the **Rename** button (  ) or by right-clicking it and selecting **Rename** from the context menu. You can also rename groups using their context menu.

You can delete a preset using the **Delete** button (  ) or from its context menu. You can remove groups using their context menu although if you delete a group, the presets within it aren't deleted - they're simply removed from that group (which is why the command to do this is called *Ungroup* rather than *Delete*).

The **Last Rename** button (  ) loads the settings for the last rename that you performed using the Rename dialog.

## Rename Modes

The top section of the Rename dialog is where you define the rename **mode** - which determines the way the two input fields, **old name** and **new name** are interpreted.



Mode: Standard Wildcard Rename

Old name: img\_\*

New name: Image\_\*

Case sensitive  Ignore extension

Use the **mode** drop-down to select the rename mode.

- [Standard Rename](#): enter a new name literally or rename using a simple wildcard system.
- [Find And Replace](#): specify a text string to search for and another to replace it with (this works just like search and replace in a text editor).
- [Regular Expressions](#): enter a new name literally or rename using a regular expressions search and replace.
- [Regular Expressions + Find and Replace](#): enter a regular expression to search for and the text to replace it with (combines the above two modes).

The **reset** button provides a quick way to clear out the other fields of the Rename dialog and revert to a "blank slate".

The **old name** field specifies the input pattern that's applied to each old (original) filename. When renaming using wildcards or *Regular Expressions*, this field is used to specify the "from" or "search" pattern. When using *Find And Replace* mode this field specifies the find string.

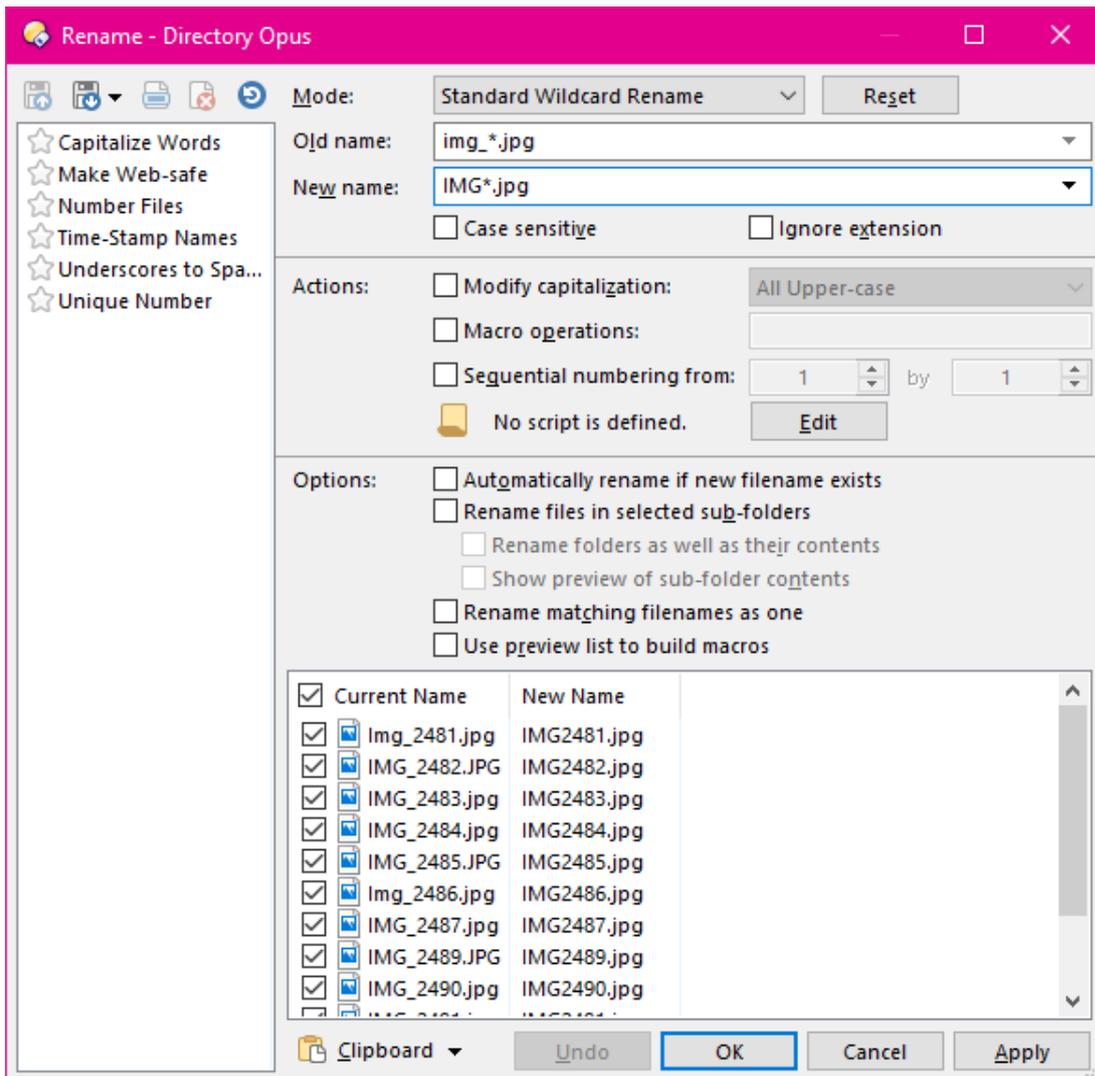
The **new name** field specifies the output pattern that's applied to generate each new filename. When using *Find And Replace* mode this field specifies the replacement string.

The **case sensitive** option is used to specify case sensitive pattern matching or *Find And Replace* searching. When it's turned on, the upper/lower case of an existing filename has to match the supplied pattern exactly - for example, "d\*" won't match "Dog" if the **case sensitive** option is turned on.

The **ignore extension** option is used to totally exclude the file extension from the rename operation - if it's turned on, the file extension won't be modified, and you don't have to (and shouldn't) account for it in any wildcard patterns. You should normally leave this option enabled since it makes wildcards (particularly regular expressions) a lot simpler, and it's rare to need to modify filename extensions.

## Standard Rename

The *Standard Rename* mode lets you batch rename files using a simple wildcard system. The \* character (asterisk) is used to specify one or more parts of the existing filename that are to be retained in the new name.



In the above example, you can see that the original filenames had a mix of *Img\_* and *IMG\_* prefixes, and the file extensions were also a mix of upper and lower-case.

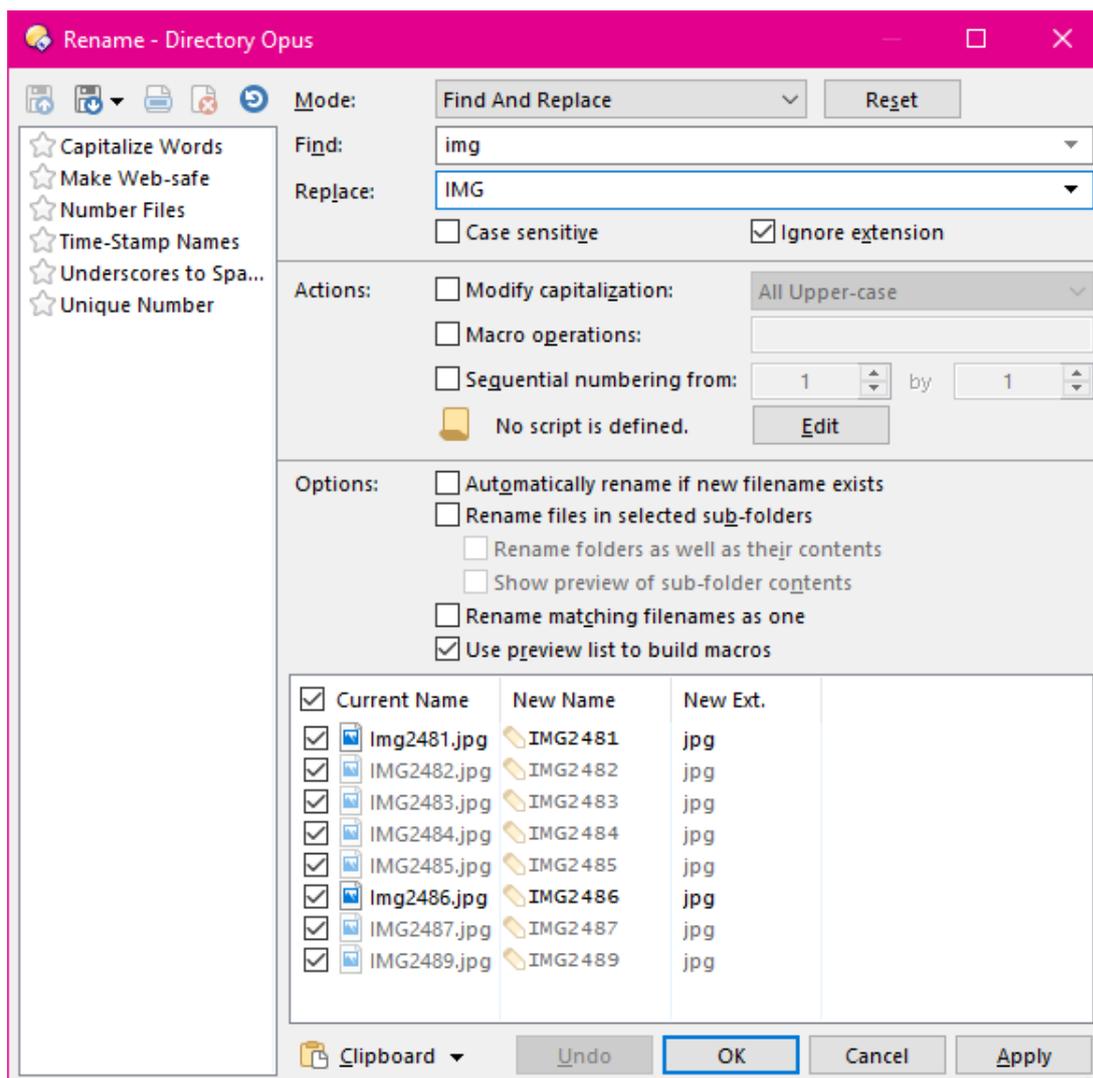
The **Old name** pattern of *img\_\*.jpg* tells Opus to match any filename beginning with *img\_* and ending with *.jpg* (case does not matter as the **Case sensitive** option was not enabled). The **\*** in the pattern is used to mark the portion of the filename that will be carried through to the new name. The **Ignore extension** option has been turned off so that the wildcard will operate on the full filename including the file extension.

You can use as many instances of the **\*** character in the patterns as you like, as long as there are at least as many in the **Old name** field as there are in the new. For example, if you wanted to preserve the case of the file extension in the above example, you could specify an **Old name** pattern of *img\_\*.\** and *IMG\*.\** for the **New name** pattern.

You can't use this system if you want to change the order of preserved strings (as one asterisk looks very much like another) - to do that, you need to use [Regular Expressions](#) mode, which lets you refer to preserved strings by number.

## Find and Replace

Similar to the way the Find and Replace function in a text editor works, *Find and Replace* mode lets you perform a batch rename by specifying a text string to search for and another to replace it with.

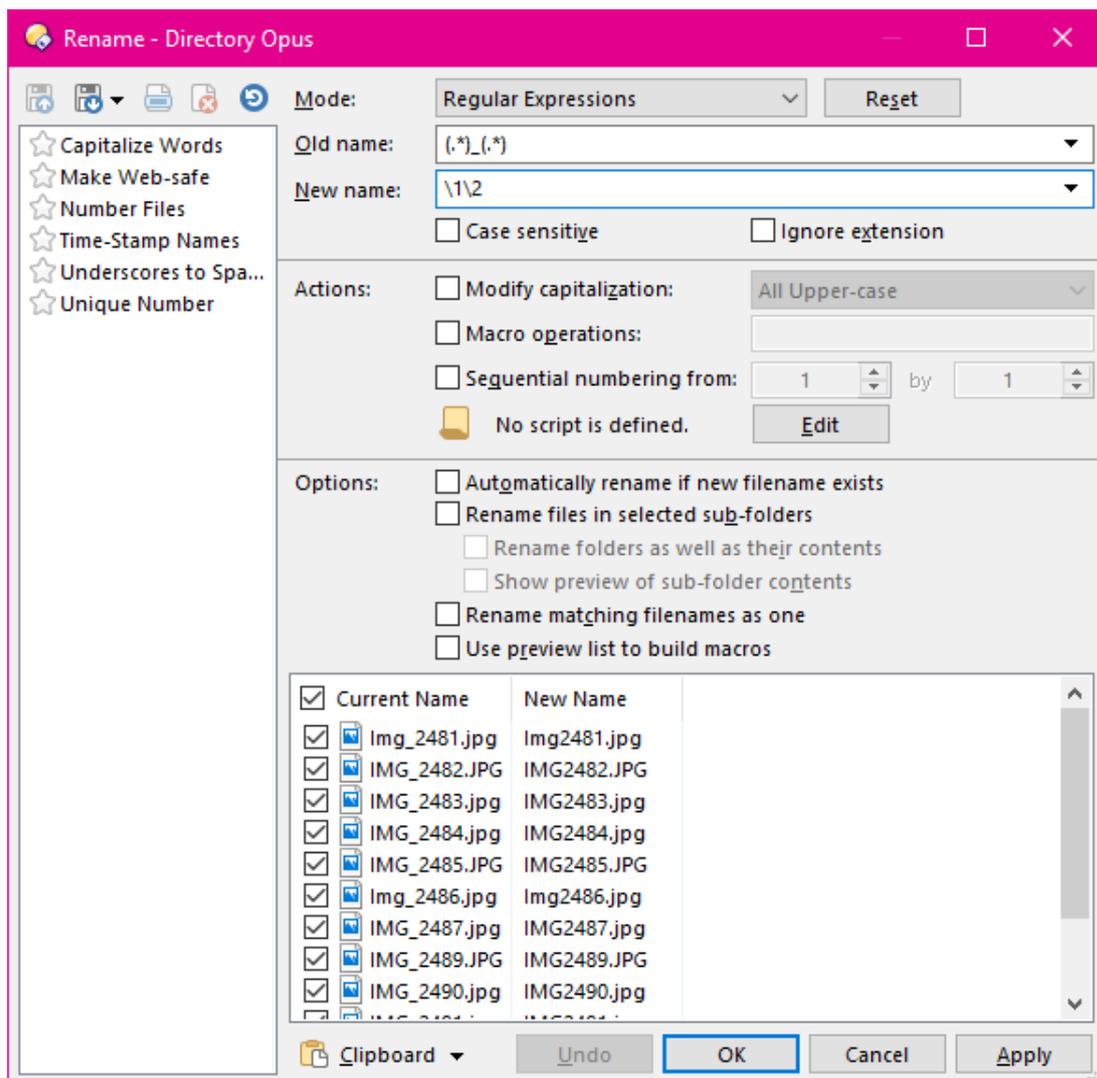


This example has the **Find** string (note the **Old name** field has been replaced with **Find**) set to *img*, and the **Replace** string is set to *IMG*. The preview list indicates the outcome of the batch rename - as you can see, only two of the displayed files will actually change as a result of the rename (they are the two displayed in black - the rest, in grey, won't change).

Ordinarily the search is confined to the filename stem, excluding the file extension. If the **Ignore extension** option is turned off, however, the *Find and Replace* operation will be performed over the whole filename including the extension.

## Regular Expressions

The *Regular Expressions* mode lets you perform batch renaming using regular expressions to specify search and replace patterns.



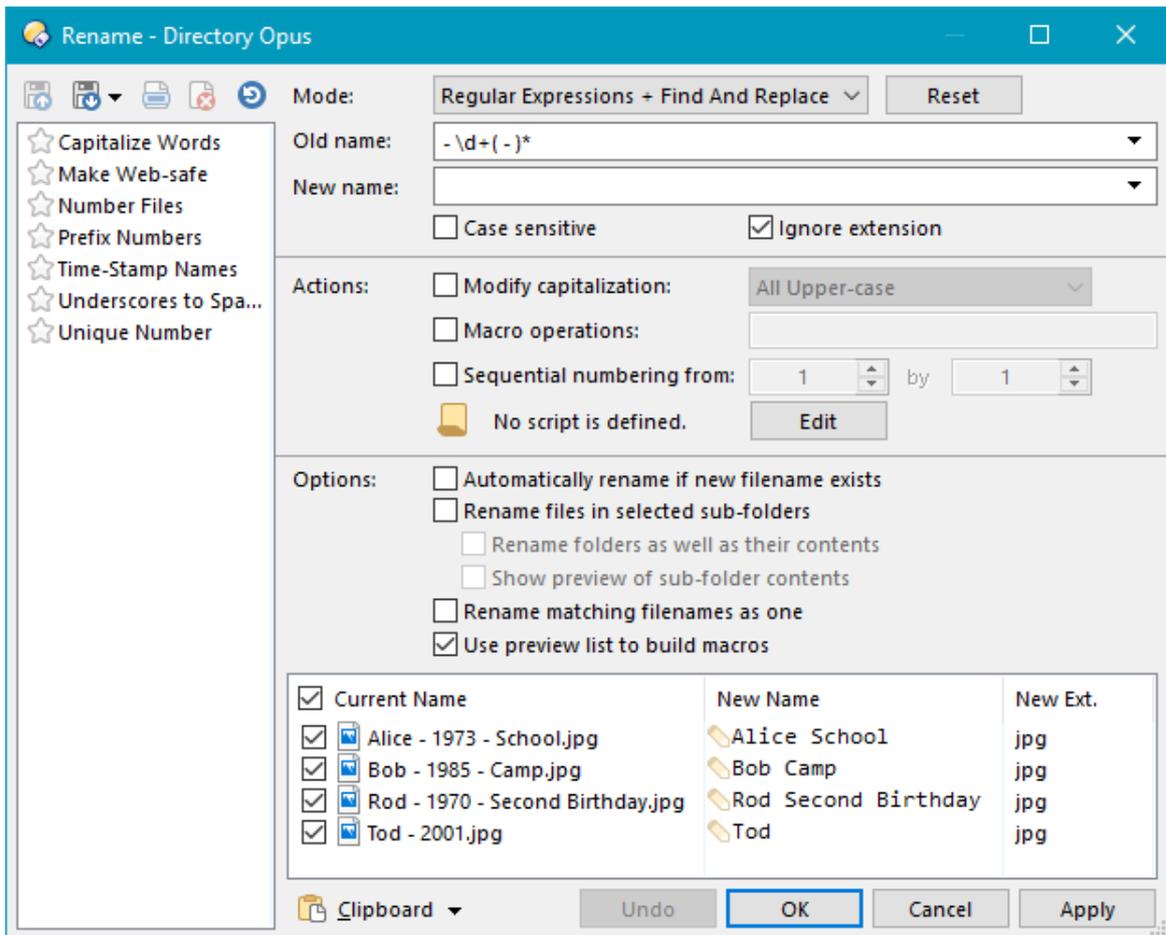
The example above shows a very simple regular expression that has been used to remove the underscore from the original filenames. The **Old name** pattern of `(.*)_*(.*)` uses parentheses to tag two "match anything" expressions, and the `\\1\\2` in the **New name** pattern inserts the values of the two tagged expressions in the new name.

An extension Directory Opus provides is the ability to perform a repeating search and replace using regular expressions. The above example will only remove a single underscore, but if you wanted to remove all underscores from the source name, you can append a # (pound/hash) sign to the **Old name** pattern. This causes Opus to repeat the search and replace until the new filename no longer changes.

See the [Regular Expression Syntax](#) reference page for more information about using regular expressions. Additional information can be found in [Microsoft's introduction to TR1 ECMAScript](#) (the flavor of regular expressions that Opus uses by default) for more information, and you can ask on the [Opus Resource Centre](#) if you need help.

## Regular Expressions + Find and Replace

The *Regular Expressions + Find and Replace* mode lets you combine the power of the *Regular Expressions* mode with the simplicity of *Find and Replace*. Rather than having to construct a regular expression that matches the whole filename, and use \1 style tags to preserve the parts of the old name that you want to keep, you can specify a regular expression to match a sub-string or part of the filename. Opus will replace any matching sub-strings with the new text you provide, and leave untouched any parts of the filename that didn't match.



The example above shows a very simple regular expression that has been used to remove the years from the original filenames. The **Old name** pattern of `- \d+( -)*` matches the initial hyphen and space following it, then any number of digits, then optionally a subsequent hyphen surrounded by spaces. The replace string is left empty since we just want to remove the year - you could, of course, replace it with something else if you wanted.

See the [Regular Expression Syntax](#) reference page for more information about using regular expressions. Additional information can be found in [Microsoft's introduction to TR1 ECMAScript](#) (the flavor of regular expressions that Opus uses by default) for more information, and you can ask on the [Opus Resource Centre](#) if you need help.

## Rename Actions

The **actions** section of the Rename dialog is where you can enable other rename actions besides the primary pattern-based operation (which is controlled by the [mode](#) section of the dialog).

Actions:

- Modify capitalization:** All Upper-case
- Macro operations:** Edit the filenames below to create a macro
- Sequential numbering from:** 1 by 1
-  **Script defined.** Hide

The options that you enable here will affect your rename operation no matter which of the primary modes you've chosen. The transformations are applied after your pattern-based operation (you can think of the rename procedure as happening from top-to-bottom down the Rename dialog).

**Modify capitalization** lets you automatically modify the capitalization of filenames - the options (with built-in examples) are: *all lower-case*, *ALL UPPER-CASE*, *Capitalize All Words*, *Capitalize first word*, *lower-case extens.ion*, *upper-case extens.ION*.

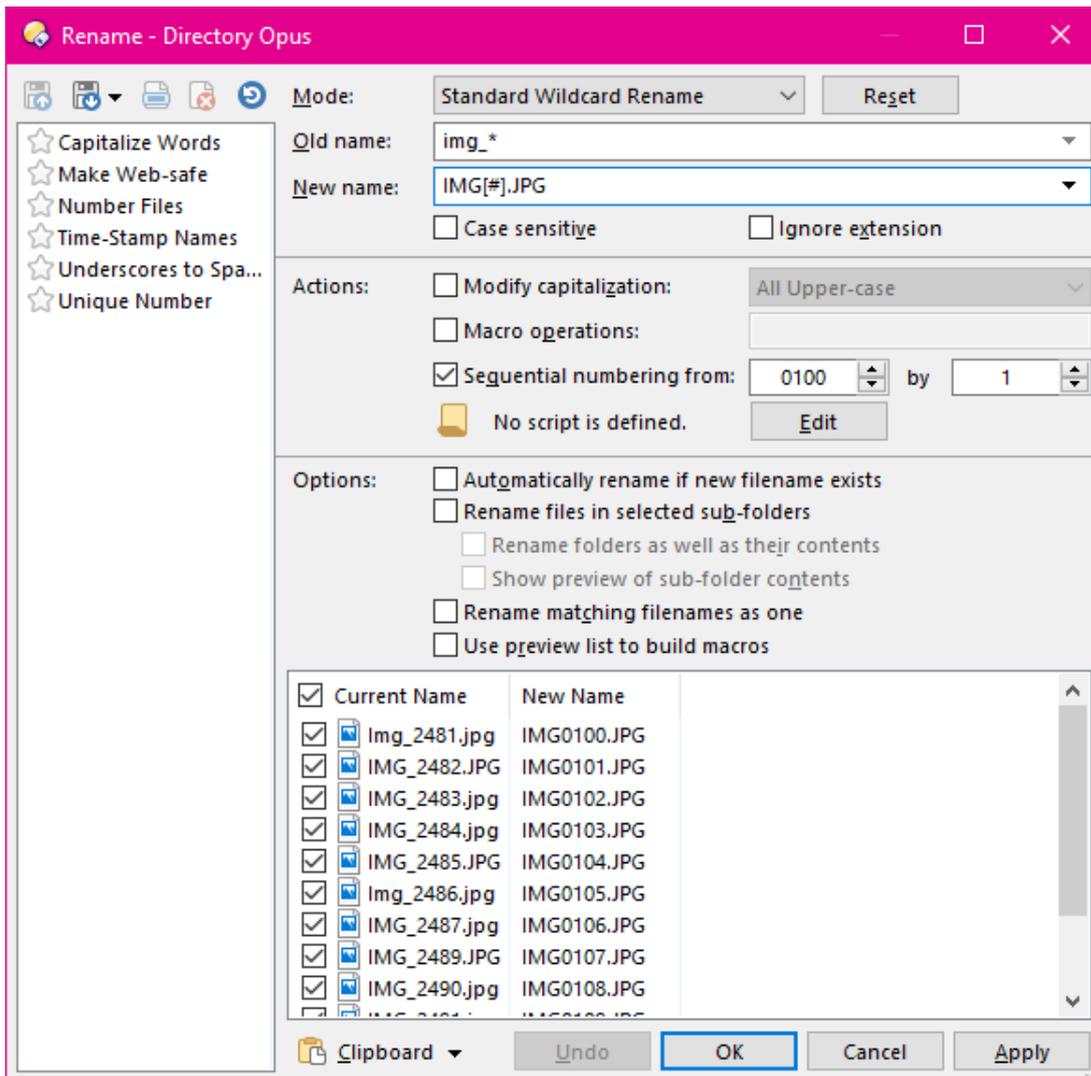
**Macro operations** lets you define a macro string that can add, remove, copy and paste characters from defined positions in each filename. The macro language is a bit unwieldy to write macros by hand, which is why there's a built-in [macro recorder](#) to do the difficult work for you.

**Sequential numbering** lets you add an incrementing number to existing files (or replace the filename entirely with a number). See [Numbering Files](#) for more details.

The **Script** option indicates if a rename script has been defined and lets you open the built-in [Rename Script](#) editor, which lets you write scripts in VBScript, JScript, etc. to completely control the rename process.

## Numbering Files

Using the **Sequential numbering** option in the Advanced Rename dialog, Opus can rename files with (or add to the existing name) an automatically incrementing number. You can specify the starting number, the number of digits (for zero-padding) and the incremental value. You can also specify the exact position in the new name where the number will appear.



In this example, we are tidying up the mixed-case names as well as renumbering the files. The special [#] code is used in the **New name** field to indicate where the number is to be inserted. The **Sequential numbering from** option specifies the starting number (in this case, 100) and the number of leading zeros lets you control zero padding. In this example, 0100 indicates that the number is to be padded to four digits.

Note that if you don't specify the [#] code, Opus will automatically insert the number at the end of the filename, before the file extension (so we could have left it out of the above example for the same results).

## Rename Macros

The **macro** feature provides a way to add and remove text to and from filenames without needing to resort to wildcards. You can use macros to add characters, remove characters, and move characters around inside filenames.

Macro operations:

The macro operations field displays the actual macro definition, but luckily you don't need to understand the macro language to use it. As well as displaying a preview of the rename operation, the preview list at the bottom of the *Rename* dialog also doubles as a macro builder. You can edit file names directly in the preview list, and doing so generates macros which batch-rename all the files in the same way.

To use the macro builder, make sure the **Use preview list to build macros** option is turned on.

Use preview list to build macros

For example, say we have a bunch of files whose names are in the format *YYYY-MM-DD\_draft.txt*, and we want to rename them as *Final DD.MM.YYYY.txt* – removing the “\_draft” suffix, swapping the order of the date fields around and inserting “Final” at the start of each filename.

To do this with wildcards would require a complicated regular expression, but a macro makes it easy. You simply pick one of the filenames in the preview list and edit it inline just like you would if you were renaming a single file – select areas of text, cut them to the clipboard, paste them in somewhere else, select another area, delete it, type some new characters, and so on.

<input checked="" type="checkbox"/> Current Name	New Name	New Ext.
<input checked="" type="checkbox"/> 2007-15-15_draft.txt	2007-15-15	txt
<input checked="" type="checkbox"/> 2012-02-10_draft.txt	2012-02-10	txt
<input checked="" type="checkbox"/> 2015-09-22_draft.txt	2015-09-22	txt

The macro builder will record your actions as a macro expression and apply the same changes to all selected files.

To build a macro, simply click on a filename in the **New Name** column of the preview list (or press the **F2** key). At this point, any keys you press will be recorded as an expression in the **macro operations** field (although you don't need to understand the macro language, it's described in the reference section for completeness).

The pencil icon (  ) indicates the current anchor position – that is, which end of the filename subsequent actions will be relative to. In the screenshot above, the anchor point is set to the right, meaning all operations are relative to the right-end of the string. This lets you, for example, delete a certain number of characters from the end of the filename without all the filenames having to be the same length.

To change the anchor position, you can double-click the pencil icon with the mouse, or position the cursor at the other end of the name and then press the cursor key (**Left** or **Right**) corresponding with that direction. You can also double-press the **Home** or **End** keys.

When you select a range of characters using the mouse or **Shift** plus the cursor keys, the equivalent range is shown as selected in all other files in the preview list – so you can check that your selection is correct before committing it.

In the above example, you can see that the original **\_draft** suffix has been removed – we did this by putting the anchor at the right end of the name, selecting left 6 characters and pressing the **Delete** key.

At the point the screenshot was taken we've selected the last two characters in the name by pushing **Shift + Left** twice, and the next step is to press **Ctrl + X** to cut those characters to the clipboard. This will generate the next macro expression **R0X2**. And so on, until the macro is complete. You can check the preview list at all stages to make sure the macro's having the desired effect on all the selected filenames – if some files are showing incorrect results you can always skip them by turning off their checkboxes.

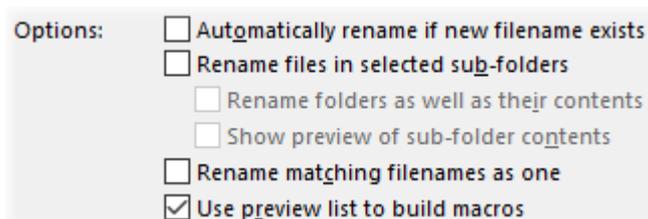
Note that in the macro builder the standard clipboard keys **Ctrl+C**, **Ctrl-X** and **Ctrl-V** will record those operations in the macro, and the operations will be treated separately for each filename. If you have some text on the clipboard that you want to paste into the macro "as-is" (rather than recording the paste operation itself), you need to press **Ctrl+Shift+V**.

If the **Use preview list to build macros** option is turned off, the preview list lets you edit the new filenames individually. This is great for making small changes to names which the macro (or other batch operations) may not have got exactly right. Once the name for a file has been edited directly it will be displayed in red and the name is locked in until the rename operation occurs (or until you clear the custom name).

If you're interested in the rename macro language, it's described in the [reference section](#) of this manual. Also note that the font used for the macro builder (which must be a fixed-width font) is configurable though [Preferences / Display / Colors and Fonts](#).

## Rename Options

The **options** section of the Rename dialog contains options which don't affect the new filenames, but instead affect how the rename operation itself behaves.



**Automatically rename if new filename exists** causes Opus to automatically append an incrementing number to the new filename if the new name is already in use. If turned off and a new filename clashes with an existing one, an error dialog will be shown.

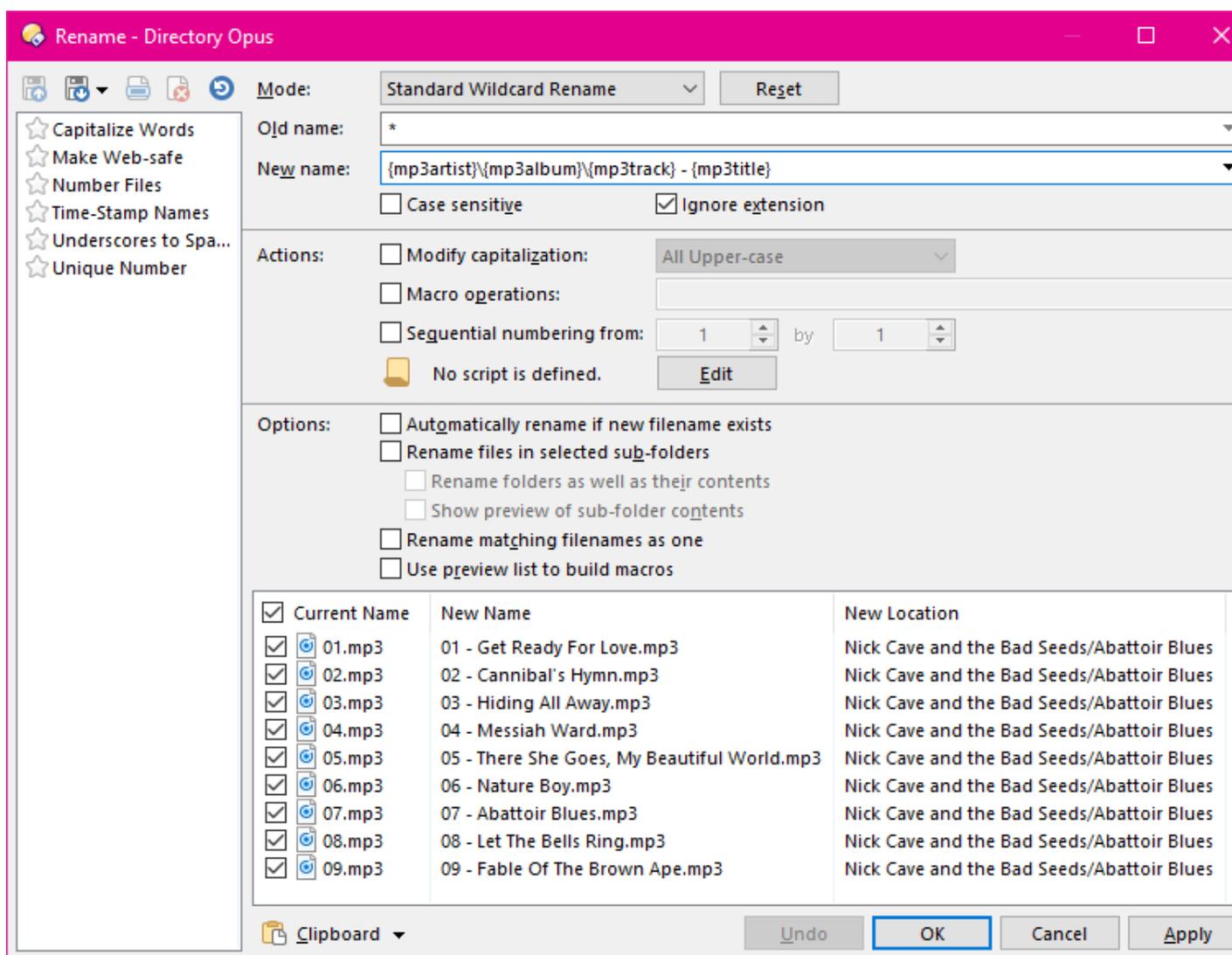
**Rename files in selected sub-folders** lets you make the rename operation operate recursively on files in selected folders rather than renaming the folders themselves. By default a recursive rename operation only renames the files within folders, not the folders themselves. Turn on the **Rename folders as well as their contents** option if you want to rename folders as well as files in the one operation. The **Show preview of sub-folder contents** causes a preview of the rename operation on the contents of any selected folders to also be shown in the preview list. Note that you can't edit or turn off individual sub-folder files like you can with files at the top-level.

**Rename matching filenames as one** causes Opus to treat multiple files with the same base filename (excluding file extension) as if they were the one file for the purpose of renumbering, wildcard or other batch renames. For example, if you had files called *01012.JPG* and *01012.WAV*, they would be given the same number as part of a renumber rename instead of being numbered sequentially.

**Use preview list to build macros** enables the macro recorder. When turned on, you can click on any "new name" in the preview list and edit the filename directly to record a macro that's then applied to all selected files. If this option is turned off, you can edit filenames individually in the preview list. This is great for making small changes to names which the macro (or other batch operations) may not have got exactly right. Once the name for a file has been edited directly it will be displayed in red and the name is locked in until the rename operation occurs (or until you clear the custom name).

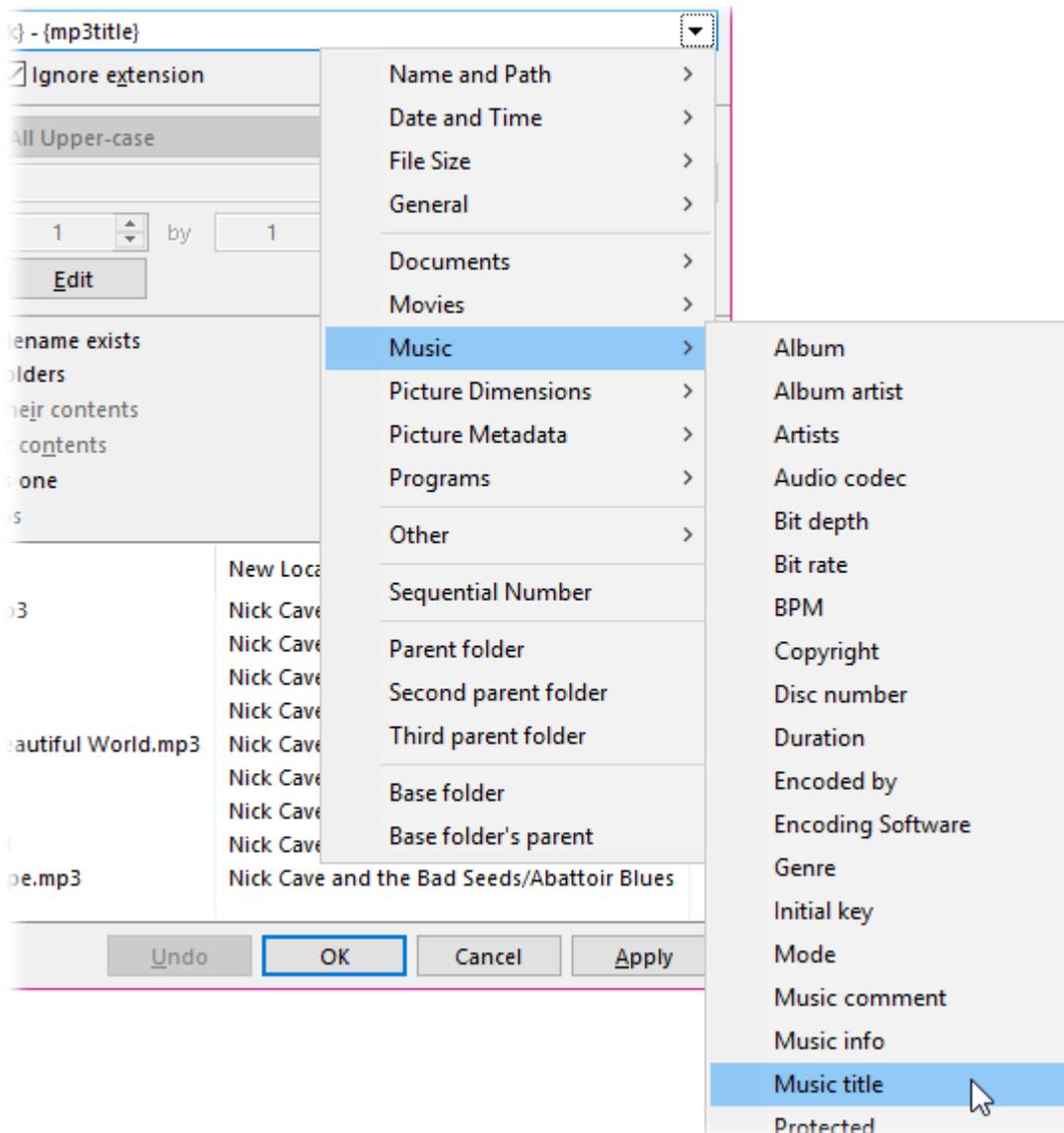
## Renaming with Metadata

You can use metadata (information about a file) to build up the new names of files by using various metadata insertion codes in the new name field.



In this example we can see that the original names of the MP3 files gave no information as to their content (*01.mp3*, *02.mp3*, etc). Luckily, MP3 files usually contain metadata describing the music. Not only are we using this metadata to rename the files, but we are using it to actually organize the music into folders.

Click the drop-down button to the right of the **New name** field to display a list of all the metadata fields that Opus supports.



These are the same fields as you can add to the file display as columns. When you pick a column from this menu (or one of the special, rename-only, options that appear at the bottom of the menu), the appropriate code is automatically inserted into the new name field at the current cursor position.

In the above example, we are using four music-related fields:

- **{mp3artist}** to retrieve the name of the artist (*Nick Cave and the Bad Seeds*).
- **{mp3album}** for the album name (*Abattoir Blues* - an excellent album I should add).
- **{mp3track}** for the track number on the album.
- **{mp3title}** for the actual name of the song.

The back-slashes that we have inserted between these fields has the effect of moving the file into sub-folders (relative to the current folder) as well as renaming it. The sub-folders will be created automatically if they don't already exist. The Rename tool can move files anywhere as long as they remain on the same drive - you can't rename from one hard drive to another.

The [Keywords for Columns](#) page contains a full list of the keywords that can be used by the **Rename** function.

As well as the full list of columns, the following codes can be used in the **new name** field:

- **{#}** (Sequential number): Lets you mark the position where the automatically incrementing number is inserted in the filename (when the **Sequential numbering** action is turned on).
- **{parent}** (Parent folder): The name of the parent folder of the file being renamed. The **Rename files in selected sub-folders** option affects the name that **{parent}** returns - if the rename operation recurses into a sub-folder, **{parent}** would return the name of that sub-folder for files within it.
- **{parent2}**: Second-level parent folder (i.e. the parent of the parent)
- **{parent3}**: Third-level parent folder, and so on.
- **{parentbase}** (Base folder): This code is similar to **{parent}** except it returns the name of the base folder rather than that of the file's parent - that is, the folder that the rename operation started in. This is most useful with a recursive rename, where the name returned by **{parent}** would change for files inside sub-folders.
- **{parentbase2}**: Base folder's parent, and so on.
- **{date}** or **{time}**: The current date or time in the local time zone.
- **{dateu}** or **{timeu}**: The current date or time in UTC.

The **{parent}** and **{parentbase}** codes can also strip the file extension from the parent name, which is usual if the parent folder is an archive file. To do this, add **|noext** to the code (e.g. **{parentbase|noext}**).

If you want to use rename to move files into new folders, adding **\$.\** at the start of the new name lets each file be moved relative to the base folder rather than its parent.

Sometimes metadata may contain characters that aren't valid in filenames. For example, the **{time}** field normally formats the time using colons (*HH:MM:SS*), and colons are not legal filename characters. By default illegal characters will be converted to the closest legal alternative (so for example, a colon will be converted to a semi-colon). However you may wish to control this process yourself:

- Date and time fields let you configure the date format, the time format, or both. For example,

**{datetaken|D#yyyy-MM-dd}** - inserts the date only in **yyyy-MM-dd** format (e.g. *2008-09-22*).  
**{modified|T#HH-mm-ss}** - inserts the modified time in **HH-mm-ss** format (e.g. *13:10:55*)  
**{datetaken|D#yyyyMMddT#HHmmss}** - inserts both the date and time as **yyyyMMddHHmmss** (e.g. *20051130154410*).

As you can see in the examples, **D#** is used to mark the date format, and **T#** is used to mark the time format. See the [Codes for date and time](#) page for information on date and time formats.

- Numeric fields let you control zero-padding. For example,

**{size|#8}** - zero-pads the size in bytes to eight places (e.g. *00045412*)  
**{mp3track|#2}** - zero-pads the track number to two places (e.g. *08*)

If for some reason you want to include a metadata code in the new filename literally (e.g. you really want to call a file *Blah{size}*) you can escape the leading **{** with another **{** character. E.g. **\*{{size}** as a new name would add the literal string *{size}* to the name of every file.

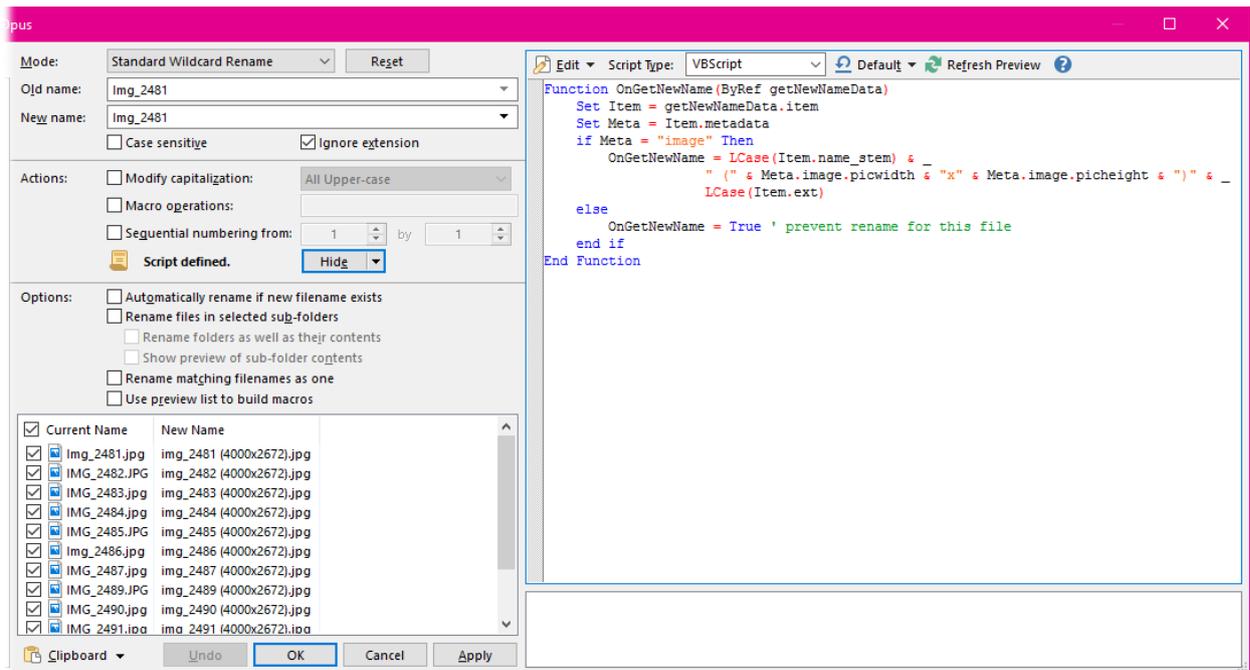
## Rename Scripts

This is by far the most powerful, but also most complicated, feature of the Advanced Rename dialog. Using VBScript, JScript or other ActiveX scripting languages, you can write a rename function that gives you complete control over the outcome of the rename operation.

Click the **Edit** button in the actions section of the Rename dialog to display the script editor.



The script editor will pop-out to the right of the dialog. The drop-down at the top of the script editor lets you choose the script language to use.



Note that a script can be defined (and have effect) even if the editor isn't visible. As shown in the screenshot above, the words **Script defined** will be displayed to indicate that a script is active. The drop-down attached to the **Hide** button gives you a clear command which lets you clear out the script contents.

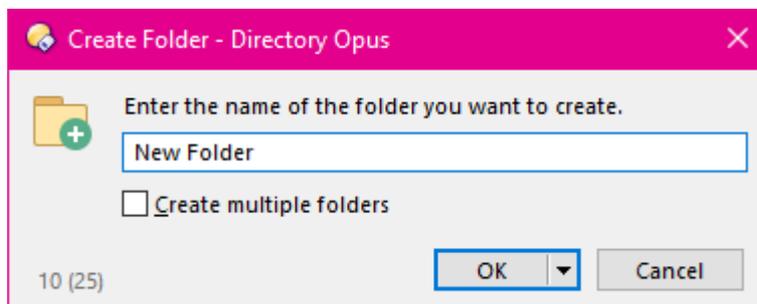
See the [Rename Scripts](#) page in the [Scripting](#) section for more information about writing rename scripts.

## Creating Folders

There are two ways to create a new folder using Opus. The first way is the same as in Explorer - right-click on the background of the file display (or on a folder in the tree) and choose **New / Folder**.

The second way is to use the **Create Folder** dialog, which as well as letting you make a new folder also offers some additional functionality:

- You can make multiple folders simultaneously
- You can make a whole sub-tree of folders at once (so a folder, a sub-folder, a sub-sub-folder, and so on, all with the one command)
- You can make multiple sub-folders under the same parent folder at once
- You can automatically navigate to the newly created folder or folders



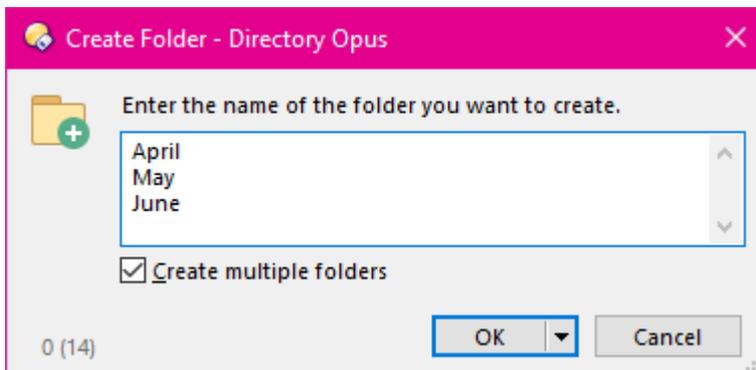
The two numbers displayed in the bottom-left corner of the dialog (as shown above) indicate the length of the name you have entered and the total length of the path once the folder is created. This can be useful to ensure that your path length doesn't grow beyond the normal maximum of 259 characters - although Opus has no problems dealing with longer paths than that, many other programs (including Explorer) will fail.

To create a folder using the **Create Folder** dialog, click the **New Folder** button on the toolbar and enter the name of the folder to create. You can create a whole tree of sub-folders at once by entering the folder names separated by slashes. For example:

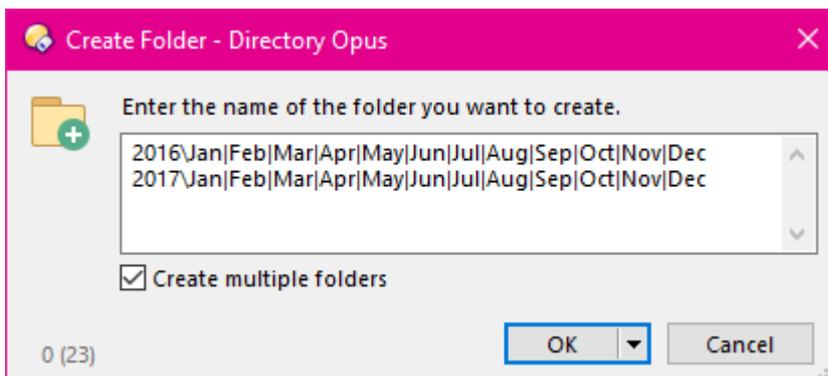
One\Two\Three\Four\Five

This would create a folder called *One* in the current location, then a sub-folder in *One* called *Two*, a sub-folder in *Two* called *Three*, and so on, all in the one operation.

If you turn on the **Create multiple folders** dialog you can create multiple folders simultaneously in the current location. If this is turned on the name field changes to a multi-line text box that lets you enter as many names as you like (one per line):



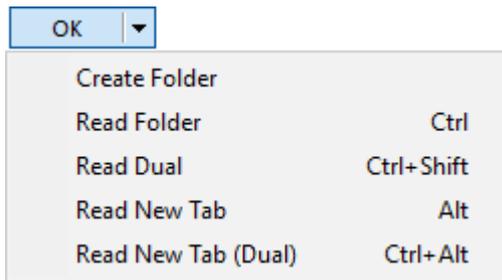
The folders will all be created relative to the current folder. You can also create multiple child-folders at the same sub-folder level using the vertical bar character (|) to separate the child folder names.



The above example will create two child folders, *2016* and *2017*, and below those create 12 sub-folders (one for each month of the year).

In **Create multiple folders** mode, the **Return** key moves to a new line in the text box, but you can still quickly *OK* the dialog by pushing **Return** twice on the last line, or by pushing **Shift+Return** on any line. Since double-tapping **Return** at the end is only one extra keystroke, even when only creating a single folder, you may find it convenient to leave the dialog in this mode all the time.

The drop-down menu attached to the **OK** button lets you choose what to do with the new folder once it's been created.

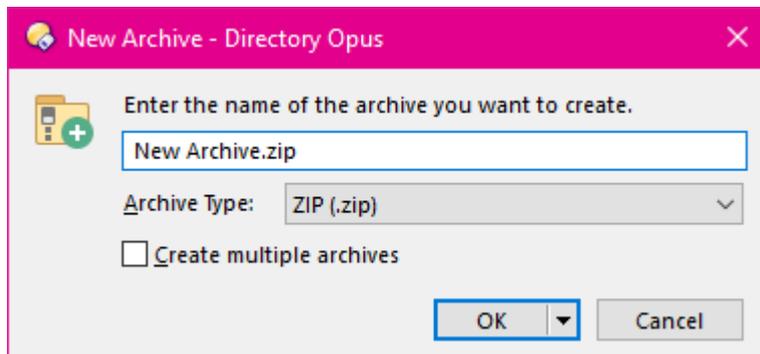


- **Create Folder:** The new folder (or folders) is created, but not read.
- **Read Folder:** The file display will automatically navigate into the new folder once it is created.
- **Read Dual:** The new folder will be opened in the other file display (the Lister will be put into dual-display mode if needed). If you are creating multiple folders, only the first folder will be read.
- **Read New Tab:** The new folder will be opened in a new folder tab. If you are creating multiple folder a new tab will be created for each new one.
- **Read New Tab (Dual):** The new folder will be opened in a new folder tab in the other file display.

You can also hold the appropriate qualifier key when clicking the **OK** button, rather than dropping down the menu.

## Creating Archives

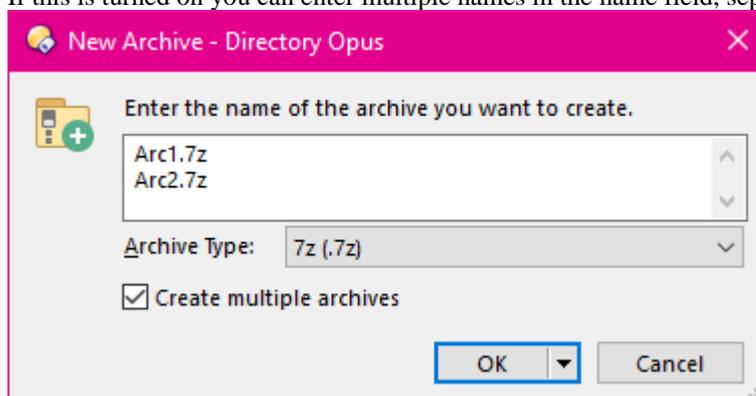
Opus supports multiple archive formats (Zip support is built-in, and support for other formats like 7zip and RAR is provided by means of a plugin). There are a number of ways to create an archive, but if you want a completely new, empty archive file, the easiest way is with the **New Archive** command in the **New Folder** dropdown menu.



This dialog works very much like the [New Folder](#) dialog. Enter the name of the archive (or accept the default name) and use the **Archive Type** drop-down to select the type of archive you want to create. The file extension of the new archive will change automatically to match the type of the archive you're creating.

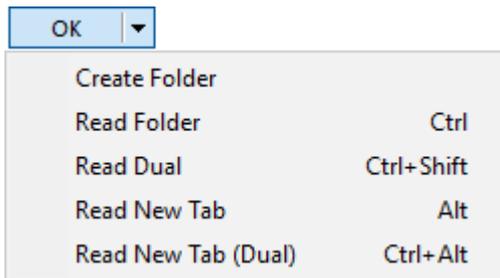
The other options in this dialog are:

- **Create multiple archives:** Select this option if you want to be able to create more than one archive at once. If this is turned on you can enter multiple names in the name field, separated by commas.



The archives will all be created in the same location.

- **Read new archive automatically:** If this is selected the file display will automatically navigate into the new archive once it is created.
- **Open in dual display:** The new archive will be opened in the other file display (the Lister will be put into dual-display mode if needed). If you are creating multiple archives, only the first archive will be read.
- **Open in new Folder Tab:** The new archive will be opened in a new folder tab. If you are creating multiple archives a new tab will be created for each new archive. You can combine this option with the **Open in dual display** option to have the new tabs created in the other file display.



The **Open in new Folder Tab** option is a "tri-state" checkbox - it can be checked, unchecked, or in the "indeterminate" state. Normally when this option creates a new tab (or multiple tabs) the first new tab will be set to active automatically, but setting this option to the "indeterminate" state has the effect of not making the newly created tab active.

Once you've created a new archive, you can add files to it in several ways. Opus treats archives exactly like a folder; this means you can:

- Drag and drop files to it. If you have the display of archives in the folder tree enabled in [Preferences](#) you can also drag and drop to archives in the tree.
- Copy files from one folder using the clipboard, then navigate to the archive and paste them in
- Open it in a dual-display and use the [Copy Files](#) function to copy files directly to the archive

The **New Archive** command creates an empty archive which you can then copy files into, but if you have one or more files that you want to add to a new archive, you can skip the **New Archive** step by using the [context menu items](#) that Opus can add. If turned on, right-clicking the files or folders will display one or more commands for [adding to a new archive](#).

## Adding to Archives

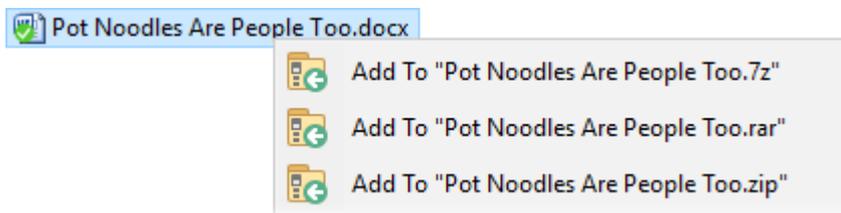
There are a number of ways to add files to an archive using Opus.

- You can [create a new archive](#) (if needed - you can also add to existing archives), then open it as if it were a folder and copy or paste files into it as normal.
- You can drag and drop files over the top of the archive to add them to it (again, this is treating the archive as if it were a folder).
- You can use the **Archive Files** command on the toolbar (or the variant commands within the drop-down attached to this button).
- You can right-click on the files you want to add and use the context menu (if enabled) to add them to a new or an existing archive.

The easiest way to access this functionality is via the context menu. In the [Archive Context Menu](#) page in Preferences you can turn on or off context menu commands for adding files to various

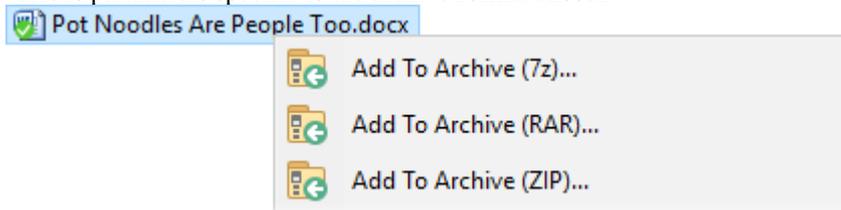
archive formats. These commands will be shown whenever you right-click on one or more files and folders. There are two types of "add" operations that can be performed via the context menu:

- **Add to Named Archive:** These commands will add the selected files and folders immediately to a new archive. The new archive will be located in the same folder as the source files, and it will be given the same name as the first selected file.



For example, if you right-click a file called *Presentation.doc* the context menu might contain a command called **Add To "Presentation.7z"** - selecting this command would create a new 7zip archive called *Presentation.7z* and add the selected file to it. The [Preferences](#) page lets you choose which formats an Add to Named Archive command will be displayed for.

- **Add to Archive:** These commands will also add the selected files and folders to a new archive, but instead of immediately creating the archive you will be given a choice of filename and path, archive type, and various parameters specific to the archive format chosen.

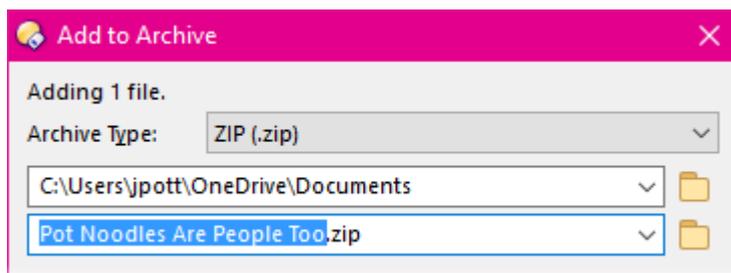


The [Preferences](#) page also lets you choose which formats to add these commands for although this selecting say **Add To Archive (RAR)** only sets *RAR* as the initially chosen format; the **Add to Archive** dialog lets you change the selected format.

Using the context menu **Add to Archive** command, or selecting the **Archive Files** command from the toolbar, displays the [Add to Archive dialog](#).

## Add to Archive Dialog

When you use the **Archive Files** button or context menu commands, the **Add to Archive** dialog box is displayed. This dialog is divided into two parts.



The top of the dialog is always the same and is where you select the archive type, path and name.

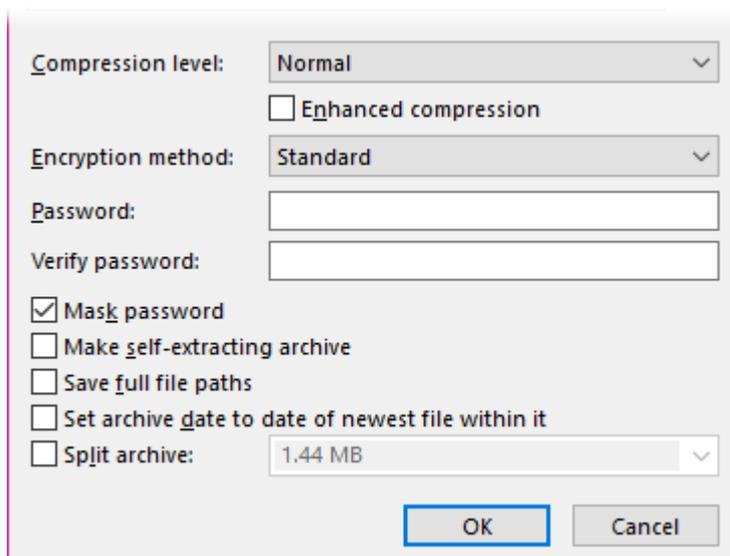
The **Archive Type** drop-down lets you choose from any of the archive formats that Opus is able to write to. (Some formats are read-only and won't be shown here.) Opus supports multiple archive formats, with Zip built-in and formats like 7z and RAR handled by a plugin. Third-party plugins can also add to this list.

The two text fields allow you to specify the location and name of the archive, respectively. Their drop-downs display your most recently used locations and archive names. (If you select an archive name from the history, its extension will be adjusted automatically if it doesn't match the type of archive you are creating now.)

As well as adding to a new archive you can also select an existing archive to add more files to it. The two **Browse** (📁) buttons allow you to locate existing folders and archives. Use the browse button next to the path to keep the name and only change the location. Use the browse button next to the name to select an existing archive; it will initially show you archives below the current path but you can also use it to select archives from other locations.

If your clipboard contains the full path to an archive you would like to use, you can paste it into the name field to automatically fill in both text fields at once.

To copy the full, combined path from both text fields, push **Ctrl-Shift-C**.



The bottom of the dialog displays options specific to the selected archive type. These options will change as you change what's selected in the **Archive Type** drop-down. The screenshot above is what you will see when the Zip type is selected.

Some archive formats do not have any options at all while other formats allow you choose things like compression modes and encryption options.

## Zip Options

When using the [Add to Archive](#) dialog to add files to a Zip archive, the following options are available:

- **Compression level:** This lets you set the compression level used when adding the files to the archive. There are six compression levels available, ranging from *Store* (which does no compression at all and so is the fastest) to *Best* (which produces the highest level of compression but takes longer to archive).
  - **Enhanced compression level:** This activates an [enhanced compression algorithm](#) that may not be backwards compatible with some Zip tools - if you are sending Zip files to other people you should make sure they can decompress such archives.
- **Encryption method:** If you want to encrypt the files in the Zip archive, you can choose the encryption method to use here.

*Standard* is the oldest, most backwards-compatible encryption method, and all Zip tools should be able to decompress archives encrypted with this method. However it's also the least secure method. If you want a more secure encryption you can select one of the *AES* options from the drop-down (the three [AES algorithms](#) are the same, just with different key lengths). If you are sending Zip files to other people you should make sure they can decompress AES-encrypted archives before using this option.

- **Password:** If you want to encrypt the files you are adding to the Zip archive, enter a password here. If the **Mask password** option is on, the password you enter here will be hidden - and so for security you must enter the password again in the **Verify password** field. If you leave the **Password** field empty no encryption will be performed.
- **Verify password:** If you have entered a password in the **Password** field you need to enter the same password here as a confirmation (unless **Mask password** is turned off).
- **Mask password:** If this option is on the passwords you enter to encrypt the files you are adding will be masked (not displayed).
- **Make self-extracting archive:** Turn this option on if you want to make a [Self-Extracting Zip File](#) out of the Zip archive once it has been created.
- **Split archive:** The Zip archive format supports the concept of split archives - a larger archive that is stored as multiple files instead of one large file.

A split archive (as long as you have all the pieces) can be accessed just like it was all the one file, and it may be useful to use this when emailing large archives (or for backup on removable media, etc). Turn this option on if you want to make a split archive. You can choose a pre-defined chunk size from the drop-down list, or enter your own size. If you enter a manual size you can specify it in kilobytes (**KB**), megabytes (**MB**) or gigabytes (**GB**) by entering the number followed by the appropriate suffix. For example, **1.87 MB**. If no suffix is given, the chunk size is taken to mean bytes.

- **Save full file paths:** If this option is turned on then the full paths of files you add to the archive will be stored in the archive. If this option is off the path is stored relative to the current folder.

For example, say you go into **C:\Test**, select a folder called **Reports** that contains one file (**Sales.docx**) and add it to an archive. If this option was off, the path of **Sales.docx** in the archive would be stored as **Reports\Sales.docx**. If the **Save full file paths** option was on, the path stored would be **C:\Test\Reports\Sales.docx**.

## 7-Zip Options

When using the [Add to Archive](#) dialog to add files to a 7-Zip archive, the following options are available:

- **Compression level:** This lets you set the compression level used when adding the files to the archive. There are six compression levels available, ranging from *Store* (which does no compression at all and so is the fastest) to *Best* (which produces the highest level of compression but takes longer to archive).
- **Compression method:** This lets you choose the compression method used when adding the files.
  - LZMA: [LZMA](#) is the default method for 7z compression. It offers a high compression ratio with good decompression speed and low memory requirements.
  - LZMA2: This is an improved version of LZMA.
  - PPMD: This is a variant of the PPM ([prediction by partial matching](#)) compression technique.
  - BZip2: [BZip2](#) is an older compression method that is generally slower than LZMA.
- **Solid mode:** This option enables [solid compression](#), which can improve compression when storing multiple similar files. You can specify the solid block size using the drop-down control. Compressing in solid mode imposes some restrictions on accessing the archive. You may find it takes a long time to view or extract individual files within solid archives because the data before the file you requested may need to be decompressed first. Features such as thumbnails are generally disabled within solid archives.
- **Password:** If you want to encrypt the files you are adding to the archive, enter a password here. If the **Mask password** option is on, the password you enter here will be hidden - and so for security you must enter the password again in the **Verify password** field. If you leave the **Password** field empty no encryption will be performed.
- **Verify password:** If you have entered a password in the **Password** field you need to enter the same password here as a confirmation (unless **Mask password** is turned off).
- **Mask password:** If this option is on the passwords you enter to encrypt the files you are adding will be masked (not displayed).
- **Encrypt file names:** As well as encrypting the file data stored in the archive, this option will encrypt the file headers - so that the names of the files in the archive are also encrypted.
- **Use multiple threads:** If this option is on then Opus will use multiple threads when adding to the archive - on a multi-CPU machine this should make the compression process quicker. The number of CPUs used varies depending on the compression algorithm.
- **Store full file timestamps:** This option causes the full timestamp (created, last modified and last accessed) of the files to be stored in the archive.

## RAR Options

When using the [Add to Archive](#) dialog to add files to a RAR archive, the following options are available:

- **Compression level:** This lets you set the compression level used when adding the files to the archive. There are six compression levels available, ranging from *Store* (which does no compression at all and so is the fastest) to *Best* (which produces the highest level of compression but takes longer to archive).
- **Solid compression:** This option enables [solid compression](#), which can improve compression when storing multiple similar files. Compressing in solid mode imposes some restrictions on accessing the archive. You may find it takes a long time to view or extract individual files within solid archives because the data before the file you requested may need to be decompressed first. Features such as thumbnails are generally disabled within solid archives.

- **Password:** If you want to encrypt the files you are adding to the archive, enter a password here. If the **Mask password** option is on, the password you enter here will be hidden - and so for security you must enter the password again in the **Verify password** field. If you leave the **Password** field empty no encryption will be performed.
- **Verify password:** If you have entered a password in the **Password** field you need to enter the same password here as a confirmation (unless **Mask password** is turned off).
- **Mask password:** If this option is on the passwords you enter to encrypt the files you are adding will be masked (not displayed).
- **Encrypt file names:** As well as encrypting the file data stored in the archive, this option will encrypt the file headers - so that the names of the files in the archive are also encrypted.
- **Send passwords to WinRAR via command-line:** When Opus invokes WinRAR (see note below) this option allows Opus to pass it the password you entered above on the command line. This is more convenient - the alternative is for WinRAR to prompt you for a password itself - but it may be less secure, as the password you enter is visible in Task Manager for the duration of the operation.
- **Use multiple threads:** If this option is on then Opus will use multiple threads when adding to the archive - on a multi-CPU machine this should make the compression process quicker.
- **Store full file timestamps:** This option causes the full timestamp (created, last modified and last accessed) of the files to be stored in the archive.

While Opus can read and decompress RAR archives by itself, creation and modification of RAR archives requires you to have [WinRAR](#) installed on your system. This is because RAR compression is a proprietary system that has not been released by its creators for other developers to use and Opus is only able to add files to RAR archives by invoking WinRAR in the background.

### **TAR BZip2 Options**

When using the [Add to Archive](#) dialog to add files to a **.tbz2** archive, the following options are available:

- **Compression level:** This lets you set the compression level used when adding the files to the archive. There are five compression levels available, ranging from *Fastest* (the quickest, but poorest compression level) to *Best* (which produces the highest level of compression but takes longer to archive).
- **Use multiple threads:** If this option is on then Opus will use multiple threads when adding to the archive - on a multi-CPU machine this should make the compression process quicker.

### **TAR GZip Options**

When using the [Add to Archive](#) dialog to add files to a **.tgz** archive, the following options are available:

- **Compression level:** This lets you set the compression level used when adding the files to the archive. There are five compression levels available, ranging from *Fastest* (the quickest, but poorest compression level) to *Best* (which produces the highest level of compression but takes longer to archive).

## Zip Files

Unlike 7-Zip, RAR and other archive formats that are implemented via a plugin system, Opus includes built-in support for Zip files. Although you don't really need to care about the distinction, it's worth mentioning because you may wonder otherwise why it seems like Zip archives are treated differently in some ways to other archive formats. For example, options affecting Zip files are found on a separate page in Preferences to other archive formats.

While you can normally work with Zip files and other archive formats the same way, there are a few special features provided for Zip files that don't apply to the other formats:

- **Zip Comment:** Explicit support is provided for setting the embedded [comment](#) inside Zip archives
- **Read-Only mode:** A special mode that applies only to Zip files - they can be opened as "[read-only](#)" to prevent accidental changes to the contents of the archive
- **Self-Extracting Zip Files:** Opus can convert Zip files to a [self-extracting](#) format - an executable file that automatically extracts its contents when run.

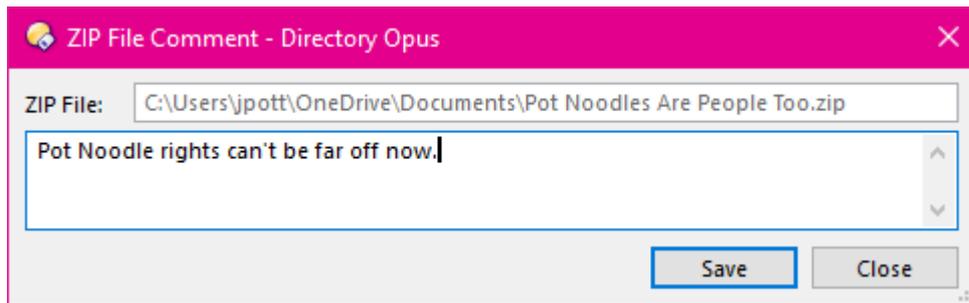
Opus also uses Zip files internally for some of its own file formats. For example, if you [backup your configuration](#), Opus creates a **.ocb** file - this is really a Zip archive, and if you rename the file to **.zip** you can open it like a normal archive.

### Zip Comment

Opus lets you assign a comment to a Zip file that is stored within the Zip data structure itself. In Opus you can see this comment by turning on the Description column in the file display - other Zip software will display it differently (for example, some programs will pop up a dialog showing the comment when you open the Zip file).

Name ▲	Size	Description
 Pot Noodles Are People Too.zip	12.9 KB	Pot Noodle rights can't be far off now.

To edit the Zip comment, navigate into the Zip file itself, then right-click on an empty area of the file display (or on the status bar) to show the [Lister Zip Context Menu](#), and choose the **ZIP Comment** command.



## **Read-Only mode**

*Zip Read-Only* mode lets you make Zip files read-only (in Opus, that is) to prevent accidental changes to the contents of the Zip file.

The **Open Zip files as read-only by default** option on the [Zip & Other Archives / Zip Files](#) page in Preferences activates this mode by default. If you have this option on and double-click on a Zip file to read its contents in Opus, you will find that you're not able to add items to or delete items from the archive - instead, an error will be displayed.

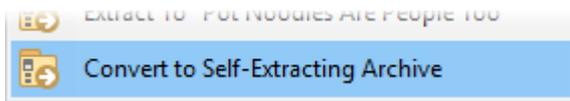
To make proper use of this mode (unless you just want to have it on all the time, and never make changes to Zip files), you will need to add the **Read-Only** button to a toolbar or menu. This lets you toggle *Read-Only* mode on or off when in a Zip file. This command can be found in the [Edit category](#) on the **Commands** tab in the **Customize** dialog - see the [Customize](#) section of this help file for more information.

## **Self-Extracting Zip Files**

Opus lets you convert a Zip archive into what's called a self-extracting archive. This turns an archive file (which needs another program to decompress it) into an executable program that you can send to people. When they run the program, an easy-to-use interface will guide them through extracting the contents of the archive.

Now that Windows itself comes with built-in support for Zip files self-extracting archives are not as widely used as they once were, but there are still some uses for them.

To create a self-extracting archive, you must first [create the source Zip archive](#). You can use the [Add to Archive](#) dialog to create a Zip archive and convert it to self-extracting format in one step, or you can create the archive and then convert it separately using the context menu command that Opus adds automatically to the context menu for **.zip** files.



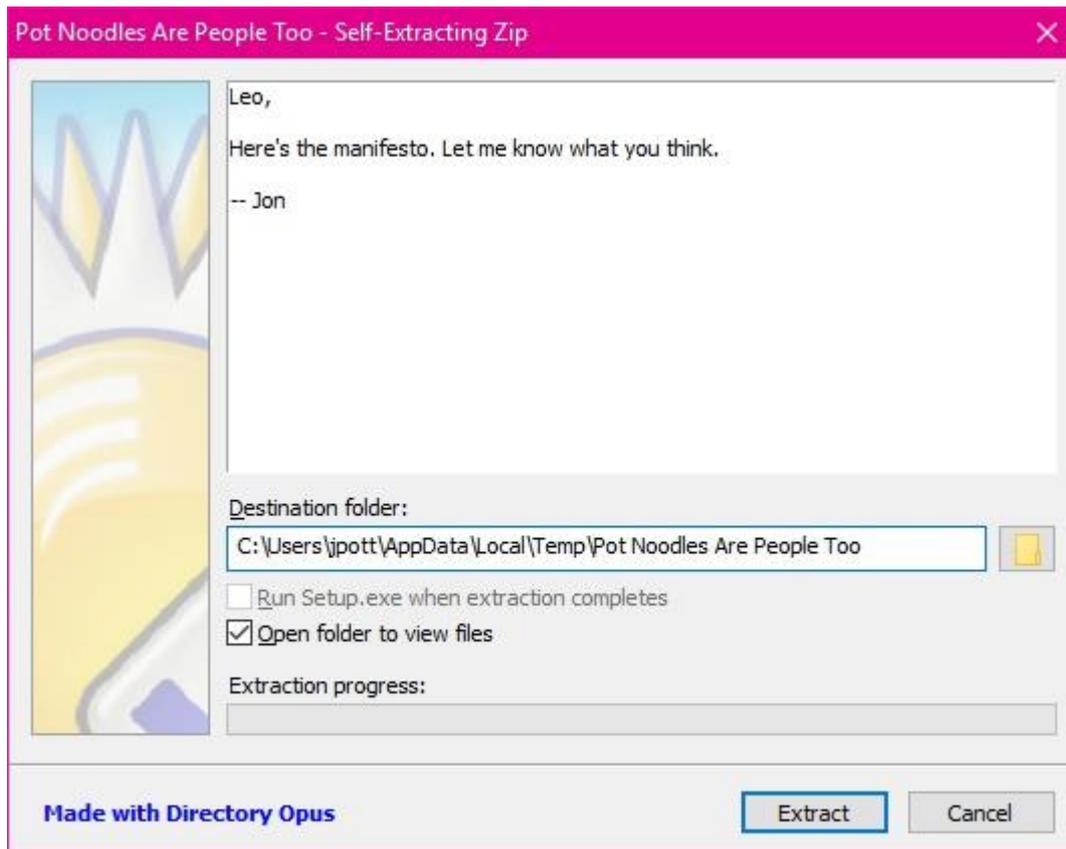
This context menu can be disabled through the [Archive Context Menu](#) Preferences page, so if you don't see it when you right-click on a Zip file you may need to go and turn it on.

The **Create Self-Extracting Archive** dialog gives you several options that control the final self-extracting archive:

- **From:** This field displays the name of the Zip archive you are converting.
- **To:** This specifies the path of the **.exe** file the conversion process will create. This defaults to the same location and name as the source archive, but with a **.exe** instead of **.zip** extension.
- **Name:** This lets you configure a name that is displayed in the title bar of the self-extractor when it's run.
- **Custom instructions:** If you check this box you are able to select an external text file (**.txt**) to provide instructions or a message that will be displayed in the self-extractor when it's run. If you don't provide a file then default instructions are displayed.
- **Custom image:** If you check this box you can select an external image file that is displayed on the left-hand side of the self-extractor window. If you don't select this option a standard image is used. If you turn this option on but don't select an image file then no image at all will be used, which can reduce the size of the final **.exe** file slightly.
- **Default path for extraction:** This lets you specify a location that will be used as the default path the files in the archive are extracted to when the self-extractor is run. The end-user will still be able to modify the destination path; this is simply a default.
- **Auto-run after extraction:** If your archive contains any executable programs, you can select one from the drop-down list to have that file automatically run after the archive has been extracted. For developers, you could use this to create a simple "installer" that extracts several files and then runs an included Setup.exe-type program to perform the actual installation.

Once you have chosen your parameters, click the **OK** button to perform the actual conversion process. You should be left with a **.exe** file slightly larger than the original Zip archive.

When someone runs your self-extracting archive, they will see a window similar to this:



From this window you can deduce the parameters we set when creating the extractor:

- The image on the left of the window is the default - we didn't specify anything for **Custom image** when creating the self-extractor.
- The displayed text is the result of specifying a **Custom instructions** file.
- We didn't specify a **Default path** for extraction, so the default behavior is to extract to the temporary folder.
- We also didn't specify an **Auto-run after extraction** file, so that option is disabled.

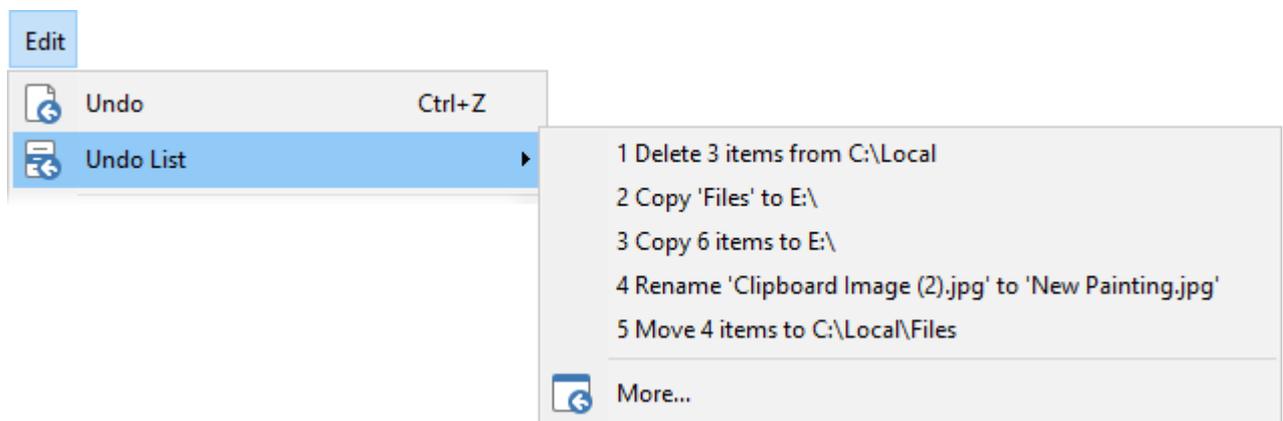
The **Open folder to view files** option is user-controllable; if they select that then a window (from their default folder browser) will be opened automatically to show the contents of the extracted archive.

The extraction function handles UAC automatically under Vista and Windows 7. If the user opts to extract to a UAC-protected location (like **C:\Program Files**) they will be asked to elevate to approve the action.

## Tracking and Undoing File Operations

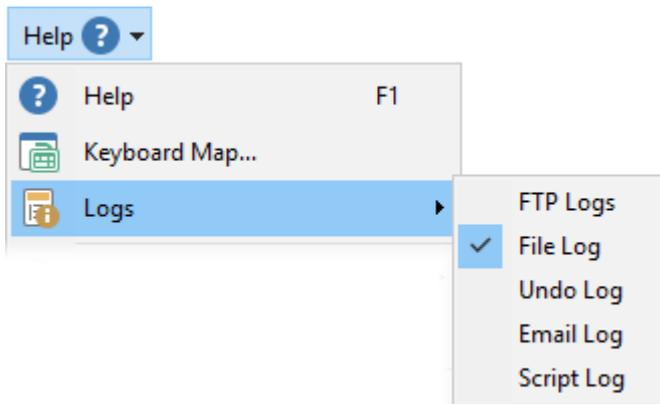
There are two features in Opus that help you keep track of changes you have made to files and folders.

The **Undo** command, as you would expect, can usually be used to undo the most recent action. Most actions can be undone - for example, if you rename a group of files incorrectly with a wildcard, you can undo the rename fairly easily. File deletes, however, can not be undone unless the delete was done to the recycle bin. You can find the **Undo** command in the **Edit** menu in a Lister (or press **Ctrl+Z**).



Opus actually maintains a history of undoable functions. From the **Undo List** sub-menu, you can see the ten most recent actions, and undo any of them by selecting it from the list. Selecting the **More...** command from that sub-menu displays the [Utility Panel](#), showing the **Undo Log** page. This page displays a full list of undoable actions, and you can undo one, many or all actions from this page at the click of a button.

While the undo log only shows the most recent undoable actions in the current session, the file log maintains a persistent log of all file operations (undoable or not). You can access the **File Log** from the [Utility Panel](#), or select the command directly from the **Help** menu.



By default the file log records the last 1000 file operations. You can see what sort of action occurred, the file involved, when it happened, and in the case of a copy or move, the destination folder.

Utility Panel: File Log ▾

Action	File	Destination	Date
Copy	lib://Pictures/Argentina_0005.jpg	C:\Home Shots\Argentin...	7/07/2016 3:18:09 PM
Docume...	C:\Users\jpott\AppData\Local\Virt...		6/07/2016 5:14:46 PM
Move	C:\Users\Public\Documents\Cats a...	C:\Users\Public\Docume...	6/07/2016 5:02:46 PM
Move	C:\Users\jpott\Documents\Best Ne...	C:\Users\jpott\Documen...	6/07/2016 5:02:38 PM
Move	C:\Users\jpott\Documents\1001 Th...	C:\Users\jpott\Documen...	6/07/2016 5:02:38 PM
Delete	lib://Documents/en_office_profess...		6/07/2016 5:00:19 PM

The file log can be extremely useful at times - if you've ever found that a file you knew was there has mysteriously disappeared, you can often find out what happened to it through the log.

The  button at the top of the log lets you export the log as a text file - useful if it's quite large and you want to search it in a text editor. The  button is used to clear the log. You can configure the maximum size of the log, and control which operations are recorded, from the [File Operations / Logging](#) page in Preferences.

# Changing Attributes

The **Change Attributes & Times** dialog lets you modify both the attributes of a file and its timestamps. To access this function, select the files and folders you wish to modify and choose the **Attributes** command from the **Properties** drop-down menu.

*Attributes* are the set of "flags" that all files and folders have, that control certain properties of the file entry as part of the filesystem. File attributes are displayed in the **Attr** column in the file display (in details and power mode).

Name ▲	Attr	Created	Modified	Size	Type
 Img_2481.jpg	r-h---	22/09/2015 8:48 AM	22/09/2015 8:48 AM	199 KB	JPEG image
 IMG_2483.JPG	-a----	8/01/2013 10:59 AM	8/01/2013 11:00 AM	1.91 MB	JPEG image

The attributes that you can change through this dialog are:

- **Archive:** Indicated by an **A** in the **Attr** column, this attribute is usually used to indicate that a file needs to be archived. The **A** attribute is normally set automatically when a file is saved or edited, and backup tools can use this to tell that a file needs to be backed up (and they would then clear it automatically after doing so).
- **Compressed:** Indicated by a **C** in the **Attr** column, this attribute indicates that the file is compressed. Only NTFS supports file compression - FAT/FAT32 formatted disks don't offer this feature.

If a folder is set as compressed then all new files created in that folder will be compressed by default. This option is exclusive with **Encrypted** (you can't both compress and encrypt the same file).

- **Encrypted:** Indicated by an **E** in the **Attr** column, this attribute indicates that the file is encrypted. Only NTFS supports file encryption - FAT/FAT32 formatted disks don't offer this feature.

When a file is encrypted, only the users who have been associated with the file will be able to decrypt it - for example, if your laptop is stolen, the thief might have access to the files on the hard drive, but without your login password would not be able to decrypt the data.

If a folder is set as encrypted then all new files created in that folder will be encrypted by default. This option is exclusive with **Compressed** (you can't both encrypt and compress the same file).

- **Hidden:** This is indicated by an **H** in the **Attr** column. When a file is set to hidden its icon will be shown as ghosted (as in the image above), but the file itself is not necessarily hidden from the display. You can configure whether Opus shows hidden files and folders through the **Global hide filters** options in the [Folders / Folder Display](#) Preferences page. The display of hidden files can also be controlled on a per-folder basis using the [Folder Formats](#) system.
- **Non-Indexed:** This is indicated by an **I** in the **Attr** column. The file's content will not be indexed by the system (for searching), only its name and other metadata.
- **Read-only:** This is indicated by an **R** in the **Attr** column. When a file is set to read-only, it normally can't be overwritten or deleted. For example, if you set a **.txt** file to read-only and then try to edit it in Notepad,

Notepad would not be able to save over the existing file. Read-only files can still be deleted to the recycle bin as normal.

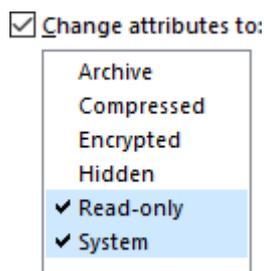
- **System:** Indicated by an **S** in the **Attr** column, the system attribute is generally used in conjunction with the **H** attribute to mark files and folders that "belong" to the operating system. By default, files marked **H+S** will be hidden from the display, although you can enable the display of these items by turning off the **Hide protected operating system files** option on the [Folders / Folder Display](#) Preferences page.
- **Offline:** Indicated by an **O** in the **Attr** column, this indicates that the file is "offline" - that is, it resides on a network or other computer and will only be copied to your computer when you try to access it. This is most commonly used by Microsoft OneDrive.
- **Pinned:** Indicated by a **P** in the **Attr** column, this indicates that the file is "pinned" - that is, marked to remain permanently on this machine rather than being moved to offline storage. This is most commonly used by Microsoft OneDrive.

*Timestamps* are the various time and date values stored for each file and folder. The timestamps you can modify through this dialog are:

- **Creation time:** The timestamp that represents when the file was first created.
- **Last modified time:** The timestamp that represents the last time the file was modified.

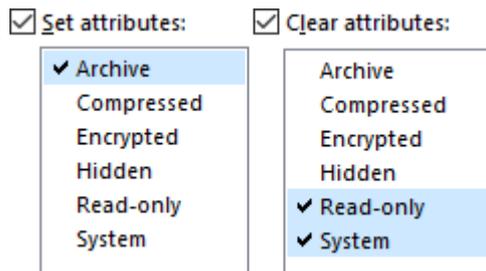
The **Change Attributes & Times** dialog lets you modify either the attributes, or the timestamps, or both. The **Attributes** section of the dialog lets you modify attributes in two ways:

- **Change attributes to:** Select this option if you want to specify the absolute attributes the files are to have.



Click the attributes in the list to toggle their checkmarks on or off. The file's existing attributes will be replaced with the new set of attributes you select.

- **Set and Clear attributes:** Select either or both of these options if you want to set and/or clear specific attributes while leaving others unchanged.



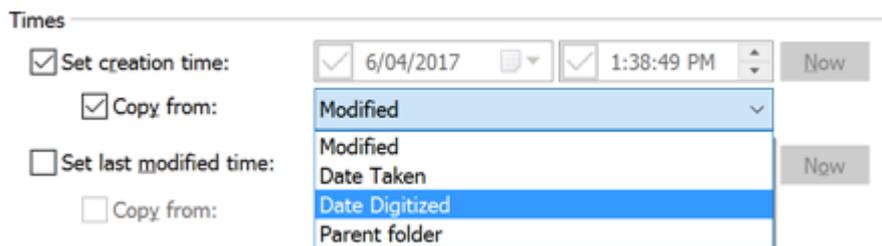
In the above screenshot, the **A** attribute would be set and the **R** and **S** attributes would be cleared, but the other attributes of the files (**C**, **E** and **H**) would not be changed either way.

The **Times** section of the dialog lets you choose which timestamps you wish to modify.

- **Set creation time:** Turn on this option to modify the files' creation timestamp.
- **Set last modified time:** Turn on this option to modify the files' last modified timestamp.

For both creation and modified time you can enter a date, a time, or both. Use the checkboxes in the date and time fields to turn them on or off - turning a field off will cause that element in each file's timestamp to be preserved.

Click the **Now** button to set the timestamp to the current date and time. You can also use the **Copy from** option to copy from another timestamp.



You can copy timestamps from the created and modified timestamps, in the case of pictures from the date taken and date digitized EXIF fields, and also the timestamp from the parent folder. You can also copy from various document-related timestamps.

The **Sub-folders** section of the dialog lets you control what happens how folders are processed by this function.

- **Make the same changes to files within selected folders:** If you turn this option on, the **Change Attributes & Times** function will recursively set the attributes and times you've specified to all files within selected

folders (and to files within sub-folders, and so on). If this option is off, any selected folders will have their attributes and times modified as normal but their contents won't be changed.

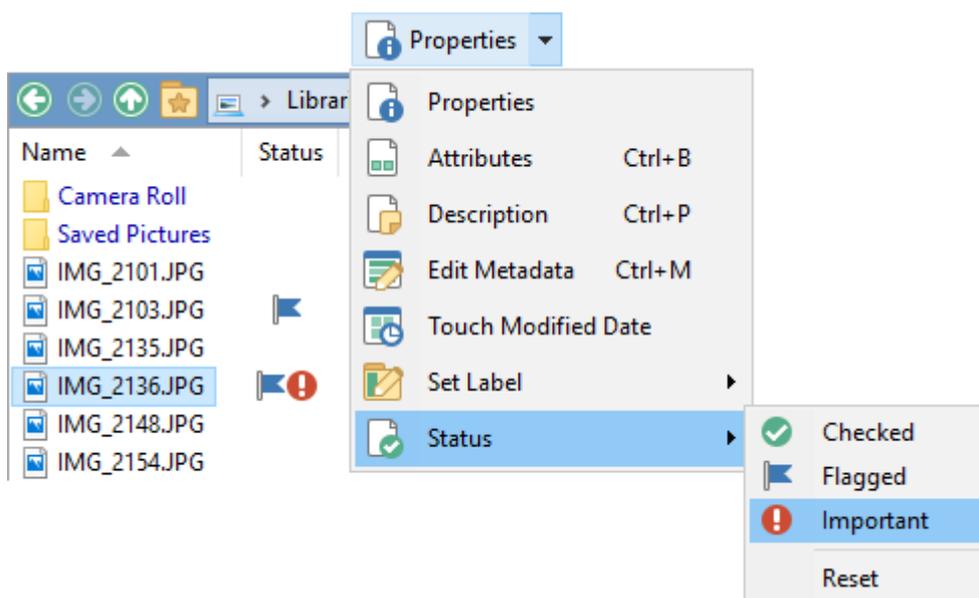
- **Use filter:** Lets you specify a filter to control which files within sub-folders are modified. You can either enter a [wildcard pattern](#) directly, select a [pre-configured filter](#) from the drop-down, or click the **Define** button to [define a new filter](#).

When you click **OK**, Opus will proceed to make your requested changes to the selected files and folders. All files and folders in the current file display that were selected when you invoked the **Change Attributes & Times** function will be modified. The contents of sub-folders will also be modified if you have the **Make the same changes** option (above) turned on.

# Labels

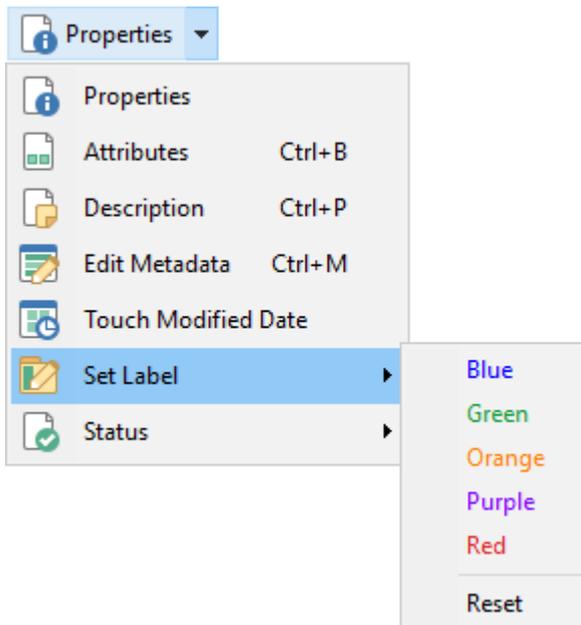
The **Labels** system lets you assign your own labels to files and folders. A label has a name, and can define colors and font styles that modify the appearance of labeled files. Labels can also be used to override the default icon image for a file or folder, or define a status icon that can be shown in a separate column to flag important files. They can also cause a file or folder to become "pinned" to the top of the file list.

By default Opus defines five labels - *Blue*, *Green*, *Orange*, *Purple* and *Red*, and four status icons - *Checked*, *Flagged*, *Important* and *Pinned*. You can use these labels (what you interpret them to mean is up to you) or define your own. For example, you could define a label called *Really Important* that displays files in red italics.



Labels are defined through the [Favorites and Recent / Labels](#) page in Preferences.

You can apply labels to files and folders on an ad-hoc basis, using the **Properties** drop-down menu on the default toolbar. The *Set Label* and *Status* sub-menus display a list of your defined labels and status icons, and lets you apply them automatically to selected files and folders.



To label a file, select it, and then choose the appropriate label from this drop-down. Depending on the label definition, the file will change color instantly to match the assigned label.

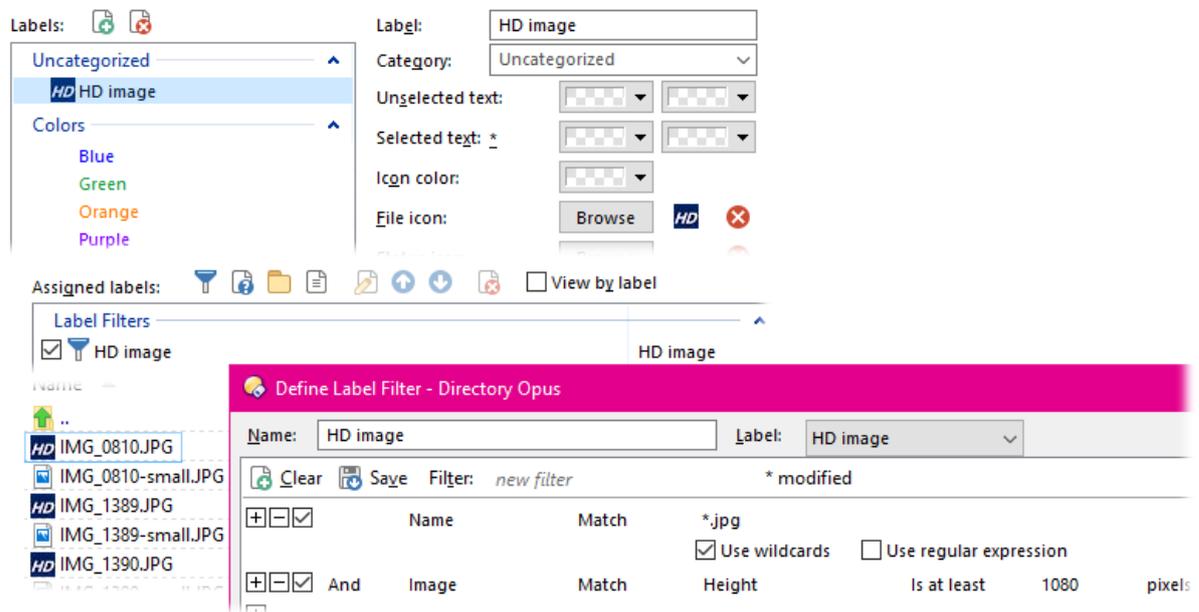
You can apply more than one label at a time to a file - that is, they can “stack” on top of each other. For example, you could have one label that colors the names of all JPEG files green, and another that bolds the filenames of all images that are 1920x1080 pixels or larger. Any 1920x1080 JPEG files would have their filenames shown as bolded green. Each label definition now has a “stop on match” flag which lets you prevent this behaviour on a per-label basis.

The **Label Assignments** page in Preferences lets you apply labels to files and folders using wildcard patterns. For example, you could cause all Word documents to be labeled *Green* by creating a wildcard label assignment for \*.doc files. Labels can also use [filters](#) to perform tests based on attributes other than file name or path.

By default explicitly applied labels override wildcard labels, but the Preferences setting *Favorites and Recent / Labels / Apply wildcard and label filters to explicitly labeled files and folder* lets wildcard labels stack on top of explicitly-assigned ones.

In the image below, you can see an example of a label filter that is used to override icons for particular files. A label has been created (called **HD Image**) that defines a custom icon. A label filter has also been created that matches JPG files (it could be used to match more filetypes if

desired) with a height of at least 1080 pixels. You can see in the Lister image at the bottom that the high resolution images are displayed with the custom icon, whereas the smaller images have the standard JPG icon.



There are two ways that Opus can store labels that are assigned to a specific file or folder (as opposed to a wildcard or filter-based label).

- On an NTFS-formatted drive (which is the most common type of file system these days) the label information is stored in an alternate data stream with the actual file itself. The advantage of this is that when you copy, move or rename the file, the label goes with it (if enabled on the [File Operations / Copy Attributes](#) page in Preferences).
- On other file systems, or if the option to store labels in the file system is turned off (on the [Labels](#) page in Preferences), Opus stores a list of the full filename and path of labeled files. That means that if you move a labeled file, the label will not go with it. You can view a list of all files labeled in this way from the **Label Assignments** page in Preferences, and use the controls there to clean up the labeled file list.

## TortoiseSVN status

For developers using the free TortoiseSVN source code control utility, Opus can optionally display a file's SVN status as an icon in the *Status* column. This is controlled by the *Folders / Folder Display / Show TortoiseSVN status icons in the Status column* option in Preferences. If turned on, Opus will automatically add the appropriate SVN status icon to any other label icons displayed in the *Status* column.

Status	Name ▲	Size	Type
	 Add to Archive Dialog.xml	2.84 KB	XML Docume
	 Adding a new Site.xml	5.78 KB	XML Docume
	 Adding Cover Art.xml	3.36 KB	XML Docume
	 Adding to Archives.xml	3.74 KB	XML Docume
	 Adding, Removing and Editing Clauses.xml	2.91 KB	XML Docume
	 Additional Functionality.xml	2.45 KB	XML Docume

Other than the status icon appearing larger and more distinctive than the usual icon overlay that Tortoise uses, this can help with the problem of limited icon overlays. Windows only allows a maximum of 15 icon overlays in the system, and Tortoise normally uses 8 or more of these for itself, crowding out other shell extensions. Using this option in Opus gets around the limit as the status is taken directly from Tortoise rather than via the icon overlay.

In Preferences there's also an option to disable the Tortoise icon overlay handler in Opus completely (as shown above – no point showing the same status icon twice), and an option to choose the SVN status icon set.

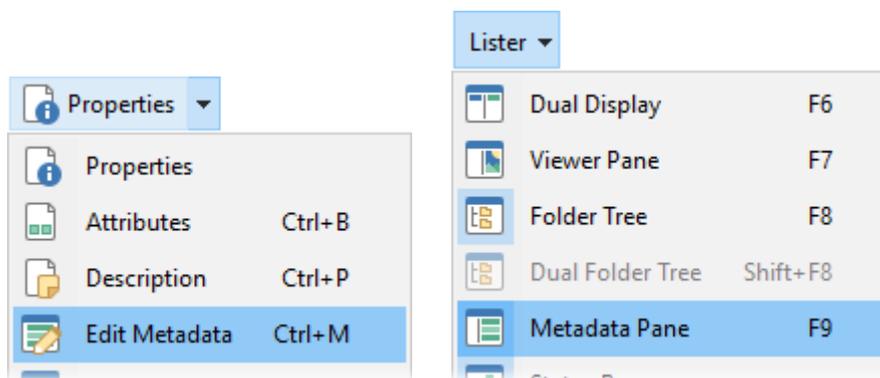
Note that this feature requires TortoiseSVN version 1.9.3 or greater.

## Editing Metadata

Metadata is information that is stored in a file that describes the file itself. For example, photos you take with your digital camera contain metadata like the date and time you took the picture, and the exposure settings used to take it.

Opus lets you edit file metadata in three ways; the easiest are with the [Metadata Pane](#), and with the **Set Metadata** dialog. The actual process of editing metadata is the same in both cases. The **Metadata Pane** lets you view metadata and make changes for selected files directly in the Lister or image viewer, whereas the **Set Metadata** dialog works like a typical file operation - it displays a dialog letting you choose the changes to make, and then when you click **OK** the changes will be applied to all selected files.

The third way of editing metadata is [programmatically](#) - you can use the internal [SetAttr META](#) command to create buttons or hotkeys that automatically modify the metadata of selected files.



To invoke the **Set Metadata** dialog, select the files you wish to edit, and choose the **Edit Metadata** command from the drop-down **Properties** menu. To display the **Metadata Pane**, select the **Metadata Pane** command from the **Lister Configuration** drop-down menu. In the standalone [image viewer](#), select **Edit Metadata** from the **Edit** menu. In the Lister, the **Metadata Pane** will update in real time as you select and deselect files in the file display.

Whichever method of editing you choose, the same display is used to both display the current metadata settings and to let you make changes.

☐ <b>Non-editable Properties</b>		
Color space:	RGB	
Dimensions:	4000 x 2672 x 24	
☐ <b>Document Properties</b>		
Authors:	Greg Perry	
Copyright:		
Description:	Vineyards	
Subject:		
Title:		
☐ <b>Picture Properties</b>		
Aperture:		
Camera make:	Panasonic	
Camera model:	DMC-G1	
Contrast:	Hard	
Creation software:	Ver.1.5	
Date digitized:	7:07:37 PM	30/11/20
Date taken:	7:07:37 PM	30/11/20

The list of metadata properties is divided into a number of categories. The categories and properties shown will change, depending on the type or types of files you have selected. Individual categories can be collapsed to hide their contents by clicking the small ☐ button to the left of the category name.

- **Non-editable Properties:** These are *intrinsic* properties of the files that are shown here for information only - they can't be edited. These are only shown in the **Metadata Pane**, not in the **Set Metadata** dialog.
- **Document Properties:** Properties that apply to documents (author, copyright, subject, etc.). Some non-document file types support some of these properties too - for example, an image file can have authors and a subject just like a Word document.
- **Picture Properties:** These are properties that apply to images. They are stored in **EXIF** format and so the image format must support the use of **EXIF data** (e.g. **JPEG**, **PNG**, **TIFF**).
- **Music Properties:** These are properties that apply to music files like **MP3** and **WMA**.
- **Video Properties:** These are properties that apply to video files like **AVI** and **WMV**.
- **Music & Movie Properties:** These are properties that apply to both music and video files. This category will only be shown if you have both music and video files selected at once.
- **Extended Properties:** These are properties that are nominally available for all file types, like **Comment**, **Rating** and **Tags**. If the selected file formats support those properties natively (for example, **JPEG** files do) then the information will be stored in the file itself. If the selected file formats don't support these properties (like **.txt** files, for example) then Opus will attempt to store the information in an **Alternate Data Stream** attached to the file. Because only NTFS file systems support the ADS system, these properties won't be available for files stored on other file systems. If the *Rating* column is displayed in the file display, you can edit a file's rating directly by clicking the stars (rather than going through the Metadata panel).
- **Standard Properties:** These are properties that are applicable to all files - attributes and timestamps. You can modify these properties through the metadata system as an alternative to using the **Change Attributes & Times** dialog. The two timestamp properties have a **Select operation** field that lets you enter a **time shift** string that will perform relative adjustments to the current value of the date fields.

You can edit the metadata of multiple files simultaneously simply by selecting multiple files at once (for the **Metadata Pane**), or by having multiple files selected when invoking the **Edit Metadata** command. Only properties that all selected files have in common can be edited. For example, if you select four image files at once, all the **Document** and **Picture Properties** fields will be shown as normal, but if you select four image files and one music file, only the few **Document** fields they have in common will be available.

The metadata display shows the current values, if any, of the various properties of the selected files. If multiple files are selected and they have different values for a given property, the indication *multiple values* is shown.



This indicates that the selected files did not all have the same initial value for that property. If you edit a property when multiple files are selected, the new value will be applied to all files, so (except in a few special cases) editing a field where *multiple values* is displayed will result in the loss of the individual properties for the selected files.

To edit a value in the metadata display, simply click it to invoke the editing controls. The type of control used will depend on the field being edited. For example, for a string this will be a text edit field, but for a property with only certain options to choose from this might be a drop-down list.



To accept changes you have made to a field, simply click out of it (or on another field). You can also move between fields using the keyboard - press the **Tab** key to move to the next field, or **Shift+Tab** to move to the previous one. You can press the **Enter** key to accept changes you have made to a field, or press **Escape** to undo changes to the current field.

When a property has been modified it is indicated with a red triangle to the left of the field name.



In the **Metadata Pane**, making one or more changes also enables the **Apply** and **Cancel** links in the pane's titlebar. Once you have finished editing the metadata for the selected files, click the

**Apply** link to save the changes. You can also click the **Cancel** link to discard any changes and reset all properties to their original values. The arrow buttons (← →) in the title let you quickly move through a list of files - click → to go to the next file and ← to go to the previous file. If you hold the **Shift** key down when you click these arrows, any changes you have made to the current file's metadata will be automatically saved.

In the **Set Metadata** dialog, once you have finished editing the metadata, click the **OK** button to apply the changes to the selected files.

Using drag-and-drop, you can copy certain combinations of properties from one file to one or many target files. When you are editing the metadata for one or more files in the metadata pane or the separate dialog, you can drag-and-drop another file over the properties list. If you drag with the left button, a default set of properties will be copied; dragging with the right mouse button displays a popup menu that lets you choose which properties you want to copy from the dropped file.



## Document Properties

The **Document Properties** fields that the metadata editor supports apply primarily to Office documents (Word, Excel, etc.). Some fields are supported for PDF files, and some document fields also apply to image and music files.

- **Authors:** This field lets you specify one or more authors of the document.



To specify multiple authors, separate them with a semi-colon (this method of supplying multiple values is used in a number of different metadata fields). Note that when this field is displayed for a music or a video file it is labelled as **Artists** instead of **Authors**, and will be grouped with the other **Music** or **Video Properties** instead of in a separate **Document Properties** category.

- **Category:** Specify a category for the document, for example "Documents from my manager".
- **Company:** Specify the name of the company that owns or authored the document.

- **Content Status:** The status of the document, for example "Draft" or "Final".
- **Content Type:** The type of the content, for example "Status Report".
- **Copyright:** Specify a copyright notice for the document.
- **Language:** Specify the primary language the document is in, for example "English".
- **Last saved by:** The name of the person who last saved the document. This is usually updated automatically when you save a document in Word, etc.
- **Manager:** In an office setting, the manager of the team or organisational unit that created the document.
- **Subject:** A description of the document's subject, for example the name of the project it is being written for.
- **Title:** The document's title.

## Picture Properties

The **Picture Properties** metadata fields apply to image files in formats that support [EXIF](#) tags - **JPEG**, **PNG**, **TIFF** and Photoshop (**PSD**) images are currently supported.

- **Aperture:** The aperture setting (expressed as an F-number) when the image was taken.
- **Camera make:** The make (manufacturer) of the camera.
- **Camera model:** The model of the camera that took the image.
- **Contrast:** The contrast setting when the image was taken.
- **Creation software:** A string identifying the software used to create or edit the image.
- **Date digitized:** The date the image was digitized. If the image was originally not in digital form (e.g. a slide you have scanned in) this would be the date it was scanned. You can specify a date and time using the controls, and you can also use the **Select operation** drop-down to perform one of several operations on the date value.
  - **Copy from date created:** This will set the value of the **Date digitized** field to the creation date of the file.
  - **Copy from date modified:** This will set the value of the **Date digitized** field to the last modified date of the file.
  - **Copy from date taken:** This will set the value of the **Date digitized** field to the value of the Date taken field (if any).

The **Select operation** field also lets you enter a [time shift](#) string that will perform relative adjustments to the current value of the date fields.

- **Date taken:** The date the image was taken. For digital camera images this is usually the same as **Date digitized**. You can specify a date and time using the controls, or use the **Select operation** field as described for the **Data digitized** field.
- **Digital zoom:** The digital zoom ratio used to record the image (if any). You can express this as a decimal value or a fraction, and can also select **Off** from the drop-down to specify that no digital zoom was used.
- **Exposure bias:** The camera's exposure bias setting when the image was recorded, specified in *stops*.
- **Exposure program:** The exposure program used to record the image (aperture priority, shutter priority, etc).

- **Exposure time:** The exposure time used to record the image, specified in seconds or fractions of a second. You can specify a whole number, a fraction (e.g. 1/5) or a decimal (e.g. 1.8).
- **F-number:** The aperture setting (expressed as an F-number) when the image was taken.
- **Flash:** Specifies whether the flash fired when the image was taken, and what special mode (red-eye reduction, etc.) was used if so.
- **Focal length:** The actual focal length of the lens, expressed in millimetres.
- **Focal length (35mm):** The focal length of the lens as a 35mm equivalent value (e.g. a digital camera with a lens focal length of 6mm may be equivalent to a 28mm focal length lens on a 35mm film camera).
- **GPS Altitude:** The altitude relative to mean sea level where the picture was taken, measured in metres. Specify a negative value if the image was taken below mean sea level.
- **GPS Latitude:** The latitude where the picture was taken, specified in degrees, minutes and seconds, in either the northern or southern hemisphere.
- **GPS Longitude:** The longitude where the picture was taken, specified in degrees, minutes and seconds, and either east or west of the prime meridian.
- **ISO speed:** The (equivalent) film speed when the image was taken, expressed using the ISO scale.
- **Metering mode:** The exposure metering mode used when the image was taken (average, spot, multi-spot, etc).
- **Rotation:** The rotation information stored in the image, usually set by the camera's orientation sensor. This reflects the orientation of the camera when the image was recorded. Select from 0, 90, 180 and 270 degrees.
- **Saturation:** The saturation of the image (normal, high or low).
- **Scene capture type:** The type of scene capture program used by the camera (standard, night, portrait, etc).
- **Sharpness:** The sharpness of the image (normal, hard or soft).
- **Shutter speed:** The speed of the shutter when the image was taken, expressed in seconds or fractions of a second. You can enter the time as a whole number, a fraction (e.g. 1/5) or a decimal (e.g. 1.8).
- **Subject distance:** The distance from the focal point to the subject, expressed in metres or fractions of a metre. You can enter the distance as a whole number, a fraction or a decimal. For example, 0.2 would correspond to 20 centimetres.
- **White balance:** The camera's white balance setting when the image was taken.

## Time Shifting

The various timestamp properties (**Date created** and **Date modified** in the **Standard Properties** category, and **Date digitized** and **Date taken** in the [Picture Properties](#) category) let you enter what's known as a **time shift** string, which lets you perform relative changes to the existing values of these fields.

The format of the time shift string is either:

```
<date-shift> <time-shift>
```

when you want to adjust both date and time, or when you only want to modify the time:

`<time-shift>`

When one string is given it's assumed to be the `<time-shift>` string, which means you can't adjust just the date without specifying a time value as well (although the time value given can be 0 which means the time won't actually change).

The `<date-shift>` portion has three allowable formats:

- `[+]Y:M:D` - year:month:day, with an optional positive or negative modifier
- `[+]M:D` - month:day, with an optional positive or negative modifier
- `[+]D` - day, with an optional positive or negative modifier

The `<time-shift>` portion has three allowable formats as well:

- `[+]H:M:S` - hours:minutes:seconds, with an optional positive or negative modifier
- `[+]H:M` - hours:minutes, with an optional positive or negative modifier
- `[+]H` - hours, with an optional positive or negative modifier

What this means is that you can add or subtract years, months, days, hours, minutes and seconds to or from the current date value. For example, say you had just taken 100 digital photos, when you suddenly realize you'd forgotten to adjust your camera's clock for the beginning of daylight savings. You could fix this by specifying `+1` (or just `1`, since `+` is the default) for the time shift string. By the above rules,

- Only one string (`+1`) is supplied, which means it's the `<time-shift>` value
- Only one number (`1`) is supplied, which means it's the `H` value - for hour

This would add one hour to the current timestamp of the selected images.

Here are a few other examples:

- `-1:0:0` - subtract one year from the timestamps
- `+8:30` - add eight and a half hours to the timestamp

- **+3 -0:45** - add three days, subtract 45 minutes from the timestamp

## Music Properties

The **Music Properties** metadata fields apply to music files - the formats that Opus currently supports are **MP3** and **WMA**. Several **Music** fields also apply to [video](#) files as well.

- **Album:** The title of the album that this music appears on.
- **Album artist:** The album artist. How you use this field is up to you - normally this is used for compilation albums where each individual track may have a different artist.
- **Artists:** The artist or artists responsible for this track. This is the same field (renamed) as the **Authors** [Document Properties](#) field. To specify multiple artists, separate them with a semi-colon (this method of supplying multiple values is used in a number of different metadata fields).
- **Author URL:** A URL associated with the artist (this might be the URL of the band's website, for example).
- **BPM:** The tempo of the music, in beats-per-minute.
- **Composers:** The composer or composers of the music. To specify multiple composers, separate them with a semi-colon.
- **Conductors:** The conductor or conductors of the music. To specify multiple conductors, separate them with a semi-colon.
- **Content group:** This field lets you group different tracks together, for example for classical music that has distinct movements. An MP3 of the last movement of Beethoven's Ninth Symphony might have "Sym. No. 9 in D" for **Content Group**, while **Title** would be "Presto-O Freunde, nicht diese Tone-Allegro assai".
- **Copyright:** A copyright message for the music.
- **Cover art:** This field lets you [add and edit album cover art](#) - images representing the "cover art" (CD cover, etc) of the music.
- **Disk number:** For multiple-disk sets, this lets you specify the disk number that this music appeared on.
- **Encoded by:** This field is generally used for the name of the person who encoded the track, or the name of the software application they used to do it.
- **Genre:** The genre of the music. You can select from the drop-down list of a number of common genres, or enter your own.
- **Initial key:** This is used to indicate the initial musical key of the track, e.g. **G#m**.
- **Mood:** Use this to specify the mood of the music (e.g. "background music", "party", etc.)
- **Original artists:** The original artists of the track. This can be used to specify the name of the original artist or artists in a cover version that was recorded by someone else. To specify multiple artists, separate them with a semi-colon.
- **Publisher:** The publisher of the music.
- **Release date:** The date the music was originally released.
- **Subtitle:** The **Subtitle** description is used to provide additional information relating to the **Title**, for example if the title was "Another Brick in the Wall, Part 2", the subtitle might be "Live in Berlin".
- **Title:** The title of the music.
- **Track number:** The track number of the music if it forms part of a multi-track work (i.e. an album). You can specify just a track number, or also specify the total number of tracks (e.g. the track number could be **5**, or it could be **5 / 15**).

▶ **Track number:**      01 / 11       **Increment**

The **Increment** option in this field lets you assign automatically incrementing track numbers when you have multiple files selected. The track number you enter will be assigned to the first selected file, and the number will be incremented once for each additional file. You can also specify that you want to the track number zero-padded by entering a leading zero in the **Track number** field.

- **Year:** The year the music was originally released.

### ***Adding Cover Art***

The **Cover art** metadata field shown for music files lets you add, remove and extract cover art from music files.

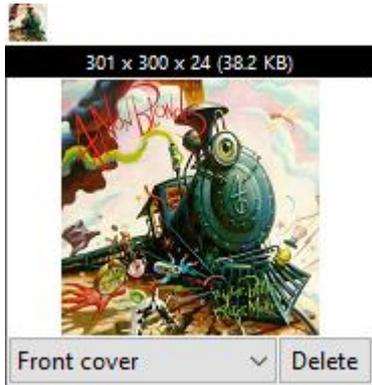


The **Cover art** field displays a (tiny) thumbnail of any cover art images currently in the selected file. Music files can store more than one image; you can theoretically add an unlimited number of images, although it would be very unusual to have more than one or two.

To add a new cover art image, click the **Add** button and select the image file you want to use. If you have an image on the clipboard you can also click the **Paste** button to add it directly instead of saving it to a file first.

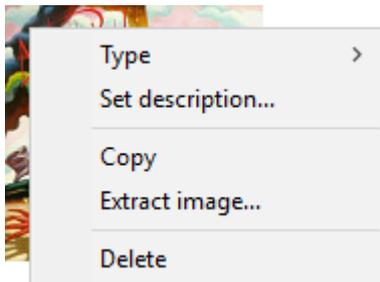
Once the image has been added you can move the mouse over it as described above and set the type and description of the image as required. Cover art images can only be **JPG** or **PNG** format, so if you choose an image in another format Opus will convert it to **JPG** automatically.

Moving the mouse over one of the tiny cover art thumbnails displays a larger image in a pop-up window. From the keyboard you can bring this up by putting the focus on the thumbnail (with the **Tab** key) and pressing the **Space** key.



The pop-up displays the resolution and size of the image at the top. The drop-down at the bottom lets you change the "type" of the image. Cover art can be assigned a type to represent its role in the real world as cover art ("back cover", "front cover", etc). The type is used when choosing a thumbnail to show for the music file in a Lister - if there are multiple image types defined, usually the "front cover" is the one that's shown for the file's thumbnail.

The **Delete** button at the bottom lets you remove the cover art image.



You can also right-click on the pop-up window to display a context menu with additional actions:

- **Type:** The **Type** sub-menu lets you change the image type (as an alternative to the drop-down at the bottom of the image).
- **Set description:** This command lets you assign a description string to the image.
- **Copy:** The **Copy** command will copy the image data to the clipboard - from there you can paste it into a paint program, or into a Lister to create a new image file.
- **Extract image:** Extracts the image to a disk file.
- **Delete:** Removes the cover art image.

## Video Properties

The **Video Properties** metadata properties apply to video files - the formats that Opus currently supports are **AVI** and **WMV**. Several of these properties also apply to [music](#) files (and their use with a video file would really only make sense for a music video).

- **Artists:** The artist or artists responsible for this video. This is the same field (renamed) as the **Authors** [Document Properties](#) field. To specify multiple artists, separate them with a semi-colon (this method of supplying multiple values is used in a number of different metadata fields).
- **Author URL:** A URL associated with the artist (for a music video, this might be the URL of the band's website, for example).
- **BPM:** The tempo of the music, in beats-per-minute. This would mainly be applicable to music videos.
- **Composers:** The composer or composers of the music. To specify multiple composers, separate them with a semi-colon.
- **Conductors:** The conductor or conductors of the music. To specify multiple conductors, separate them with a semi-colon.
- **Content group:** This field lets you group different videos together, for example for classical music that has distinct movements. An music video of the last movement of Beethoven's Ninth Symphony might have "Sym. No. 9 in D" for **Content Group**, while **Title** would be "Presto-O Freunde, nicht diese Tone-Allegro assai".
- **Copyright:** A copyright message for the video.
- **Directors:** The director or directors of the video. To specify multiple directors, separate them with a semi-colon.
- **Encoded by:** This field is generally used for the name of the person who encoded the video, or the name of the software application they used to do it.
- **Genre:** The genre of the video or music video. You can select from the drop-down list of a number of common genres, or enter your own.
- **Mood:** Use this to specify the mood of the video or music video (e.g. "background music", "party", etc.)
- **Producers:** The producer or producers of the video. To specify multiple producers, separate them with a semi-colon.
- **Publisher:** The publisher of the video.
- **Subtitle:** The **Subtitle** description is used to provide additional information relating to the **Title**, for example if the title was "Another Brick in the Wall, Part 2", the subtitle might be "Live in Berlin".
- **Title:** The title of the video.
- **Writers:** The writer or writers of the video. To specify multiple writers, separate them with a semi-colon.
- **Year:** The year the video was released.

## Extended Properties

The extended metadata properties are properties that theoretically can apply to any file. If the selected file formats support those properties natively (for example, **JPEG** files do) then the information will be stored in the file itself. If the selected file formats don't support these properties (like **.txt** files, for example) then Opus will attempt to store the information in an

[Alternate Data Stream](#) attached to the file. Because only NTFS file systems support the ADS system, these properties won't be available for files stored on other file systems.

- **Comment:** Specify a comment for the file. If the file format supports a comment field natively (and Opus supports metadata for that file format), then the comment will be stored in the file itself. If not, Opus will use an ADS stream to store the comment. You can also choose to use the **description** comments system - this lets you save comments for all file systems, but results in an additional data file being placed in the same folder. See the [File Descriptions](#) page for more information about file comments.
- **Tags:** Specify tags, or keywords, for the file.

Tags:                    cat; kitten; feline

To specify multiple tags, separate them with a semi-colon.

- **Rating:** This lets you assign your own "star rating" to the file. You can use this to keep track of your favorite videos, or rate songs in order of preference, etc.

Rating:                    ★★★★★ ✕

You can select a rating from one star to five stars, by clicking the desired star. Click the little **X** button to clear the rating.

## Programmatic setting of Metadata

Using the [SetAttr](#) **META** command it is possible to automate changes to file metadata.



For example, you could set up a series of buttons to apply different ratings to files.

This menu was created from six different commands, each of which applies a different rating to selected files.

- The command for *Clear rating* is **SetAttr META rating:0**
- The command for *1 star* is **SetAttr META rating:1**
- The command for *2 stars* is **SetAttr META rating:2**

and so on.

The general template for this command is:

**SetAttr META *keyword*:<value> *keyword*:<value> ...**

You can specify as many **keyword:**<value> pairs on the command line as you like. The keyword must be one of the metadata keywords listed on the [Keywords for SetAttr META](#) page. <value> will vary depending on the field you are setting.

If the <value> part is missing, the specified keyword will be cleared. For example,

```
SetAttr META album
```

It is important to remember that if <value> contains any embedded spaces, the entire **keyword:**<value> pair must be enclosed with quotes. For example,

```
SetAttr META "album:Dark Side Of The Moon" "albumartist:Pink Floyd"
```

The specified keyword can also contain [wildcards](#), to apply the same value to multiple metadata fields at once. You could, for instance, set the **artist**, **albumartist** and **origartist** fields all at once with **\*artist**. You can also use this to clear all metadata fields:

```
SetAttr META *
```

Some metadata values support more complex data than a simple string. For example, you can apply an offset to the EXIF *date taken* field, to adjust the current value rather than replacing it completely. To subtract one hour, the command would be:

```
SetAttr META datetaken:-1
```

See the [Keywords for SetAttr META](#) page for a full list of keywords and the values they can accept.

You can also copy certain combinations of metadata from a source file to the target files, using the **copyfrom** keyword. The template for this is:

```
SetAttr META copyfrom:<type>,<filename>
```

<type> is a numeric value that represents which type of properties should be copied. The available values for <type> are:

Type	Category	Copies
1	Music	Artist, Album, Year, Genre
2	Music	All except Title and Track
3	Music	All
4	Documents	Author and Ownership
5	Documents	All except Title and Subject
6	Documents	All
7	Movie	Director, Producer, Writer
8	Movie	All except Title
9	Movie	All
10	Picture	Author and Ownership
11	Picture	GPS
12	Picture	GPS and Camera
13	Picture	Author, Ownership, Camera
14	Picture	Author, Ownership, GPS
15	Picture	All

<file> is the source file to copy the specified metadata from. Remember that if the filename contains spaces, the **keyword:**<value> pair must be enclosed with quotes. For example,

**SetATTR META "copyfrom:8,/myvideos/my home movie1.avi"**

# File Descriptions

File Descriptions are a user-defined string that can be assigned to a file or folder, often used to keep notes about the file contents. Unfortunately Windows does not provide a standardised system for adding a user-defined description to files and folders. Opus does allow you to do this, and provides two different mechanisms for storing these descriptions:

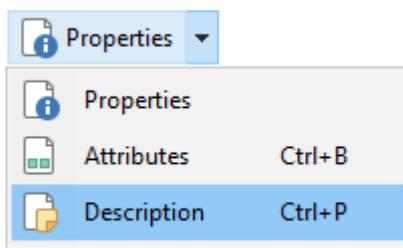
- The *descript.ion* system is an old, pseudo-standard, that saves all the descriptions for the items in a folder to a separate file (called "descript.ion") in the same folder. You can enable support for this system using the option on the [Folders / Folder Behaviour](#) page in Preferences.
- The *NTFS comments* system uses the [Alternate Data Stream](#) feature of the NTFS file system to store the description for a file in a separate data stream attached to the file.

Each of these two systems have their own advantages and disadvantages. The *descript.ion* system may be supported by other programs - for example, some image viewing tools support it, and you may want to enable this in Opus so that descriptions added by one program can be seen in the other. The disadvantages of *descript.ion* are that it is less efficient - every time a description is read or written, the *descript.ion* file has to be opened and parsed. The *descript.ion* files can also clutter up your file listings, although Opus does give you the ability to hide them if desired.

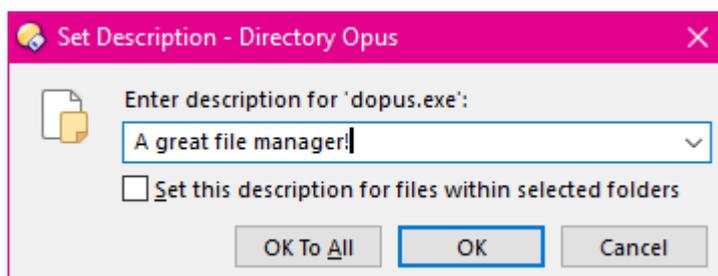
Using the *NTFS comments* system is more efficient than *descript.ion* and as alternate data streams are not normally visible, it can give a more elegant and unified feel to using file comments. The main disadvantage is that they are only supported on NTFS-formatted drives - if you're using another file system like FAT32, this system doesn't work.

Whichever system you elect to use, you can assign a description to a file in two ways:

- Using the **Description** command in the **Properties** drop-down on the default toolbar.



You can also press **Ctrl+P**. This displays the Set Description dialog, which displays the current description (if any) and lets you clear or edit it.



The **Set this description for files within selected folders** switch causes the function to operate recursively - all files inside any selected folders will be given the same description.

- Using the [Metadata Pane](#) or the [Set Metadata](#) dialog. These let you edit the description for selected files using the **Comment** field in the **Extended Properties** category.



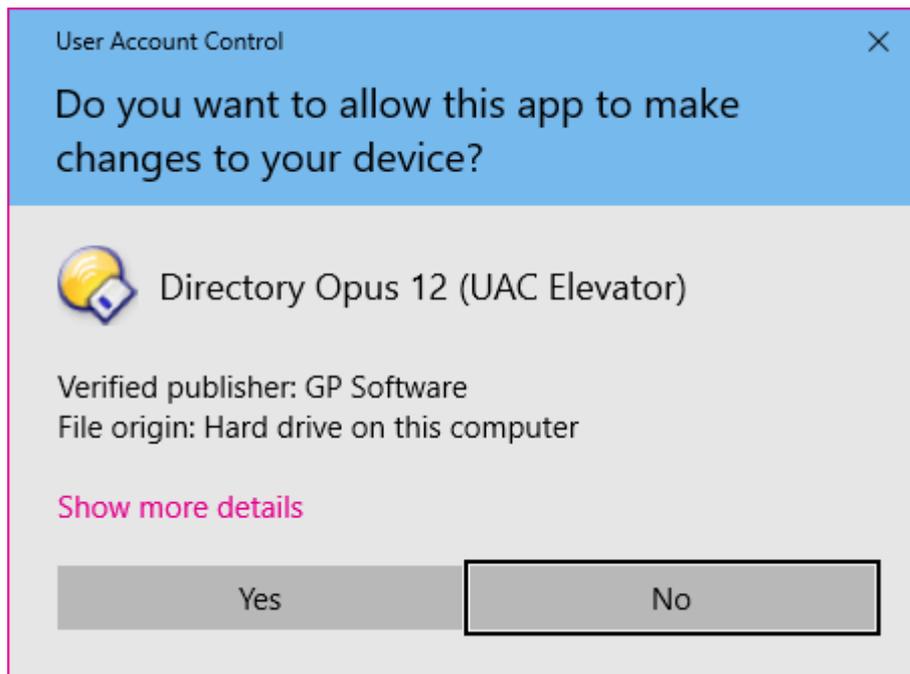
See the [Editing Metadata](#) page for more information on using the metadata editor.

By default Opus does not preserve file descriptions when you copy a file, as (particularly when using *description*) this can slow down the file copy process. The **Preserve the descriptions of copied files** option on the [File Operations / Copy Attributes](#) page in Preferences lets you enable this if desired.



## UAC and Administrator Mode

UAC ([User Account Control](#)) is a security feature in Vista and above. It aims to make it much more practical to run as a standard user than in Windows XP. When UAC is enabled (which it is, by default), users and the programs they launch will have standard user privileges by default. This means that important operating system locations (like the *Windows* folder) and registry keys are protected against accidental or malicious modification - in order to make changes to these locations, you need to *elevate* the task by means of a UAC prompt.



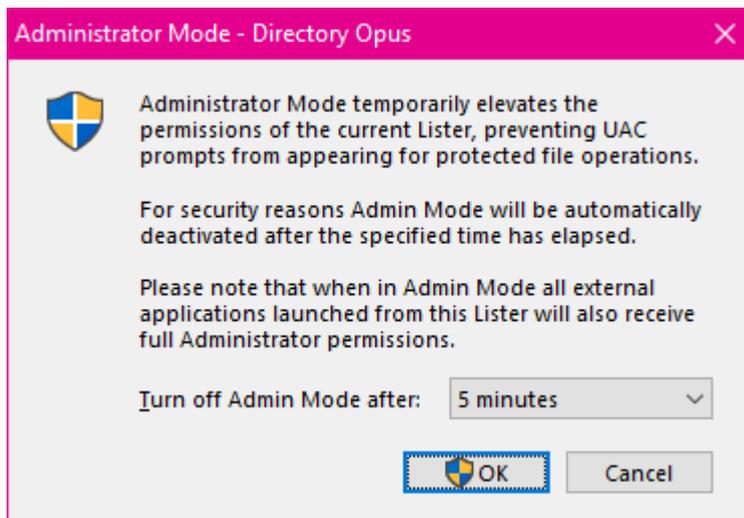
Unfortunately Microsoft somewhat [botched](#) the implementation of UAC in Explorer. Vista was widely derided for the sheer number of UAC prompts you needed to click through to perform routine or mundane tasks. Simply creating a folder could result in four individual prompts, leading many people to simply turn UAC off altogether.

In Windows 7, Microsoft over-reacted to the criticism and have dramatically toned down UAC - unfortunately to the detriment of your system security. The default UAC settings (which use a [white-list](#) to prevent any UAC prompts at all for Explorer and other Windows components) reduce it to the mere appearance of a security system - and at the same time, disadvantaging other third-party tools that don't have the ability to be white-listed.

We have always viewed UAC as a commendable (if poorly-executed) attempt to improve the security of Windows, and we recommend both a) leaving it turned on, and (in Windows 7 or above) b) changing the level in *User Account Control Settings* to **Always notify**. Opus natively supports UAC, in a much more sensible way than Explorer, and you won't find performing administrative tasks in Opus difficult or irritating at all.

- When performing a file operation in a protected location, like *C:\Program Files*, Opus will display a UAC prompt as in the screenshot above. It tries to be smart about this - it will only prompt once per function, unlike Explorer which can sometimes prompt up to four times for the one operation.
- If you are performing multiple administrative tasks, you can use the **Administrator Mode** function to temporarily elevate the current Lister.

To activate Administrator mode, click the  **Admin** button on the default toolbar. Opus will display a dialog that lets you specify a timeout after which Administrator mode is automatically deactivated. This is for security reasons, as it avoids the risk of accidentally leaving a Lister elevated longer than necessary.



Select the timeout, and click OK. Opus will then display a UAC prompt to elevate, and after that the Lister itself will remain elevated for the specified period (or until you turn the mode off manually). The title bar of the Lister window changes to indicate this:



Any operations you perform in an Administrator mode Lister will be automatically elevated - no further UAC prompts will need to be shown. Note though that any programs you launch from that Lister will also be elevated.

- You can configure a toolbar button or hotkey that launches an external program to launch the program elevated using the [@admin command modifier](#).

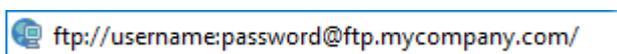


# FTP

Directory Opus has a built-in [FTP](#) client that lets you access remote FTP servers as if they were local folders. You can copy, rename and delete files, create folders, and even view pictures as if they were files stored on your hard drive instead of on a remote FTP site.

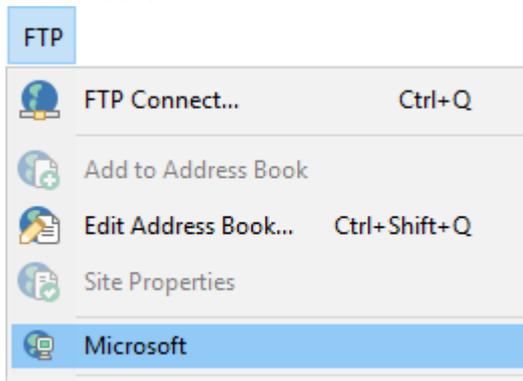
There are a number of ways to connect to an FTP site in Opus:

- You can make an ad-hoc connection using the location field. For example, to connect anonymously to the Microsoft FTP server, click in the location field and enter <ftp://ftp.microsoft.com/> and press the **Enter** key. You can also use this method to connect with a username and password, for example:



ftp://username:password@ftp.mycompany.com/

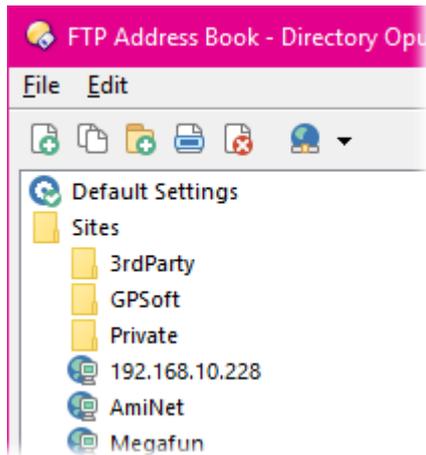
- You can use the [FTP Connect](#) dialog to enter the site and user details and connect to the site.
- You can create and maintain an [Address Book](#) of FTP sites. When a site is listed in the address book, you can connect to it using the list displayed in the FTP drop-down menu on the toolbar.



Your FTP Address Book entries can also be displayed in the [folder tree](#), and in the [Jump List](#) for Windows 7 users.

# FTP Address Book

The FTP Address Book lets you maintain a collection of FTP sites, each with their own login details and other configuration. If you have lots of sites to keep track of you can also organise them into folders. To display the FTP Address Book, select the Edit Address Book command from the FTP menu on the default toolbar, or from the right-click context menu for the FTP item in the folder tree.



The left-hand side of the FTP Address Book dialog displays your address book entries. The toolbar at the top is used to add or edit the entries; the **New FTP Site** (🌐+) button [adds a new site to the list](#), **Duplicate FTP Site** (📄) duplicates an existing site, the **New Folder** (📁+) button creates a folder that you can use to organise your sites, the **Rename** (📄) button renames an existing site or folder, and the **Delete** (🗑️) button deletes the selected site or folder.

The drop-down **Connect** (🌐 ▾) button lets you establish a connection to the selected site in the address book. There are many other ways to connect to a site; this is mainly just provided as a convenience for when you are editing the address book.

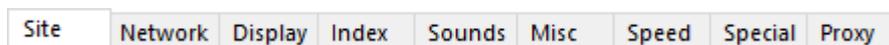
The **File** menu at the top of the dialog contains the following commands:

- **Import:** Import a previously exported address book file. You can use the export/import functions to copy your address book from one machine to another. When you import an address book the options are:
  - **Import to a new folder:** The contents of the imported address book are placed in a new folder in your existing address book.
  - **Merge with existing entries:** The contents of the imported file will be merged. Any site entries that clash in your existing address book will be overwritten. You might use this option when updating a laptop with new entries you created on your desktop, for example.
  - **Replace all existing entries:** Your existing address book will be discarded completely and the imported file will replace it.
- **Export:** Export your FTP address book to a file. You can then use the import function to import it on another machine. The options when exporting are:

- **Export all entries:** Your whole address book is exported.
- **Export selected item only:** If you select a folder or site from the address book before running the **Export** command, you can export just that folder or site.
- **Remove FTP passwords from exported file:** If you turn this option on, any stored passwords will be omitted from the exported file. You might want to do this if you are sharing a folder of FTP sites with a work colleague; they can import the site configurations and put their own passwords in on their machine, and you won't be revealing your passwords to them.
- **New FTP Site:** Create a new FTP site (same as using the  button).
- **Duplicate FTP Site:** Duplicate the selected FTP site (same as using the  button).
- **New Folder:** Create a new folder in the address book (same as using the  button).
- **Create Shortcut:** This command creates a shortcut to the selected FTP site and places it on your desktop. Whether double-clicking the shortcut will open the site in Opus or not depends on whether Opus is configured as the system [default FTP handler](#). You can also create a shortcut to a site using drag and drop.
- **Rename:** Rename the selected folder or site (same as using the  button).
- **Delete:** Delete the selected folder or site (same as using the  button).

The **Edit** menu contains the standard **Copy / Cut / Paste** commands; you can use these to copy the settings from one FTP site to another, or to move sites between folders. You can also re-arrange the address book using standard drag and drop techniques. All sites and folders listed in the address book also have a context menu accessed by clicking them with the right mouse button - this provides another way of accessing the above commands.

The special [Default Settings](#) entry in the address book is not a real site; instead it lets you configure default settings that apply to all FTP sites unless the site entries override them. The configuration for each site is broken into a number of pages, each with one or more sections, and the individual sections for a site can be set to use the defaults or to override them at the site level.



The pages for each site entry are:

- [Site](#): Defines the name, host, user and password details for the site. For the **Default Settings** item, the **Site** page is replaced by the **Global** page.
- [Network](#): Various options to do with connecting to the site.
- [Display](#): Defines which messages from the site are displayed and whether logging is enabled.
- [Index](#): Controls whether each directory's index file is to be downloaded (if it exists).
- [Sounds](#): Lets you configure a number of event-triggered sounds; for example, you can have a sound play when a connection times out or an error occurs.
- [Misc](#): Miscellaneous settings to do with server features and communication.
- [Speed](#): Lets you choose to limit upload and/or download speed if desired.

- [Special](#): Various advanced settings for server-specific features.
- [Proxy](#): Lets you configure a proxy server to use when connecting.

## Default Settings

The **Default Settings** item in the [FTP Address Book](#) is a special entry that lets you configure default parameters that apply to all your FTP sites. Individual sites can then override these parameters if desired.

When the **Default Settings** item is selected the **Site** tab changes to the **Global** tab, which lets you configure several global options that affect FTP in general.

- **Anonymous password**: This is the password used when connecting to an anonymous FTP site; some sites have a policy that you should use your own email address as the anonymous password, so you may want to change this setting.
- **Global password**: This lets you configure a global password for the FTP address book. You can enter this password to reveal the details of individual site passwords, which are normally hidden. This defaults to **password** which obviously isn't very secure - we recommend you change this.
- **Save password for newly added sites**: If this option is enabled and you add a new FTP connection to the address book using the **Add to Address Book** command in the **FTP** menu, Opus will automatically save the password to the address book entry.
- **Cache site passwords**: This lets you enable or disable the caching of site passwords in memory, and how long they are cached for. If you don't store your passwords in the FTP address book, but instead prefer to enter them manually when connecting to a site, this option is useful as it can prevent you being repeatedly asked for the same password over and over again.
- **Show site passwords in plain text**: If this option is turned on, the password field on the **Site** page for individual sites will display the stored password in plain text instead of masking it. You will need to enter the *global password* before this option can be enabled.
- **Remember Quick Connect passwords**: When you connect to an FTP site using the [FTP Connect](#) command, Opus normally stores information about the last connection so that it can be quickly recalled next time you use the **FTP Connect** command. If you turn off this option Opus will not store the password for the last connection.

## Site Page

The **Site** page is where you specify the basic configuration for a site entry in the [FTP Address Book](#).

- **Site name:** This specifies the name of the site as shown in the address book.
- **Comment:** This lets you specify a comment for the site entry; as well as being displayed here, it is displayed in the **Description** column in the file display if you select the FTP folder in the tree.
- **Connection:** This is how you specify the connection type for the FTP site. The available options are:
  - **Standard Connection:** A normal, unencrypted [FTP](#) connection.
  - **Secure TLS Explicit:** An encrypted (secure) FTP connection. This uses [Explicit FTPS](#) over TLS.
  - **Secure SSL Implicit:** An encrypted (secure) FTP connection. This uses [Implicit FTPS](#) over SSL.
  - **Secure SFTP via SSH:** An encrypted (secure) FTP connection. This uses [SFTP via the SSH](#) protocol.

Please note that the secure FTP options require the purchase of the optional *Advanced FTP* feature.

- **Host address:** The host address of the FTP site. This can be specified as either its domain name (e.g. [ftp.microsoft.com](#)) or its IPv4 address (e.g. **64.4.17.175**).
- **Port:** The port that the remote FTP server listens on. The default ports (which are the most commonly used) are **21** for standard FTP and explicit FTPS, **990** for implicit FTPS and **22** for SFTP via SSH. You can change the port number if needed.
- **Anonymous login:** Turn this option on if you wish to make an anonymous connection to the site (if the site allows it). If you make an anonymous connection you don't need a username and password to login. The **Anonymous password** specified in the [Default Settings](#) will be used when making the connection.
- **User name:** If not using an anonymous login, specify your user name (login name) for the FTP site here.
- **Authentication:** Use this to select the authentication method if not using an anonymous login. You can choose from the following options:
  - **Ask for password:** Select this if you want Opus to ask you for your password whenever you connect to the site.
  - **Use stored password:** You can enter your password in the **Password** field below and Opus will authenticate connections automatically. Note that the system administrator can disable the storing of passwords; see the bottom of this page for details.
  - **Use private key file:** For SFTP connections you can specify a private key file that's used to authenticate. If the key file has a passphrase you will be prompted for the passphrase during the connection process and given the option to save it for the future.
- **Password:** As an alternative to the above option, enter your password for the FTP site here if you want Opus to remember it in the configuration for the site. The password will be masked when displayed unless the **Show site passwords in plain text** option is turned on on the [Default Settings](#) page.
- **Private key file:** For SFTP connections, you can set the **Authentication** method to **Use private key file**, and then use this field to specify the filename of the key file to use.
- **Initial directory:** If you specify a directory path here, Opus will attempt to automatically change directory to this folder whenever it connects to the site. If you leave this empty the starting directory on the site will

be defined by the remote FTP server. If you turn on the **Keep last directory** option on the [Network](#) page, Opus will automatically update this field with the current directory when you disconnect from the site.

- **Adjust directory and file dates for site time zone:** Set this to have file timestamps automatically adjusted to compensate for the timezone of the remote FTP server. If turned on you must select the timezone of the remote server from the drop-down list. You can also turn on the **Adjust for daylight saving changes** option to have Opus automatically compensate for daylight saving time (summer time) when adjusting timestamps.

The first entry in the drop-down list is **Automatic (Serv-U)**. This is a special option for when the remote FTP site is running the *Serv-U* software, which has special provisions for timezone handling.

For security in a corporate environment, it's possible to prevent Opus from saving FTP passwords, using an administrator-only registry setting. To do this, set the **PreventSaveFTPCredentials** value under **HKEY\_LOCAL\_MACHINE\Software\GPSSoftware\Directory Opus** to **DWORD:1**.

## Network Page

The **Network** page is where you specify various connection-related options for a site entry in the [FTP Address Book](#).

The three sections this page is divided into (**Connection**, **Reconnect** and **Site-to-Site**) can all be individually enabled or disabled for any given site. When a section is set to **Use defaults**, the settings for that section come from the equivalent page in the [Default Settings](#).

The **Connection** section specifies various connection-related options for the site:

- **Allow special directory names beginning with space:** Some FTP servers allow files and folders to be made inaccessible if they begin with a space character. With this option enabled Opus will compensate for leading spaces in file names and let you access them.
- **Keep last directory:** If this option is turned on, Opus will update the **Initial directory** field on the [Site](#) page with the current directory when you disconnect from the site. This lets you return to your most recently visited location the next time you reconnect to the site.
- **Rescan directory after change:** After you have copied files to a remote FTP site, it's possible for the idea Opus has of the remote directory contents to get out of sync with the server. This can especially be the case if the remote site doesn't support the setting of file timestamps. If you turn this option on, Opus will automatically refresh the directory listing whenever you copy a file to the server.
- **Use PASV (passive) mode:** When Opus initiates a file transfer to or from a remote server, a separate data connection is established to the site. This can be established in [two ways](#). *Active* (PORT) mode is where Opus tells the remote site to connect to a specific data port on your machine - the remote server connects to you for the data transfer. *Passive* (PASV) mode is where Opus asks the server for a port to connect to - in this case, Opus connects to the server for the data transfer. Passive mode is often needed when you are behind a firewall, or your local network is using [NAT](#) (network address translation) - any time the remote server may not be able to establish a direct connection to you, you should use passive mode.

- **Timeout:** This lets you modify the local network timeout for FTP transfers. You can try adjusting this if you experience excessive timeouts on active (PORT) connections. We recommend using PASV mode in most cases.

The **Reconnect** section specifies the behavior when automatically reconnecting a connection that has been lost:

- **Retry count** and **Delay:** If a connection attempt to the FTP server fails, this specifies how many times the connection will be retried, and how long between retries, before giving up and reporting an error.
- **Automatically reconnect if connection lost:** If the connection to a site is lost unexpectedly (e.g. due to a timeout at the server end, or a network glitch), Opus will try to automatically reconnect to the site if this option is on. If turned on Opus will display an error and you will need to attempt the reconnection manually.
- **Keep link alive** and **Delay:** This option lets you attempt to work around server timeouts by enabling a "keep alive" mode. If enabled, Opus will regularly send a dummy packet to the remote server, which should prevent the other end from timing out the connection. The **Delay** option lets you specify the delay between keep alive packets in seconds. Many FTP sites discourage keep-alive mechanisms; it's generally viewed as more polite to only stay connected to the site when needed, rather than indefinitely.

The **Site-to-Site** section lets you enable site-to-site transfers for the site. A site-to-site transfer can be used when you are copying files from one remote FTP site to another FTP site - if the remote servers support it, the file transfer can go direct from one site to the other without the data having to be relayed through your local computer. Not all servers support this behavior so if you find site-to-site transfers aren't working you can turn it off.

- **Attempt for uploads:** Attempts site-to-site transfers when uploading to this site from another FTP site.
- **Attempt for downloads:** Attempts site-to-site transfers when downloading to this site from another FTP site.

## Display Page

The **Display** page is where you specify which messages from an FTP server are displayed and whether logging is enabled for a site entry in the [FTP Address Book](#).

The two sections this page is divided into (**Display** and **Log**) can all be individually enabled or disabled for any given site. When a section is set to **Use defaults**, the settings for that section come from the equivalent page in the [Default Settings](#).

The **Display** section lets you control which site messages are displayed and how Opus interprets certain aspects of the directory listing.

- **Display progress windows:** This option causes Opus to display intermediate messages from the site during certain activities (login, change directory and folder listing).
- **Hide files beginning with '.':** Unix-based systems often have special files that begin with a dot character. If this option is enabled these files will be hidden from the list.
- **Show directory messages:** This option causes the "directory information" messages to be displayed in the log (if logging is enabled). These are messages that the server administrator can configure for individual folders on the server. If enabled Opus will display these in the log when you change directory.
- **Show startup messages:** This option causes the site's "welcome" message to be display in the log (if logging is enabled). This is a message that the server administrator can configure the site to display upon connection. If enabled Opus will display this in the log when you connect to the site.
- **Treat unknown links as:** Unix systems often use links to make files and folders appear to be located in multiple places. Depending on the software the FTP server is running it can sometimes be impossible for Opus to distinguish between links to files and links to folders. In these cases, the link will be interpreted according to this option. You can choose **automatic** (Opus makes its best guess), or specify **file** or **folder**.
- **Logical parent directory:** Directories on FTP servers are often accessed via links, so the path that Opus sees as the current directory may not be the absolute path to the folder. This can cause problems when going up the folder tree as the "true" parent of a folder may not be the "logical" parent (i.e. simply stripping off the last component in the path may not give you the path of the true parent folder). When this option is turned off, Opus will send an FTP **UP** command when going up in the folder tree - when turned on, Opus strips off the last component and sends an FTP **CD** command to change to the logical directory.

The **Log** section lets you control logging for the FTP site.

- **Enable log:** Enables logging for connections to this site. The basic log option shows commands sent to the FTP site and responses received.
- **Enable debug:** Enables debug logging - diagnostic information will be logged as well as the normal commands/responses. Use this when trying to diagnose problems with the connection.
- **Display log automatically upon connection:** This option causes the log for the site to be automatically displayed in the Lister when you connect to the site.

The log for an FTP site can be displayed using the **Display FTP Logs** command in the drop-down **FTP** menu, or from the **Logs** sub-menu in the **Help** menu on the toolbar. You can also use the **Display log automatically** option above to have the log automatically displayed whenever you connect to the site.

## Index Page

The **Index** page controls whether each directory's index file is to be downloaded for a site listed in the [FTP Address Book](#). The index file is an optional file that FTP server administrators can place in each folder on a site to provide descriptions for the files in that directory. If index downloading is enabled, whenever you change directory on an FTP site Opus will look for the index file, and download it in the background. Once the index file has been downloaded the file

display will be updated to show the file descriptions. The index is downloaded on a background thread so you don't have to wait for it to finish before you can access the site.

This page has two different forms. For the [Default Settings](#) entry you can configure global settings that help Opus recognise and interpret index files. For each individual site entry you can selectively enable index downloading.

The options that are common to both the **Default Settings** and individual sites are:

- **Enable index file downloading:** This option causes Opus to look for an index file whenever you change directories. Index files often have names like INDEX, FILES.BBS and 00\_INDEX.txt, but you can also configure your own filenames to look for (see below).
- **Download automatically if less than:** If an index is found, this option causes it to be downloaded automatically in the background. You can specify an upper size limit in KB, above which the index won't be downloaded automatically.
- **If index not downloaded automatically:** If an index is found, but it wasn't automatically downloaded (because the **Download automatically** option was off, or the index file was too large), this lets you choose whether Opus should then ignore the index, or ask you (via a dialog) if it should be downloaded.

The global options that are only available for the **Default Settings** entry are:

- **Match:** This lets you add additional filenames that Opus will recognise as index files. To add a filename to the list, enter it in the field below this list and click the + button. To remove a filename from the list, select it and click the - button. You can use [standard wildcards](#) for the filenames in this list.
- **Ignore:** This lets you configure filenames that are exceptions to those in the **Match** list. You would use this if you've specified a wildcard string but then wish to specifically except some filenames from being identified as indexes.
- **File start position:** For each custom entry (i.e. not one of the default items) in the **Match** list you can specify an offset within the index file that Opus skips to before it begins parsing the file. Normally this would be set to 0.

## Sounds Page

The **Sounds** page lets you enable sound effects to be played for various FTP-related events when connected to a site in the [FTP Address Book](#).

The events that you can assign sound effects to are:

- **Login success:** This sound is played whenever a successful connection to the site is established.
- **Login failure:** This sound is played when a connection to the site fails for some reason.
- **Copy success:** This sound is played whenever a file copy operation succeeds.
- **Copy failure:** This is played when a file copy operation fails for some reason.
- **Error:** This sound is played whenever any other error occurs (i.e. not one of the other error conditions with a specific sound).

- **Lose connection:** This sound is played whenever the connection to a site is lost.
- **Timeout:** This sound is played when a network timeout occurs.

Some of the sound effects also let you configure an **Activation Time** value. This lets you specify a minimum time threshold for the event. If the event occurs (either completes or fails, as appropriate) in less than the configured number of seconds, the sound won't be played. So you can for example have a sound played at the end of a long copy operation, but short file copies wouldn't play anything.

The actual sound files that are used for these events are configured in the [Miscellaneous / Sounds](#) page in [Preferences](#).

## Misc Page

The **Misc** page lets you configure various server and connection-related settings for a site entry in the [FTP Address Book](#).

The **Server Commands** section has the following options:

- **Use MLST:** This option specifies if Opus should attempt to use the new, more modern **MLST** command to retrieve directory listings. If the server doesn't support **MLST** it will fall back to the old **LIST** mechanism controlled by the options below.
- **Use custom LIST command:**
  - If this option is not enabled then Opus will attempt to determine the format of **LIST** command (used for retrieving the remote directory listing) based on the type of server it is connected to. If the remote server isn't recognized correctly, you can turn this option on and provide your own custom **LIST** command.
  - If this option is enabled you can enter a custom **LIST** command that will be used to retrieve directory listings from the FTP server. The value you provide here is the full command that is sent to the FTP server, including any arguments. For example, **LIST -alF** is a very common list command, but you might use this to change to just **LIST** with no arguments if the server doesn't support the **-alF** mode.
- **Use MDTM:** Some FTP servers support the **MDTM** command to change the timestamps on remote files. Opus uses this when copying files to the FTP server so that your original timestamps are preserved.
  - This is a three-way checkbox - you can choose from **on**, **off** or **automatic** mode. When in automatic mode, Opus will only attempt to modify file dates on servers it recognizes, even if the server claims to support **MDTM**. Set this checkbox to the **on** state to force the use of **MDTM** on all server types.
  - Although the FTP standard specifies that the **MDTM** command takes a timestamp specified in UTC, some FTP servers have misinterpreted the standard and expect the timestamp to be given in local time. The drop-down menu lets you change how Opus sends timestamps to the server. You

can select *Automatic* to have Opus attempt to determine the format automatically, or you can choose from *UTC* or *local time*.

- **Use FEAT:** This specifies whether the server supports the **FEAT** command, which is an FTP command that lets clients determine which other features a server supports. This is a three-way checkbox - you can choose from on ( **Use FEAT** ), off ( **Use FEAT** ) or automatic mode ( **Use FEAT** ). When in this third state, Opus will attempt to automatically determine if the command is supported.
- **Use RESUME:** This specifies whether the server supports the **RESUME** command, which is an FTP command that lets a previously started FTP file transfer be resumed from where it left off. This is also a three-way checkbox, with on, off and automatic settings.
- **Use UTF8:** This specifies whether the server supports sending directory listings using UTF8 character-encoding. Historically FTP was always an ASCII-based protocol; directory listings only supported filenames that could be encoded using plain 7-bit (or sometimes 8-bit) ASCII, which meant support for non-Western alphabets was very limited or completely unavailable. UTF8 is an 8-bit encoding mechanism that allows the full Unicode character set to be used, which means filenames in languages like Chinese or Arabic are possible. This is also a three-way checkbox, with on, off and automatic settings.

The **Transfer Mode** section lets you control how files are transferred to and from this FTP site. The FTP protocol allows files to be transferred either as **ASCII** (with the idea that they are plain text files - the FTP server will automatically convert line endings between LF and CR/LF depending on the computers involved in the transfer), or as **Binary** (files are transferred as-is, with no changes to the data). Normally you would want files transferred as **Binary**, but you can use this option to switch to **ASCII** mode, or also select **Automatic**.

In automatic mode, Opus will switch modes automatically based on the file extension of the file being transferred. You can configure which file extensions are treated as ASCII in this mode from the **Misc** page for the [Default Settings](#) entry. There you can configure a list of file extensions that will be recognised as plain-text files and transferred in ASCII mode.

## Speed Page

The **Speed** page lets you configure upload and download speed limits for a site entry in the [FTP Address Book](#).

You can configure upload and download speeds separately. When the **Limit Upload Speed** or **Limit Download Speed** options are enabled, Opus will limit the transfer speed to the specified maximum rate. This lets you stop FTP transfers from hogging all your available bandwidth when connected to fast servers. You can optionally choose a time when the speed limits are applied, so for example you could have the speed limited during business hours but allow transfers to run at full speed overnight.

For both uploads and downloads, the options available are:

- **Maximum speed:** Specify the maximum transfer rate in KB/s (kilobytes per second).
- **Scheduled:** Turn this on to have the speed limit only applied during certain hours.
  - **From:** If the **Scheduled** option is on, use this to specify the start of the scheduled time. For example, *9:00 AM*.

- **To:** Use this to specify the end of the scheduled time, for example *5:00 PM*.

## Special Page

The **Special** page lets you configure certain advanced options for an FTP site entry in the [FTP Address Book](#).

The available options are:

- **Compression:** The use of zlib compression can be enabled to improve FTP throughput. You can choose from *None* (no compression), *Zlib for LIST only* (directory listings are compressed, but not file transfers) and *Zlib for all transfers* (both directory listings and file transfers are compressed). Note that when transferring binary files such as **.zip** or **.jpg** or other files that are already compressed, zlib compression will have little to no effect and may even make the transfer slower! When using compression you can choose the level (how much compression is applied) from 1 to 9. The higher the level of compression the greater the potential size saving but the more CPU power is required.
- **SSL Data Channel - Use Clear Data:** When using secure FTP via SSL, you have the option to encrypt just the control connection (which is where your login details are transmitted), or both the control and data connections. There's usually no reason to encrypt the data transfer - it's normally just your username and password that you want encrypted, and encrypting the data can slow down file transfers. Turn on the **Use Clear Data** option to specify that you don't want the data connection encrypted.
- **VMS Settings:** Lets you configure several options specific to VMS servers.
  - **Display all file versions:** VMS filesystems may store multiple versions of each file. Turn this on to see all of the versions.
  - **Escape characters:** Some VMS servers add extra escape characters before spaces and dots. Turn this on to correct for the additional escape characters.
  - **Show file versions as description:** Show VMS file version numbers in the file display's Description column.
- **Limit data ports:** This lets you limit the range of data ports Opus will use for data connections when transferring files.

## Proxy Page

The **Proxy** page lets you configure Opus to use a proxy server when connecting to a site in the [FTP Address Book](#). A proxy server is an "intermediary" server that accepts a connection from one computer (the "client") and establishes an outgoing connection to the target server (the "host") on the client's behalf. They are often used in corporate networks to provide security or caching features. You should consult your network administrator if you're not sure what proxy settings you need to use.

Turn on the **Connect via FTP Proxy** to enable the use of a proxy server. When this is enabled, the options for the proxy are:

- **Proxy type:** Use this drop-down to specify the type of proxy server being used. Most proxy types specify the use of a standard FTP proxy server, and let you control how your username and password are sent to the server (there are several variants like **USER username@host**, **USER user@proxyuser@host**, etc). The **Custom Template** option lets you have complete control over the way your user details are sent. You can also choose to use a **SOCKS4** or **SOCKS5** proxy from this drop-down.
- **Proxy address:** Specify the host name or IPv4 address of the proxy server here.
- **Port:** Specify the port number that the proxy server listens on. For standard FTP proxies this is normally port **21**, and for SOCKS proxies this is normally **1080**.
- **Delimiter:** Specify the delimiter character used when sending your login details to the proxy server. This is normally the **@** character.
- **Proxy user name** and **Proxy password:** For some options in the **Proxy type** drop-down, the **Proxy user name** and **Proxy password** fields will appear. These let you supply different login credentials for the proxy server, instead of using the username and password for the remote FTP server.
- **SOCKS 5 Authentication:** When *SOCKS5* is chosen from the **Proxy type** drop-down, this option enables the display of the **Proxy user name** and **Proxy password** fields. Use this option if your SOCKS5 proxy requires authentication.

When **Custom Template** is selected from the **Proxy type** drop-down, you can fully control the connection string used to connect to the FTP proxy. This connection string is built from control codes that are used to insert your login details into the string. The codes you can use are:

- **%h:** Host name of the remote FTP site
- **%u:** User name for the FTP site
- **%p:** Password for the FTP site
- **%x:** User name for the proxy server
- **%y:** Password for the proxy server

You can specify separate templates for both the **USER** and **PASS** commands. See the examples given in the actual dialog for more information.

## Adding a new Site

To add a new FTP site entry in the [FTP Address Book](#), click the **New FTP Site**  button or right-click on the site list and choose the **New FTP Site** command. This displays the **Create New FTP Site** dialog, which lets you configure the basic parameters for the new FTP site.

The options available when creating a new site are:

- **Site name:** This specifies the name of the site as it will be shown in the address book.
- **Comment:** This lets you specify a comment for the site entry; as well as being displayed in the address book, it is displayed in the **Description** column in the file display if you select the FTP folder in the tree.
- **Connection:** This is how you specify the connection type for the FTP site. The available options are:
  - **Standard Connection:** A normal, unencrypted [FTP](#) connection.
  - **Secure TLS Explicit:** An encrypted (secure) FTP connection. This uses [Explicit FTPS](#) over TLS.
  - **Secure SSL Implicit:** An encrypted (secure) FTP connection. This uses [Implicit FTPS](#) over SSL.
  - **Secure SFTP via SSH:** An encrypted (secure) FTP connection. This uses [SFTP via the SSH](#) protocol.

Please note that the secure FTP options require the purchase of the optional *Advanced FTP* feature.

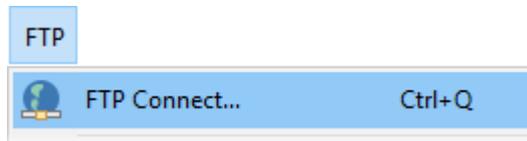
- **Host address:** The host address of the FTP site. This can be specified as either its domain name (e.g. **ftp.microsoft.com**) or its IPv4 address (e.g. **64.4.30.34**).
- **Port:** The port that the remote FTP server listens on. The default ports (which are the most commonly used) are **21** for standard FTP and explicit FTPS, **990** for implicit FTPS and **22** for SFTP via SSH. You can change the port number if needed.
- **Anonymous login:** Turn this option on if you wish to make an anonymous connection to the site (if the site allows it). If you make an anonymous connection you don't need a username and password to login.
- **User name:** If not using an anonymous login, specify your user name (login name) for the FTP site here.
- **Ask for password:** Set this checkbox if you want Opus to ask you for your password whenever you connect to the site. You might want this if you don't want to store your site passwords in the Opus configuration. This is a "tri-state" checkbox - it has the normal states of on () and off () , as well as a third state (). The third state means "use default" - the state of the global **Always ask for password for login** option on the FTP Address Book's [Default Settings](#) page will be used.
- **Password:** As an alternative to the above option, enter your password for the FTP site here if you want Opus to remember it in the configuration for the site. The password will be masked when displayed unless the **Show site passwords in plain text** option is turned on on the FTP Address Book's [Default Settings](#) page.
- **Initial directory:** If you specify a directory path here, Opus will attempt to automatically change directory to this folder whenever it connects to the site. If you leave this empty the starting directory on the site will be defined by the remote FTP server.
- **Adjust directory and file dates for site time zone:** Set this to have file timestamps automatically adjusted to compensate for the timezone of the remote FTP server. If turned on you must select the timezone of the remote server from the drop-down list. You can also turn on the **Adjust for daylight saving changes** option to have Opus automatically compensate for daylight saving time (summer time) when adjusting timestamps.

The first entry in the drop-down list is **Automatic (Serv-U)**. This is a special option for when the remote FTP site is running the *Serv-U* software, which has special provisions for timezone handling.

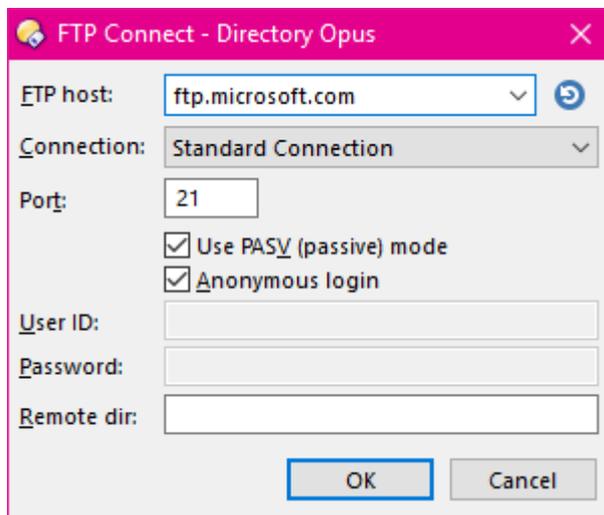
All these site settings (and others) can be changed through the address book once the site has been added.

# FTP Connect

The FTP Connect dialog lets you make a quick connection to an FTP site without going through the [FTP Address Book](#). To access this function, select the **FTP Connect** command from the FTP menu on the toolbar.



You can also right-click the FTP item in the tree and choose **FTP Connect** from the context menu.



The options available in the **FTP Connect** dialog are:

- **FTP host:** This is the host address (domain name or IPv4 address) of the FTP site to connect to. The drop-down attached to this control displays the contents of your [FTP Address Book](#), and selecting a site from here will populate the fields of the dialog with the information for that site. For example, you could use this to connect to a site from your address book with a different username to the one stored in the site entry.
- : This is the *recall* button. Clicking this button will recall the details of the last site you connected to through the **FTP Connect** dialog.
- **Connection:** Select the FTP connection type from the drop-down. The available options are:
  - **Standard Connection:** A normal, unencrypted [FTP](#) connection.
  - **Secure TLS Explicit:** An encrypted (secure) FTP connection. This uses [Explicit FTPS](#) over TLS.
  - **Secure SSL Implicit:** An encrypted (secure) FTP connection. This uses [Implicit FTPS](#) over SSL.
  - **Secure SFTP via SSH:** An encrypted (secure) FTP connection. This uses [SFTP via the SSH](#) protocol.

- **Port:** The port that the remote FTP server listens on. The default ports (which are the most commonly used) are **21** for standard FTP and explicit FTPS, **990** for implicit FTPS and **22** for SFTP via SSH. You can change the port number if needed.
- **Use PASV (passive) mode:** When Opus initiates a file transfer to or from a remote server, a separate data connection is established to the site. This can be established in [two ways](#). *Active* (PORT) mode is where Opus tells the remote site to connect to a specific data port on your machine - the remote server connects to you for the data transfer. *Passive* (PASV) mode is where Opus asks the server for a port to connect to - in this case, Opus connects to the server for the data transfer. Passive mode is often needed when you are behind a firewall, or your local network is using [NAT](#) (network address translation) - any time the remote server may not be able to establish a direct connection to you, you should use passive mode.

This is a "tri-state" checkbox - it has the normal states of on () and off () , as well as a third state (). In the third state the passive-mode setting will come from the site's address book entry (if a site was selected from the drop-down list) or from your default FTP settings, defined at the top of the FTP address book.

- **Anonymous login:** Turn this option on if you wish to make an anonymous connection to the site (if the site allows it). If you make an anonymous connection you don't need a username and password to login.
- **User name:** If not using an anonymous login, specify your user name (login name) for the FTP site here.
- **Password:** If not logging in anonymously, enter your password for the FTP site here.
- **Remote dir:** If you specify a directory path here, Opus will attempt to automatically change directory to this folder after connecting to the site. If you leave this empty the starting directory on the site will be defined by the remote FTP server.

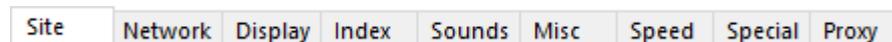
As well as the **FTP Connect** command, you can also make an ad-hoc connection to an FTP site by simply entering the URL in the location field. For example, to connect anonymously to the Microsoft FTP server, click in the location field and enter <ftp://ftp.microsoft.com/> and press the **Enter** key. You can also use this method to connect with a username and password, for example:

 <ftp://username:password@ftp.mycompany.com/>

## Site Properties

When you are displaying the contents of a remote FTP site in the file display, you can choose the **Site Properties** command from the **FTP** drop-down menu, or right-click on the background of the file display or on the Lister's status bar, and choose the **Site Properties** command to view connection properties for the site.

The **Site Properties** dialog has a number of tabs across the top that correspond with the pages in the [FTP Address Book](#).



Please see the various sections of the FTP Address Book documentation for information on the settings on these pages. At the bottom of the **Site Properties** dialog are two options that affect how changes you make through this dialog are applied:

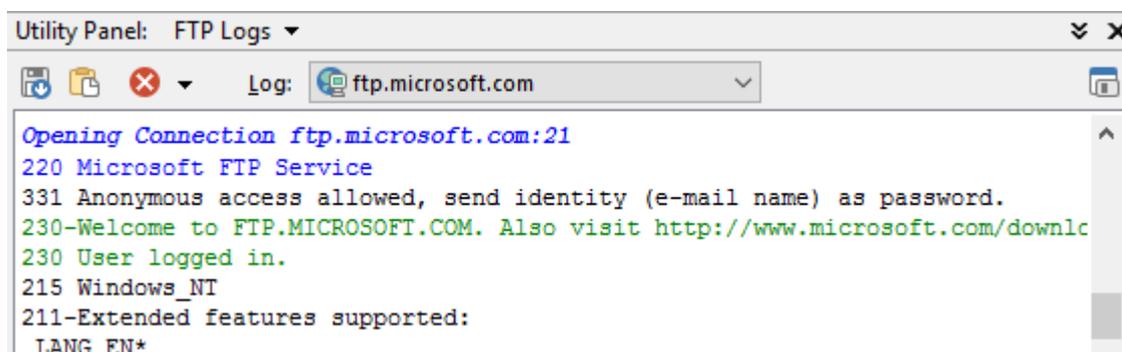
- **Apply changes to this session only:** The parameters you change will only apply to the current connection. If you connected via the address book, the original address book entry will not be modified.
- **Save changes permanently for this FTP site:** If you connected to the site using an address book entry, the site entry will be updated with the new configuration. If the current connection doesn't correspond with an entry in the address book, you will be prompted for a name to create a new address book entry.



## FTP Log

Opus can maintain a separate log for each FTP site you connect to. To enable logging for an FTP site, turn on the **Enable log** option (and optionally the **Enable debug** option) on the [Display](#) page for the site entry in the [FTP Address Book](#). Or to enable logging for all FTP connections, turn on the same option for the [Default Settings](#) entry.

The FTP Log is displayed in the Lister's [Utility Panel](#), which appears at the bottom of a Lister. To display the log, select the **Display FTP Logs** command in the drop-down **FTP** menu, or from the **Logs** sub-menu in the **Help** menu on the toolbar. You can also use the **Display log automatically** option for a site entry to have the log automatically displayed whenever you connect to the site.



The buttons at the top of the log display let you **Save** (📁) the log to a text file, **Copy** (📄) any selected text to the clipboard, and **Clear** (✖) the log. The drop-down attached to the **Clear** button gives you the option to clear all logs, otherwise only the currently displayed log is cleared.

The **Log** drop-down lets you select which site's log you are viewing. A separate log is maintained for each FTP site you connect to. The drop-down also contains an **All Activity** option which is a unified log showing all FTP output. If you connect to multiple sites at the same time you can keep an eye on all the FTP activity at once using the unified log.

The **Float** (📏) button on the right of the log panel lets you float the FTP log display free of the Lister. When you do this the Utility Panel will shrink to take up minimal space. This can be handy if you want to put the FTP log on another monitor to keep an eye on your FTP activity. When you close the floating log it returns automatically to the Lister it came from.

There are a couple of options in [Preferences](#) that affect the FTP log:

- The FTP log font can be configured on the [Display / Colors and Fonts](#) page.
- The maximum size of each individual log can be configured on the [File Operations / Logging](#) page.

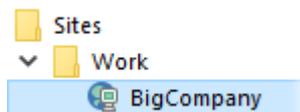
### FTP Paths

FTP locations are referenced internally using a URL-style path format, with the **ftp://** prefix. The full format of this path is:

**ftp://<user>:<password>@<host>:<port>//<folder>**

So if your username on the *bigcompany.com* FTP site were *jon* and your password were *apple*, you could make a connection with the path string **ftp://jon:apple@bigcompany.com/**. You can type this sort of path into the location field, or use it in buttons and hotkeys with the internal command set to automate your use of FTP. For example you could [set up a button or hotkey](#) to automatically copy (upload) selected files to this site using the raw command **Copy TO "ftp://jon:apple@bigcompany.com"**.

Opus internal commands also support a short-hand notation for sites listed in your [FTP Address Book](#). Instead of an **ftp://** style path, you can use the name of the address book entry prefixed by an *at* symbol.



The above command modified to refer to the address book entry would be **Copy TO @Work\BigCompany**. The advantage of using the address book in this way is that the current settings for the site are taken from the address book entry when it is used - so any changes you make to the configuration of the entry will automatically be reflected by any commands that refer to the site.



# Additional Functionality

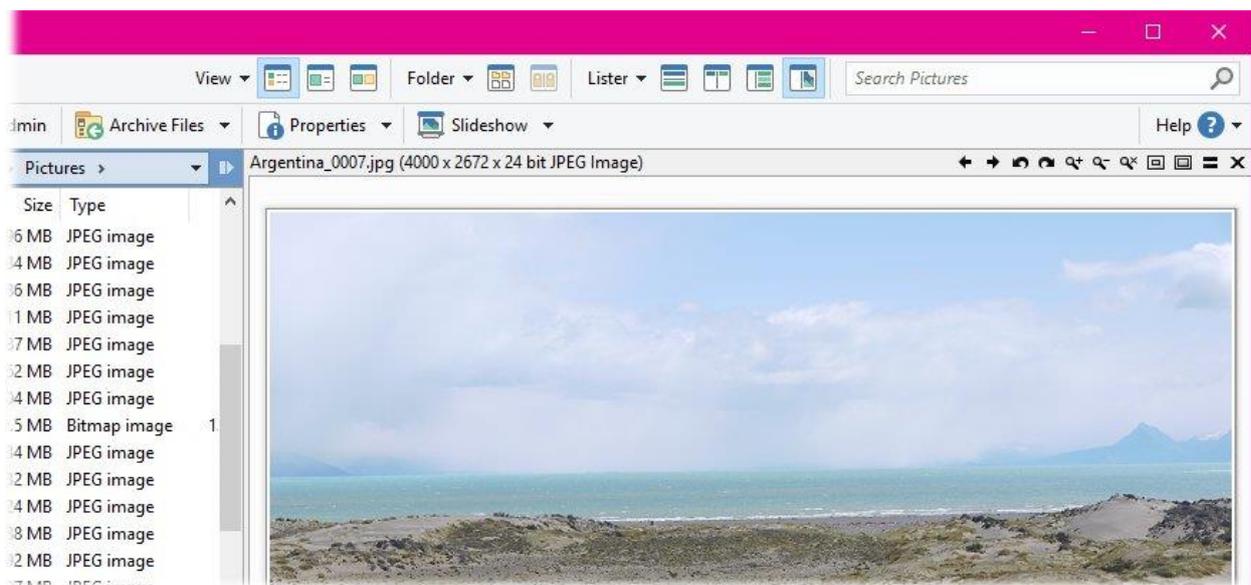
This section describes the non-core functionality that Opus provides - that is, functions that aren't directly related to everyday file management tasks, including:

- [Viewing images](#) and [playing sound files](#)
- Simple [image conversion](#) functions
- Printing and exporting [folder listings](#)
- Searching for [duplicate files](#)
- Sharing images on [Flickr](#)
- [Splitting](#) and [joining](#) files
- Creating [links, shortcuts and junctions](#)
- Using [floating toolbars](#) as a program launcher
- Configuring [system-wide hotkeys](#)
- Exporting Opus to [run from a USB drive](#)

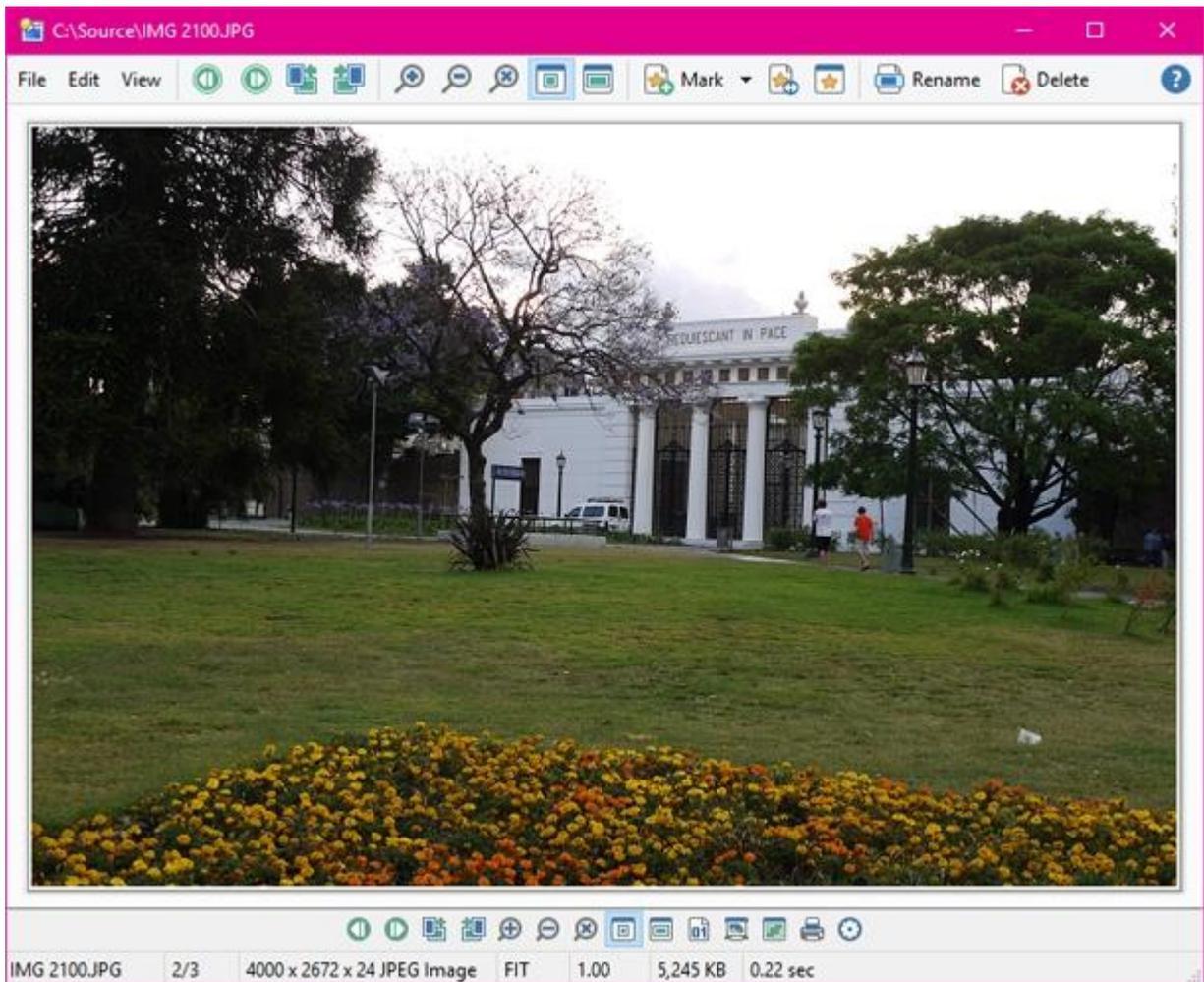
## Viewing Images

Opus has two main image viewers. Both can be used to view images, movies, documents and other formats for which a plugin exists.

There's the integrated [Viewer Pane](#), which is embedded within a Lister:



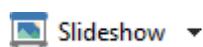
There's also a standalone (floating) image viewer.



The main difference is the [Viewer Pane](#) is tied to a Lister whereas the standalone viewer acts almost like a separate utility. The standalone viewer also has basic image editing functions which the viewer pane does not.

You can access the standalone viewer in several ways:

- If the **Use internal picture viewer** option on the [File Operations / Double-click on Files](#) page in Preferences is turned on, double-clicking on a recognized image file will open it in the standalone viewer.
- The **Slideshow** button on the default toolbar will launch a slideshow of all images in the current folder, using the standalone viewer.



The speed of the slideshow is controlled from the [Viewer / Behavior](#) page in Preferences.

- The internal [Show](#) command will display selected files in the standalone viewer. This command is available on the drop-down menu attached to the **Slideshow** button.
- From outside of Opus, you can use the **d8viewer.exe** or [DOpusRT.exe /show](#) commands to open files with the Opus viewer.

There are a number of options that control the appearance and behaviour of the standalone viewer. These can be found in the [Viewer](#) category in Preferences. By default, the viewer will:

- Auto-size to fit every picture - as you step through images, the window will resize if needed to display the picture.
- Open centered on the current monitor.
- Display a frame around the picture (as in the above screenshot).
- Show or hide the scrollbars while viewing images.
- Automatically build a list of all other pictures in the folder, when opened via a double-click on an image file (with an additional option for the list to wrap-around when you reach the start or end).
- Automatically rotate images to compensate for the EXIF orientation tag, saved by most digital cameras.

These options can all be changed from Preferences.

## Viewer Keys and Toolbar

By default, the main mouse and keyboard controls for the standalone viewer are as follows:

- **Scroll the image:** Drag the image around with the **left mouse button**, or use the **cursor keys**. While dragging with the mouse, hold **Ctrl** to toggle between 1:1 scrolling and accelerated scrolling.
- **Drag the image to another application:** Using the **left mouse button**, click and drag the window icon (top-left of the titlebar) and drop it on another program which accepts image drops (e.g. most image editors).
- **Select part of the image:** Using the **left mouse button**, hold the **shift** key and then click and drag over the image to create a selection rectangle. You can then use **Ctrl-C** to copy that part of the image to the clipboard or **Ctrl-R** to crop what the viewer displays. (Note: If you use **Ctrl-C** without a selection then it will copy the entire image to the clipboard, so you don't have to Select All first.)
- **Toggle additional information about the image:** Push **F10** and information about the image will appear in the bottom right. Push the same key again to hide it.

- **Toggle slideshow mode:** Push the **S** key. The current list of images will be cycled through on a timer.
- **Toggle full-screen mode:** Click the **middle mouse button** or push **Alt-Enter**.
- **Next or previous image:** Turn the **mouse wheel** or push the **Space** and **Backspace** keys.
- **Delete the current file:** Push the **Delete** key.
- **Zoom in and out:** Push = and - on the main keyboard, or + and - on the numeric keypad. You can also hold **Ctrl** and turn the **mouse wheel**.
- **Reset to original size (100% zoom):** Push **O** (letter-O) on the main keyboard or \* on the numeric keypad.
- **Set to specific size (25% to 800% zoom):** Push **Ctrl-1** through to **Ctrl-8**, on the main keyboard, for various zoom presets.
- **Fit to page:** Push **F**. Large images are reduced if too large for the window, but images are never enlarged. (Aspect ratio is always preserved.)
- **Grow to page:** Push **G** on the main keyboard. Large images are reduced to fit in the window; small images are enlarged to fill the whole window. (Aspect ratio is always preserved.)
- **Rotate the image to the left:** Push **L** on the main keyboard or numeric keypad.
- **Rotate the image to the right:** Push **R** on the main keyboard or numeric keypad.
- **Rotate to a specific position:** Push **1** (90 degrees), **2** (180 degrees) or **3** (270 degrees). Push **0** (zero) on the main keyboard or numeric keypad to reset the image's rotation.
- **Flip the image horizontally:** Push **H**.
- **Flip the image vertically:** Push **V**.

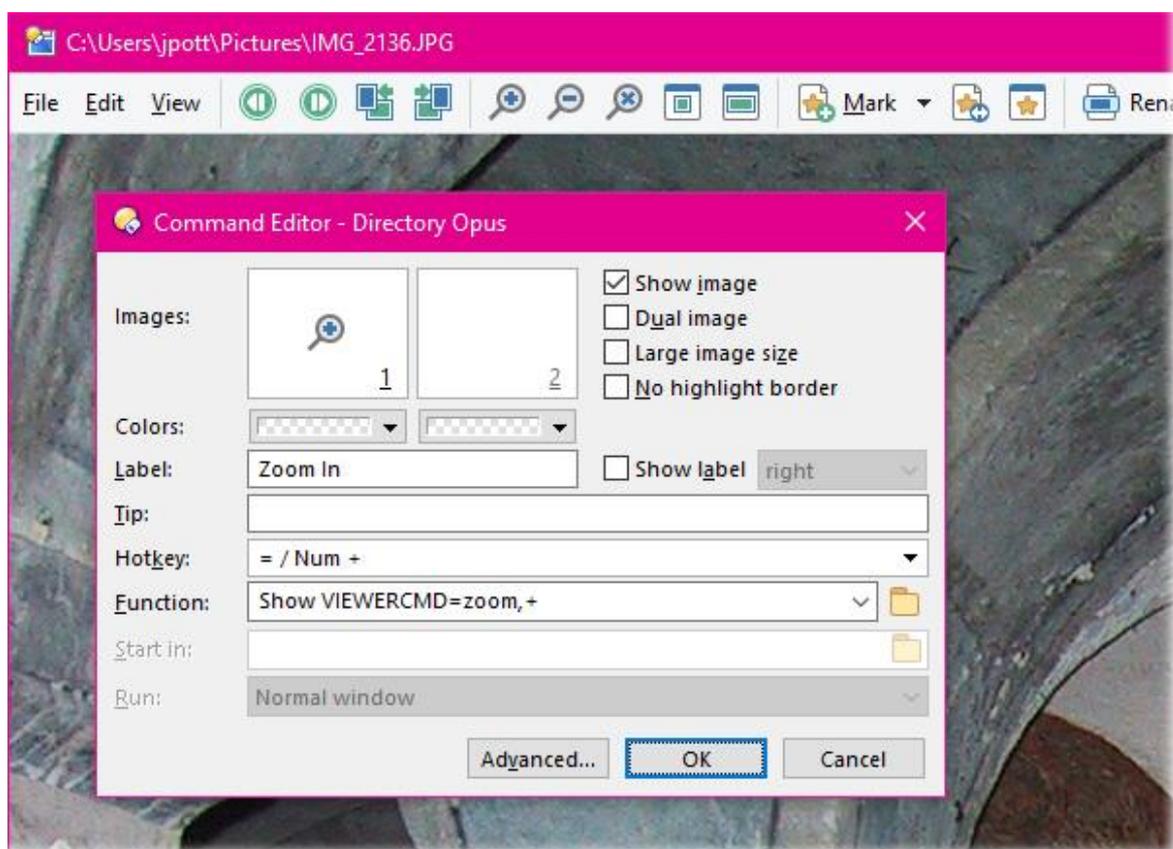
There are several other actions and hotkeys which you can find via the menu at the top of the viewer:

- **File:** The File menu contains commands to open a new image, save the current image (useful when you have cropped it, or want to convert it to a different format). You can print the current image, and launch the [Image Conversion](#) function to convert, resize or rotate the image.
- **Edit:** Contains some simple file-manipulation commands (*Copy, Move, Cut, Delete*). You can also copy the image (or a selected portion of the image) to the clipboard for pasting into other applications. It is also possible to crop the image to the current selection.
- **View:** Contains commands that let you modify the appearance of the viewer window. You can also rotate and zoom in and out of the image, apply gamma correction and selectively disable the alpha channel (if any). There is also a Full Screen command to display the image in full screen mode, and the *Show Information* command displays information about the image file in an overlaid tooltip.

**Ctrl+Tab** can be used to shift the input focus between a field in the metadata editor and the main viewer.

## Configurable Toolbar

The toolbar and context menu in the standalone image viewer are fully configurable, [just like all the toolbars and menus in the Lister](#). Additionally, you can create hotkeys that are only active in the viewer.



To edit the toolbar in the viewer, simply select the **Customize Toolbars** command from the Edit menu, just like in a Lister. The viewer context menu can be edited from the [Context Menus](#) tab in the [Customize](#) dialog, and you can also create viewer-specific hotkeys on the [Keys](#) tab.

The default viewer toolbar is called *Image Viewer*, but you can select another toolbar to use from the *Viewer / Appearance* page in Preferences. You might want to do this if, for example, you want to create your own toolbar but leave the default toolbar unchanged.

All the internal functionality of the standalone viewer is accessed using the **Show VIEWERCMD** command. These commands only work from a viewer toolbar, menu or hotkey – they will have no effect if you try to run them in a Lister. You can see from the above screenshot that the command corresponding to the *Zoom In* function is **Show VIEWERCMD=zoom,+**. A full list of **VIEWERCMD** commands is shown in the [Show command](#) reference section.

Although the **Show VIEWERCMD** command only works inside the viewer, this doesn't mean it's the only command that does – all other Opus commands and external functions also work inside the viewer. Of course, some commands (for example, **Select**) are not applicable to the viewer, but it's certainly possible to use commands like **Copy** or **Rename**, or have buttons that open the current picture in, say, Photoshop.

The **@if** directive can be used to test the state of various **Show VIEWERCMD** options when used within the viewer. For example, the following function would toggle between 100% zoom and Grow To Page modes:

```
@if:Show VIEWERCMD=zoom,reset
Show VIEWERCMD=zoom,grow
@if:else
Show VIEWERCMD=zoom,reset
```

See the [Command modifier reference](#) section for more information about **@if** command testing.

The behaviour of the mouse wheel and mouse buttons can be changed via Preferences. For example, you can make the left mouse button advance to the next image or close the viewer if you prefer.

## Control Bar

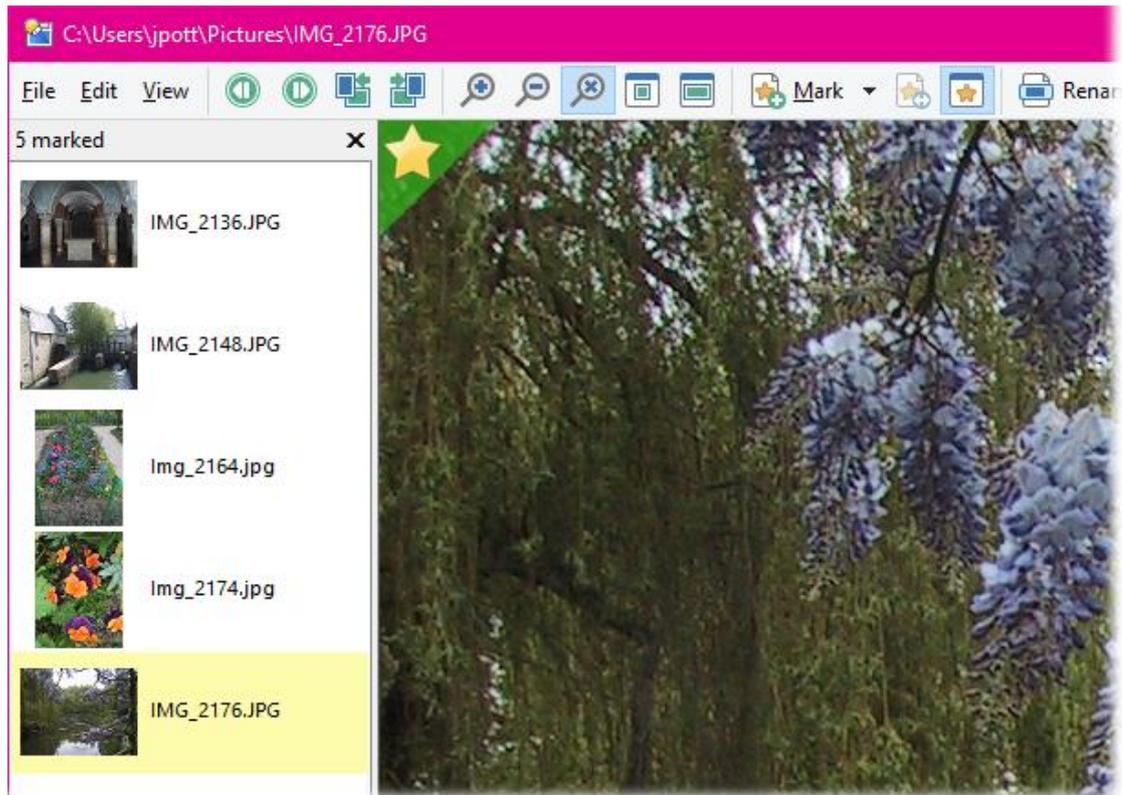
The optional control bar at the bottom of the viewer is the same as the one in the [Viewer Pane](#) - it contains buttons for commonly used functions. You can enable or disable it using the **View / Control Bar** command.



From left to right, the buttons are **Previous File** (◀), **Next File** (▶), **Rotate Left** (↶), **Rotate Right** (↷), **Zoom In** (+), **Zoom Out** (-), **Original Size** (⊗), **Fit To Page** (◻), **Grow To Page** (◻), **Hex View** (01), **Slideshow** (🎞), **Full Screen** (🖥), **Print** (🖨) and **Settings** (⚙).

## Image Marking

A common task is sorting through a bunch of digital photos, working out which ones to keep, which ones to upload, which ones to print, etc. The Image Marking feature in the standalone viewer can make this process very simple. Images you mark are automatically added to a [file collection](#), from which you can then copy, delete, upload, etc. the images you've chosen. Additionally, thumbnails of the images you've marked are shown in a separate panel in the viewer, and there are commands that make it easy to move around the marked images.



To mark the current image in the viewer, simply push the **M** key (or click the **Mark** button on the toolbar). With the default configuration, a file collection will automatically be created based on the name of the image's parent folder. On the left of the viewer window the marked image pane opens automatically, showing thumbnails of the images you've marked. When this opens it automatically checks the collection for any images you may have already marked in a previous session.

If the current image you're viewing is marked this is indicated with a star icon in the top-left corner, as shown in the above screenshot.

You can jump around the marked images by double-clicking their thumbnail. The marked image pane also lets you rename images by selecting their icon and pressing **F2**. The following keys relating to marking are also defined by default in the viewer:

- **M / Insert**: Mark / unmark the current image
- **Ctrl + M**: Show or hide the marked image pane
- **Ctrl + Left**: Jump to previously marked image
- **Ctrl + Right**: Jump to next marked image
- **Ctrl + Up**: Jump to first marked image
- **Ctrl + Down**: Jump to last marked image
- **Ctrl + Space**: Return from jump to last viewed image
- **Shift + M**: Exchange mark with previously marked image

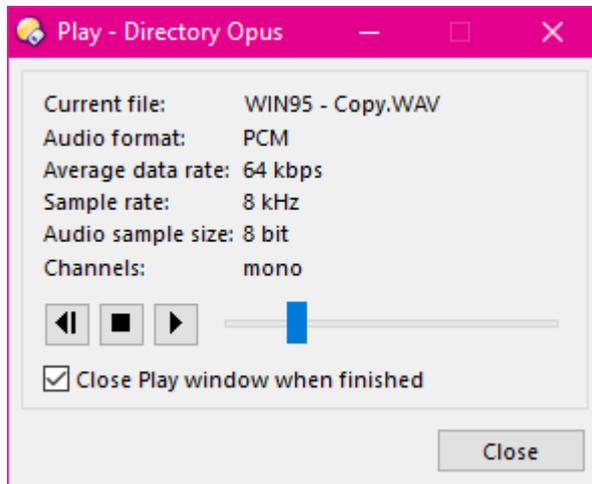
These keys make it very easy to jump back and forth between images you've marked and your current "position" in the list of images. The **Exchange mark** command comes in handy when you've got multiple photos of the same scene and you're trying to decide which is the best. You might have marked the first photo because you thought it was ok, but then two or three photos later you find one that's slightly better – simply press **Shift + M** to unmark the previous one and mark the new one instead.

If you want to take a break from your session you can simply close the viewer and come back to it later – all images that you've marked will be saved in the file collection. When you've marked one or more images and you close the viewer, Opus will automatically display the file collection for you in a new tab – you can change this behaviour in Preferences.

The name of the file collection can be configured on the *Viewer / Behavior* page in Preferences. When you configure the collection name, you can use the special code **%F** to insert the name of the parent folder, and **%D** to insert the current date. You can also have Opus ask you for a collection name before each marking session.

## Playing Sounds

Opus includes a simple audio player that lets you quickly play sound files from within Directory Opus. The only sound format that Opus supports natively is **.wav**, although formats like **.mp3** can also be played if you have a suitable codec installed in Windows.



The **Play** dialog displays some basic information about the sound file - name, format, data rate, etc. The available controls are:

- **◀**: Restart. This rewinds the sound file to the beginning.
- **■**: Stop. This stops the current sound file from playing.
- **▶**: Play. If the sound file is stopped, this starts it playing.
- **Position**: The slider indicates the current play position within the sound file. If the file allows it you can drag the slider to change the playback position.
- **Close Play window when finished**: If this option is on the Play dialog will automatically close when the final sound finishes playing. If turned off the dialog will wait for you to close it.

The internal sound player is accessed using the internal [Play](#) command. Because of its rather limited nature, and because these days all computers have media playing software as standard, it is not provided on the default toolbars. However, you can access it in two ways:

- On the [Customize](#) dialog (select the **Settings / Customize Toolbars** command), use the filter at the bottom of the [Commands](#) page to locate the **Play** command, and drag it to your toolbar or menu. You can then select one or more sound files and click the **Play** button to play them in sequence.
- Turn on the **Use internal sound player for WAV files** option on the Preferences [File Operations / Double-click on Files](#) page. If this option is on, double-clicking on a **.wav** file in Opus will automatically play it in the internal sound player.

# Image Conversion

The Image Converter is a simple tool that provides a quick way to perform basic image conversion tasks:

- It can convert from any image format Opus understands to **BMP**, **GIF**, **PNG** and **JPG** formats.
- It can rotate images in 90 degree increments, and also automatically rotate to compensate for EXIF orientation.
- It can resize or enlarge images, by a percentage or to an absolute size, optionally retaining the original aspect ratio.

To access the image converter, select the image or images you want to convert, and choose the **Convert** command from the **Tools / Convert Images** menu. The **Images file type group** also adds a command to the context menu of common image formats, so you can also access the converter by right-clicking on an image file and choosing the **Convert Image** command.



The image converter displays a thumbnail of the next image to be processed. You can enable some or all of the following options at once; for example, you can rotate and resize in the one step.

- **Convert:** Set this option on if you want to convert the image to a different format. Opus is able to save images in **Bitmap**, **GIF**, **PNG** and **JPEG** formats. If the source image is not already in one of these then you have to choose an output format to save in, but if the source image is already one of these formats then you can leave the **Convert** option disabled.

When converting to JPEG format, you must choose a **JPEG Quality** setting from **0** to **100**. JPEG is a [lossy compression](#) method, and the quality setting specifies how accurately the converted image will represent the source image data. The higher the number the better the quality will be, but also the larger the resulting file size.

When converting from a format that supports an alpha (transparency) channel, like PNG, you can specify the **Alpha conversion background color**. This color will be used as the background color if saving out in a format that doesn't support the alpha channel.

- **Rotate:** Turn this on if you want to rotate the image. You can select the amount of rotation to apply from **90**, **180** and **270** degrees - the thumbnail preview will update in real time to reflect your setting. If the source image has EXIF orientation information saved within it (e.g. a photo from a digital camera that was taken as portrait instead of landscape), you can also select **Use EXIF information** from the drop-down. If this is selected then Opus will use the EXIF information to automatically rotate the image to its correct orientation. If the image has no EXIF information or is already in the correct orientation, the output image won't be rotated.

Selecting **Reset** from the drop-down has the opposite effect to **Use EXIF information** - instead of using the EXIF orientation to rotate the image, Opus will leave the image alone and reset the EXIF orientation field to 0. Again, if the image has no EXIF information in it this option will not modify the image.

Opus supports lossless JPEG rotation if possible. Normally when you load and re-save a JPEG image, there is a reduction in quality of the new image. This can be hard to notice at first but errors accumulate, and a JPEG file that has been repeatedly decompressed and recompressed will often look quite a lot worse than the original. The JPEG format allows for image rotations to be lossless (i.e. no reduction in quality) if certain conditions are met; namely, the width and height of the image must be an exactly multiple of the "block size" the image has been saved in (usually 8x8 or 16x16). Luckily, most digital cameras do produce images of these dimensions. You don't need to do anything special in Opus to enable lossless rotation - if a JPEG image can be rotated losslessly it will be.

- **Resize:** Turn this option on if you want to resize the image. You can select some predefined output sizes, or reduction or enlargement ratios from the drop-down, or choose **Custom size** and enter your own width and height values (specified in pixels).

The **Add filename suffix** option causes Opus to automatically add an appropriate suffix to the filename when it saves the output file. For example, if you resized **IMG\_2483.jpg** to 150% of the original size, the output file would be called **IMG\_2483-150%.jpg**.

Turn on the **Preserve aspect ratio** option to avoid changing the aspect ratio of the image when it's resized. Opus will adjust the output dimensions to keep the same ratio between width and height as the source image.

- **Write converted images to destination folder:** Normally Opus will save the converted images in the same folder as the source images. If you turn this option on Opus will instead save the converted images to the [current destination](#) file display. If you leave this option off, and the output filename isn't changed because of the **Add filename suffix** option (above), the original image file will be overwritten. You will be prompted for confirmation (and given a chance to change the output filename manually) in this case.

When you have set your conversion parameters as desired, click the **OK** button to convert the current image. If you originally selected multiple images, you can apply the same conversion to all of them (as a batch operation) by clicking the **All** button instead. If you click **OK** and have multiple images selected, the first image will be processed, and the image conversion dialog will then re-open for the second image.

## Automated image conversion tasks

Using the internal [Image](#) command, it is possible to configure buttons and toolbars that automate various simple image conversion functions. That way, you can simply select the image files in question and click on a button to perform the conversion, without having to display and configure the [Image Conversion](#) dialog.

Listed below are some example uses of the **Image** command - please see the [Creating your own buttons](#) page for information on how to create a button.

- **Image ROTATE=270 HERE REPLACE**

Rotates selected image files 90 degrees to the left (i.e. 270 degrees clock-wise). The images are rotated "in-place" - that is, the rotated file will be saved over the top of the original. JPEG images will be rotated losslessly if possible.

- **Image ROTATE=90 HERE REPLACE**

Rotates selected image files 90 degrees to the right.

- **Image ROTATE=EXIF HERE REPLACE**

Rotates selected image files to compensate for the orientation value stored in the EXIF tag. This lets you "correct" photos taken on digital cameras that appear incorrectly rotated. The image will be automatically rotated to be the "right way up" and the EXIF orientation tag will be reset. This will replace the original files. If a selected file does not have an EXIF orientation tag it will be unaffected.

- **Image CONVERT=png HERE**

Converts selected image files to PNG format. The converted images will be saved in the same folder as the originals, with **.png** file extensions.

- **Image CONVERT=jpg QUALITY=90 HERE**

Converts selected image files to JPEG format, at 90% quality.

- **Image CONVERT=jpg WIDTH=256 HEIGHT=256 PRESERVEASPECTRATIO ADDSUFFIX "\_thumb"**

Makes JPEG thumbnails from selected image files. The thumbnails will be sized to at most 256x256 pixels (the original aspect ratio will be preserved, so the final dimensions will depend upon the original image). The thumbnails will be saved in the current destination folder, and will have the suffix **\_thumb** appended to their filename.

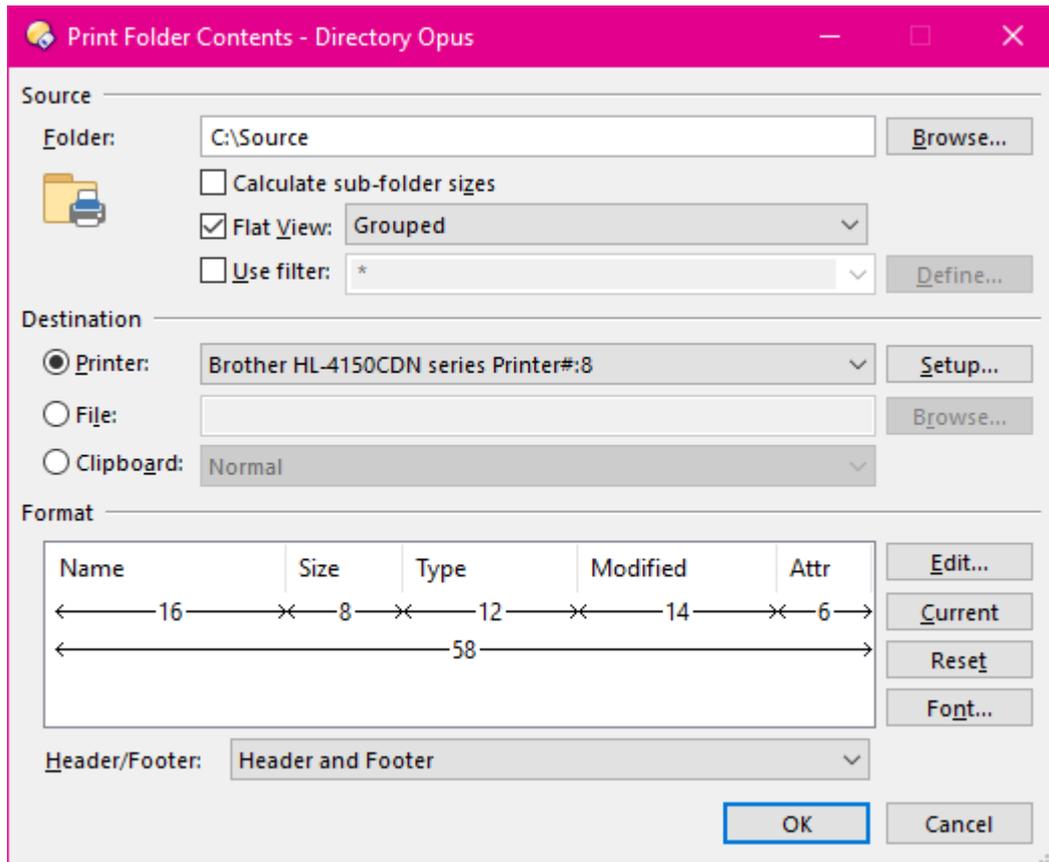
- **Image CONVERT=png HEIGHT=1200 PRESERVEASPECTRATIO HERE REPLACE**

Resizes selected images to 1200 pixels high - the width will be determined automatically based on the aspect ratio of the original image. The resized image files will be saved in PNG format - if the original images were also PNG format they will be replaced.

# Print Folder

The Print Folder function lets you print a listing of the contents of a directory. As well as printing to a printer, it can also print to a file or to the clipboard. In these cases you can choose from various export formats (plain text, tab separated or comma separated) which would then let you import the listing into a program like Excel.

To access the Print Folder dialog, select the **Print / Export Folder Listing** command from the **Tools** menu.



The Print Folder dialog is divided into three sections; **Source** (specifies which folder to print), **Destination** (specifies where to print or export it to) and **Format** (specifies the information to be included in the output).

The **Source** section contains the following options:

- **Folder:** Specifies the folder that you want to print the contents of. When you run the Print Folder command this will default to the current folder shown in the file display, but you can change this using the **Browse** button.
- **Calculate sub-folder sizes:** Select this option if you want the total sizes of any folders in the listing to be calculated. If turned off, folder sizes will not be displayed. This option also controls whether columns like *File count* and *Sub-folder count* work when added to the **Format** section of the **Print Folder** dialog. Any columns that require recursively enumerating the contents of folders will only work as expected if the **Calculate sub-folder sizes** option is switched on.

- **Flat View:** Similar to the Lister's [Flat View](#) mode, this lets you print the contents of sub-folders as well as the selected folder. The drop-down lets you pick the following modes:
  - **Mixed:** The contents of all sub-folders (and their sub-folders, and so on) are shown on the parent level, as if the directory tree was just one big folder.
  - **Mixed (No Folders):** The same as **Mixed**, but only files are included in the listing, not folders.
  - **Grouped:** Prints the folder as a nested tree structure. The contents of sub-folders will be shown indented from their parents.
- **Use filter:** Lets you specify a filter to control which files and folders are included in the listing. You can either enter a [wildcard pattern](#) directly, select a [pre-configured filter](#) from the drop-down, or click the **Define** button to [define a new filter](#).

The **Destination** section specifies where you want the folder listing to go.

- **Printer:** The folder listing will be printed to the selected printer. The drop-down shows a list of your installed printers, and the **Setup** button lets you configure the printer properties.
- **File:** The folder listing will be saved to a file. Click the **Browse** button to specify the output filename (or enter it manually in the field). The file extension (or option you choose for the **Save as type** drop-down in the browse dialog) defines the format of the file that's exported. You can choose from the following formats:
  - **Text File:** Exports the folder listing to a plain text (**.txt**) file.
  - **Comma-Separated List:** Exports the folder listing to a comma-separated (**.csv**) file. This can then be imported into a program like Excel for further processing.
  - **Tab-Separated List:** An alternative to **.csv**, this format can also be imported into Excel and similar software.
- **Clipboard:** The folder listing will be placed on the clipboard, and you can then paste it into another program as in any copy/paste operation. The **Clipboard** drop-down lets you pick from the same formats as the **File** option above.

The **Format** section defines the format of the print-out, including which columns are included in the listing.

- **Edit:** Click the **Edit** button to edit the folder format used for the print out. This displays a standard [folder options](#)-type dialog that lets you select columns, configure display and sorting options, etc.
- **Current:** The Print Folder dialog remembers its format from use to use. The **Current** button lets you update the format settings in the Print Folder dialog with those from the file display that you launched the Print Folder command from. So if you have a specific folder format set in a file display and want to print a list that looks the same, you could click the **Current** button to avoid having to reconfigure the Print Folder format manually.
- **Reset:** This resets the format to the one that was set when the Print Folder dialog was invoked. This lets you undo any changes you have made to the format (or if you accidentally clicked the **Current** button, etc).
- **Font:** This command lets you pick the font that's used to print the listing. The font is only used when printing to a printer; when exporting to a file or the clipboard, the font setting is ignored.
- **Column widths:** The Print Folder dialog displays a "preview" of the columns that have been specified in the format for printing. This lets you control how wide each column is, by clicking on the separators between the header items and dragging them to resize. The specified column widths are used when outputting to a printer and when sending the results to a text file or the clipboard using the 'Normal' format. (The widths are ignored when using the Comma-Separated or Tab-Separated output formats.)

Name	Size	Type	Modified	Attr
← 16	× 8	× 12	× 14	× 6
← 58				

Below the header are two rows of information; the first row displays the widths of each column, and the second row displays the total width.

When printing, the widths are expressed in "nominal characters" (an average character width calculated from the specified font) and you can use them as a guide to make sure your folder listing will fit horizontally across the page.

When outputting to a text file or the clipboard using the 'Normal' format, the widths are the number of fixed-width characters used for each column, and will be rounded up to a multiple of 8 characters (to match the tab stops in most text editors).

- **Header/Footer:** This option lets you enable a header, a footer, or both. The header and footer show the name of the folder being printed and the current date and time. If printing to a printer, the header and footer are printed to each page. If printing to a file, the header is printed once, at the start of the file, and the footer is printed once at the end.

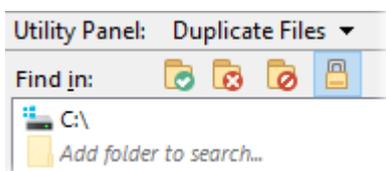
# Duplicate File Finder

This tool lets you search your drives for duplicate files. This function has two distinct modes:

1. To search for all files that appear more than once (i.e. *all* duplicate files).
2. To search for duplicates of one or more specific files.

To access the duplicate file finder, select the **Find Duplicate Files** command from the **Tools** menu. The duplicate file finder appears at the bottom of the Lister in the [Utility Panel](#).

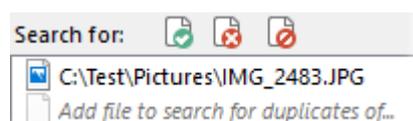
The top section of the duplicate file finder panel lets you specify where to search. You can search one or more folders (or entire drives) at once by adding entries to the **Find in** list.



Each entry in the list corresponds with a folder to search. To add a folder to the list, double-click the *Add folder to search* item. You can also use the **Select Folders to Search** (folder with checkmark) button on the toolbar - this displays a folder selection dialog with checkboxes so you can select multiple folders simultaneously. You can edit a folder path in the list by double-clicking it (or select it and press **F2** if you want to enter the path using the keyboard).

To remove a folder from the list, select it and click the **Remove Folder** (folder with X) button. The **Reset Folder List** (folder with red circle and slash) button clears the folder list, and the **Lock Folder** (folder with lock) button locks the **Find in** location to the folder displayed in the current file display. When the folder is locked, the **Find in** list will automatically reset to the current location whenever you navigate in the file display.

The bottom section of the panel lets you specify which files you want to search for duplicates of. If this list is empty, the finder will search for *all* duplicated files (mode one). If you specify one or more files in this list, the finder will only search for duplicates of those files (mode two).



Each entry in the list represents a file that the finder will search for duplicates of. To add a file to the list, double-click the *Add file to search for duplicates of* item, or click the **Browse** (document with checkmark) button on the toolbar. You can edit a file path in the list by double-clicking it (or select it and press **F2** if

you want to enter the path by hand). Use the **Remove File** () button to remove a file from the list, and the **Clear File List** () button to clear it.

The right-hand side of the panel contains options that let you control how Opus searches for duplicates.

- **Show results in:** This lets you specify the name of the [File Collection](#) that the results of the search will be displayed in. The current file display will automatically navigate to show this collection when the search begins, and as files are found they will appear in the display. The display will be automatically grouped to keep duplicate files together. If the specified collection doesn't already exist it will be created automatically.
- The **Comparison method** drop-down lets you specify what method of comparison Opus uses to determine whether two files are duplicates:
  - **Filename only:** The comparison is based on filename only. If two files have the same name they will be considered duplicates.
  - **Filename (no extension):** This will search for files with the same filename stem, ignoring their file extension. For example, "grandma.mp3" and "grandma.jpg" would match.
  - **Filename and size:** The comparison is based on filename and size. If two files have the same name and are the same size they will be considered to be duplicates.
  - **Size:** The comparison is based only on file size. If two files are the same size they'll be considered to be duplicates.
  - **MD5 checksum:** This is the slowest but most accurate method of comparison. Filenames are not considered using this method; instead, for any two files Opus first compares their sizes. If the sizes are the same Opus then calculates the MD5 checksum for both files - if the checksum matches then the two files are considered to be duplicates. You can use the **MD5 accuracy** slider to reduce the accuracy of the comparison and speed up the search.
- **Clear previous results:** If this option is on the contents of the **Show results in** file collection will be automatically cleared before the search begins.
- **Delete mode:** If the reason you are searching for duplicate files is to eliminate wasted space, this mode can be handy. When this option is turned on and you perform a search, the file display showing the results is automatically put into [checkbox mode](#). When the search is complete, Opus will automatically select all but the first file of every duplicate file group. You can then use the **Delete** button at the bottom of the panel to delete the checked files. The **Select** button lets you re-run the delete selection process without having to re-run the search. This can be useful if you want to re-sort the list to affect which files are checked for deletion.
- **Filter:** The filter field lets you define a filter to control which files and locations are searched. You can use this in two ways:
  - You can enter a [simple wildcard pattern](#) to control which files will be considered by the duplicates search. For example, enter \*.jpg to only search for duplicate **JPG** files.
  - You can click the **Define Filter** button () to define a filter using the advanced filter control (or use the drop-down to select a previously saved filter). You can use this to specify files to search for as well as control (using Sub-folder clauses) which sub-folders are searched. See the [Filtered Operations](#) page for more information on filters.
- **Number duplicate groups:** The Duplicate Finder creates groups for each set of duplicate files found, and normally these groups are named after the criteria used for the duplicate search. If you turn this option on

then each group of duplicates will be numbered (from 1 to X, in the order they are found).

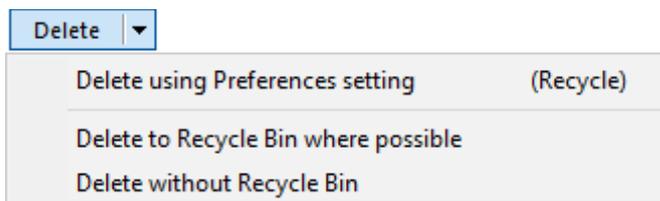
- **Search inside archives:** The function will also search the contents of any archive files that are found in the specified locations.
- **Search inside subfolders:** The function will search the contents of all sub-folders of the specified locations as well as the locations themselves. You can use the **Filter** option below to control which sub-folders are searched.
- **MD5 accuracy:** When searching by checksum you can elect to only calculate a hash for a percentage of the file. This lets you speed up the operation for large files at the expense of accuracy. Use the slider control to adjust the percentage from 1% to 100%.



- **Use MD5 cache:** Use the MD5 checksums cache for large files located on NTFS partitions. If a large file's checksum is calculated it will be cached, and the cached value used in the future if the file doesn't appear to have changed.

When you have specified the locations and parameters for the duplicates search, click the **Find** button to begin.

When using the **Delete mode** option, you can override the main Preferences Recycle Bin setting when deleting any selected duplicate files.



The **Delete** button in the bottom-right of the panel has a drop-down menu attached which lets you choose exactly how you want to delete the duplicates.

# Flickr Synchronization

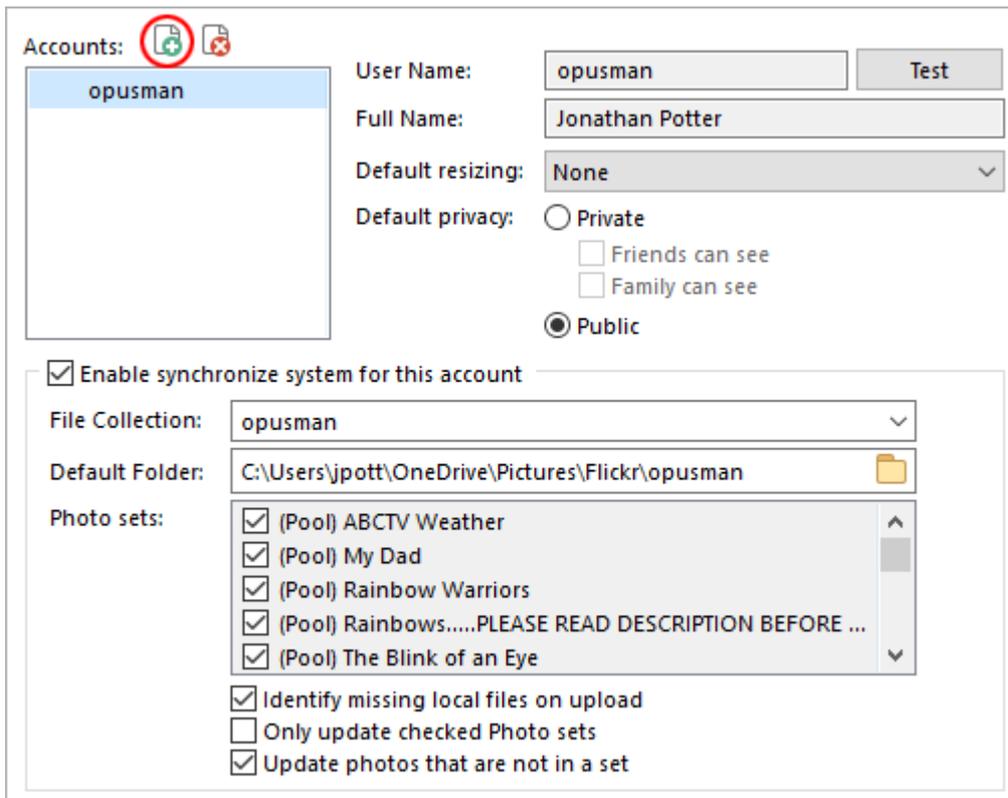
The Flickr Synchronization tool lets you keep a local copy of your [Flickr](#) photo collection, and makes it easy to keep them in sync. Flickr synchronization makes use of [file collections](#) to store a virtual catalog of your Flickr photos. The basic premise is:

1. [Configure your Flickr account](#) in Preferences. You will need to authorize Opus to access your Flickr account - the appropriate page will open in your web browser automatically.
2. A file collection is created representing your Flickr account (by default this is listed under **Flickr Photos**\<account name> - e.g. **coll://Flickr Photos/username** – but this can be changed from Preferences).
3. A local disk folder is designated as the default storage for photos downloaded from Flickr.
4. To upload new photos to Flickr, simply add them to the file collection using the **Copy Files** button or drag-and-drop. The photos themselves can stay in their original locations – copying them into the file collection does not actually copy or move the file data.
5. If you wish to place images in a Photoset, create a sub-collection (using the **Create Folder** command) with the appropriate name and add the photos to the sub-collection instead of the main one. You can place the same photo in multiple sets by adding it to multiple sub-collections.
6. Perform the synchronization. The easiest way to initiate this is by right-clicking the *username* collection in the tree and choosing *Synchronize with Flickr*.
7. You have the option of one or two-way synchronization. Opus will contact Flickr and try to establish which photos need to be uploaded and downloaded (or only one if one-way synchronization is selected)
8. Photos that are in your file collection that do not exist on Flickr will be uploaded, and optionally assigned to any photosets you have placed them in
9. Photos that are on Flickr that do not exist in your file collection will be downloaded to the default storage location, and placed in the appropriate collection (and sub-collections if they are in a Photoset).

The idea is that at the end of the process your Flickr file collection will exactly represent your Flickr account.

## **Configuring your Flickr account**

Before you can perform a synchronize with your Flickr account you need to grant access to Opus. This process is initiated from the [Photo Sharing / Flickr](#) section of Preferences.



Click the **New Account** button (highlighted above) to begin this process. Opus will display a dialog prompting you to grant authorization. When you click the **Authorize** button, Opus will open a web browser where you should log in to your Flickr account. Follow the prompts given to complete authorization. Once authorization is complete, your Flickr account details will appear in the **Accounts** list. You can register as many different Flickr accounts as you like.

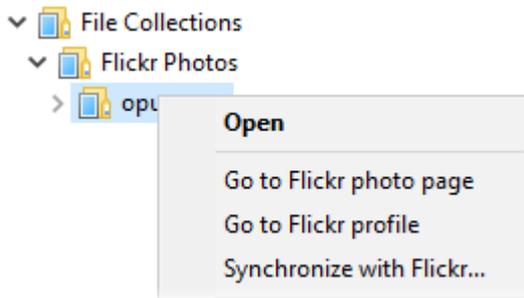
The other options on this Preferences page control the behavior of the synchronize process. You can choose to have Opus automatically resize your photos when you upload them, and configure their default privacy settings.

The **File Collection** setting lets you configure the name of the file collection used to represent your Flickr account. By default this will be the name of your account. The **Default Folder** setting lets you specify a folder on disk where any photos downloaded from Flickr are stored. Because the Flickr synchronization is a two-way process, it is possible for photos on your Flickr account to be downloaded to your local computer – the default folder is the folder these will be stored in.

You can also configure which of your Photosets are included when you perform a synchronize. The first time Opus contacts Flickr it will populate this list with the names of your Photosets and Pools. If you only want a subset of your photosets to be involved in the synchronization you can control it from this list.

## Synchronizing Photos

Once you have configured your Flickr account, you are ready to begin the synchronize process. In the *Folder Tree* you will see that a new file collection has been created to represent your Flickr account. For example, if your account name is Nudelyn, the file collection created would by default be **coll://Flickr Photos/Nudelyn** (although this can be changed from Preferences).



To initiate the synchronization, right-click on the collection name (*Nudelyn* in the above example), and choose **Synchronize with Flickr**. The Flickr Synchronize dialog will open and step you through the various stages of the process:

### 1. Select Accounts To Synchronize

It is possible to configure multiple Flickr accounts, and to synchronize one, multiple or all of them at once. The first step of the synchronization process lets you select which accounts are to be included. You can also select whether photos are to be uploaded (from your local computer to your Flickr account), downloaded (from your Flickr account to your local computer) or both.

### 2. Identify Local Files

This step is necessary if you have opted to download from your Flickr account, and Opus finds photos on your account that do not appear to exist locally. It gives you a chance to identify the images locally without having to download them, thus saving time and bandwidth.

If any photos are found that Opus does not recognize, it will first try to locate them itself, by looking in the **Default Folder** configured in Preferences. Opus tries to match photos in two ways – firstly, by filename, and secondly by EXIF shooting time.

Matching by filename is not very accurate, because Flickr does not store the original file name along with the image. For example, an image called *IMG0599.JPG* may have been renamed on Flickr as *My Pet Cat*. Opus will therefore attempt to find a file called *My Pet Cat.jpg*, but will not know that the file was originally called *IMG0599.JPG*. Therefore, the second method of matching will be used if the first fails – EXIF shooting time. Most modern cameras store the time a photo was taken inside the image, and Opus

is able to retrieve this information from Flickr. It is therefore able to search your local files for one that precisely matches the shooting time, even if the filename has changed.

If the automatic matching fails, you can use the controls on this page of the dialog to locate the files manually. Use the **Identify** button to tell Opus exactly which file it is (you can use the **Thumbnail** button to retrieve a thumbnail of the image from Flickr so you know what you are looking for.) You can also use the **Search** button to perform the automatic matching process in a location other than the default folder.

Each image that appears in the **Identify Local Files** page has a checkmark next to it. If you turn this checkmark off, Opus will mark that image as *ignored*, and will never prompt you again to identify it locally.

Any photos that you can't identify locally, and you do not mark as *ignored*, will be downloaded from your Flickr account to the default folder during the synchronization process.

Note that you can disable the **Identify Local Files** step altogether from Preferences.

### 3. **Select Photos To Upload**

This step is displayed if you have opted to upload from your local computer to your Flickr account, and Opus has found photos in your Flickr file collection that do not exist online.

For each photo displayed here you can configure the *Title*, *Description* and *Tags*. You can also choose whether to resize the image on upload, and configure the privacy settings. Note that default resizing and privacy options can be configured in preferences.

Each photo listed here has a checkmark – if you turn this checkmark off the image will not be uploaded.

### 4. **Select Photos To Download**

This step is displayed if you have opted to download from your Flickr account, and you could not identify all missing images locally in step 2. A list of unmatched online images is displayed. Each photo listed here has a checkmark – if you turn this checkmark off the image will not be downloaded.

### 5. **Synchronize Summary**

This is the final step in the process. The Summary page displays information about the actions to be performed – how many photos will be uploaded, how many photos will be downloaded, and how many photoset reassignments will occur. Once you are satisfied with the summary click the Next button to begin the synchronization.

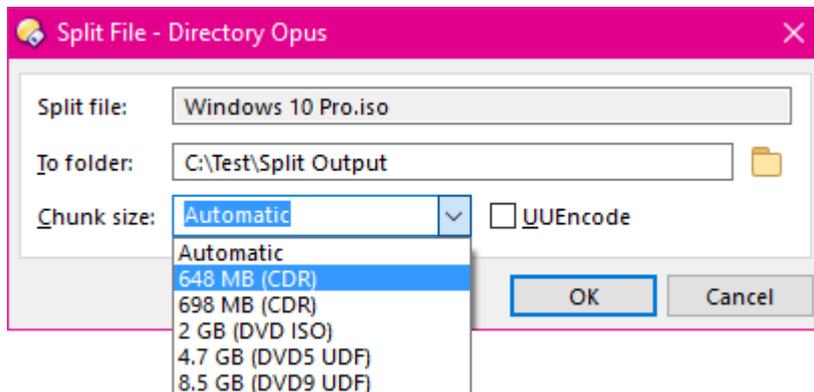
The first time Opus contacts Flickr, it downloads a list of your Photo sets, and the Pools that you are a member of, and creates sub-collections for each one. When you perform your first synchronization, Opus will add all the images found online to the collections as appropriate. After that, uploading new images to Flickr is as simple as adding them to the collection or sub-collection as desired and performing the synchronization.

Flickr supports adding the one photo to multiple Photosets, and Opus supports this too – simply add the photo to the sub-collections representing those sets. You can also create new Photo sets by simply creating new sub-collections. You can reassign photos to Photosets and pools without re-uploading them by simply moving them from one file collection to another. Or, to remove a photo from all Photosets, move it from the sub-collection to the parent collection (the parent collection represents *unassigned* photos – that is, photos that are not in a set or in a pool.)

You cannot use Opus to delete photos from Flickr – this can only be performed from the Flickr website.

## Splitting Files

The **Split File** tool lets you split a single file into multiple parts. For example, you can split a large file into parts that are small enough to fit on a CD, or to be sent via email. The [Join Files](#) tool can be used to join the separate parts back together again.



To access the **Split File** tool, select the file you wish to split and then choose the **Split File** command from the **Tools** menu.

The **To folder** field lets you define where the split parts will be written - this defaults to the [destination folder](#) (if there's no destination folder, it defaults to the folder containing the source file) but you can change it using the **Browse** (📁) button.

The **Chunk size** field lets you select the size of the split parts. You can choose a pre-defined size from the drop-down list, or enter your own size. If you enter a manual size you can specify it in kilobytes (**KB**), megabytes (**MB**) or gigabytes (**GB**) by entering the number followed by the appropriate suffix. For example, **1.87 MB**. If no suffix is given, the chunk size is taken to mean bytes.

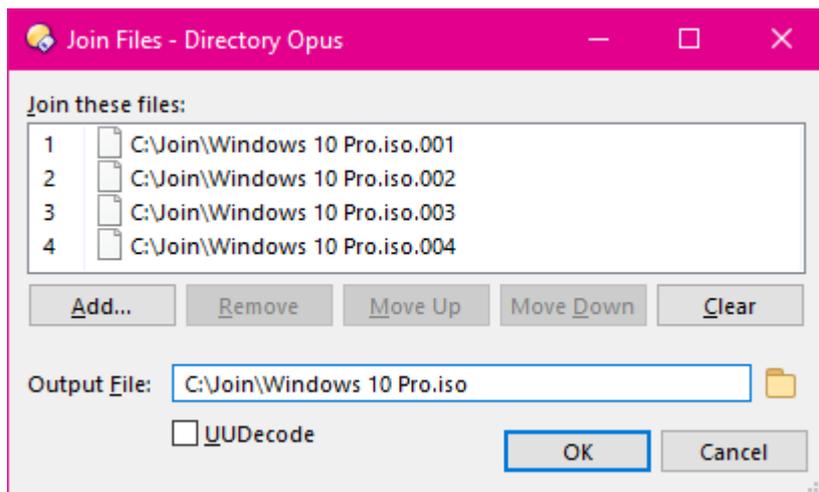
If the **Chunk size** is set to **Automatic** then it will use the destination's free space, if it is a removable device, and a fixed size of 100 MB otherwise.

If the **UUencode** option is turned on then the split parts will be [uuencoded](#). Uuencoding is a form of encoding used to transmit binary files (like programs or video files) as plain-text ASCII files. For example, some UseNet newsgroups are used to distribute binary files that have been uuencoded.

## Joining Files

The **Join** tool lets you join two or more files together. For example, if you have previously used the [Split](#) tool to split one large file into multiple parts, the Join tool can join them back together again.

To access the Join tool, select the files you want to join and choose the **Join Files** command from the **Tools** menu. You can also run the Join Files command without any files selected, and then add files to the join list manually.



The Join Files dialog displays a list of the files you are joining together, in the order that they will be joined. When you add files to the list, the tool tries to determine the proper order based on their filenames, but if needed you can reorder the list using drag and drop, or by selecting an item and then moving it up or down the list with the **Move Up** or **Move Down** buttons.

Use the **Add** button to add additional files to the join list. You can also add files by dragging them from a Lister and dropping them on the tool. The **Remove** button lets you remove the selected file from the list, and the **Clear** button clears the list completely.

The **Output File** field lets you specify the name of the output filename. This will default to joining to the current destination folder (if any), but you can enter a different path or use the **Browse** (📁) button to select the output location.

Turn on the **UUDecode** button if the files you are joining together have been [uuencoded](#). Uuencoding is a form of encoding used to transmit binary files (like programs or video files) as plain-text ASCII files. For example, some UseNet newsgroups are used to distribute binary files that have been uuencoded.

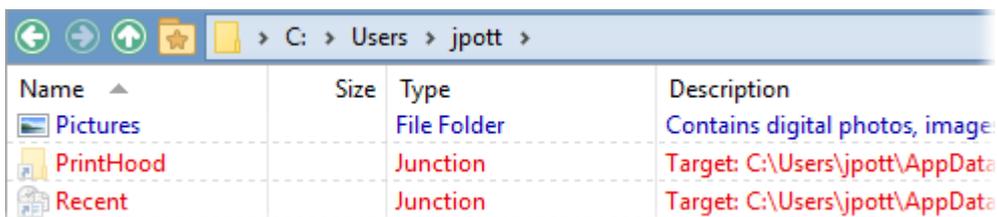
# Making Links and Junctions

A link in general terms is a file or folder that points, or redirects to another file or folder. Windows supports four distinct types of link:

- **Shortcuts:** The ones you're probably most familiar with, these are implemented as a separate .lnk file (although the file extension is often hidden), that stores the details of the file or folder it points to. Shortcuts can refer to a file or folder on any drive, including on a network share.
- **Hard links:** Hard links can only point to a file. They are implemented at the file system level. Technically speaking, all files are "hard links" - there's no concept of an original file. Once a link has been created, there is no way to tell which one is older. A file is only deleted when the last hard link that points to it is deleted. Hard links can only be created on the same drive partition as the original file.
- **Junctions:** Junctions can only point to a folder. They are implemented at the file system level.
- **Soft links:** Soft links are only available in Vista and later. A soft link can point to either a file or a folder, and they are able to span hard drives. Creating a soft link requires administrator permissions - Opus will display a [UAC](#) prompt if needed. Soft links can be *absolute* or *relative*.

Note that hard links, soft links and junctions are only supported on NTFS-formatted drives.

Windows (particularly in Vista and above) uses links and junctions quite a lot. If you turn off the **Hide protected operating system files** option on the [Folders / Global Filters](#) page in Preferences, you will notice several folders appear in red in various locations (the root of C: for one, and your user profile folder). Microsoft uses these hidden junctions to maintain backwards compatibility with software that has hard-coded the names of legacy paths. For example, in Windows XP, user data is stored in *C:\Documents and Settings*, but in Vista and above it's in *C:\Users*. By default Windows creates a hidden junction called *Documents and Settings* that points to the new *User* folder, so that software that specifically references the old path will still operate.



Name	Size	Type	Description
Pictures		File Folder	Contains digital photos, image
PrintHood		Junction	Target: C:\Users\jpott\AppData
Recent		Junction	Target: C:\Users\jpott\AppData

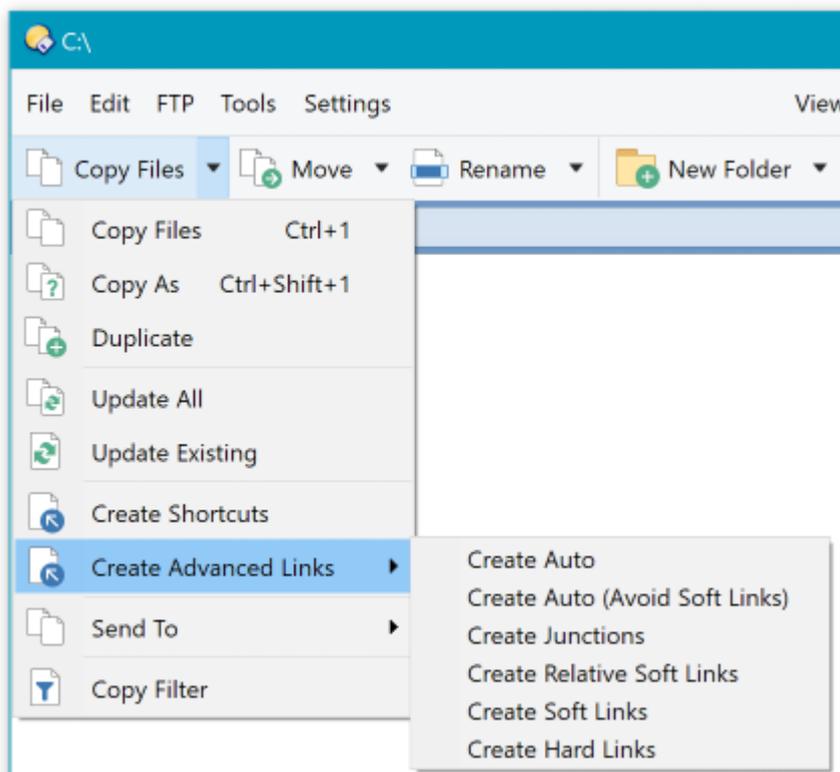
If you turn on the **Description** column in a Lister you can see the target of links and junctions. See the [Folder Options](#) section for information on adding columns to the display.

Opus is able to create all these types of links for you. Creating a shortcut is the simplest, and you can do this in Opus just as you can in Explorer, either by:

- Dragging and dropping an item with the right mouse button. The popup menu will contain a **Create Shortcut Here** command.
- Copying a file or folder to the clipboard, and then right-clicking in a folder and choosing the **Paste Shortcut** command from the context menu.

Aside from basic shortcuts, other types of links and junctions are generally used only rarely and by experienced users (at least on Windows). If you are not familiar with how they work, be careful and test with copies of both target and source folders first. Be aware that most actions in most software are not "link aware". For example, be careful when deleting or overwriting links that you are not deleting or overwriting the things they point to, unless that is your intention. Test how the things behave first before risking important data!

Commands to create the other types of links and junctions can be found in the default toolbars under the **Copy Files** button's menu, within the **Create Advanced Links** sub-menu:



Like the main **Copy Files** button itself, the menu items work between dual file displays, taking the selected items in the source folder and creating junctions or links in the destination folder which point to them.

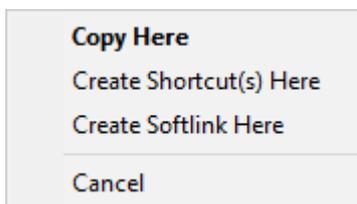
Behind the default menu items is the internal [Copy](#) command, which is used to create shortcuts as well as the other types of links. For example, you the command to create a button or menu item to create soft links in the destination folder to selected files in the source is: **Copy MAKELINK=softlink**.

When creating soft links, there are two ways the target of the link can be stored:

- *Absolute* links store the entire path name of the link target. Absolute links can reference files or folders on different drives as well as the same drive.
- *Relative* links store the path of the target relative to the location the link is created. For example, consider the following:
  - **Link path:** *C:\Test\Folder\Link.txt*
  - **Target path:** *C:\Test\Original.txt*

Creating *Link.txt* as a relative link would store the target path as *..\Original.txt* - the *..* notation indicates the parent folder. The advantage of a relative link is the common parent folder (in this example, *C:\Test*) can be renamed or moved, and the link will still work. Relative links are created using the **Copy MAKELINK=relsoftlink** command. If the target path can not be expressed relative to the link path then a regular, absolute link will be created instead.

See the documentation on the [Copy](#) command for more information, and the [Customize](#) section for information on how to configure buttons and hotkeys. You may also like to see the [File Types Drop Menu](#) page - it's possible to add commands to the drag-and-drop menu, and you could, for example, add a **Create Softlink** command to go along with the **Create Shortcut** command that already appears on this menu.



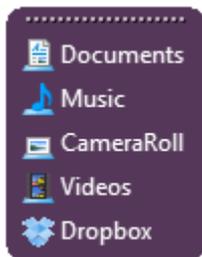
## Floating Toolbars

As well as toolbars (and menus - they're all interchangeable) attached to a Lister, you can have toolbars that float free in their own window. You can use a floating toolbar for many things, for example:

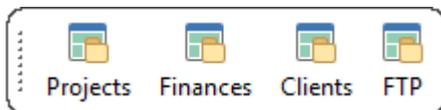
*As a program launcher:*



*For quick access to your commonly used folders:*



*To quickly load a Lister layout:*



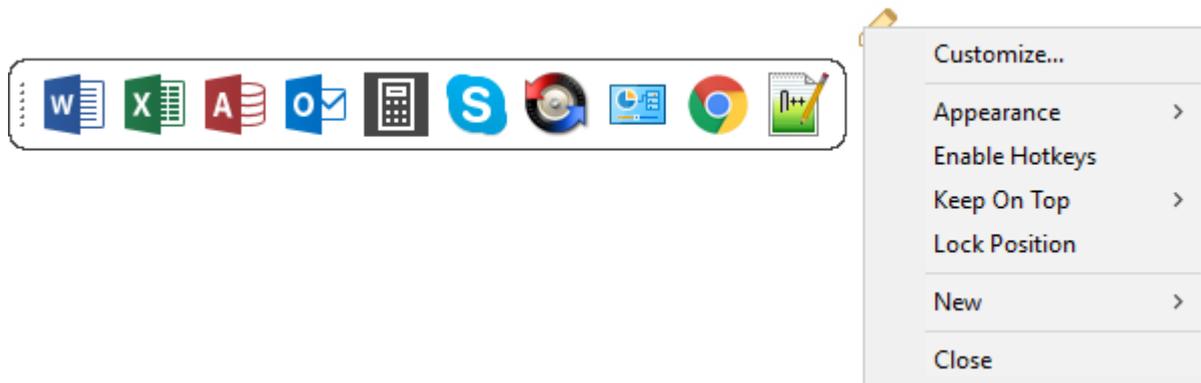
Floating toolbars can be shown whenever Opus is running in the background - you don't need a Lister open to make use of them. They can be set to dock with the edge of the screen, and auto-hide if desired, so they take up little if any space. Any Opus internal command can be used from a floating toolbar, and external programs can be launched just as easily (the same as with a regular toolbar).

As you can see from the screenshots above, you have a lot of control over the appearance of floating toolbars - see the [Controlling Floating Toolbars](#) page for more information.

## Controlling Floating Toolbars

You can change a number of settings that affect the appearance and behavior of a floating toolbar.

When you are in [Customize](#) mode, moving the mouse cursor over a floating toolbar displays a small icon (🔧) will appear to the top-right of the toolbar. Clicking this icon displays the following context menu.



If you're not in Customize mode, you can also display this menu by right-clicking an empty part of the toolbar itself (although this may not be that easy, if for example the border has been turned off).

- **Customize:** Provides a quick way to get back to the Customize dialog.
- **Appearance:** This sub-menu has a number of options affecting the appearance of the floating toolbar.
  - **Frame:** The toolbar will have a solid background and a frame (optionally with rounded corners).



- **No Frame:** The toolbar will have a solid background but no frame - buttons will be hard against the edge.



- **Transparent:** The toolbar will have no background at all - it will appear as if the buttons on it are floating on the desktop. Because transparent toolbars are harder to manipulate the toolbar won't actually appear transparent until you leave Customize mode - instead it will be rendered with a checkerboard pattern. Out of Customize mode the toolbar will display its normal frame if you hold

the **Shift** key down as you move the mouse over it, which lets you reposition transparent toolbars.



- **Glass:** The toolbar will be rendered with a translucent glass background on Vista and Windows 7. Microsoft went off the idea of translucency in Windows 8 so from Windows 8 onwards you get the system "tint" colour without any translucency.



- **Taskbar:** The toolbar will be rendered in the same style as the Windows taskbar (and so obviously looks different in every version of Windows).



- **Rounded Edges:** When the toolbar is set to a type that has a frame, it will have rounded corners (as in the images above) rather than square ones.
- **Same-sized Buttons:** Make all buttons in the floating toolbar the same size. When this is on all toolbar buttons will be the same size as the largest button.
- **Vertical:** The floating toolbar will have a vertical orientation rather than horizontal.
- **Auto-Hide:** This option is only available when the toolbar is docked to the side of the screen. If this is turned on the toolbar will hide out of view unless the mouse moves over it.
- **Enable Hotkeys:** When a toolbar is floating its hotkeys become system global - that is, they work anywhere in Windows without Opus or the toolbar having to be active. This is very useful but also potentially confusing, as the hotkeys will override the use of the same key press in other programs. For example, if a toolbar button had Ctrl-C defined as a hotkey and it was made global, you would no longer be able to press Ctrl-C to copy to the clipboard in any other program! Therefore, by default hotkeys defined for toolbar buttons are disabled when the toolbar is floating. You can enable them with this option.
- **Keep on top:** The toolbar will appear on top of other windows.
- **Lock position:** Lock the position of the toolbar on-screen so it can't be dragged or resized.
- **New:** Create a new toolbar button.
- **Close:** Close the floating toolbar.

You can float multiple copies of the same toolbar if desired, and Opus will remember the settings for each one individually.

# System-wide Hotkeys

A hotkey is a key or combination of keys that you can press to trigger an action - usually the running of a command or launching of a program. Opus lets you create several types of hotkeys.

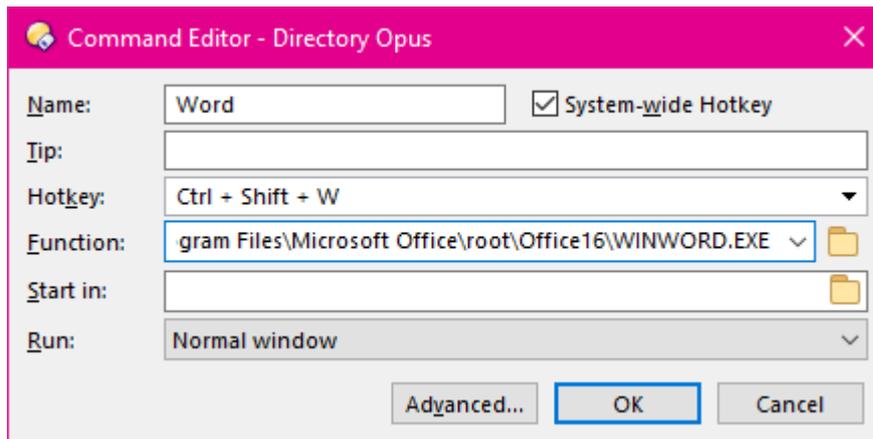
- Toolbar and menu buttons can have a hotkey associated with them. Pressing that key in a Lister will activate the button's function.
- Folders listed in your [Favorites](#) can have hotkeys attached to them. Pressing that key in a Lister will read the favorite folder.
- You can create hotkeys that aren't attached to a button or a toolbar or any other item - they are "pure" hotkeys, that exist in their own right.

True hotkeys are created through the [Hotkeys](#) page of the [Customize](#) dialog. You will find that you can create two types of hotkeys this way:

- Local hotkeys (the default) can only be used when a Lister window is active.
- Global, or system-wide hotkeys, work anywhere in the system - as long as Opus is running in the background.

<input checked="" type="checkbox"/> Ctrl + Shift + Q	Edit Address Book...	Opens the FIP Address B...
<input checked="" type="checkbox"/> Ctrl + Shift + T	New Tabs for Selected	Open a new tab for each
<input checked="" type="checkbox"/> Ctrl + Shift + W	Undo Close Tab	Reopen the last tab or t...
<b>Hotkeys (System-wide)</b>		
<input checked="" type="checkbox"/> Win + Shift + E	New Lister	Open a new Lister
<b>Image Viewer (Keys)</b>		
<input checked="" type="checkbox"/> Backspace / Page Up	Previous Picture	Show VIEWERCMD=pre...

One use for system-wide hotkeys is as shortcuts for launching programs - for example, you could reassign **Ctrl+Shift+W** to launch Microsoft Word.



Because system-wide hotkeys are trapped by Opus no matter which program you are using, you do have to be careful when choosing which keys to use. For example, you could certainly create a system-wide hotkey that activates whenever you press the **Space** key, but then you wouldn't ever be able to type a space! For that reason, we generally recommend using multiple-key combinations for system-wide hotkeys. Opus even allows you to reassign the **Windows**-key hotkeys that Explorer normally reserves for itself. For example, you could remap **Windows+E** to specifically open an Opus Lister even if [Explorer Replacement](#) mode is turned off.

See the [Hotkeys](#) page in the [Customize](#) section for more information about creating hotkeys.

## Exporting to USB

If you purchase the optional USB export licence, you are able to export your installation and configuration to a USB device. This lets you make a "portable" version of Opus that you can run on other computers without needing to install Opus on them. The portable version is entirely self-contained on the export device, and it doesn't make any registry or other changes to the system it's run on.

The USB export feature is an extra cost because it actually grants you an extra licence to run Opus simultaneously on another machine. If you didn't select it when you purchased Opus, you can add it on at any time from the [My Account](#) page on the GP Software website. You can check if the USB export feature is enabled from the **Licence Manager** (from the **Help** menu).

To begin the USB export procedure, select the **Backup and Restore** command from the **Settings** menu. Click the **Export to USB Flash Drive** option, and click **Next** to continue.



The dialog displays a list of your USB devices. As well as normal flash drives and external USB hard drives, Opus also supports [U3](#) devices (although this standard is more-or-less defunct now) - any U3 devices will be indicated with a different icon. Opus also supports devices with the [PortableApps](#) menu system installed on them.

You can choose several different ways of exporting to the USB drive:

- The default behaviour is to export Opus to the selected drive. The program and configuration files will be copied to a new folder in the root of the selected device.
- The **Use this drive as a dongle** option lets you export to a different location - the selected USB drive will only be used to validate the exported licence. For example, a [TrueCrypt](#) encrypted container does not appear to Opus as a USB drive, and so you would not ordinarily be able to select it for export. Using the *dongle* option, you can export Opus to the encrypted drive, and the "parent" USB drive itself will only be used for licencing purposes.
- U3 devices will show in the list as two separate entries - the U3 device (with a U3 icon), and a normal USB drive. These represent the two separate partitions on a U3 device. If you want to make a **.u3p** bundle (which can be installed into the U3 program launcher), you should select the U3 device entry. (Note: U3 is only available on 32-bit computers, due to limitations of the discontinued U3 software itself.)
- [PortableApps](#) devices are detected automatically. When you select a device with PortableApps installed on it, Opus will automatically create the necessary folder structure to integrate the portable version into the PortableApps launcher menu.

Select the drive you want to export to, and **click** Next to continue. At this point, if you selected the **Use this drive as a dongle** option, you will be prompted for the real location to export Opus to.

You will next be given three choices as to which version(s) of Directory Opus to export:

- **Directory Opus 32-bit (x86):** This will export a 32-bit copy of Directory Opus. The exported copy will run on both 32-bit and 64-bit Windows but will have slightly limited functionality on 64-bit. (See above.)
- **Directory Opus 64-bit (x64):** This will export a 64-bit copy of Directory Opus. The exported copy will only run on 64-bit versions of Windows; it will not work at all on 32-bit versions of Windows.
- **Both versions:** This will export both a 32-bit copy and a 64-bit copy of Directory Opus to the same device. The small DOPUS.EXE program placed in the root of the device will automatically select the best copy when you run it. This is the recommended option unless you need to minimize space usage or know that you'll always be using the USB stick with either 32-bit or 64-bit computers.

You will also be asked to select which elements to include with the exported copy, allowing you to reduce space usage and the duration of the export process by excluding unwanted components:

- **Plugins:** Choose which viewer and VFS plugins to include. Only "USB safe" plugins are listed; that means plugins designed not to leave anything behind on the host machine (e.g. in the registry) when running from USB. Some third-party plugins are not USB safe and will not be listed.
- **Languages:** English is always included, but this lets you select additional language files to include if desired.
- **Help files:** Includes this help file.
- **Current configuration:** You can choose to export your configuration as well as the program. If this option is not selected, the exported version will begin with the default configuration. You can also choose which

elements of your configuration to include.

You may be offered the option to export image and sound files used by your configuration. This only applies to files stored below the appropriate **Images** and **Sounds** Opus configuration directories. You can locate these directories by typing the `/dopusdata` [folder alias](#) into the location field. (They may also be stored below the shared `/dopusglobaldata` directory. Elements of the private and shared configurations are merged when exporting to USB.) Images or sounds located in other places will not be copied to the USB drive; the configuration can still point to them, but the exported copy will obviously only be able to load them if they exist in the same place on the computer you run the exported copy of Opus on.

The final confirmation page displays the total size required on the export drive. Click **Next** to proceed with the export.

Normally [Explorer Replacement](#) mode is not available when running the USB version. This is because Explorer Replacement mode requires making changes to the system registry, including some changes that can only be made as an administrator (e.g. from the program installer). However, you can enable a limited form of **Explorer Replacement** mode for the USB version by editing the INI file that is created alongside the portable launcher. When Explorer Replacement mode is turned on in the USB version, some (but not all) of the modifications to the system registry are made, and the old values restored automatically once the portable version exits. You may find that Explorer Replacement mode does not work quite as well in the portable version as in the normal version of Opus - for example, folders launched by some third-party applications may continue to open in Explorer.

The INI file can be found (or created) in the same folder as the USB launcher (called *DOPUS.ini*, normally in the root of the device) for a normal USB export, and in the **F:\PortableApps\DirectoryOpusPortable\App\AppInfo** folder for a PortableApps export (called *DirectoryOpusPortable.ini*). Below

is an example of the format of this INI file.

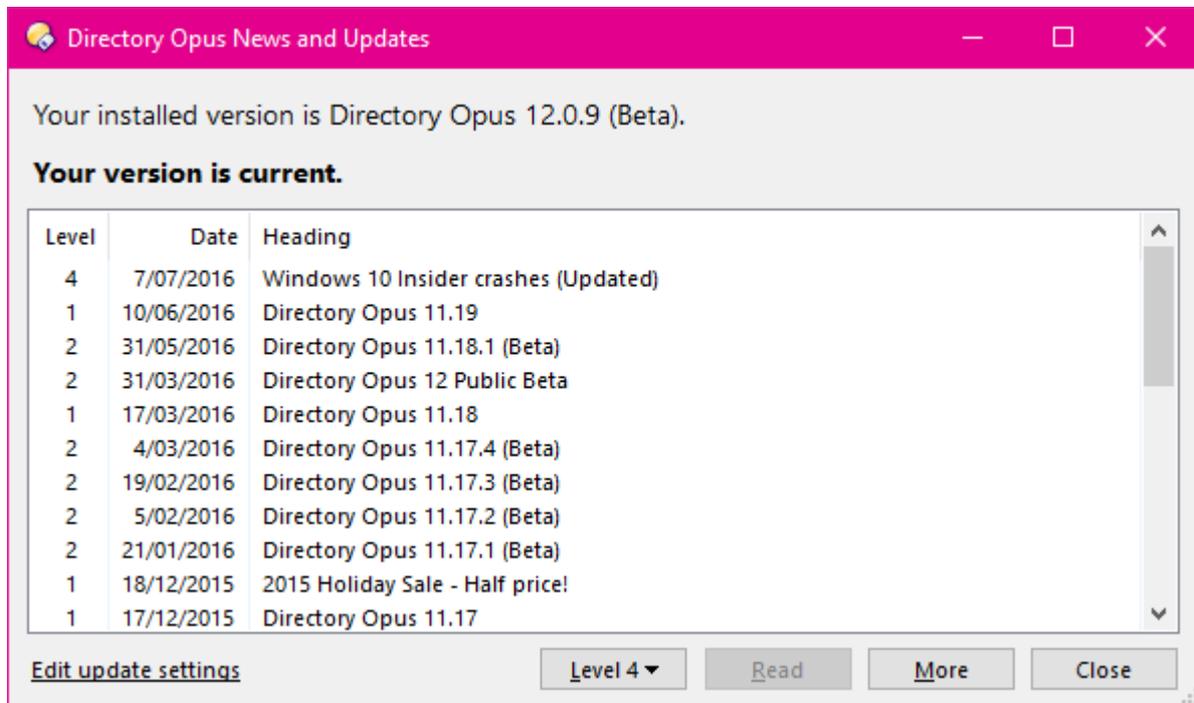
```
# This is an example INI file for the Directory Opus portable
launcher.
#
# The options you can set in this INI file are:
#
# AllowExplorerReplacement: Determines whether the Explorer
Replacement option is available in the portable version.
# "true" or "false" (default "false")
#
# RunFromTempFolder: If AllowExplorerReplacement is true,
determines whether the launcher is copied to the %TEMP% folder
# on the host system and run from there. Allows more secure clean-
up of Explorer Replacement registry changes.
# "true" or "false" (default "true")
#
```

```
# AutoLister: Enable or disable 'auto-Lister' semantics when the
portable version is started.
# "true" or "false" (default "true")
#
```

```
[DirectoryOpusPortable]
AllowExplorerReplacement=false
RunFromTempFolder=true
AutoLister=true
```

# Update Checker

The **Directory Opus News and Updates** tool can check for updates to your installed version of Opus, download them if found, and also displays the contents of the [Opus News](#) RSS feed.



The status of your version (whether it is current, or out-of-date) is displayed at the top of the window. If a new version is found and you have not elected to automatically download updates, a **Download** button will be shown that lets you download the update manually. The progress of any download will be shown, and if an update has already been downloaded and is awaiting installation an **Install** button will be shown. You will also be given the option to **Save** the downloaded update in a location of your choosing, so you can install it later at your leisure.

Below that the most recent items from the RSS news feed are shown. To read a news item either double-click it, or select it and click the **Read** button - it will open in your default web browser. The **Level** drop-down can be used to filter the list of displayed items - the available "importance" levels are:

- **Level 1:** Only the most important items relating to updates to the main program
- **Level 2:** Information about main program updates, and new or modified plugins
- **Level 3:** All important information - program updates, plugins, plus information about bugs and problems, security issues, etc.
- **Level 4:** Everything - all of the above, plus hints and tips and other information.

By default only the most recent items are displayed, but you can click the **More** button to fetch the full list of available items.

Note that the update checker only identifies official releases. There are often Beta versions made available through the [Opus Resource Centre](#), and the update checker will not detect these - although announcements about them will be shown in the RSS feed.

You can configure the update checker using the options on the [Internet / Updates](#) page in Preferences.

- You can set Opus to automatically check for updates (either weekly or monthly).
- The update check can be performed silently, in which case you will only be notified if there is a new version - otherwise, the update checker will be displayed when it starts to check. You may want to leave the silent option turned off if you're interested in seeing the news items from the RSS feed.
- You can have Opus automatically download updates - in this case, you will be prompted to install them when they are ready.

You can also trigger the update check at any time using the **Check for Program Updates** command in the **Help** menu (  ) on the default toolbar.

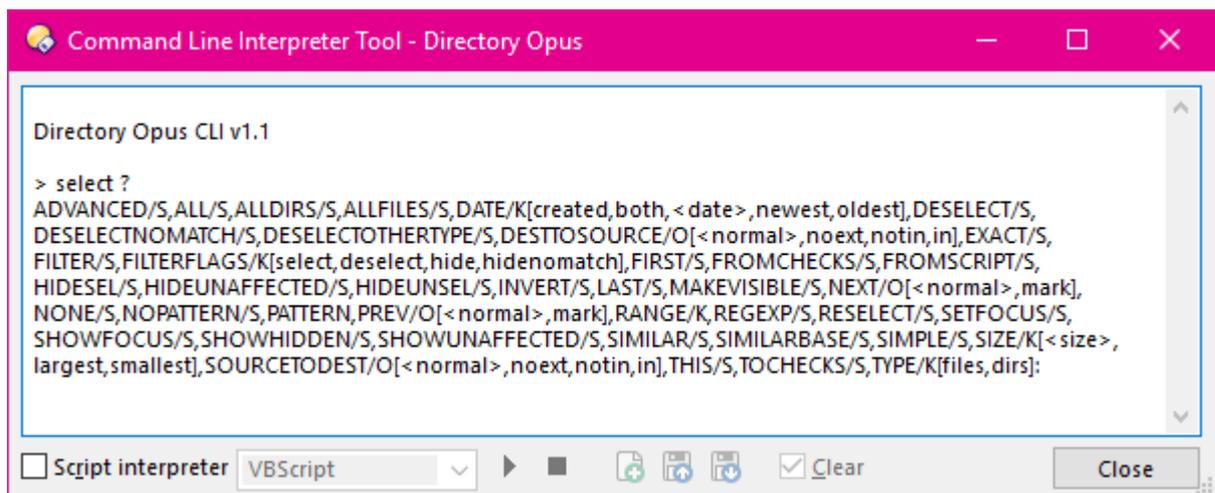


# CLI

The **Command Line Interpreter** (CLI) tool has two main functions:

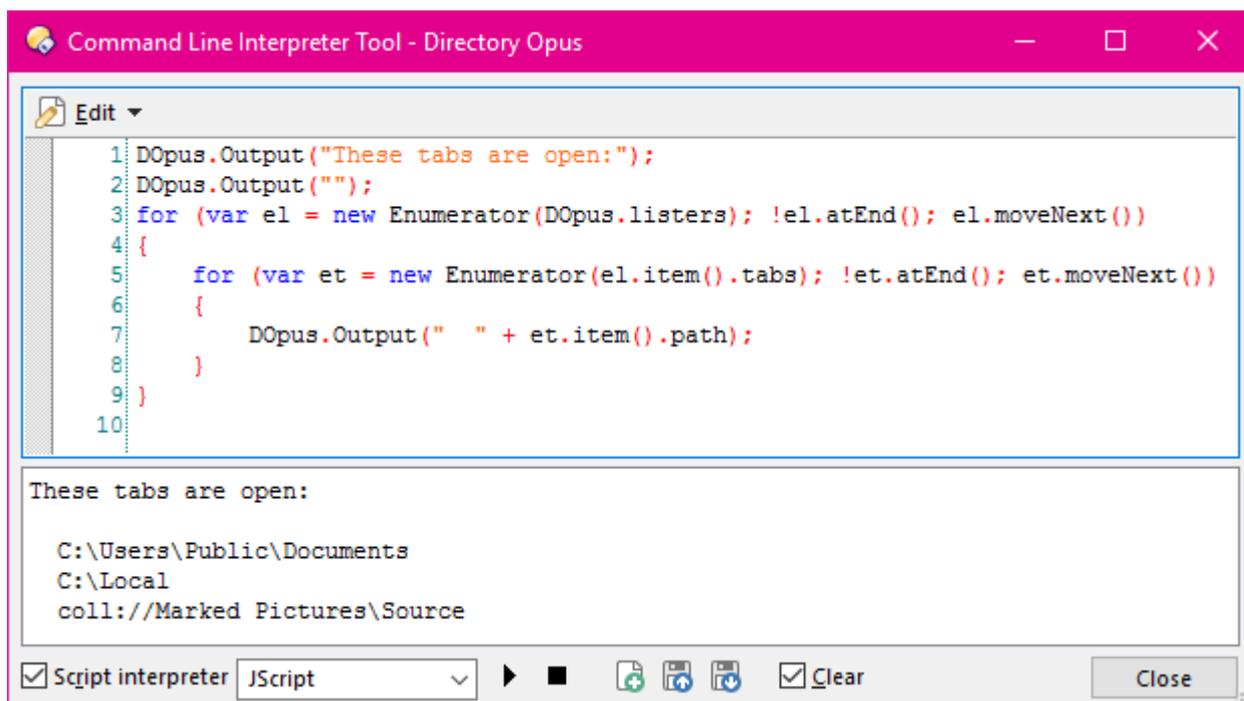
- It provides a very simple keyboard interface to the Directory Opus command set.
- In Script mode, it allows you to design and test ad-hoc [scripts](#).

You can open the CLI using the **Tools / Opus CLI** menu command.



The command line lets you run [internal commands](#) directly without having to configure a button or hotkey first. It is similar to using the [find-as-you-type](#) field in command mode. You can also display the full template for any internal command by entering the command name followed by a ? symbol (as in the screenshot above).

Turning on the **Script interpreter** option puts the CLI in Script mode, which allows you to design and test ad-hoc scripts. The drop-down field lets you select the script language - if the language you wish to use isn't listed, simply type the name in. You will need to consult the documentation for third-party languages as to the name of the "script engine".



Press **F5** or click the **Run** button to run your script. Any errors, warnings or text output from the script (e.g. via the **DOpus.Output** method) will be displayed in output field below the editor.

(Accelerator keys mentioned below are for the English version of Directory Opus. Hover the mouse over buttons to discover their accelerators in your language.)

The **Abort (Alt+A)** button can be used to stop a script which is caught in a loop.

The **New (Alt+N)** button clears both the script and output field so you can start afresh.

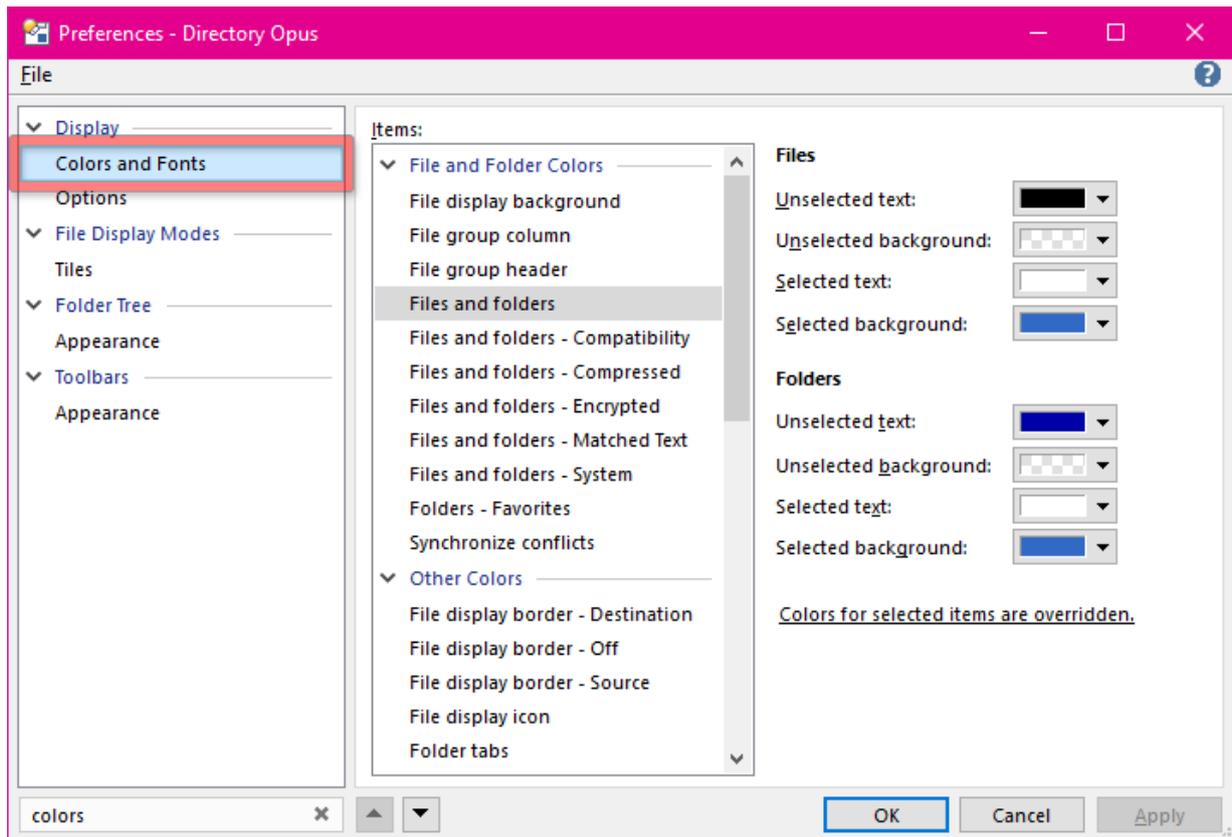
The **Load (Alt+L)** and **Save (Alt+S)** buttons allow you to load and save scripts for use at a later date.

If the **Clear** option is checked, the output field will be cleared each time a script is run; otherwise, output from each run will accumulate.



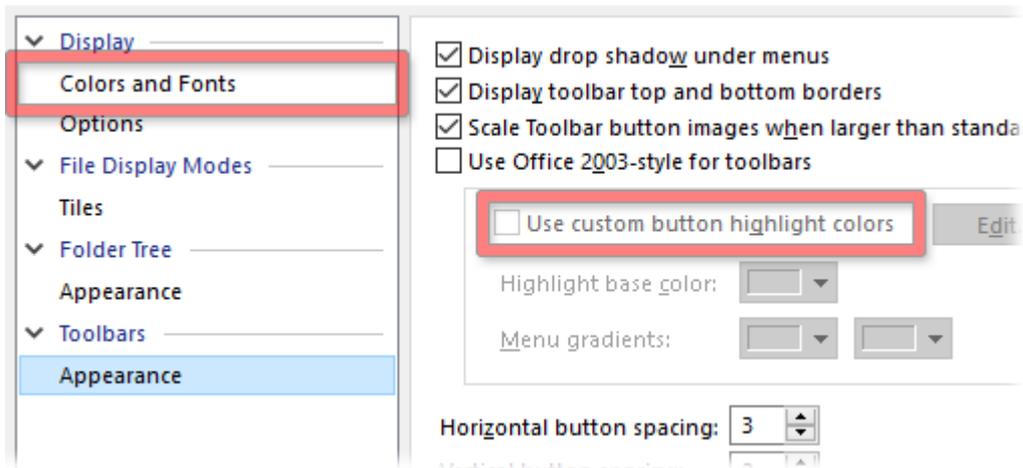
# Preferences

The Preferences dialog is the main interface for configuring Directory Opus. Almost all the options that control how Opus looks and behaves can be found in here. And there's no denying there are a **lot** of options. Once you start to drill through the various categories and pages the number of options swiftly becomes daunting. Luckily there's a nifty feature that can make it easier to find the option you're looking for.



The *Filter* field at the bottom-left of the dialog can be used to search Preferences for those pages and options that match one or more supplied keywords. In the above screen shot we have used the filter to search for items relating to color settings. To use the filter, click in the field (or press **F3**) and type one or more keywords, separated by spaces, and then press enter to activate the filter. All pages that don't match are filtered out, and any matching terms on the remaining pages are highlighted.

In the above screen shot, the entry for the *Colors and Fonts* page is highlighted because the whole page matches the filter. If the whole page doesn't match, but individual controls on a page do, then those controls will be highlighted:



The *Toolbars / Appearance* page didn't match the filter itself, but one control on the page did.

To clear the filter and display all pages, click the little **X** button at the right of the field or activate the field and press the **Escape** key.

At the very top of the Preferences dialog is the **File** menu; this contains a number of useful commands:

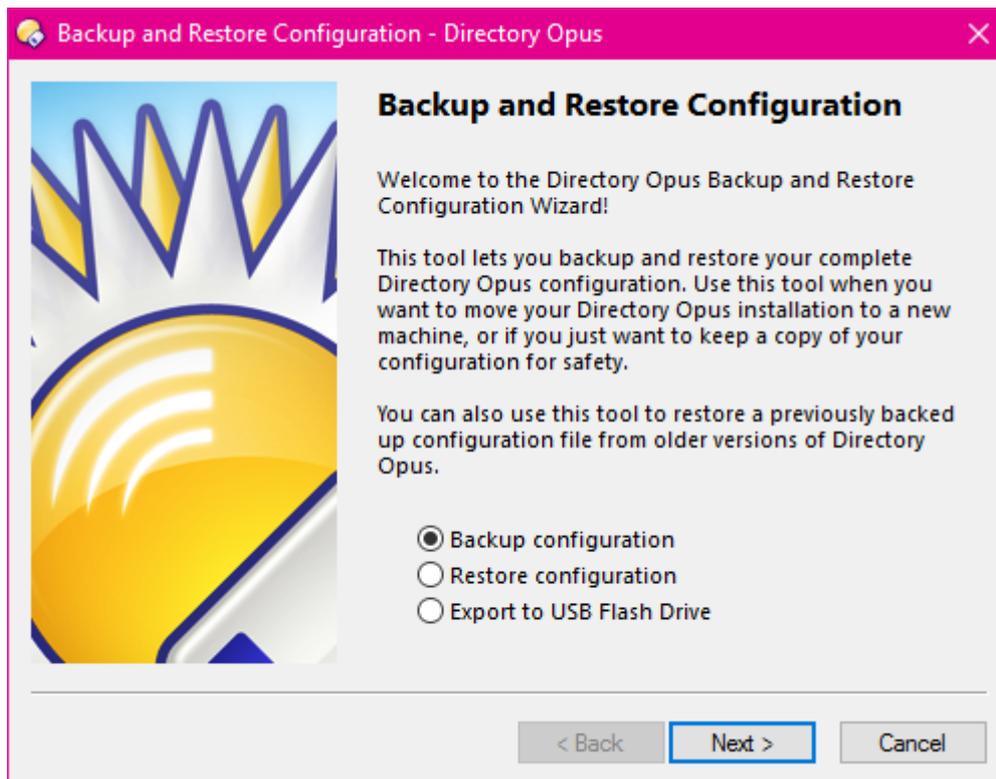
- **Restore Page Settings:** Restores all the settings on the current page to their values when Preferences was first opened. This even works if you have clicked the **Apply** button to apply your changes - you can undo any changes at any time as long as you don't close the Preferences dialog.
- **Reset Page To Defaults:** Resets all the settings on the current page to their default values.
- **Restore All Settings:** Restores all the settings on all pages to their original values (from when Preferences was first opened).
- **Reset All To Defaults:** Resets all the settings on all pages to their default values.
- **Change Configuration Mode:** Lets you change between the two different configuration modes that Opus supports.
  - **Private configuration:** Each user of the computer will have their own Opus configuration, completely separate from every other user's.
  - **Shared configuration:** There is only one global Opus configuration, and all computer users share it (so changes one user makes to the configuration affects all other users).

You can change modes at any time by selecting this command - when Opus restarts it will be operating in the other mode.

- **Backup & Restore:** Access the Preferences [backup and restore](#) system.

## Backing up and Restoring Preferences

Opus stores its Preferences settings on disk, so backing up your Preferences can be as simple as making a copy of the files / folders in question. However Opus includes an integrated Backup tool that automates this process for you, and results in a single backup file that can be used to restore your Preferences in the event that Opus ever has to be reinstalled. If you ever install Opus on a new computer you can also use this to move your configuration across.



To access the *Backup and Restore Configuration* dialog, select the option from the File menu in the main *Preferences* dialog, or from the Settings menu in a Lister.

The three different modes of this dialog are:

- **Backup configuration:** make a backup of your current configuration.
- **Restore configuration:** restore a previously backed-up configuration.
- **[Export to USB Flash Drive](#):** if you have purchased the optional USB export license, export your Opus installation and configuration to a portable USB device.

Note that the *Backup and Restore Configuration* dialog is not available when running an exported portable copy of Opus. If you wish to update the portable copy with a newer configuration from your normal install of Opus, simply export the normal install to the USB device again to overwrite it. (You will be asked whether you want to update just the program or both the program and the configuration.) If you wish to backup the configuration stored on a USB device you can simply archive the configuration directories below the DOPUS folder on the device.

As you click through the wizard interface you'll see there are several options available when backing up that determine how much of your configuration is included in the backup. **Backup images** will include any image files you are using in your configuration, and **Backup sounds** does the same for any sound files. **Backup miscellaneous data** will backup data that, while not forming part of your configuration as such, may still be important - things like your [File Collections](#), [Libraries](#) and [Flickr photoset information](#) for example. Finally, **Backup local state data** will include data that may really only make sense on the local machine - things like your remembered window positions and so on.

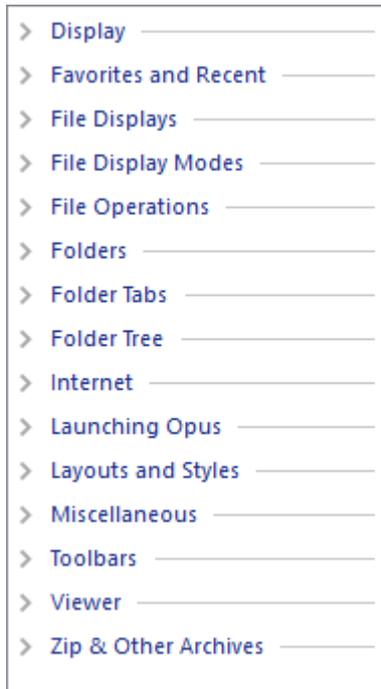
You are also given the option to encrypt the backup file, which you may wish to do to protect things like your stored FTP site passwords.

The same options (images, sounds, miscellaneous data and local state data) are also available when restoring a previously backed up configuration, as well as the **Replace existing configuration completely** option. If this option is not selected, the configuration you restore will be "merged" with your current configuration - if selected, the option causes Opus to delete your existing configuration before restoring the new one.



# Preferences Categories

The various Preferences pages have been categorized in an attempt to group conceptually-related topics. Another advantage of dividing the pages into categories is that individual categories can be collapsed to hide their contents, which is very useful with so many different pages of options!



- [Display](#): Options relating to the appearance of Opus (colors, fonts, etc)
- [Favorites and Recent](#): Options relating to favorite folders or folders that are otherwise important
- [File Displays](#): Options that control the behavior of the area of the Lister that displays files
- [File Display Modes](#): Options specific to certain display modes (Thumbnails, Details, etc)
- [File Operations](#): Options that control file operations (copying, deleting, etc)
- [Folders](#): Options that relate to the display of specific folders (a certain folder can always appear sorted by size, that sort of thing)
- [Folder Tabs](#): Controls relevant to the behavior of Folder Tabs
- [Folder Tree](#): Options that relate to the Folder Tree
- [Internet](#): Email, proxy, update checking, other settings relating to how Opus uses the Internet
- [Launching Opus](#): How Opus is launched, what it does when it's launched, and whether to replace Explorer or not
- [Layouts and Styles](#): Options to control saved Lister Layouts (arrangements of one or more pre-defined Listers) and Styles (pre-defined configuration of an existing Lister)
- [Miscellaneous](#): These are the options that wouldn't fit anywhere else. The page to change the user interface language is in this category.
- [Toolbars](#): Options relating to toolbars, and toolbar icons
- [Viewer](#): Options relevant to viewing images either in the Lister or in the standalone image viewer
- [Zip & Other Archives](#): Options that relate to various archive formats that Opus supports

When the input focus is on the category / page list, you can navigate it using the keyboard in the same way as you can with a tree control - **Cursor Up/Down** to move up and down and **Cursor Right/Left** to expand or collapse a category. You can also press \* to expand all categories at once, or - *twice* to collapse all categories.

## Display

This category contains options relating to the appearance of Directory Opus.

- [Colors and Fonts](#): Edit the colors and fonts used by most elements in the user interface
- [Fields](#): Assign specific colors and font styles to the various information fields that can be displayed in Details mode
- [Images](#): Display images instead of a solid background color in parts or all of the user interface
- [Language](#): Lets you change the user interface language Opus uses.
- [Options](#): Among others, options relating to the way different colors are combined (blended)
- [Status Bar](#): Configure what information is displayed on the status bar at the bottom of the Lister
- [Transition Animations](#): Configure user interface animations that can enhance your experience (i.e. eye candy!)

### Colors and Fonts

This page lets you configure the colors and fonts that are used throughout much of the program. The list of items that can be configured is divided into three categories:

- **File and Folder Colors**: These are the (default - see [Note 1](#)) colors that are used for the display of files and folders in the File Displays.
  - **File display background**: Ok, that's not actually the color for a file or a folder - it's the background color of the file display area. You can also set a different color that's used when a file display is in [destination](#) mode (either in a single file display Lister or as part of a dual display Lister), also when it's off (i.e. a single file display Lister that isn't either the source or destination). You can also set separate colors for when the [quick filter](#) is active. If you leave the destination or off background colors set to *transparent* the main background color will be used for both source and destination file displays (and similarly, if the 'filtered' colors are set to *transparent* the main colors will be used instead).
  - **File group column**: When the file display is grouped, you have the option of showing the group names as headings between each group or in a column on the side. If you choose to use the group column, this defines its colors.
  - **File group header**: When the file display is grouped, and you opt for group names to be displayed as headings between each group, this defines the text and separator color for the group headings. The glyph colors are used for the icons you click to expand or collapse a group, in display modes which allow collapsing, with the "hot" color used when the mouse is over the glyph.
  - **Files and folders**: The text and background colors for files and folders that don't derive a color from some other setting. You can specify colors for when these items are selected as well as in their unselected state. Using visual styles to draw items and selection boxes may override some of these colors; if this is happening, a clickable message will appear below the last color, and clicking it will take you to the option you can turn off if you don't want this to happen.
  - **Files and folders - Compatibility, Compressed, Encrypted and System**: Optional colors for various types of files that will override the defaults if activated. [Compatibility files](#) are those from "compatibility folders" when Opus displays the contents merged with the main folder (that's a bad

explanation, follow the link for a proper discussion of this concept). Compressed files are those with the **C** attribute, encrypted files are those with the **E** attribute, and system files are those with both the **H** and **S** attributes set.

- **Files and folders - Matched Text:** Colors used to highlight text matches in filenames when using the [Find-as-you-Type](#) field (and the [Highlight matches](#) option is turned on).
- **Folders - favorites:** Optional colors that are used for any folders you have added to the [Favorites](#) list.
- **Synchronize conflicts:** Colors that are used with the Synchronize tool to indicate files which are in conflict.
- **Other Colors:** These are colors used elsewhere in the user interface - they're not related to file displays themselves, but to other parts of the Lister.
  - **File display border - Destination, Off, Source:** Defines the text and fill colors used for the File Display border (the "title" bar displayed at the top of the file display) in each of the [three Lister states](#).
  - **File display icon:** Defines the color used for the Opus logo icon that is shown in the File Display border, as well as that used for the Lister's window icon. You can define different colors for when the Lister is set as source, destination or off, as well as a separate color for when the Lister is in dual-display mode.
  - **Folder tabs:** Text and background colors for folder tabs, as well as the background color for the bar (empty space) behind the tabs. You can also configure a separate "hot" color for when the mouse is over a tab.
  - **Folder tabs (linked):** Lets you configure the colors used when folder tabs are linked. You can configure eight separate linked tab colors. Each pair of linked tabs uses the colors in the order they are specified here.
  - **Folder tree:** Text and background colors for items in the folder tree, if not overridden by above colors (see [Note 2](#)). The glyph colors are used for the icons you click to expand or collapse a branch; the hot glyph color only applies when using visual styles.
  - **Folder tree (destination):** When a dual-display Lister has two trees, or a single-display Lister with a tree is set as the destination, you can use this to specify different colors for the tree associated with the destination file display.
  - **Graphs - Free Space:** Colors used to draw the free-space graphs in places like the This PC (aka My Computer) folder and the status bar. The "warning" color is used when space gets below a threshold (usually 10% free). You can set the background color to transparent if you don't wish to have one.
  - **Graphs - Relative Age:** Colors for the Relative Age graphs Opus can display next to things in Details and Power modes. There are separate colors for files and folders. You can set the background color to transparent if you don't wish to have one.
  - **Graphs - Relative Size:** Colors for Relative Size graphs Opus can display next to things in Details and Power modes. There are separate colors for files and folders. You can set the background color to transparent if you don't wish to have one.
  - **Jobs Bar:** The text and background color used for the [jobs bar](#).
  - **Metadata pane:** Colors for the metadata pane (text, background, and the color used to provide alternating colored stripes).
  - **Pane borders:** Colors used for the borders surrounding the various panes (sub-windows) in the Lister (Folder Tree, Viewer pane, etc). Also has options to **Use for lister column headers** and **Use for lister scrollbars**. (The scrollbar option is only available if Windows visual styles are active, and is also disabled if WindowBlinds is detected).
  - **Status bar:** Default text and background color for the Lister status bars (see [Note 4](#)). Normally, visual styles are used to draw the status bar's frames and borders; however, if a dark background

color is chosen, Opus will draw the frames and borders itself using the **Pane borders** colors, since visual styles usually do not look good on dark backgrounds.

- **Thumbnail relative dimensions:** Color used to render the [thumbnail relative dimension bars](#).
- **Toolbar and menu defaults:** Colors used to draw toolbars and menus.
  - If **use system colors** is turned on, Opus uses visual styles to draw most toolbar and menu elements; if it is off, Opus still uses visual styles to a degree (re-coloring them as needed) but also draws some elements itself where it makes sense. Choosing a light text color or dark background color can influence how much is custom-drawn vs drawn using visual styles, since visual styles tend to only look good with dark-on-light colors.
  - The text and background colors can be [overridden on a per-toolbar basis](#).
  - The "shadow" and "highlight" colors are used to render toolbar borders and separators, and you can set them to transparent to avoid one or both borders entirely.
  - "Selected" and "Toggled" are used to render the background of toolbar buttons and menu items in different states. If you set them to "use default" they will use the system colors.
  - "Menu background" allows you to use a different background color for menus; choosing "Use Default" will give menus the same background color as toolbars. Note that some menus are drawn by Windows itself and will not be affected by any of these colors.
  - "Menu frame inner" and "Menu frame outer" are used to draw the frame around the outside of pop-up menus, one inside the other. You can set the inner color to transparent if you only want a single-color, one pixel border. Setting both inner and outer colors to transparent will remove the border entirely.
  - "Field background" lets you change the background color of the breadcrumbs path field and search field on Lister toolbars. If no background color is specified, the fields are drawn using Windows visual styles. If a background color is specified, Opus takes over and also uses the other toolbar colors to draw the fields (e.g. the text color also applies to text in the fields, if the field's background color is being overridden).
  - "Field glyphs" lets you change the color of the arrows between path components on the breadcrumbs path field.
- **Tree highlight path to selection:** Colors used in the folder tree to highlight the path to the currently selected folder (if the appropriate options are enabled).
- **Viewer pane background:** Background color for the Viewer pane when it's empty (see [Note 5](#)).
- **Fonts:** These control the fonts that are used in various parts of the user interface.
  - **File display:** The font used to display files and folders in the File Display in all modes except Thumbnails
  - **File display border:** The font used in the File Display border at the top of the file display
  - **File display groups:** The font used for group headings when the file display is grouped
  - **File display header:** The font used for the File Display header in Details mode
  - **Folder tabs:** The font used for Folder Tabs
  - **Folder tree:** The font used for the Folder Tree
  - **FTP log:** The font used for the FTP output log
  - **Info Tip:** The font used for file and folder info tips (tooltips shown when hovering the mouse over a file or folder)
  - **Jobs Bar:** The font used for the [jobs bar](#).
  - **Metadata pane:** The font used in the metadata editor.
  - **Pane borders:** The font used for the title of the various panes (sub-windows) in the Lister (Folder Tree, Metadata pane, etc)

- **Rename macro builder:** The font used for the [rename macro builder](#) (must be fixed-width)
- **Status bar:** The font used for the Lister status bar
- **Thumbnail labels:** The font used for item labels when the File Display is in Thumbnails mode

Some notes relating to the above:

1. You can specify colors for files and folders on a per file-type basis or for specific files and folders, using the [Labels](#) system - this will override colors specified on this page.
2. The Folder Tree will use the colors set in the **File and Folder Colors** section by default, but you can set it to ignore these color settings by turning off the [Use configured file display colors for tree items](#) option, in which case the colors in **Other Colors / Folder tree** will be used.
3. On Vista and above, the file displays and folder trees use the Windows Theme by default to display item highlights by default. This overrides configured background colors for *selected* files and folders (text color, and non-selected background colors can still be changed). You can turn off the use of the Windows Theme if you like with the [Use visual style to draw items](#) option.
4. On Vista and Windows 7 (but not later versions) the status bar will be set to use "glass" mode by default, which overrides the background color setting - turn off the [Show glass status bar option](#) if you prefer a more traditional status bar.
5. The viewer pane background color is only used when the Viewer pane is empty. When the viewer is displaying a picture, and the picture doesn't fill the display completely, or has transparent areas, a separate color is used to fill the background. This is configured on the [Viewer Pane](#) page.

## Fields

The Fields page lets you configure colors that are used for the display of specific columns when the Lister is in Details or Power mode. For example, you could have the Type field displayed with a different background color, or the Size field in bold. For each column (if it's enabled), you can specify text and background colors, and select from bold, italic and underline styles.

The **Default width** option lets you configure a default width for every field. This is especially useful in configuring the default width of the *Thumbnail* column.

The **Reverse standard sort direction** option allows you to change the way each field sorts by default. When the option is off, as is the case for most fields, the field will sort from smallest (at the top) to largest (at the bottom) the first time you click its column header. The same field will sort from largest to smallest if you click its column header a second time. For example, the filename field defaults to sorting A-to-Z on the first click and Z-to-A on the second. When **Reverse standard sort direction** is turned on for a field, you get the opposite: Largest to smallest on the first click; smallest to largest on the second. Date fields default to reverse sorting, giving you the most recent (largest date) files at the top on your first click. Size fields also do this, giving you the largest files at the top on your first click.

The **Alignment** option allows the alignment (left/right/center) of individual columns to be modified from the defaults.

The **Invert graph** option (shown for columns that display bar graphs) lets you invert the meaning of the graph (e.g. instead of the widest graph indicating the oldest file, it would indicate the newest file).

The **Current sort field** entry is a special set of colors and styles that applies to whichever field the file list is currently sorted by (or if multiple sort fields are defined, the primary sort field).

The list of fields allows you to see the field colors and states at a glance. If the checkbox is not set for a field then its colors, while visible in the list itself, are not currently enabled and will not affect the colors used to display files. If an arrow appears on the right of a field's name then that field is set to sort in reverse. The checkboxes and arrows are independent of each other; that is, you can enable reverse sorting without having to turn on special colors, and vice versa.

## **Images**

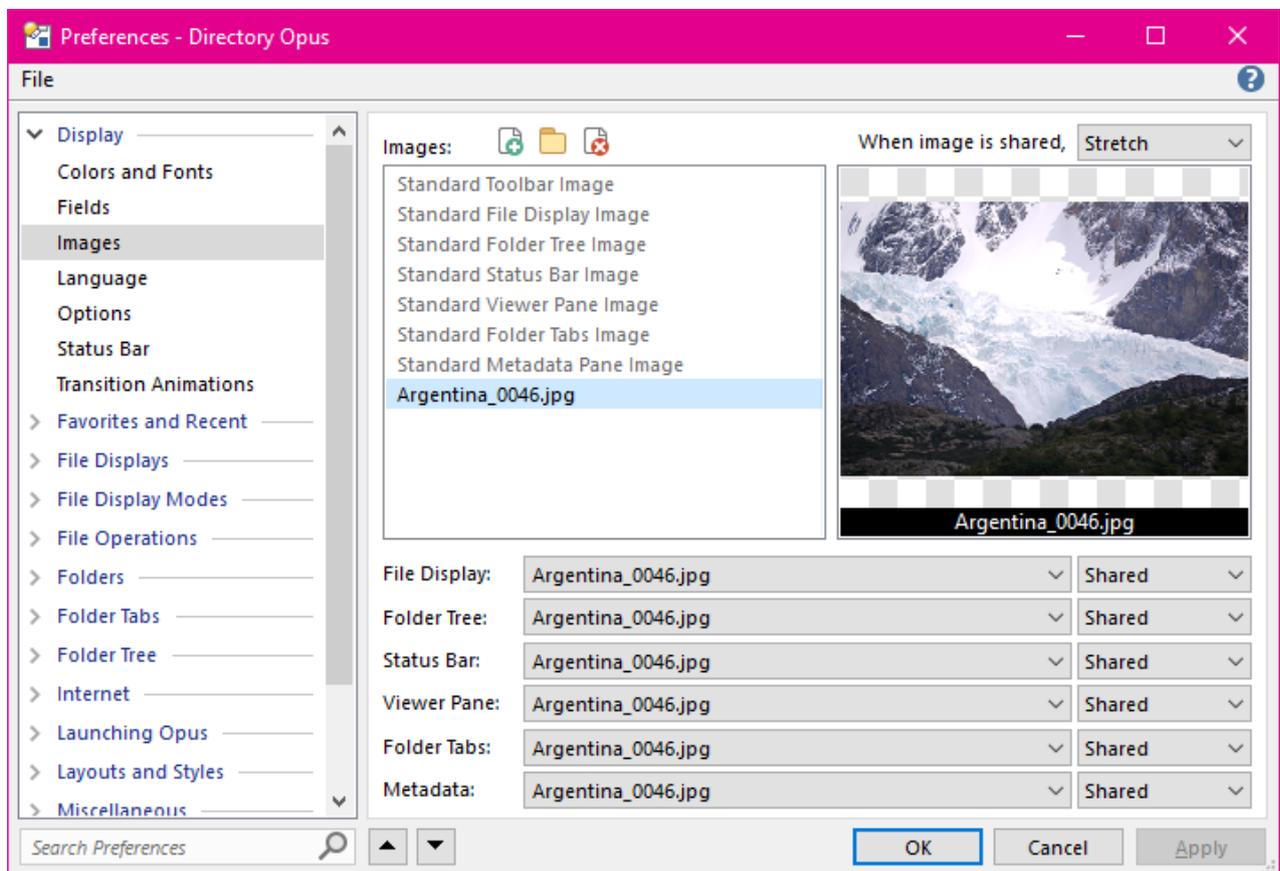
You can use the Images page to define images that Opus will display instead of a solid background color behind elements of the Lister. An image file on disk (e.g. a JPEG file) has to be added to the list of Images in Preferences before it is available for a Lister element to use. There are two types of images in this list (although in practical terms they are the same):

- **Standard** images are placeholders for each of the elements that can have an image assigned to it. Although no images are configured by default, each element is set by default to use its standard image. For example, the File Display is set by default to use the *Standard File Display Image*. There are two reasons for this:
  - To assign an image to a Lister element, generally all you need to do is edit the standard image setting for that element.
  - [Lister Themes](#) are able to make changes to toolbar images without needing to know what toolbars you have turned on (a theme can define an image for a specific toolbar, but you may not have that toolbar turned on in your configuration - whereas, if all toolbars are set to use the standard image, a theme is able to change them automatically).
- **User-added** images are images that you have added to the Images system yourself. These appear in the Images list and can be assigned to any Lister element just like the Standard images.

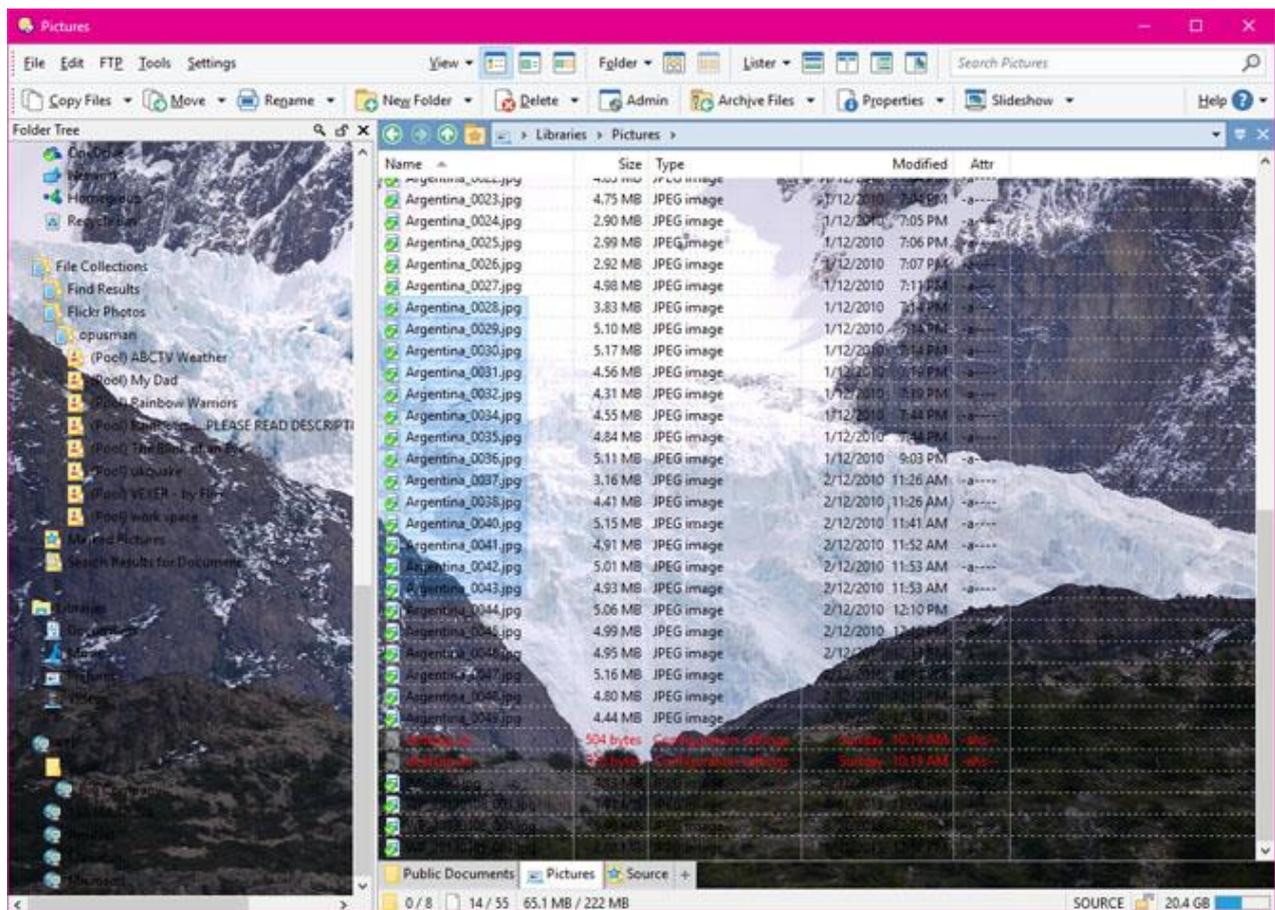
Images can be added, assigned or removed using the three buttons above the list, in the top half of the dialog. You may also drag & drop an image file to the list to add a new image, or to the thumbnail on the right to replace the currently selected image.

Once an image file has been added to the list (or assigned to one of the Standard image types) it can be assigned to one or more of the Lister elements using the drop-downs at the bottom of the page. For each element, you can specify how the image is used:

- **Tiled:** The image is tiled across the element; if it is smaller than the dimensions of the element, it is repeated to fill the bounds of the element
- **Stretched:** The image is stretched to the dimensions of the element (either smaller or larger)
- **Top/Left:** The image is anchored to the top/left corner of the element, and is clipped if it is larger than the element
- **Top/Right:** The image is anchored to the top/right corner
- **Bottom/Left:** The image is anchored to the bottom/left corner
- **Bottom/Right:** The image is anchored to the bottom/right corner
- **Center:** The image is centered within the element
- **Shared:** The image is shared with other elements across the whole Lister



The screen shot above shows an image that has been added to the list, and set to be **Shared** by all elements. When an image is set to be shared by an element, the image is displayed relative to the whole Lister. In effect, this lets you have one single image that is displayed behind the whole Lister, rather than behind individual elements of the Lister. When an image is shared the way it is drawn is governed by the setting for the **When image is shared** drop-down at the top of the page - images can either be tiled across the whole Lister, or stretched to fill the Lister completely, as in the screen shot below:



Images for toolbars must be added to the list in Preferences like those for other elements, but they are assigned to toolbars using the [Customize](#) dialog. You can also assign images to specific folders using the [Folder Formats](#) system - so for example, you could make any folders containing mostly music files display a music symbol as a background image.

## Language

This page lets you change the user interface language used by Opus. To change language, select the desired language from the list, and then click the **Change Language** button below. When Opus restarts, the language will be changed. Opus attempts to automatically translate your toolbars and menus as much as possible - any commands from the default toolbars that you haven't edited the label for will be translated to the new language.

## Display Options

This page contains various options relating to the appearance of Listers. For all items that include an "opacity" setting, this is used to specify the level of blending that is performed, from 1% (nearly transparent) to 99% (nearly opaque).

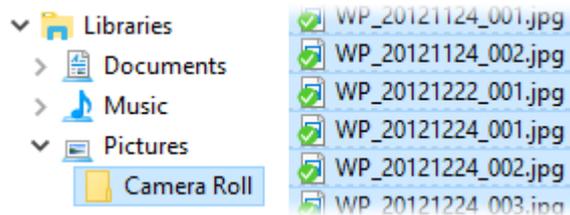
- **Blend file background colors with background:** Useful when you have defined a background image or color for the File display, this option causes the solid background colors of files and folders to be blended with the background of the display. Note that if visual styles are in use on Vista or Windows 7, this option will have no effect for selected items (as the visual style will override the item's selected background color).
- **Blend file selection rectangle:** When you click in an empty area of the file display and drag out the selection rectangle ("marquee") it will be drawn as a filled rectangle blended with the background, rather than just as an outline.
- **Blend row and column background colors:** In Details and Power mode, when you have defined colors for various fields, this option causes those colors to be mixed with the ordinary file and folder background colors rather than overriding them.
- **Blend selected items with column colors:** Blends sort-column and other custom field/column colors with each other and with the background colors (if any) of unselected files.
- **Enable background images in virtual folders:** With this option on, Opus will try to use your background image settings when it is displaying a [virtual folder](#) (e.g. Network Neighborhood). As these folders are provided by the operating system and Opus merely hosts them as a container, it won't necessarily work!
- **Fade selected item colors when file display does not have focus:** When the file display isn't the active window, the background colors of selected files and folders will be faded to indicate the non-active nature of the window. The opacity setting is only effective when not using visual styles - when styles are in use, this option is a simple on/off switch.
- **Fade selected item colors when tree does not have focus:** The same as the above option, except it applies to the Folder Tree rather than the file display.
- **Use visual style to draw items:** Except on XP (or when themes are disabled), this option renders files and folders in both the tree and the file display using the Windows Explorer theme. Turning this on will override settings for file background colors for selected items (as the selection indicator now comes from the theme rather than from Opus). See below for an example of the difference.
- **Lister title bar:** You can define what is displayed in the Lister title bar. By default this just shows the name of the current folder, but you can opt to display the full path name of the current folder, and/or the name of the [layout](#) that the Lister came from (if any).
- **Custom title:** This option lets you completely configure the string used in the Lister title bar. If you turn this option on, the text you provide in the *Custom title* field will be used to generate the Lister title string - you can even leave the field empty if you don't want a title!

You can use several special "tokens" in the title string to insert various pieces of information:

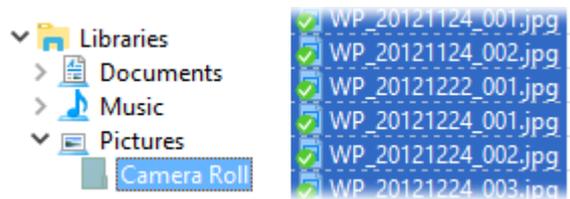
- **%P** - full path of the current (source) folder
- **%N** - name of the current (source) folder
- **%R** - drive root of the current (source) folder
- **%D** - full path of the destination folder
- **%M** - name of the destination folder
- **%G** - target if the folder is a junction or softlink
- **%1** - full path in the left file display
- **%2** - full path in the right file display
- **%3** - folder name in the left file display
- **%4** - folder name in the right file display
- **%L** - name of the Layout the Lister came from (if any)
- **%T** - complete original title (useful for simply adding a prefix or suffix to the title)
- **%%** - insert a literal % character

- **Drag image opacity:** Specifies how transparent files and folders are when they are being dragged.

Use visual style to draw items on:



Use visual style to draw items off:



## Status Bar

You can tell Opus exactly what information to display on the status bar. By default a count of files, folders (and how many are selected) is shown, but you can configure the display to include much more information, including total playing time for music files and bar graphs to represent things like the proportion of space selected files would take up on a DVD. The text that tells Opus what to display on the status bar is known as its *definition*.

There are a three different definition styles available. You can choose either:

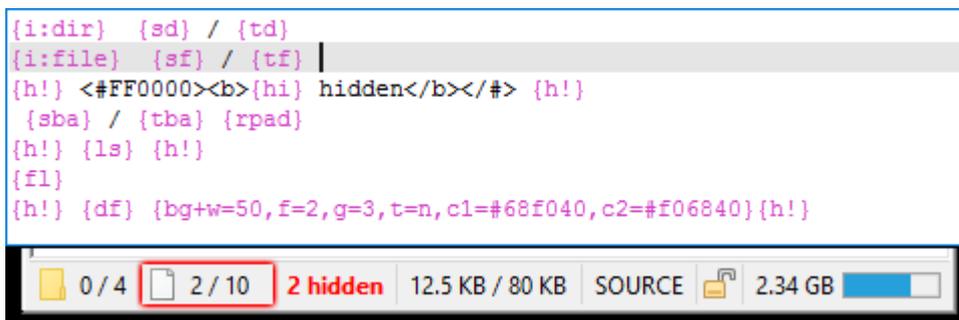
1. A single status bar *definition* that's used everywhere
2. Two *definitions* - one that's used in single display mode, and one that's used in dual display mode
3. Two *definitions* - one that's used for the left file display and one that's used for the right file display

The options at the top of the page determinate which combination you want to use:

- **Use two independent status bars when in dual display mode:** If this is turned on, then a dual-display Lister will show two separate status bars, one for the left/top and one for the right/bottom file displays. If turned off, a Lister only shows a single status bar in both single and dual-display modes.

- **Keep status bars at the bottom of the Lister:** In a dual-display Lister with two separate status bars, the status bars are normally displayed at the bottom of the file displays themselves. Often this means they are at the bottom of the physical window, but this isn't always the case - for example, a toolbar could be positioned between the status bars and the bottom of the window. This option lets you move them to the very bottom of the Lister. If the option to use two independent status bars is turned off then the single status bar is always displayed at the bottom of the Lister.
  - **Glass background:** If the status bar is at the bottom of the Lister it can be shown with a "glass" background to match the title bar of the window.
- **Separate definitions:** The exact label and meaning of this option changes depending on the state of the **Use two independent status bars** option. If this option is turned off, Opus uses definition style #1 as described above - otherwise:
  - **...for single and dual display modes: With independent status bars turned off,** this option sets Opus to use definition style #2 as described above.
  - **...for left/top and right/bottom file displays: With independent status bars turned on,** this option sets Opus to use definition style #3 as described above.

Depending on the state of the **Separate definitions** option there will either be one or two multi-line text fields on this page, for you to edit the definition text. Each line of the status bar definition corresponds to a "section" on the status bar. You can align or pad sections, and even have sections hidden based on simple conditions.



This image shows the default status bar definition. There are seven lines in the text field, which means seven separate sections in the status bar. As you can probably tell, you tell Opus which information to display using a series of `{..}` codes. We won't document all those codes here - there's a [full list in the reference section](#). Luckily you don't actually need to know most of the codes, as the **Codes** drop-down at the top of the dialog provides a full list with descriptions of their meanings.

Below the text field is a small preview of the status bar - this updates in real time, so as you make changes to the status bar definition you can get an idea for how it will look in real life. The red highlight indicates the section you are currently editing.

The first line of text in the status bar definition shown above corresponds to the first section in the preview, and so on.

- `"{i:dir} {sd} / {td}"` displays the number of folders selected and the total number of folders. `{sd}` corresponds to the number of selected folders (selected **directories**) and `{td}` to the total number (**total directories**). `{i:dir}` causes a folder icon to be displayed - you could label this section with text if you like (e.g. **Folders: {sd} / {td}** or similar) but using an icon saves space (and looks neater!)
- `"{i:file} {sf}/{tf} files"` similarly displays the number of files, both selected and total, with a standard file icon as a label.
- `"{h!} <#FF0000><b>{hi} hidden</b></#> {h!}"` displays the number of hidden items, but only if there are any. The `{h!}...{h!}` codes are a special marker that means "if all of the codes between these two markers evaluate to zero, hide the entire phrase". So if there are no hidden items, nothing will be displayed at all. The `<#FF0000>` sets the text color to red and the `<b>` tag makes the font bold.
- `"{sba} / {tba} {rpad}"` displays the total size of all selected items and the total size of all items. The code `{sba}` stands for "selected **bytes automatic**" - that is, selected byte count, displayed automatically as bytes, kilobytes, megabytes, etc. depending on the actual size - similarly, `{tba}` stands for "total **bytes automatic**". `{rpad}` is a code that means "right-pad this section". The section will be expanded to fill all available space in the status bar (so you could say this also has the effect of right-justifying any subsequent sections)
- `"{h!} {ls} {h!}"` is another possibly hidden section. `{ls}` displays the current [source or destination state](#) of a single-display Lister (in the preview above, *SOURCE*). In a dual-display Lister this code returns an empty string and so the `{h!}` markers will cause the section to be hidden.
- `"{fl}"` is the code responsible for displaying the [format lock](#) icon, which you can click to quickly lock or unlock the current folder format.
- `"{h!} {df} {bg...}{h!}"` displays information about drive space in the final section. `{df}` displays the amount of free space on the current disk as a number, and the `{bg}` code is responsible for displaying the bar graph. See the [Bar graphs and Percentages](#) page for more information about configuring bar graphs.

See the [Status Bar Codes](#) reference section for the full list of codes you can use.

## Transition Animations

On Vista and Windows 7, Directory Opus uses the desktop composition feature to provide transition animations when doing things like changing folder, opening or closing Lister panels, etc. These are purely cosmetic (eye candy!), but because they use the DWM they are hardware accelerated and so won't slow down your computer. You are free to turn them off using the options on this page if you like.

You can configure the animations used for six basic actions. When you open one of the drop-downs and move the mouse over the list of animation types, a small preview is displayed of the animation effect.

- **Navigate Into:** When you navigate into a folder by double-clicking it.
- **Navigate Up:** When you go up to the parent folder (e.g. by using the Up button).
- **Navigate Back:** When you go back to the previous folder (e.g. by using the Back button).
- **Navigate Forward:** When you go forwards to the "next" folder (e.g. by using the Forward button).

- **Navigate Other:** When you navigate by other means, e.g. by selecting a folder in the Folder Tree, or a folder from the Favorites list.
- **Prefs Pages:** This affects the Preferences dialog, not the Lister - it defines the animation used when changing from one Preferences page to another.

Transition animations (and previews) are not available on Windows XP or when desktop composition is disabled (e.g. Aero Basic).

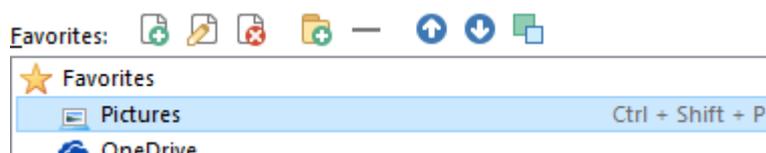
## Favorites and Recent

This category contains options relating to favorite folders or folders that are otherwise important to you. You can add folders to your favorites list, assign aliases for quick access, and control the color of important files and folders.

- [Favorites List:](#) Create and maintain a list of your favorite folders that can be easily accessed from the Favorites menu in a Lister.
- [File And Folder Labels:](#) Assign colors and font styles to important files and folders.
- [Folder Aliases:](#) Assign aliases to important folders to make them easier to access throughout Opus.
- [Jump List:](#) Configure the "jump list" menu for Directory Opus on Windows 7 (this is in this category because you can add folders to the jump list).
- [Recent List:](#) Control the behavior of the "recent list", a list of your most recently visited folders.
- [SmartFavorites:](#) Control the Directory Opus "SmartFavorites" system, that attempts to learn your most used folders automatically.

### Favorites

This page lets you edit your Favorites list, which is basically the list of folders that is displayed in the Favorites menu in an Opus Lister. You can add as many folders to this list as you like, and they can be arranged in branches (sub-folders) if desired. You can also assign hotkeys to the folders from the list, which lets you access a folder by pressing a key without having to define a separate hotkey for it.



You can see here that we have added a favorite folder. You can assign names to your favorites, so that the full path doesn't have to be displayed in the Favorites menu - here we have named this favorite *Pictures* - the folder it actually points to is displayed at the bottom of the page. We have also assigned a key to it.

The toolbar buttons at the top are how you make changes to the favorites list:  (add new favorite),  (edit existing favorite),  (delete favorite),  (create a branch for the favorites menu),  (insert a separator in the menu),  (move favorite up the list),  (move down the list) and  (sort the favorites list). You can also use drag and drop to rearrange the list.

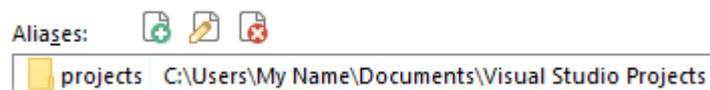
When you edit a favorite (double-click it, or select it and click the  button) you can assign a name to the favorite, assign a hotkey that lets you access the favorite folder from the keyboard, and also change the path that the favorite refers to.

If you put '&' before a character in the name, that character will become an accelerator key. For example, if you name a favorite *&Videos* and then open the Favorites menu, the V key will activate that item. This is different to adding a hotkey to a favorite: Accelerators only work when the menu is open, while hotkeys are always active even if the menu is closed. If you wish to use a literal & character, use two next to each other: *Cats && Dogs* will display as *Cats & Dogs* in the menu.

By default the favorites list isn't sorted, and new favorites are added to the bottom of the list. You can use the  button to sort the list at any time, or turn on the **Automatically sort newly added Favorites** option at the bottom of the page to have new favorites sorted automatically.

## Folder Aliases

The Folder Alias system lets you assign aliases to folder paths that can then be used elsewhere in Opus. This lets you refer to long, complicated paths using a simple, easy to remember name. For example, instead of a function that reads **Go C:\Users\My Name\Documents\Visual Studio Projects** you could assign an alias to this folder, and replace the function with **Go /projects**. Aliases can be used when defining toolbar icons, background images, you can type an alias into the [Find-As-You-Type](#) field or the location field to browse to that location - basically anywhere in Opus that you would ordinarily use a full pathname, you can use an alias.



Use the  button to add a new alias, the  button to edit an existing one and the  button to delete one.

Directory Opus also includes a large number of built-in aliases that refer to special system folders. For example, **/desktop** is an alias for the desktop folder, and **/dopusdata** is an alias for the main Opus configuration directory. These aliases can't be edited and by default are hidden from the list in Preferences, but you can display them by turning on the **Show built-in aliases** option at the bottom of this page.

## ***Jump List***

This page lets you configure the Windows 7 Jump List - the menu that is displayed when you right-click on the Directory Opus icon on the taskbar, or that appears in the start menu when Directory Opus is pinned there.

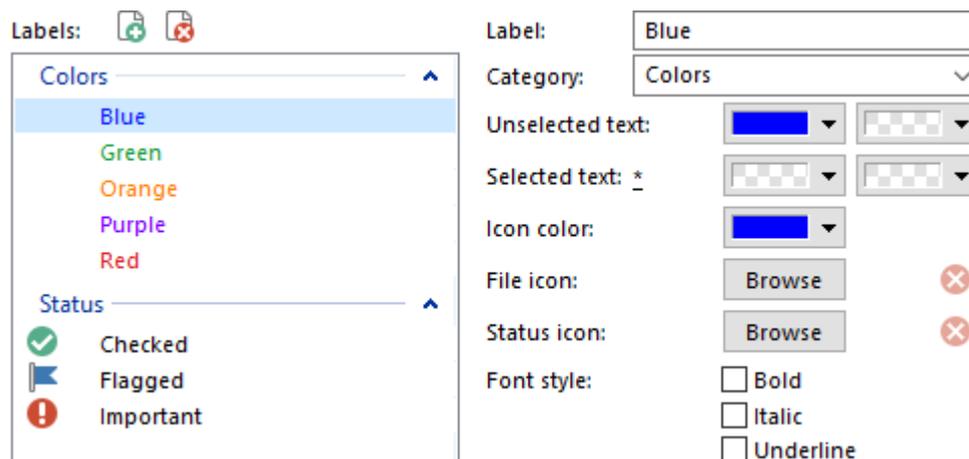
This page is basically one big scrolling list of options that can be turned on or off:

- **Recent Folders:** Adds your most recently used folders to the menu
- **Favorites:** Adds your favorite folders to the menu. You can add all your favorites or individually select folders to be added. The Jump List doesn't support sub-menus so you can't add any favorite sub-folders unfortunately.
- **Smart Favorites:** Adds your Smart Favorites to the menu.
- **FTP Sites:** You can select individual FTP sites from the Address Book or add all sites.
- **Other Folders:** This lets you add additional folders to the Jump List that don't come from any of the above locations. To add a folder to this list, you have to right-click the **Other Folders** entry and choose **Add Folder** from the context menu. Folders that have been added this way can be removed from the list by right-clicking them and choosing **Remove Folder** from the context menu.
- **Layouts:** Adds some or all of your configured [Lister Layouts](#) to the Jump List.
- **Commands:** Lets you add your configured [User commands](#) to the menu, as well as certain pre-defined commands.

Note that the size of the Jump List is not infinite, so even if items are turned on they may not actually be displayed in the Jump List - this is a limitation that Windows imposes on Jump List menus.

## ***Labels***

This page lets you create and edit [Labels](#) - combinations of colors and font styles that you can apply to files and folders. Any file or folder can be assigned a label, and you can use wildcards or [filters](#) to automatically label based on filename (so, for example, you could automatically label all your .doc files).



For each label, you can specify text and background colors for when the item is both selected and unselected. In the above screen shot you can see that the *Blue* label specifies that items assigned this label will display blue text when not selected. No other text colors have been defined (this is indicated by the checkerboard pattern) and so the normal colors for the items will be used for its background color, as well as the color when selected. If items are displayed using visual styles on Vista and Windows 7, any selected background color defined here won't be used (see [Display Options](#) for more information).

A label can also specify a color that is applied to labeled items' icons. This color is applied algorithmically to the existing icon - it doesn't make Opus show a different icon for the items - so your results may vary as to how effective this is. For most icons however it works fine:



A label can also be used to specify custom icons that are used for the labeled files or folders. To specify a custom icon for a label, click the **Browse** button. You can use icon files (*.ico*, *.icl*), icons from programs (*.exe*, *.dll*) or any image file format that Opus supports. If you specify an image Opus will automatically scale and convert it to icon format. Labels can also define a status icon that can be shown in a separate column to flag important files.

You can use the **Category** field to arrange your labels in categories - the list of labels shown in the **Properties** drop-down menu on the default Lister toolbar will be sorted into categories by default.

There are two ways that Opus can store labels that are assigned to a specific file or folder (as opposed to a wildcard or filter-based label).

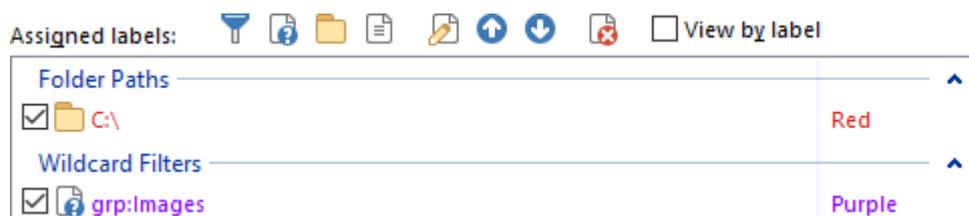
- On an NTFS-formatted drive (which is the most common type of file system these days) the label information is stored in an alternate data stream with the actual file itself. The advantage of this is that when you copy, move or rename the file, the label goes with it (if enabled on the [File Operations / Copy Attributes](#) page in Preferences).
- On other file systems, or if the option to store labels in the file system is turned off, Opus stores a list of the full filename and path of labeled files. That means that if you move a labeled file, the label will not go with it.

If the **Enable label storage in the file system** option is turned on, Opus will look for labels stored in the file system. Turn on the **Automatically store labels in the file system** option to store labels like that by default. If the first option is turned on but the second one is off, you can still store a label in the file system using the arguments to the [Properties](#) command.

By default explicitly applied labels override wildcard labels, but the **Apply wildcard and label filters to explicitly labeled files and folders** option lets wildcard labels stack on top of explicitly-assigned ones.

## Label Assignments

This page shows a list of all files and folders that have [labels](#) assigned to them. You can label files and folders using this interface; another way to do it is to use the *Set Label* command in the *Properties* drop-down on the default Operations toolbar. Note that if the label is stored in the file system (depending on the options on the [Labels](#) page), they will not appear in this list.

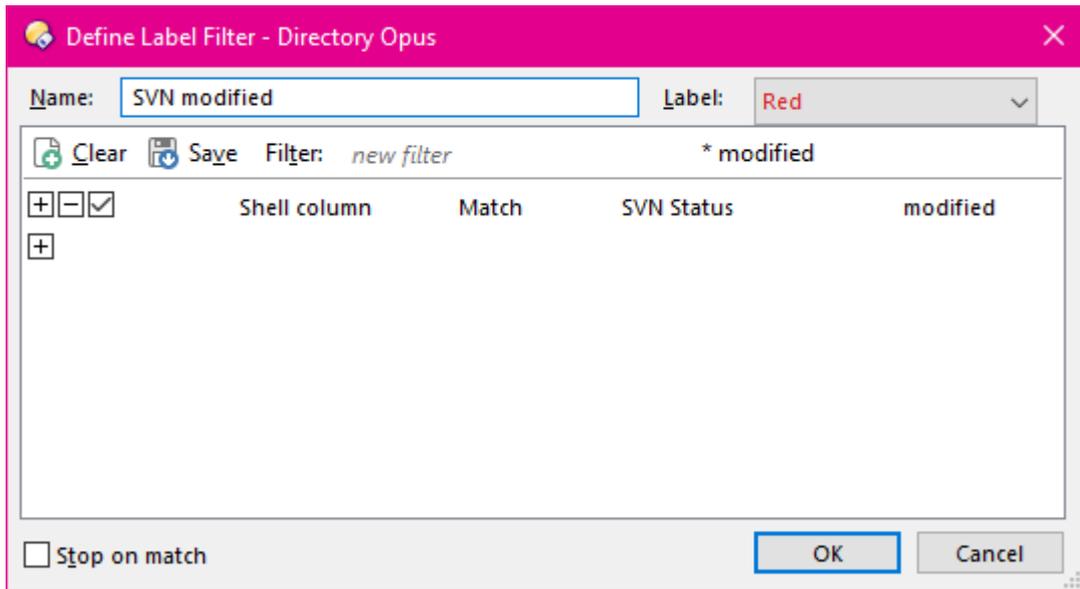


In this screen shot one folder has been assigned a label, and a wildcard label assignment has also been made, that will assign the *Purple* label to all image files. The checkbox next to each item allows you to temporarily disable a label assignment without deleting it.

Use the toolbar buttons at the top to assign new labels:

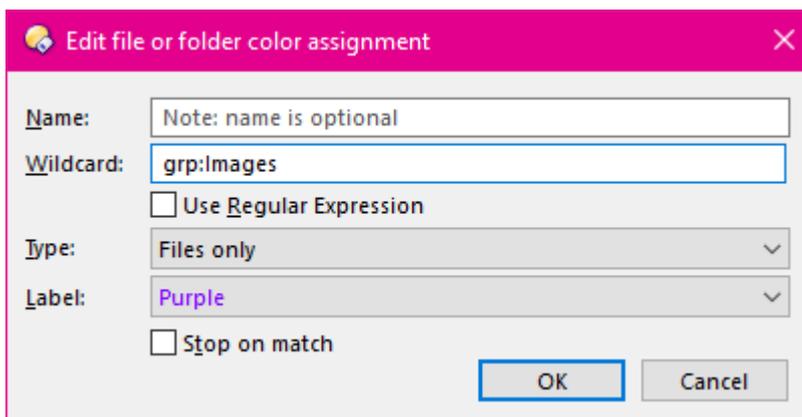
-  **Create a new label filter:** You can use [filters](#) to assign labels based on attributes other than filename or location - they can be based on [any criteria supported by the filter system](#), although you should keep in

mind that the filter will be compared against every file in every directory you visit - potentially slow filter clauses like *Contains* should be used with caution.



The above screenshot shows a label filter that will color red any files or folders that are under SVN control and have been modified. It uses a shell column provided by [TortoiseSVN](#) to match the SVN status.

-  **Create a new wildcard label:** Labels files and folders based on their names or locations. You can specify a pattern using standard [Opus wildcards](#) or [regular expressions](#), and choose whether to restrict the assignment to files, folders or both. The pattern can be used to match just the filename or a whole path; for example, you could have a wildcard pattern that matched **.doc** and **.docx** files in your Documents folder:



-  **Assign label to a specific folder path:** Lets you assign a label to a folder by its path. That is, the label will be attached to the path and name of the folder, not the folder itself.
-  **Assign label to a specific file:** Assign a label to a file by its path. That is, the label will be attached to the path and name of the file, not the file itself.

By default filter and wildcard labels assignments will "stack" on top of each other. One label could change the color of an icon, and another could change the icon itself. You can use the **Stop on match** option on both wildcard labels and label filters to prevent this from happening. Note that filter and wildcard label assignments can be rearranged to control their priority. The **up** (↑) and **down** (↓) buttons let you move label assignments above or below each other.

## **Recent List**

The Recent List is a list of your most recently used folders, displayed in the Go menu. The options here let you control how many folders are saved in this list:

- **Enable Recent List, maximum size:** This option can be used to disable the feature, or when enabled, specify how many folders are remembered. When the limit is exceeded, the oldest folder is removed from the list when a new folder is visited or used.
- **Include virtual folders:** Turn this option off if you want to prevent virtual folders like My Computer from being added to the recent list.
- **Ignore filesystem folders that are only transited:** If this option is on, a folder will only be added to the recent list if you actually do something in it. If you simply pass through a folder on your way to somewhere else, it won't be added to the list.

This page also lets you configure the size of the **Lister History** list; this is a separate list of recently visited folders that is local to a Lister. It is the list shown in the drop-down on the Back and Forward buttons on the Location toolbar, and also defines how many folders back and forward you can go when you click these buttons.

## **SmartFavorites**

The SmartFavorites system attempts to learn which folders you most often use and display them in the Favorites menu automatically. This page lets you configure how this system works. SmartFavorites is based on a points system. A number of points are allocated to certain activities that are performed in folders, and Opus keeps track of the points that individual folders have earned. When a folder's points exceeds a configurable limit it is deemed an important folder, and will appear in the SmartFavorites list.

- **Enable SmartFavorites system:** You can use this option to turn SmartFavorites off if you don't want to use this function.
- **Number of SmartFavorites folders:** This is the maximum number of folders that will be displayed in the Favorites menu. Folders will be displayed in order based on their accrued points.
- **Folder Activity Point threshold:** The number of activity points that a folder must earn before it is considered important.

- **Reference table size:** The number of folders that Opus will track activity points for - the topmost folders from this table are the ones that will be displayed in the menu.
- **Check on startup for folders that no longer exist:** When this is turned on, Opus will launch a low-priority background task when it starts to check for folders in the reference table that no longer exist.
- **Folder Activity Point configuration:** This table lets you configure the points that are awarded to a folder for certain activities. You may want to change these if you personally consider some activities more important than others.

## File Displays

This category contains options that control the behavior of the main part of the Lister, the area that displays files and folders.

- [Border](#): Options that affect the [File Display border](#) (the "title" bar of the file display)
- [FAYT and Filter Bar Keys](#): Configure the keys that control the [Find-As-You-Type](#) field and the [Filter Bar](#).
- [FAYT and Filter Bar Options](#): Options that control the [Find-As-You-Type](#) field and the [Filter Bar](#).
- [Mouse](#): Options relating to the behavior of the mouse in file displays
- [Options](#): General options that control the behavior of the file displays

### *File Display Border*

These options affect the [File Display border](#) - the toolbar or title bar that is displayed above the file displays in the Lister.

There are two main choices to make on this page - whether you want the file display border displayed as a toolbar, or displayed as a static header (to mimic versions of Opus prior to 11).

- **Display as a toolbar:** This option is on by default, and enables the use of one or more toolbars at the top of each file display. The factory toolbar used for this is [File Display](#), but you can use the lists to select your own toolbars. The list on the left displays your available toolbars, and the list on the right shows those that have been selected for use in the file display border.
  - **Click on destination's toolbar only changes state:** If this is turned on, and you click on the toolbar of the destination file display, your click will set that file display to the source but won't actually activate the toolbar button that you clicked on.
- **Display as a static header:** This option disables the file display toolbar, and instead displays a static header that displays the current folder path and several small navigation and control buttons. While the displayed path has some functionality (e.g. you can click individual segments similar to the [breadcrumbs location](#)

[field](#)), it is nowhere near as flexible as a toolbar.

- **Allow docking of Listers:** When this is enabled, you can drag one single display Lister over the title bar of another to dock them together (the Lister you dragged is closed, the Lister you docked it with is changed to dual display mode, and the folder tabs it was showing are opened in the new display). You can also split a dual display Lister into two single displays by dragging from the [file display border](#).
- **Enable hot paths in file display border:** When this is turned on, you can click individual path segments in the border to browse to the current folder's parent locations.
  - **Only when file display is the source and the Lister is active:** Applies to the above option - path segments are only "hot" in the source display. This lets you click the destination display to switch source / destination without accidentally clicking a path segment.
- **File display border Up button does "Up Back":** When turned on, clicking the *Up* button on the file display border behaves as if the **Go UP BACK** command was used (e.g. if the parent folder is in the path history, Opus goes "back" to it, preserving file selections and calculated folder sizes).
- **Show file display border in single display mode:** The borders have to be displayed in dual mode (to indicate [which side is the source and which is the destination](#)), but you can turn it off in single display mode to save space.

The following options apply to both styles of file display border:

- **Align other Lister element headers with the file display border:** This is a cosmetic option most useful when a file display toolbar is enabled. Because a toolbar is usually slightly higher than the static header, this option causes other Lister elements that are "inline" with the file display borders (like the folder tree's header) to be resized to match the height of the toolbar. Note that this option is disabled if you select more than one toolbar for the file display border.
- **Display close button:** Displays a close button at the end of the border that allows the tab or file display to be closed.
  - **Close button only closes active tab:** If turned on, clicking the close button will close the current tab, but other tabs will be unaffected. The file display will only close when the last tab closes.
- **Display icon:** Displays a Directory Opus icon at the left of the border. This icon allows you to create a shortcut to the current folder by drag-and-drop, and you can right-click it to display the context menu for the folder.
- **Enable file display Copy and Swap border buttons:** Enables additional buttons in the border that let you copy one file display to another (puts the Lister into dual mode if it isn't already), or swap the position of the two file displays (when already in dual mode).
- **Enable horizontal/vertical layout button:** Displays a button that lets you toggle the layout of dual file displays between horizontal (one above the other) to vertical (side-by-side). This is disabled by default, as the buttons in the default [Menu toolbar](#) already provide access to this functionality.
- **Set file display to source when Up/Back/Forwards buttons are used:** If turned off, the source/destination state of the file display will be unaffected when these buttons are used to navigate.

## **FAYT (*Find-As-You-Type*) and Filter Bar Keys**

The options on this page affect the [Find-As-You-Type](#) field, a multi-purpose field that appears at the bottom of the file display when you type, and the [Filter Bar](#), a popup field that makes it easy to filter the contents of current folder.

This page lets you configure the keys that invoke the various modes of the FAYT field and the Filter Bar. For each of the FAYT modes you can configure a different text and background color, which can provide an important visual cue as to which mode the field is in. For most modes you can also configure the activation key - the key that you press in the file display to show the bar in that mode. For example, the default key for *Command* mode is > so pressing > in a file display would open the FAYT in *Command* mode.

The **FAYT close timeout** option lets you specify how long the FAYT field remains displayed for once you have stopped typing. Not all modes auto-timeout; for example, in *Command* mode the field will remain open until you press **Enter** or **Escape** or click out of the field.

The **Default mode** option lets you configure which option is the "default" - that is, what happens if you press a key that's **not** one of the configured FAYT/Filter Bar keys. The default is *Find* mode, which means if you press, say, the **A** key, the FAYT will open in *Find* mode and the selection will jump to the first filename beginning with an **A**. The activation key field is hidden for the default mode.

See the [Find-As-You-Type](#) documentation for more detail on the FAYT field including a description of its various modes.

## **FAYT (*Find-As-You-Type*) and Filter Bar Options**

The options on this page affect the [Find-As-You-Type](#) field, a multi-purpose field that appears at the bottom of the file display when you type, and the [Filter Bar](#), a popup field that makes it easy to filter the contents of current folder.

The **Find-As-You-Type Find Mode Options** section contains options pertaining to *Find* mode:

- **Enable all cursor keys:** By default, the **Left/Right** cursor keys as well as **Home** and **End** are passed through to the file display, which lets you move the selection around in modes like List or Thumbnails

using these keys. If you turn this option off those keys will instead be used to move the caret in the FAYT field.

- **Highlight matches:** Turns on the highlighting of text within filenames that matches the entered string. You can configure the highlight color on the [Colors and Fonts](#) Preferences page.
- **Minimize scroll to make item visible:** This option changes the way the file list is scrolled when jumping to files that are currently out of view. The normal behavior is to scroll the list as little as possible, which often results in the matching file being positioned at the bottom of the window. If this option is turned off then the list will be scrolled (if possible) far enough that the matching file is positioned in the middle of the window.
- **Search anywhere in the filename:** This option enables partial matching when searching filenames. When turned off the entered text must match the beginning of the filename; when turned on, the entered text can match anywhere within the filename (although priority is still given to filenames that match at the beginning).
- **Prioritize shorter filenames:** The normal behaviour of Find mode is to jump to the next matching filename, however when the list isn't sorted alphabetically this may be undesirable. When this option is on, the shortest filename that matches the entered text will be selected, no matter where in the list it is.

The **Find-As-You-Type Filter Mode** section contains options pertaining to *Filter* mode:

- **Allow return key to open selected item:** In the *FAYTFilter* mode, pressing the **Return** key will open the item with focus rather than simply closing the *FAYT* field.
- **Select first matching item:** In the *FAYTFilter* mode, the first item that matches the filter will be automatically selected.

The options in the **Filter Bar** section affect the [Filter Bar](#), the bar that can be displayed at the bottom of a file display to filter the displayed contents of the current folder.

- **Display Filter Bar:** Lets you choose when the Filter Bar displayed.
  - **Always:** The bar is always displayed, whether a filter is active or not.
  - **Whenever a filter is set:** The bar is displayed when a filter is active, and hidden otherwise.
  - **Only when editing the filter:** The bar will only be displayed when you activate it to edit the filter - it will close again when you finish editing, whether a filter is active or not.
- **Automatically type \* when activating the filter bar:** When turned on, a \* will be typed into the Filter Bar if the activation key is pushed and no filter is currently set. Since the activation key is usually \* this means you can type things like \*.txt directly into the file display to do common filters on suffixes. With the option turned off, you would have to type \*\*.txt instead. (One \* to activate the Filter Bar, another to type the wildcard.) This option is particularly useful when **partial matching** is turned off.
- **Clear Quick filter automatically when changing folders:** If this option is turned off, any filter you assign via the Filter Bar will remain in effect when you change to another folder.
- **Match any word:** This option treats all words you enter as separate patterns. For example, you can type “moo cow” and it would automatically match a file called “moo” or a file called “cow”. This saves you having to build up complex *OR* wildcards (the equivalent wildcard would be “(moo|cow)”).

- **Partial matching:** When turned on, the filtering will automatically match on sub-strings, so typing **og** would match any file containing those letters, e.g. **dog.txt** or **google.png**. When off, the pattern you enter must match exactly - so you would need to type **\*og\*** for the same effect. Note that if you explicitly add a **\*** at the start or end of the pattern then Opus will assume you do not wish to use partial matching even if it is switched on. This allows you the convenience of partial matching most of the time while still being able to filter by the start or end of things when you need to.
- **Real-time filtering:** When turned on, the file display will be filtered in real-time, as you edit the filter. If turned off, you must press the **Enter** key to activate the filter you have typed.
- **Use regular expression:** When turned on, the pattern you enter into the *Filter Bar* will be treated as a [regular expression](#) rather than a simple wildcard.

See the [Find-As-You-Type](#) documentation for more detail on the FAYT field including a description of its various modes.

## Mouse

This page contains various options relating to the behavior of the mouse in file displays.

- **Allow file selection when clicking to activate Lister:** If a Lister is not the active window, clicking on a file display to activate it can have the unintended consequence of altering the file selection state. Turn this option off to prevent a click that activates the Lister from selecting or deselecting files.
- **Allow file selection when clicking to switch source/destination state:** This option is similar in nature to the **Allow file selection when clicking to activate Lister** option. If turned on, and you click on a file in the destination file display, the file display will be set to source and the file will be selected. If this option is turned off then although the file display would still be set to source, the file selection in that file display would be unchanged.
- **Allow drag and drop into sub-folders:** Normally when you drag a file over a sub-folder you can drop it to move or copy it to that folder. If you turn this option off, sub-folders won't appear as drop targets unless you are also holding down the **Shift** (to move), **Control** (to copy) or **Alt** (to make a shortcut) keys. This can be useful if, for example, you have trouble with drag and drop due to physical disability - there's nothing worse than seeing a file vanish into an unknown sub-folder because you dropped it on it accidentally.
- **Double-click on file display background:** Lets you configure what happens when you double-click the left mouse button on a blank area of the file display. You can enter any Opus command line into the field. For example, **Go UP BACK** to make double-click automatically go back to the parent folder.
- **Middle double-click on file display background:** Similar to the previous option, lets you configure a command that is run when you double-click the middle mouse button on a blank area of the file display.
- **Single click to open an item:** This option activates "single-click" mode. In this mode, you select files by hovering over them with the mouse, and open files or folders by single-clicking them, like links on a web page.
  - **Underline items on hover:** When single-click mode is active, this lets you control whether items you hover over with the mouse have their label underlined or not.
- **Single middle-click runs Middle double-click File Type event:** This option makes Opus treat a single middle-click on a file or folder as if it were a double middle-click. For example, if you turn this option on and configure the *Middle double-click* [File Type event](#) for a folder to open it in a new tab (**Go NEWTAB**), you can make Opus emulate the behavior of web browsers where middle-clicking a link opens it in a new tab. If this option is turned off then single middle-clicking a file or folder toggles its selection status (which is handy, since it doesn't affect the selection status of other files).

- **Smooth scrolling with mouse wheel/keyboard:** This option enables smooth scrolling of file displays.

## File Display Options

This page contains various options that let you adjust the behavior of File Displays.

- **Automatically select first file in folder:** When navigating to a new folder, this option causes the first item in the list to be selected automatically. This option does not apply to Power mode file displays - there is a separate option for them on the [Power Mode](#) page.
- **Enable file InfoTips:** InfoTips are the "tooltips" that appear when hovering over files and folders with the mouse. This option lets you enable or disable their display, and also choose for which display modes and which types of drives they appear. For example, you may not want them to appear when on a network folder or FTP site. The **Keyboard delay** option lets you display InfoTips when using Opus with the keyboard - if you hold the **Control** key down for the configured length of time, the InfoTip for the item with focus will be displayed.
- **Enable "slide-out" navigation buttons:** If you turn this option on, Opus will display tiny navigation buttons that slide out from the corners and/or sides of the file display when you hover the mouse there for the specified time. These buttons let you go up, back and forwards without having to move the mouse all the way up to the toolbar at the top of the Lister.
- **Hover to switch source/destination state:** Hovering the mouse cursor over a file display for the specified time (in milliseconds) will set it as the source.
- **Reset focus entry when sorting file list:** Normally when you re-sort the file list by clicking on a column header (in Details mode), the input focus is kept on the same file, even if the file changes position in the list. With this option turned on, input focus is maintained at the same relative position (so for example, if the fifth file down had input focus before you sorted the list, the fifth file down - whatever it is - will still have it afterwards).
- **Set new window to source when switching to dual file display:** When you switch a Lister into dual display mode, this option causes the newly opened file display to become the source, and the original file display the destination.
- **Specify initial folder when switching to dual file display:** Normally when you switch a Lister into dual display mode, the folder that was already open is opened in the second file display. This option lets you specify a specific folder that will be the default for the newly opened display (so you could, for instance, always have the second file display open to display the Desktop). The new **Tab group** option lets you configure a tab group to be opened automatically when switching to dual display mode (instead of just a folder).
- **Use tab key to switch source/destination in dual-display Listers:** Normally pressing the Tab key in a Lister moves the focus through the various elements of the Lister. If this option is on then and the lister is in dual display mode then the Tab key simply switches the source and destination states of the two displays; it doesn't move the focus to any other element.

## File Display Modes

This category contains options that relate to the various display modes that File Displays can use.

- [Details](#): Options specific to Details mode displays
- [Power Mode](#): Options specific to Power mode displays

- [Power Mode Buttons](#): Options specific to mouse button behaviour in Power mode displays
- [Thumbnails](#): Options specific to Thumbnails mode
- [Tiles](#): Options specific to Tiles Mode displays
- [Toolbars](#): Options for turning on extra toolbars in specific display modes

## Details Mode

This page contains options that control the appearance and behavior of File Displays when they are set to Details mode.

- **Display grid lines**: Lets you configure the display of horizontal gridlines. There are a number of different line patterns available, as well as a solid fill which gives the effect of alternating solid lines behind the text. You can set the color and opacity of the gridlines (the opacity setting determines how solid the lines appear, and how much they are blended with the background color or image). If the **Grid lines only in source file display** option is enabled, the gridlines (or fill) will only be displayed when a file display is set as [source](#).
- **Display icons**: Lets you turn off the display of file icons in Details mode.
  - **Hide icons when thumbnail column is visible**: If the *Thumbnail* column is displayed, you may not want to show file icons as well - turning this option on will hide the icons automatically when the *Thumbnail* column is present.
- **Extra line padding**: Lets you add additional padding inside each line, making each line taller and a larger area to click on. The padding you specify will be scaled if your config is used in a different DPI.
- **Extra line spacing**: Lets you add additional spacing between each line. A particular use for this is to add one extra pixel between lines if you dislike the way selection boxes overlap (producing a thin line between each selected item) in some versions of Windows. If you add more than one pixel of spacing, you will start to get gaps between items where clicking them is like clicking the folder background. Unlike padding, the spacing value does not change when you move to different DPIs.
- **File selection / highlighting style**: Lets you control both how selected files are displayed and which areas of the line respond to mouse clicks for selecting files.
  - **Filename only**:

Name ▲	Size	Type	Modified	Att
 ..		Parent Folder		
 IMG_0810.JPG	1.69 MB	JPEG image	16/10/2011 8:29 PM	-a--
 IMG_1389.JPG	1.64 MB	JPEG image	16/10/2011 8:29 PM	-a--
 IMG_1390.JPG	1.47 MB	JPEG image	16/10/2011 8:29 PM	-a--
 IMG_1391.JPG	1.61 MB	JPEG image	16/10/2011 8:29 PM	-a--
 IMG_1910.JPG	1.33 MB	JPEG image	16/10/2011 8:29 PM	-a--
 IMG_2100.JPG	1.05 MB	JPEG image	16/10/2011 8:29 PM	-a--

Only the filename is active - clicking anywhere to the left or right of the filename will not select the item

- **Full width of the Name column**:

Name ▲	Size	Type	Modified	Att
↑ ..		Parent Folder		
IMG_0810.JPG	1.69 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1389.JPG	1.64 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1390.JPG	1.47 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1391.JPG	1.61 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1910.JPG	1.33 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_2100.JPG	1.06 MB	JPEG image	16/10/2011 8:29 PM	-a---

The entire Name column is active - clicking anywhere to the left or right of that column will not select the item

- **Full row:**

Name ▲	Size	Type	Modified	Att
↑ ..		Parent Folder		
IMG_0810.JPG	1.69 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1389.JPG	1.64 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1390.JPG	1.47 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1391.JPG	1.61 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1910.JPG	1.33 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_2100.JPG	1.06 MB	JPEG image	16/10/2011 8:29 PM	-a---

The entire row is active - you can click anywhere on the item, in any column, to activate it.

- **Always highlight full row:** Normally the option above controls both the active area for selection and the width of the highlight that is displayed for selected items. If this option is on, the full row will be highlighted when an item is selected, irrespective of which option is selected for the active area.
- **Sort-field specific key scrolling:** Normally, when you type in the file display (and the [Find-As-You-Type](#) field comes up in Find mode), Opus will search for items by name and scroll to show them. If this option is on, the current sort field can affect this behavior - for example, if the list is sorted by the Type column, the FAYT field will search for items by file type rather than by name.
- **Thumbnail column aspect ratio:** This option controls the aspect ratio of the *Thumbnail* column. The width of the *Thumbnail* column is set like the width of any other column - by resizing the column header using the mouse (or using the **Default width** option on the [Fields](#) page). The aspect ratio you specify controls the row height. For example, setting the aspect ratio to 1:1 would mean the row height of each line would be the same as the width of the column.
- **Show thumbnail borders:** This option lets you choose whether thumbnails in the *Thumbnail* column have a border displayed for them, and if so, what color the frame is filled with.

## Power Mode

This page contains options that control the appearance and behavior of File Displays when they are set to Power mode. Power mode is like Details mode except that you have far more control over the behavior of the file display in this mode.

- **Activate Keyboard Mode when a partial filename is typed:** Power mode file displays are not in "keyboard" mode by default - the cursor keys scroll the list rather than moving the focus from one item to another. If this option is turned on, typing into the file display to activate the Find-As-You-Type field will also place the file display in keyboard mode. When in keyboard mode, the cursor keys behave as they do in a Details mode file display.
- **Always in Keyboard Mode:** When this option is on, power mode file displays are always in keyboard mode - the cursor keys act like they do in a details mode file display.
- **Automatically select first file in folder:** When navigating to a new folder, this option causes the first item in the list to be selected automatically.
- **Display grid lines:** Lets you configure the display of horizontal gridlines. There are a number of different line patterns available, as well as a solid fill which gives the effect of alternating solid lines behind the text. You can set the color and opacity of the gridlines (the opacity setting determines how solid the lines appear, and how much they are blended with the background color or image). If the **Grid lines only in source file display** option is enabled, the gridlines (or fill) will only be displayed when a file display is set as [source](#).
- **Display icons:** Lets you turn off the display of file icons in Power mode.
  - **Hide icons when thumbnail column is visible:** If the *Thumbnail* column is displayed, you may not want to show file icons as well - turning this option on will hide the icons automatically when the *Thumbnail* column is present.
- **Enter key opens all selected files:** When the Power mode file display isn't in keyboard mode, the **Enter** key normally does nothing - with this option on, any selected files will be opened when **Enter** is pressed. When the file display is in keyboard mode, this option controls whether all selected files are opened when **Enter** is pressed, or only the file that currently has input focus.
- **Extra line padding:** Lets you add additional padding inside each line, making each line taller and a larger area to click on. The padding you specify will be scaled if your config is used in a different DPI.
- **Extra line spacing:** Lets you add additional spacing between each line. A particular use for this is to add one extra pixel between lines if you dislike the way selection boxes overlap (producing a thin line between each selected item) in some versions of Windows. If you add more than one pixel of spacing, you will start to get gaps between items where clicking them is like clicking the folder background. Unlike padding, the spacing value does not change when you move to different DPIs.
- **File selection / highlighting style:** Lets you control both how selected files are displayed and which areas of the line respond to mouse clicks for selecting files.
  - **Filename only:**

Name ▲	Size	Type	Modified	Att
..		Parent Folder		
IMG_0810.JPG	1.69 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1389.JPG	1.64 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1390.JPG	1.47 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1391.JPG	1.61 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1910.JPG	1.33 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_2100.JPG	1.06 MB	JPEG image	16/10/2011 8:29 PM	-a---

Only the filename is active - clicking anywhere to the left or right of the filename will not select the item

- **Full width of the Name column:**

Name ▲	Size	Type	Modified	Att
↑ ..		Parent Folder		
IMG_0810.JPG	1.69 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1389.JPG	1.64 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1390.JPG	1.47 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1391.JPG	1.61 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1910.JPG	1.33 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_2100.JPG	1.06 MB	JPEG image	16/10/2011 8:29 PM	-a---

The entire Name column is active - clicking anywhere to the left or right of that column will not select the item

- **Full row:**

Name ▲	Size	Type	Modified	Att
↑ ..		Parent Folder		
IMG_0810.JPG	1.69 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1389.JPG	1.64 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1390.JPG	1.47 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1391.JPG	1.61 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_1910.JPG	1.33 MB	JPEG image	16/10/2011 8:29 PM	-a---
IMG_2100.JPG	1.06 MB	JPEG image	16/10/2011 8:29 PM	-a---

The entire row is active - you can click anywhere on the item, in any column, to activate it.

- **Always highlight full row:** Normally the option above controls both the active area for selection and the width of the highlight that is displayed for selected items. If this option is on, the full row will be highlighted when an item is selected, irrespective of which option is selected for the active area.
- **Range selection:** Normally the way you select a range of files in power mode is by clicking and dragging with the mouse. With this option on, you can perform range selection similar to Details mode by holding down the specified qualifier key (e.g., if set to *Shift*, you could click on the first file, then hold down the **Shift** key and click on the last file to select the range).
- **Sort-field specific key scrolling:** Normally, when you type in the file display (and the [Find-As-You-Type](#) field comes up in Find mode), Opus will search for items by name and scroll to show them. If this option is on, the current sort field can affect this behavior - for example, if the list is sorted by the Type column, the FAYT field will search for items by file type rather than by name.
- **Thumbnail column aspect ratio:** This option controls the aspect ratio of the *Thumbnail* column. The width of the *Thumbnail* column is set like the width of any other column - by resizing the column header using the mouse (or using the **Default width** option on the [Fields](#) page). The aspect ratio you specify controls the row height. For example, setting the aspect ratio to 1:1 would mean the row height of each line would be the same as the width of the column.
- **Show thumbnail borders:** This option lets you choose whether thumbnails in the *Thumbnail* column have a border displayed for them, and if so, what color the frame is filled with.

## Power Mode Buttons

The Power Mode Buttons page lets you control exactly what all three mouse buttons do in a Power mode file display. You can select actions for what happens when you click on a file, and also what happens when you drag and drop files with each of the three buttons.

For each of the three buttons, the options for clicking are:

- **Auto-deselect:** Makes the file display behave like a Details mode one - clicking to select a new file, or clicking in an empty part of the display, will automatically deselect any already selected items.
- **Context Menu:** Selects the file under the mouse, and displays its context menu.
- **Context Menu (no select):** Displays the context menu for any currently selected items, but does not select the one under the mouse.
- **Context Menu (select):** Lets you drag-select one or more files, and then displays the context menu for them.
- **Deselect only:** Deselects the file under the mouse and any you drag-select.
- **Disabled:** The mouse button will do nothing.
- **Full toggle:** Clicking an item, or drag-selecting over items, will invert their selection state.
- **Inline Rename:** Immediately enters inline rename for the item under the mouse.
- **Normal drag select:** Inverts the selection state of the item under the mouse, and then applies the same state to any items you drag-select over.
- **Partial toggle:** Clicking an unselected item and then drag-selecting will select all the items; clicking an already selected item and then drag-selecting will deselect all the items.
- **Select only:** Selects the file under the mouse and any you drag-select.

The options for drag and drop are:

- **Disabled:** Drag and drop can not be initiated with this mouse button.
- **Immediate action:** Dragging an item and releasing it will immediately perform the default drop action.
- **Show menu:** Dragging an item and releasing it will display the drag and drop action menu.

## Thumbnails Mode

This page contains options that affects the appearance of thumbnails in file displays, as well as options that control how thumbnails are generated. The **Appearance** section contains the following settings:

- **Size:** Specify the size in pixel of thumbnail images (*width x height*). Thumbnails will be scaled down to fit within these boundaries, preserving their aspect ratio. The Make square button provides a quick way to set the width and height to the same values.
- **Spacing:** Specify the spacing between thumbnails (*horizontal x vertical*). (Note that if labels are enabled and set to more than one line, the vertical spacing shares its space with the extra label lines and will not have much effect on the actual spacing until the value is very large.)
- **Label:** Specify the maximum number of lines of text that the thumbnail's label can display.

- **Fill color:** Specify the color used to fill the background of thumbnail images.
- **Make square:** Locks the two size numbers together, so changing one changes both.
- **Display borders:** Draw borders around thumbnail images.
- **Display labels:** Display labels below thumbnails. If this option is on the filename will be displayed below each thumbnail.
- **Display date taken in label:** If this option is turned on Opus will display the date taken (in the case of digital photos), or the last modified file date, in a smaller font below the thumbnail's label.
- **Display image and file size in label:** If this option is turned on Opus will display the size of the image (and the size of the file) in a smaller font below the thumbnail's label.

The other options on this page are:

- **Cache thumbnails:** This lets you control whether Opus caches thumbnails or not. Caching thumbnails takes up some disk space but can result in much quicker loading of thumbnails once they have been generated and cached. If caching is enabled, use the **Adjust cache settings** link to control the following cache options:
  - **Use lossless compression:** This makes Opus use a lossless compression method when storing cached thumbnail images - it will result in slightly higher quality thumbnails, but they'll take up more space and it can take longer to cache thumbnails the first time they are generated.
  - **Cache thumbnails for images located on:** This lets you specify which types of drives thumbnails will be cached for (e.g. you can turn caching off for fast local drives but leave it on for networks).
  - **Maximum cache size:** This lets you specify the maximum size of the thumbnail cache. The current size is displayed below, and you can use the **Empty** button to clear the cache completely.
- **Display thumbnails for folders:** This option enables the display of thumbnails for folders as well as for files. If folder thumbnails are enabled, use the **Adjust folder thumbnail settings** link to control the following options:
  - **Generate folder thumbnails via the shell:** On Vista and Windows 7, this option makes Opus ask the system for thumbnails for folders. This results in the modern-looking 3D folder thumbnails that Explorer displays. Otherwise Opus uses its own flat 2D-style folder thumbnail.
  - **Generate folder thumbnails from images inside folders:** If this option is turned off, folder thumbnails will show a generic image representing a folder. With this option on, Opus will display miniature thumbnails for the first few files it finds within the folder, in the actual folder thumbnail itself.
    - **Choose most recent images:** When Opus is generating its own folder thumbnails, this option makes it use the most recent images in the folder to generate the thumbnail. When turned off the contents of the folder will be looked at alphabetically.
    - **Single image:** When Opus is generating its own folder thumbnails, this option makes it only use a single image (the first one found) rather than multiple images to generate the thumbnail. You can use a [standard wildcard pattern](#) to control the filenames that Opus will look for.
  - **Display folder frame:** When Opus folder thumbnails are displayed, this option lets you turn off the "folder" image used by default. If turned on, you can use the options below to control its color.
- **Load all thumbnails in a folder automatically:** Normally Opus only generates thumbnails when it needs to display them, which means when you scroll through a large folder of images, there may be a small delay

before thumbnails are loaded. With this option turned on, Opus will automatically begin loading thumbnails for all files in the folder when you navigate to a new location.

- **Overlay rating:** Opus will overlay the thumbnail image with stars to indicate its assigned rating (if you have assigned the file a rating, that is). Ratings are assigned using the [metadata](#) functions.
- **Overlay relative dimension bars:** Opus will draw small horizontal and vertical bar graphs over the thumbnail to represent the dimensions of that image file relative to the largest image in the current directory. This is particularly useful when you have multiple copies of the same image saved at different resolutions - their thumbnails will ordinarily all look the same, and the bar graphs let you tell at a glance which relative size each file is. This option can also be changed from a button or hotkey using the **RELDIMENSIONOVERLAYS** argument to the internal [Set](#) command.
- **Overlay thumbnail with file type icon:** Opus will display a small icon for the type of file in question in the bottom corner of the thumbnail image (e.g. a JPEG image will have the generic **.jpg** icon displayed in the corner of the thumbnail).
  - **Hide file type icon if type not registered:** With the above option on, the file type icon will not be shown if the file type is not registered. This mostly applies to image files that may have no file extension - Opus is still able to display thumbnails for them as it looks at the file contents to determine their type, but there would be no valid icon registered in the system that could be shown.
  - **Hide file type icon if thumbnail is too small:** If the generated thumbnail is too small, the file type icon will not be overlaid. This prevents the problem where the file type icon may actually be larger than the thumbnail and would obscure it completely.
- **Use EXIF information to auto-rotate images:** Most modern digital cameras contain an orientation sensor that records into the image the orientation of the camera when the picture was taken. If this option is turned on Opus will read the orientation information from the image and automatically rotate the displayed thumbnail so that it appears the right way up.
- **Thumbnail threads:** This determines how many background threads Opus will use to generate thumbnails. As thumbnail generation is mostly a CPU intensive task (decoding JPEG images, for example), increasing the number of threads can greatly speed up the generation of thumbnails, although you may see diminishing returns as shared resources like storage become the bottleneck. We don't recommend setting this to a number higher than the number of logical CPUs in your computer.

## Tiles Mode

This page contains options that control the appearance of the file display in Tiles mode. The options in the **Size and Layout** section are:

- **Tile size:** This specifies the size of individual tiles (or more accurately, of the text part of individual tiles). It is given as the number of characters wide and the number of lines of text high.
- **Tile spacing:** This is the spacing between tiles (*horizontal x vertical*), in pixels.
- **Expand tile with focus if needed:** When a tile has the input focus, it will be expanded to overlap other adjacent tiles if its label is bigger than the size allowed in the **Tile size** setting. This lets you read the full name of the selected item when it may otherwise be clipped. This option is overridden if [visual styles](#) are enabled for file display items.

- **Vertical layout:** Normally the Tiles display has a horizontal layout - tiles are drawn from left-to-right, top-to-bottom, and if a scrollbar is needed the list scrolls vertically. With this option on, the layout is switched to a vertical one - tiles go top-to-bottom, left-to-right, and scrolling is horizontal, similar to List mode.

You can optionally have Tiles mode also display thumbnails for files - the options that control this are:

- **Show Thumbnails in Tiles mode:** Turn this on to enable the display of thumbnails in tiles mode. Thumbnails are generated subject to the settings on the [Thumbnails](#) page.
- **Thumbnail size:** Specifies the size of thumbnails in pixels.
- **Show file type icon in tile:** Because the thumbnail replaces the display of the normal file icon, this option lets you display a small version of the icon in the bottom corner of the tile. Either the frame or fill options (below) must be turned on for this to work - it's disabled otherwise, because the file type icon would end up floating in space.
  - **Hide file type icon if type not registered:** With the above option on, the file type icon will not be shown if the file type is not registered. This mostly applies to image files that may have no file extension - Opus is still able to display thumbnails for them as it looks at the file contents to determine their type, but there would be no valid icon registered in the system that could be shown.
- **Show thumbnail borders:** This option makes Opus display a frame around thumbnails in the tile. You can also specify a **Fill color** when the thumbnail is framed.

Tiles can be displayed with or without frames, and with the interior filled or unfilled. The options for **Frame and Fill** are:

- **Show tile frames:** This option displays a frame around each tile. You can select two colors for the frame - the top and left edges use one, and the bottom and right edges use the other.
- **Fill tile interior:** This option fills the tile's background. You can select two colors - if they're different, a gradient fill will be performed.

## **Toolbars**

This page lets you configure a toolbar or toolbar set that will be turned on automatically whenever a file display is set to the specified [display mode](#).

By default Opus uses this to show the [Images](#) toolbar whenever the file display is in *Thumbnails* mode. For each display mode, you can select a toolbar or [toolbar set](#) that's displayed automatically when that mode is selected. Toolbar sets are indicated in the drop-down list by an icon to distinguish them from toolbars. If you select a toolbar (rather than a set), you can also choose where in the Lister it should appear.

The automatically triggered toolbar or set will be turned off again automatically if you change to another mode.

In a dual-display Lister, the toolbar for a particular mode will be displayed when either file display is in that mode. If instead you want the toolbar to be shown only when the source file display is in that mode, turn off the **In a dual-display Lister, activate toolbar when either file display is in that mode** option.

## File Operations

This category contains options that control file operations (copying, deleting, etc.) within Directory Opus. In most cases these only define the default behavior of the file operations - most options can be overridden at the command level with the addition of command-line parameters to the appropriate command.

- [Copy Attributes](#): Controls which file attributes are copied by default (attributes, metadata, security permissions, etc.)
- [Copy Options](#): Other options that control the copying of files and directories (what to do when files already exist, that sort of thing).
- [Deleting Files](#): Options that control what happens by default when you delete files through Directory Opus.
- [Double-click on Files](#): A few options that control what happens when you double-click on certain files in a Lister. The [File Types](#) system controls this to a far higher degree.
- [Filters](#): Provides a convenient place where you can define and edit filters that can be used by various Opus functions (filtered copy/delete operations, the Find tool, etc.)
- [Inline Rename](#): Controls the behavior of the [inline rename](#) function.
- [Logging](#): Controls where Opus should keep a log of your file operations or not.
- [Metadata](#): Options relating to the editing of file metadata.
- [Options](#): General options that didn't fit on any other page!
- [Progress Indicators](#): Options that control the progress indicators that Opus can show during file operations.

### Copy Attributes

The options on this page control which file attributes are copied (or preserved) when you copy files in Directory Opus. These options can all be overridden at the command level by supplying different parameters to the [Copy](#) command.

- **Clear read-only flag when copying from CDs**: Files that reside on CDs and DVDs are implicitly read-only, and normally this attribute is preserved when you copy these files. If this option is on, the **R** attribute will be cleared when these files are copied. This can be overridden by the **CLEARREADONLY** argument to the **Copy** command.

- **Copy metadata:** Copies file metadata that is stored in an NTFS *Alternate Data Stream* (ADS). This metadata is things like comments, tags, rating information and [labels](#). This only applies to metadata that is stored in a separate data stream - for example, Office documents store this information as part of the file itself, and so this metadata would always be copied anyway. Turning this option on may make the file copy process slightly slower. This can be overridden by the **COPYPROPERTIES** argument to the **Copy** command.
- **Copy all NTFS data streams:** If **Copy metadata** is enabled, this causes Opus to copy **all** data streams contained in the file, not just the metadata ones.
- **Copy owner:** This option copies file owner information. Normally newly copied files will be owned by the user making the copy, but with this option on they would be owned by the owner of the original file. Because setting the file owner requires elevation under Vista / Windows 7, this will cause a UAC prompt to be displayed if the original owner is not the current user. You can opt to do this on **Local drives only** if desired. This can be overridden by the **COPYOWNER** argument to the **Copy** command.
- **Copy security permissions:** This copies the original files security permissions - when turned off, the copied file will inherit the permissions of the target folder. This setting can be overridden by the **COPYSECURITY** option to the **Copy** command.
- **Copy sparse files as sparse:** Sparse files are a special feature of the NTFS file system that allows regions of a file to be marked as "empty" (containing all zero bytes) and have them not occupy space on the disk. For example, a 1GB file consisting of all zeroes could actually occupy no disk space. They're used in specialized applications like virtual machines and some download tools will also use them when downloading large files. Normally if you copy a sparse file with Opus it will be "de-sparsified" (not really a word), meaning the copy will occupy its full size on disk. With this option turned on, regions in the source file that are marked as sparse will be recreated in the copied file. This option can be overridden with the **COPYSPARSE** argument for the **Copy** command.
- **Mark copied files as archived:** This sets the **A** attribute on newly copied files. You can use this if have a backup solution that uses the **A** attribute as an indication that a file has been backed up. This can be overridden by the **MARKDESTARCHIVE** argument to the **Copy** command.
- **Mark original files as archived after being copied:** This sets the **A** attribute on the original files after they have been copied. This can be overridden by the **MARKSOURCEARCHIVE** argument.
- **Preserve the attributes of copied files:** This preserves the attributes of files when they are copied. Without this option attributes will be reset on the new files. This can be overridden by the **COPYATTR** argument.
- **Preserve the descriptions of copied files:** This preserves any description assigned to the copied files. This option only applies to the **DESCRIPT.ION** file comment system that Opus uses - descriptions that are stored in file metadata are subject to the **Copy metadata** option instead. This can be overridden by the **COPYDESC** argument to the **Copy** command.
- **Preserve the timestamps of copied files:** This option preserves the timestamps of copied files - if turned off, newly copied files will have their timestamps set to the current time and date. This can be overridden by the **COPYFILETIMES**, **COPYDIRTIMES** and **COPYCREATIONTIMES** argument to the **Copy** command. The [Miscellaneous / Advanced](#) page in Preferences also contains some overrides (**no\_copy\_creation\_time** and **no\_copy\_dir\_dates**).
- **Update permissions/encryption to match the destination when moving files:** Normally when you move a file on the same hard drive (an operation that doesn't involve a new file being created), it will keep its original permissions. This can cause problems when you move files into a folder with different permissions - for example, if you moved a private file into a shared public folder, the file would keep its original permissions and not be accessible to users of the share. Turning on this option causes Opus to update the

permissions of the moved file to match the folder it has been moved into. This can be overridden by the **UPDATESECURITY** argument to the **Copy** command.

## Copy Options

This page controls options that affects how Directory Opus copies files. These options can all be overridden at the command level by supplying different parameters to the [Copy](#) command. Several conditions require Opus to ask for confirmation before copying files; the options that affect this are:

- **Ask for confirmation before overwriting existing files:** If files you are copying already existing in the destination, Opus will prompt you before overwriting them. This can be overridden by the **WHENEXISTS** argument to the **Copy** command.
- **Ask for confirmation before overwriting read-only files:** This option causes an additional confirmation to be displayed if the existing file has its read-only (**R**) attribute set.
- **Ask for confirmation before merging existing folders:** Unlike a file, where copying over an existing file deletes the original, copying over an existing folder has the effect of merging the contents of the two folders. Only files within the target folder that clash with files in the source folder will actually be deleted, and so this is generally a non-destructive operation. This option causes confirmation to be displayed before Opus merges one folder with another. This can be overridden by the **WHENEXISTS** argument to the **Copy** command.
- **When copying in Flat View mode:** Files displayed in Flat View can exist in physically different folders. If you select files from multiple folders for a copy operation, Opus gives you the choice of copying them all to the target folder, or of recreating their original folder structure in the target. You can choose *ask how to copy*, and Opus will prompt you each time, or select *copy to single level* or *recreate source folder structure*. This option can be overridden by the **FLATVIEWCOPY** argument to the **Copy** command.
- **When copying folders to a Collection:** When you copy (add) a folder to a File Collection, you have the option of adding the folder to the collection directly (in which case it appears like any other collection member), or adding it as a sub-collection. In this second case, the folder will appear as a collection in its own right, and the folder's contents will be added to the sub-collection. This option lets you choose *ask how to copy*, *add to collection as a member* or *add to collection as a sub-collection*. This can be overridden by the **COPYTOCOLL** argument.

Additional options on this page are:

- **Automatically select newly copied files:** When newly copied files are added to the file display, they will be automatically selected.
- **Automatically manage file copy queues:** This option enables automatic copy queuing. When this option is on, Opus will automatically create copy queues based on the source and destination locations and the types of drive involved. If you turn this off, copy queuing can be used manually with the **QUEUE** argument to the **Copy** command.
  - **Display confirmation when a job is queued:** If a copy is queued automatically, this option causes Opus to display a prompt confirming the fact. You can disable this prompt by turning this option off, or by specifying the **QUEUE=quiet** argument to the **Copy** command.
- **Sort newly created and copied files:** Normally when files are copied to or created in a file display, they are automatically sorted into their proper position (based on the current sorting parameters). If you turn this option off they will instead be added, unsorted, to the end of the list.

## Deleting Files

This page contains options that affect how Opus deletes files. Some of the options here can be overridden at the command level by arguments supplied to the [Delete](#) command.

- **Ask for confirmation before commencing delete:** This causes an initial confirmation prompt to be shown at the very beginning of a delete operation. It will give you a summary of the number of files and folders selected for delete. This can be disabled by turning this option off or with the **QUIET** argument for the **Delete** command.
  - **Skip confirmation when deleting to Recycle Bin:** Because the Recycle Bin itself normally displays a confirmation dialog, you can choose to disable the above confirmation when using the Recycle Bin. If you have configured the Recycle Bin to not display a prompt, you may wish to leave this option off to allow the Opus confirmation to be shown.
- **Automatically select next file after deleting:** If this option is turned on, the file in the list following the deleted files will be automatically selected. This is useful in conjunction with the [Viewer Pane](#) when you want to inspect and optionally delete multiple images, for example.
- **Delete to Recycle Bin where possible:** With this option on, Opus will delete to the system Recycle Bin whenever it's able to. If the recycle bin is disabled (globally, or for a particular drive), or when deleting from locations like network folders that don't support the Recycle Bin, files will be deleted directly (and therefore no undo would be available in those cases). This option can be overridden with the **RECYCLE** and **NORECYCLE** arguments, and the **SHIFT** argument can also affect whether the Recycle Bin is used or not.

When Opus deletes a file directly, as opposed to moving it to the Recycle Bin, the following options affect the behavior:

- **Ask for confirmation for each selected file before deleting:** With this option on, you will be prompted for each selected file, with the option to delete or skip it. You will only be prompted for files selected at the top level; you will not be prompted for any files within any selected folders. This option can be overridden by the Delete command's **QUIET** argument.
- **Ask for confirmation for each selected folder before deleting:** With this option on, you will be prompted for each selected folder, with the option to delete or skip it. You will only be prompted for folders selected at the top level; you will not be prompted for any files or folders below those folders. This option can be overridden by the Delete command's **QUIET** argument.
- **Delete read-only files automatically:** With this option on, read-only files will be treated like any other - their **R** attribute will be cleared automatically so they can be deleted without showing an error. This option can be overridden with the **FORCE** argument to the **Delete** command.
- **Use Secure Wipe:** If this option is enabled, Opus will perform a secure wipe of each file before deleting it. This theoretically makes it impossible (or at least much harder) to recover the file contents. You can select the number of overwrite passes to perform. This will slow down deleting greatly - you will probably not want to turn it on here, but instead use the **Secure Delete** command (the **SECURE** argument for the **Delete** command) when appropriate.

The bottom of this page displays the current size of the Recycle Bin, and lets you empty it and access the system Recycle Bin configuration dialog.

## Double-click Files

The options on this page let you control what Opus does when you double-click on certain types of files in a file display. The [File Types](#) system gives you far more control over double-click actions than this page.

- **Open unregistered file types in text viewer if they appear to be plain text:** If you double-click on a file of an unregistered type (or one without a file extension), Opus will examine its contents, and if it looks like a plain text file it will treat it as one, and open it in the default text viewer (usually Notepad). This is particularly handy for files from other file systems that don't necessarily have a **.txt** extension. If you want to use a text viewer other than the default you can select one to use.
- **Use internal sound player for WAV files:** If this option is on and you double-click on a **.wav** file, Opus will play the file in its own simple sound player rather than opening the normal media player application.
- **Use internal picture viewer:** Similarly, you can choose to have Opus use its own picture viewer when you double-click on image files rather than launching whatever the default application is. You can choose to have this happen for all recognized image files, or only unregistered file types.

## Filters

This page lets you define and edit filters that can be used by various functions in Opus (for example, **Copy File** and **Delete** can use filters to control which files in sub-folders are affected, the [Advanced Find](#) tool uses filters to define what to search for, etc.). While you can usually edit filters directly when using them in the various functions, this page provides a central repository of stored filters.

Use the toolbar buttons at the top of the page to manage filters:  (add a new filter),  (duplicate an existing filter),  (rename a filter),  (assign a description to a filter) and  (delete a filter). Filters that you create in here will be available for selection via a drop-down list in the various tools that use them, and some commands also accept the name of a pre-defined filter given via command line argument.

For instructions on how to actually define a filter, please see the [Filtered Operations](#) page.

## Inline Rename

This page contains options that control how the [inline rename](#) function behaves.

- **Automatically number files when names clash:** When using inline rename, if you try to rename a file to a name that is already in use, this option causes Opus to automatically append a number to the new filename. For example, if you enter the new name "Manuscript.txt" and a file of that name already existed, Opus would automatically rename the file to "Manuscript (1).txt" instead of showing an error.
- **Retain cursor position when moving to next/previous file:** When this option is on and you press the **Up** or **Down Cursor** keys to move from one file to another in inline rename mode, the cursor position within the file will be maintained. See the [Inline Rename](#) section for a more detailed discussion of this subject.

- **Default selection mode:** This lets you change which part of the filename (if any) is automatically selected when inline rename mode begins. If the **Retain cursor position** option is turned on, the selection will also be applied to subsequent files as you move through the list with the **Up** or **Down Cursor** keys.

## Logging

The options on this page let you control which file operations Opus will keep a log of. The log can be quite handy at times - anyone who's ever dropped files into the wrong folder by mistake and then spent 20 minutes looking for them will know what we're talking about! You can view the file log by selecting the **Logs / File Log** command from the **Help** menu. You can also set the maximum size of FTP logs from this page.

- **Maximum size of FTP Logs:** This specifies the maximum size of FTP logs. Opus maintains a separate log for each FTP site you connect to, as well as a unified log for all FTP activity. Each log will be no larger than the number of kilobytes specified here.
- **Log file functions:** This option enables the file function log, and lets you specify the maximum number of events Opus will remember.
- **File Function Logging Level:** This lets you select the amount of detail Opus stores in the file function log. *Standard* will only log copies and moves initiated by drag and drop, and delete operations. *Maximum* will log everything and *Custom* lets you choose exactly what operations to log.

## Metadata

These options define what happens when Opus saves certain metadata to files (via the [Metadata Pane](#) or the [Edit Metadata](#) command).

- **Write APE tags:** When writing metadata to MP3 files, Opus will save tags in APE format as well as ID3v2.
- **Write ID3v1 tags:** When writing metadata to MP3 files, Opus will save an ID3v1 tag as well as ID3v2.
- **Apply changes automatically in the metadata panel:** When this is turned on, any changes you make to the metadata of a file via the panel will be saved automatically when a new file is selected or the panel is closed.
- **Update last modified file dates when setting metadata inside files:** Normally the modification date of a file is unchanged when Opus saves metadata - if this option is on, the timestamp will be set to the current time. This option deals with metadata changes which are written into the files themselves, where the file format has inherent support for it, such as ID3 tags in MP3 files and EXIF tags in JPG files.
- **Update last modified file dates when setting metadata in NTFS ADS:** If this option is on, file timestamps will be set to the current time when metadata changes are saved into NTFS ADS (alternate data streams). With this option on, a file's date will change if you place or modify a label on it (if labels are stored in the filesystem rather than a central configuration file). Changes to file descriptions, ratings, and the general "tags" field will also bump a file's date, for file formats where Opus does not have a way to store them in the main file itself (e.g. text files). You would normally want this option off unless you are using a backup tool which preserves NTFS ADS and uses the modified timestamp to decide what to back-up.

## Options

This page contains miscellaneous options relating to file operations.

- **Append " - Shortcut" when creating shortcuts:** Lets you control whether a suffix is automatically added to the names of shortcuts that Opus creates using the **Copy MAKELINK** command (e.g. when you drag & drop a file with the right button and choose *Create Shortcut Here* from the context menu).
- **Deselect files used in functions:** Normally when you select one or more files and then invoke some sort of function on them (e.g. copy them to another folder), Opus deselects the files involved in the function automatically. If you turn this option off, the files will remain selected so that you can perform additional operations on them without having to reselect them.
- **Postpone file deselection until end of function:** In conjunction with the above option - the files will not be deselected until the function has finished. With this option off, the files would be deselected as soon as the function is invoked. When this option is on, the function only maintains a "weak" connection to the files involved - as soon as you click in the Lister or do anything else that involves selecting or deselecting files, the link is broken and the files will no longer be automatically deselected.
- **Detect external file changes on network drives:** This option causes Opus to monitor network folders for changes made by other programs. For example, if you use a third-party archiving tool in a network folder, Opus would not detect any files it created in a network share unless this option was on. Monitoring network drives like this can result in reduced network performance (some file systems are worse than others) and so you may want to turn this off - you can always manually refresh a file display by pressing **F5**.
- **Re-upload modified files:** This option lets you (sort of) edit files directly on FTP sites. When you double-click on a file (say, a Word document) on an FTP site, Opus will download it to a local temporary copy, and then open the local copy in Word. If you make changes to the file and save it to disk, Opus can detect this and re-upload the new version of the file to the FTP site automatically. There are a number of different options for this monitoring:
  - **Check for modifications when launched process exits:** This is the default option, as it involves the lowest system overhead. Opus waits for the program it launched to exit, and then checks the temporary file for changes. Unfortunately this option is not always reliable - many programs when run simply send a message to an existing instance of themselves and then exit immediately. If you have this option on and you find Opus isn't detecting your changes, try one of the other options below.
  - **Monitor file until it changes:** With this option, Opus will continually monitor the temporary file until it changes exactly once. Subsequent changes will not be detected.
  - **Monitor file for X minutes:** Opus will monitor the file for any changes that occur within the specified time. When the timer expires, Opus will stop monitoring the file.
  - **Monitor indefinitely:** Opus will monitor the file for any changes, and keep monitoring it until Opus itself exits (or you restart or shutdown the computer).
  - **Ask before uploading:** With this option on, Opus will ask you before it re-uploads a file it has noticed a change in. If this is turned off the file will be uploaded automatically.
  - **Use file checksum to detect modifications:** Normally Opus detects a file has changed by comparing its timestamp against the original, but with this option on Opus will also calculate "before" and "after" checksums and compare those. This prevents unnecessary uploads of files that may have been re-saved but haven't actually changed.

## Progress Indicators

These options control the progress indicators that Opus displays when copying or moving files, or for other file operations.

- **Count files in folders before copying:** The way file systems work is that you can only find out how many files a folder (and its sub-folders) contain by reading the folder contents and counting the files. Unfortunately we need to do this if we want to provide an accurate progress indication when copying folders. You might select a single folder that contains 10,000 files but unless Opus had counted its contents it would only see it as a single item. This option lets you control whether Opus counts files in folders before copying or not. Counting files can slow down the copy operation slightly, depending on the number of files involved and what drives you are copying from. You can choose to enable counting for only some drive types, all drives or none.
- **Count files in folders before deleting:** Similar to the option above, this controls whether Opus counts files in folders before deleting them. This only applies when not deleting to the recycle bin, as Windows displays its own progress dialog for recycle bin deletes.
- **Delayed progress indicators:** This option is used to stop Opus displayed progress indicators for very quick operations. The indicator for a function won't open at all if the function finishes within the specified time (in milliseconds).
- **Hide progress indicators from taskbar when jobs bar is visible:** When the jobs bar is visible, progress indicators will no longer show a "footprint" on the taskbar, and won't appear in the **Alt+Tab** window list. This means that when progress indicators are minimized they will *only* be accessible from the jobs bar. If the jobs bar is closed and there are hidden progress indicators they will automatically be displayed on the taskbar again, so there is no risk of "losing" a job.
- **Minimize progress indicators:** If this option is turned on then progress indicator dialogs will be automatically minimized when they open. Depending on the state of the previous two options, this might mean that the only indication of a new job is a new button appearing on the jobs bar, and so Opus will display a small arrow animation pointing to the newly created jobs bar button to provide feedback that the operation was successfully started.
  - **Only when the jobs bar is visible:** Progress indicators will only be minimized if the jobs bar is displayed - if not, they will open in front of the Lister as normal.
- **Show speed graph in file copy progress indicators:** If this option is on the progress indicator will display a graph representing the transfer speed over time.
- **Show percent complete in progress bar titles:** Opus will display the overall percent complete for a function in the title bar (on XP / Vista this means you can minimize the progress indicator and still keep an eye on the progress in the taskbar - on Windows 7 the progress is displayed directly in the taskbar icon and so this option isn't as useful).
- **Slide animation when queued jobs begin:** If on, and a progress job completes while another job is queued behind it, the old job will slide off the top of the progress dialog as the new jobs slides on the bottom. This adds a slight delay between jobs, and can be turned off if undesired. (Regardless of this setting, the slide animation is automatically suppressed if using Remote Desktop or if client-area animations are disabled system-wide.)

## Jobs bar

These options control the optional jobs bar, which provides a way to monitor and control file operations in the background.

- **Always display the jobs bar:** The jobs bar will always be shown at the bottom of Listers, even when no jobs are running.
- **Display the jobs bar automatically when starting a new job:** If this is enabled, the [jobs bar](#) will open automatically when an operation that displays a progress indicator opens. If you don't have the jobs bar opening automatically you can still show it manually using the [Set JOBSBAR](#) command.
  - **Show in all Listers:** The jobs bar will be displayed in all existing Listers, not just the one that started the job.
- **Resize Lister to display the jobs bar:** If this is turned on then the Lister will be resized vertically to accommodate the jobs bar when it's displayed (and its original size restored when the jobs bar closes). This is only possible if the new size fits on screen - for example, if the Lister window is maximized it can't be resized and in that case the jobs bar will take space out of the from the existing window.
- **Same size buttons on the jobs bar:** Turn this option on to make all buttons on the task bar the same width - if turned off, they will automatically size to fit their contents. You can also specify a **Fixed size** in pixels.

## Folders

This category contains options that relate to the how folders are read and displayed (including display options that can be configured for specific folders).

- [Auto-Loading](#): Options that affect which types of folders are automatically loaded when re-opening saved windows.
- [Folder Behavior](#): Options that affect what happens when you navigate into a new folder.
- [Folder Display](#): Options that control how files and folders are displayed.
- [Folder Formats](#): Lets you configure the display format for specific folders, or folders containing certain types of files.
- [Global Filters](#): Options affecting how all file displays and folder trees will hide and sort items.
- [Virtual Folders](#): Controls how Opus handles virtual folders like the Desktop and My Computer.

### ***Auto-Loading***

The options on this page allow you to prevent Opus from automatically loading certain types of folders (e.g. when a Lister opens, it will normally automatically load a folder).

This can be very useful if you have Opus set to re-open the last location when it runs, or to open a specific Lister layout that displays network drives or another slow drive type. It stops Opus from attempting to automatically read the specified types of folders when a Lister first opens. For example, you could have a layout that reads a network drive - with this option on, opening that layout would not automatically connect to the network drive, but instead you will be prompted to click a link in order to read the folder.

- **CD-ROM/DVDs:** Folders located on CDs and DVDs will not be automatically loaded.
- **Drives that are powered down:** Hard drives that are in sleep mode (have powered down) will not be automatically loaded. This stops Opus from waking them up until you manually read the folder.

- **Floppy discs:** If you still use floppy discs for some reason, this option stops them being automatically loaded.
- **FTP sites:** Connections to FTP sites will not be automatically reconnected until you manually load the folder.
- **Network drives:** Folders located on network drives will not be automatically loaded.
- **Removable devices:** This applies to things like USB thumb drives and some (but not all) external hard drives.
- **Zip files and other archives:** Irrespective of the device they reside on, archives will not be automatically loaded.
- **All other drive types:** Applies to everything not listed separately.

For each folder type, you can select from three options:

- **Load normally:** Loads the folder.
- **Prevent loading:** Prevents automatic loading of the folder. A message will be displayed at the top of the file display with a link you can click to trigger the read.
- **Load on tab activation:** Prevents the loading initially if the tab isn't the active one, but will load the folder automatically the first time the tab is activated.

## ***Folder Behavior***

The options on this page are mostly involved in what happens when Opus reads a new folder.

- **Allow manual sorting in all folders:** Some folders, like FAT32-formatted drives, don't support saving of [manual sort](#) information. If this option is turned off, you cannot even turn on manual sorting when on a drive where it cannot be saved, to prevent you from spending time organizing things only to find that you cannot save them. If the option is on, Opus will let you turn on manual sorting in such a drive, and you can re-arrange the files, but the sort order will be lost when the window is closed.
- **Automatically save manual sort order where possible:** If this option is on Opus will automatically save the manual sort order (if it can) whenever you rearrange files. If turned off, you need to specifically select the Save Sort Order command to make the sort order permanent.
- **Calculate folder sizes automatically:** When you navigate to a new location, Opus will launch a background thread that calculates the total size of all sub-folders. If this option is off you can always use the **Calculate Folder Sizes** command in the Edit menu. When this option is on you can choose what type of drives to apply it to. The **Skip junctions and softlinks** option can be used to stop Opus automatically calculating the size of any links or junctions that point to folders - only "real" folders would have their size calculated automatically.
- **Cancel Checkbox mode when folder is changed:** If the file display is in [checkbox mode](#) and you navigate to a new location, checkbox mode will be automatically turned off.
- **Cancel Flat View mode when folder is changed:** If the file display is in [Flat View](#) mode and you navigate to a new folder, Flat View will be turned off.
- **Enable Folder Content Type detection:** This option enables the [Content Type](#) detection system for the specified types of drives. When this is on and you navigate to a new folder, Opus compares the contents of the folder against the configured [Content Type formats](#) and automatically changes the display format if

appropriate. For example, the display might automatically switch to thumbnails mode when reading a folder containing mostly images.

- **Ignore junctions and softlinks when calculating folder sizes:** When calculating the size of a folder (either due to the automatic option above, or when triggered manually via the **GetSizes** command), this option causes Opus to ignore and junctions or softlinks within the folder.
- **Select previous folder when going Up:** When this option is on and you go up to the previous folder, the folder you just came from will be selected. Note that the **Up** button on the Location toolbar actually runs the **Go UP BACK** command - the **BACK** argument makes **UP** act like **BACK** if the previous folder in the history is the parent folder. Because file selection is preserved when going back and forward in the location history, the parent folder will most likely be selected irrespective of the state of this option. So long story sort, if you turn this option off and find that when you go up the parent is still selected, that's why!
- **Sort shortcuts to folders like folders:** With this option on Opus will treat shortcuts to folders as if they were the folders themselves when sorting the file list (meaning they'll be grouped with the folders rather than with the files).
- **Path completion in path fields:** This enables path completion in any string field that is used to enter a folder path (e.g. the location field on the toolbar).
  - **Automatically, as I type:** Path completion occurs automatically - as you type into the field, the first folder name that matches what you have typed so far will be automatically filled in for you. You can ignore it and keep typing, or press a path separator key (:, / or \) or **Enter** to accept the completed path component, or press the **Up** or **Down** cursor key to scroll through other matching folder names.
  - **When I press the cursor keys:** Path completion does not occur until you press the **Up** or **Down** cursor key, at which time the first matching folder name will be completed as above.
  - **Complete paths for local drives only:** Path completion is enabled for local drives (hard drives, USB drives etc) only and not for network shares etc.
  - **Complete network server names:** When path completion is enabled for network paths, this option specifies whether or not Opus should enumerate the computers on your network and do path-completion for the computer names. For example, if you type \\a you will see computer names starting with the letter a. The enumeration is done in the background but you may still wish to turn it off if your network is slow or has a large number of computers.
  - **Display popup list:** A popup list will appear above or below the path field showing possible matches for what you have typed so far. You can use the up and down cursor keys, or the mouse, to select items from the list. If you find the list gets in your way, you can turn it off.

As well as regular paths Opus also auto-completes paths that you begin with a **%** character (for environment variables), paths that begin with a **/** (aliases), and URL-style paths (**coll://** or **ftp://**, for example). When the cursor is at very beginning of the field there are also some special characters that will be expanded to certain locations as a shortcut:

- **~ (tilde):** Your personal profile folder
- **` (back-tick):** The desktop directory
- **\$ (dollar sign):** Your documents folder
- **\* (asterisk):** Your favorites folder
- **} (right brace):** The Windows folder
- **^ (caret):** The Windows System folder
- **# (hash/pound sign):** The program files folder

- **Use 'descript.ion' file comments system:** Opus lets you assign descriptions to files using the **Set Description** (and **Edit Metadata**) commands. If this option is turned on, those descriptions are stored using the semi-standard DESCRIPT.ION system, where a file called "descript.ion" is created in the folder to hold the descriptions. If you turn this option off, Opus will instead store descriptions using NTFS metadata - this only works on NTFS-formatted drives but means you don't have a lot of extra files cluttering up your folders.
  - **Hide 'descript.ion' comment files when marked as Hidden:** In conjunction with the above option, when the DESCRIPT.ION files have their **H** attributes set, Opus will hide them from the file display (even if other **H**-marked files are not hidden).

## Folder Display

This page contains options that affect how Opus displays the contents of folders.

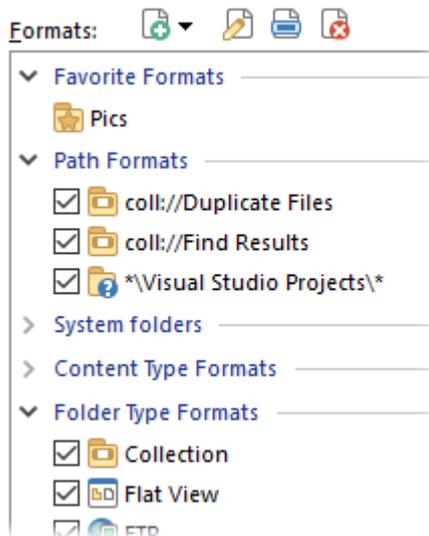
- **Add 'Group' column automatically when file display is grouped:** Automatically adds the **Group** column when the file display is [grouped](#). The column prevents the group headers from appearing between groups, using horizontal space instead of vertical space to indicate where each group starts and ends. If the option is off, you can still add the column explicitly, allowing you to use both styles of grouping in different situations.
- **Display Compatibility Files where possible:** When this option is on Opus will automatically merge the contents of a folder's [compatibility folder](#) when it has one. For example, the compatibility folder for *C:\Program Files* is *C:\Users\<Your Name>\AppData\Local\VirtualStore\Program Files*. With this option on, the display of *C:\Program Files* would automatically include the contents of the compatibility folder as well, with items from that folder displayed in a different color. This option doesn't apply to Windows XP.
- **Display localized folder names:** Vista introduced the concept of localized folder names. In Windows XP, the *C:\Program Files* directory (for example) is physically renamed in other languages - e.g. on French systems the folder is *C:\Programmes*. In Vista and above, the folder is always called *C:\Program Files*, but its name is translated when it's displayed in the user interface. With this option on Opus will display the localized name for such folders instead of the real name in the filesystem.
- **Hide the '.lnk' extension for shortcuts:** Shortcut files have the extension **.lnk** - if you turn this option on, the **.lnk** extension will be hidden in file lists.
- **Highlight previous folder on Up/Back:** If this option is on and you go up (or back) to the parent folder, the child folder you came from will be highlighted with a blinking underline for the specified time.
- **Modify display of all-uppercase filenames:** This lets you modify the display of filenames that are all upper-case, for example legacy files copied from non-long filename filesystems like FAT-formatted drives. The filenames themselves are not altered, just how Opus displays them in the file list.
- **Show '..' parent item in folders:** When this option is on Opus adds a **..** item to the top of file lists. The **..** is a long-used symbol for the parent folder, and the **..** item lets you go up to the parent folder by double-clicking it as if it were a child folder. You can also drag and drop to the **..** item to copy or move files to the folder's parent.
  - **Hide when file display is grouped:** When the file display is [grouped](#), you may not want the **..** item displayed - if it is, it ends up in the *Unspecified* group, which is normally at the very bottom of the file list, and so its presence there may not be as useful as when it's at the top. This option lets the **..** be automatically hidden whenever the display is grouped.
- **Show shortcut arrows and other icon overlays:** If this option is off, Opus will not display icon overlays such as arrows in the corners of shortcut icons, shared folder indicators, or status indicators from source-control and folder-syncing shell extensions (e.g. TortoiseSVN and DropBox).
- **Show day names in date columns:** If this option is on and a file's timestamp is within the past week, the day of the week (or *Today* or *Yesterday*) will be displayed instead of the actual date.

- **Show relative graphs behind modified date columns:** Displays bar graphs behind the date columns indicating a file's timestamp relative to the oldest file in the folder. These are the same graph shown when the separate *Modify (relative)* and *Create (relative)* columns are turned on.
- **Show relative graphs behind size columns:** Displays a bar graph behind the *Size* column indicating a file's size relative to the largest file in the folder. This is the same graph shown when the separate *Relative Size* column is turned on.
- **Show seconds in time columns:** Opus will show seconds (*hh:mm:ss*) in time columns with this option on - otherwise, it only displays the time using hours and minutes. If seconds are displayed you can also turn on the **Show milliseconds** option to display timestamps with millisecond resolution (to the limit of the underlying file system).
- **Show generic icons:** Most file types use the same icon for all files of that type - for example, all *.txt* files generally have the same icon. However some file types (most notably *.exe* files) can have a different icon for individual files. Loading this icon can slightly slow down the display time of folders, particularly on network drives, so you can use this option to control whether Opus displays generic icons rather than per-file icons for these types of files.
- **Show cloud storage status icons in the Status column:** In Windows 10, when inside a OneDrive or Dropbox folder, this will cause an icon representing the file's synchronization state to be shown in the *Status* column automatically. Similar status icons will also appear next to files in display modes like List and Thumbnails.
- **Show TortoiseSVN status icons in the Status column:** For developers using the free TortoiseSVN source code control utility, Opus can optionally display a file's SVN status as an icon in the *Status* column. Other than the status icon appearing larger and more distinctive than the usual icon overlay that Tortoise uses, this can help with the problem of limited icon overlays. Windows only allows a maximum of 15 icon overlays in the system, and Tortoise normally uses 8 or more of these for itself, crowding out other shell extensions. Using this option in Opus gets around the limit as the status is taken directly from Tortoise rather than via the icon overlay. You can also select the icon set to use, and optionally disable the icon overlay handler completely (so that the *Status* column is the only place the TortoiseSVN status icons will appear). Note that this feature requires TortoiseSVN version 1.9.3 or greater.

## Folder Formats

This page lets you configure the [Folder Formats](#) system, which lets you define folder formats (display parameters) for specific folders, folders containing certain types of files, or defaults for certain types of folders. For example, you can make it so that Opus automatically sorts a particular folder by date in reverse, or switches into thumbnails mode in any folder containing mostly images.

The Folder Formats system is powerful but rather complex. There's an [FAQ on it at the Resource Centre](#) that we recommend you have a read of as it may explain things in a bit more detail than this help file does.



The Formats list is grouped into different sections:

- **Path Formats:** This group contains formats that have been defined for specific folders. You can also define a format for a path using a wildcard string - in the screenshot above, a format has been defined for any path below a folder called *Visual Studio Projects*, as well as for the folder *C:\Test*.
- **System folders:** Contains formats that can be defined for certain special locations. There are currently only two formats in this group - the **Computer** folder, which is used when Opus displays My Computer itself rather than hosting Explorer, and the **Portable Devices** folder, which is used when viewing the root of the [mtp://](#) namespace.
- **Content Type Formats:** The Content Type section contains formats that are defined for [file type groups](#). When the [Content Types](#) system is enabled, these formats will be used automatically when a folder containing enough of those types of files is read. For example, the Content Type format for the Images group can be set to switch the display into thumbnails mode. If you navigated to a folder that contained mostly images, this format would be applied automatically.
- **Folder Type Formats:** This section defines the default formats that are used for certain types of folder when a specific format hasn't been defined. The default types are:
  - **Collection:** The default format for [File Collection](#) folders.
  - **Flat View:** This specifies a format that is applied whenever a file display is put into [Flat View](#) mode.
  - **FTP:** The default format for FTP sites.
  - **Local Drives:** The default format for folders located on local drives (fixed hard drives like C:).
  - **Network Drives:** The default format for folders located on network drives.
  - **OneDrive:** The default format for OneDrive synchronized folders (only under Windows 10).
  - **Portable Devices:** The default format for folders located on portable devices (like phones and cameras).
  - **Removable Drives:** The default format for removable drives like USB flash drives.
  - **Search Results:** The default format for a File Collection that is used to present search results. This will override the **Collection** format for these collections.
  - **Synchronize:** This is a format that is applied whenever the [Synchronize](#) tool is used.

- **ZIP:** The default format for ZIP files.
- **Favorite Formats:** These are formats that aren't linked to a specific folder, but instead can be applied quickly to any folder through a drop-down list in the Lister's *Folder Options* menu.
- **Default Format:** Contains the *User Default* format. This is the over-all default format that is used if no other format has been defined (see below).

You'll notice that some formats in the list have checkboxes - this lets you turn a format on or off (in the case of Path Formats, it lets you temporarily disable a format without deleting it from the list).

Whenever you navigate to a new folder, Opus consults the Formats list to work out what format to display it with. The list is searched from top-to-bottom:

1. First, if a **Path Format** has been configured for that specific folder, or the folder matches a configured wildcard format, it will be used.
2. Secondly, the **Content Types** system is checked if enabled. If the folder contains enough files to meet the requirements of one of the Content Type formats, that format will be used.
3. Next, if no other format has matched, the **Folder Type Formats** for that type of folder or drive is used if it's turned on.
4. Finally, if a **Folder Type Format** hasn't been configured (or is turned off), the **User Default** format is used. This is a "catch-all" format that can never be turned off and represents the underlying default format for all folders.

(As a side note: There are some cases when the above doesn't happen when you navigate to a new folder. If you have edited the format in the current file display with the [Folder Options](#) dialog, step 4 does not take place - your manually edited format will be maintained until you navigate to a folder with its own specific format defined. And if you have the [format lock](#) turned on, the format will **never** automatically change).

To edit the formats, use the toolbar buttons at the top of the page:  (add a new format),  (edit an existing format),  (rename a Favorite format) and  (delete a format). Clicking the **Add** button displays a drop-down menu that lets you select the type of format to add:

- **Path Format:** a format for a specific path.
- **Wildcard Path Format:** a format for any path that matches the specified wildcard pattern. You can define the pattern using the standard Opus [wildcard system](#), or using [regular expressions](#). If the **Expand aliases** option is turned on then Opus will attempt to expand folder aliases and environment variables in the entered string before performing the pattern matching. For example, `/$Data` (which is a folder alias for a drive called *Data*) would let you create a folder format that applies to a drive labeled *Data* no matter what drive letter it has.

- **Content Type Formats:** Actually this option doesn't do anything except open the [File Types dialog](#) for you. Content Type formats are based on [File Type Groups](#), and so to add a new Content Type you have to use the File Type editor to add a new group. Once you add the group through the File Type editor, it will appear in the Content Types section in the Formats list and you can then define the format for it.
- **Favorite Format:** You will be prompted for the name of the new format.

As well as using the toolbar buttons, you can also:

- Double-click a format to quickly edit it
- Right-click a format to copy its definition and paste it into another, an existing format to your favorites list, or copy the definition of a favorite format into another format.



When you edit a format Opus displays the **Edit Format** dialog. This dialog contains several tabs with options that control various aspects of the folder display. Each tab contains several sections and these sections can be enabled or disabled separately. When a section is disabled the options within that section are not applied to the display even when the format itself is used. This means that "more specific" formats (like a Path Format for a certain folder) can inherit settings from "more general" formats like one of the Defaults. For example, the wildcard format for *Visual Studio Projects* in the screenshot above does nothing except set the file display to be grouped by Date modified. The rest of the display parameters will come from the applicable default format.

See the [Folder Options](#) page for a full discussion of the options available for folder formats. The **Options** tab has a couple of settings that only apply to saved folder formats:

- **Include columns from other matching formats:** This option affects which columns are displayed when this format is in use. If turned off (and the checkbox for the **Columns** page is enabled), the only columns displayed in details or power mode will be the columns defined by this format. If this option is turned on, then the columns defined by this format will be displayed, *as well as the columns for any other formats that match the current folder*. In this case, the method used to find a folder format described above is modified slightly. The first matching format is still used, but the search doesn't stop immediately - Opus will continue down the order of priority looking for other formats that match, and apply any columns (and only columns - no other settings) that those formats define. This process stops as soon as a matching format is encountered that doesn't set the Include columns from other matching formats option.
- **Use as the default format for all sub-folders:** This option is used with path formats - if turned on, the format defined for a path will also be used for any sub-folders under that path, unless they have their own specific path format defined as well.

## Global Filters

This page contains options that let you specify files that Opus will never show in file displays or folder trees. While you can configure filters on a per-folder basis using the [Folder Formats](#) system (specifically, the Show and Hide [Filters](#) pages), this provides an easy way to hide files globally without needing to configure folder formats.

- **Enable global wildcard filters:** With this on you are able to define wildcard filter strings for both files and folders that are to be hidden globally. You can match files by name or file extension, either using the Opus [pattern matching syntax](#) or, with the **Use regular expression** option turned on, [regular expression syntax](#). For example, to never display desktop.ini files or thumbs.db files, you might specify the pattern **(desktop.ini|\*thumbs\*.db)**.
- **Hide hidden files:** This option causes Opus to hide files and folders that have **H** attribute set and do not have the **S** attribute set. You can toggle this option quickly via the **Show Hidden Files** item in the **Folder Options** menu on the [Menu toolbar](#).
- **Hide protected operating system files:** This option causes Opus to hide files and folders that have both the **H** and **S** attributes set, which generally indicates they are a special operating system files and folders. You can toggle this option quickly via the **Show System Files** item in the **Folder Options** menu on the [Menu toolbar](#).
- **Ignore prefix when sorting:** This is similar to the **Ignore prefix** option in the [Folder Format](#) editor - it lets you configure one or more filename prefixes that Opus ignores when sorting filenames alphabetically. This option takes effect globally in all file displays and also in folder trees. Multiple prefixes should be separated with a vertical bar character. For example, **The |A |An** would ignore those three prefixes when sorting.

All filtering can be quickly disabled in a folder tab by toggling [Show Everything](#) mode.

## Virtual Folders

This section lets you configure how Opus handles certain [virtual folders](#). These are folders that do not relate to a physical location on a disk - that is, they can not be represented by a traditional path like *C:\Users\*. There are many virtual folders provided by Windows - some common examples are *My Computer* (or *Computer* under Windows 7), the *Desktop*, *Network* (used to be called *Network Neighborhood*) and so on. Third parties can also implement virtual folders via so-called "namespace shell extensions" - for example, your mobile phone may have come with software to access it through Explorer, and this would most likely be implemented as a virtual folder.

For most virtual folders, Opus provides a display of the folder by "hosting" Explorer within the file display. This lets you see the folder and interact with it in the same way as you would in Explorer itself (primarily via drag and drop, cut and paste, and context menus).

For the two most commonly used virtual folders - *Computer* and *Desktop* - Opus has the option of providing its own display (known as "native display") instead of hosting Explorer. The advantage of this is that then the full gamut of Opus functionality is available, including folder formats, configurable colors and images, and (for *Desktop*) Opus commands like Advanced Rename, etc. For the most part the Opus-provided displays of these folders will look exactly the same as the hosted form, so you may not even realise it is Opus and not Explorer providing the display.

The **Native display of the Desktop** option lets you enable or disable native display for the virtual Desktop folder. When native display is on, you can choose which desktop contents are included:

- **Documents / Profile folders:** Includes folders for your personal files. The actual folders vary depending on your OS version. For example, on Windows XP this will include the *My Documents* folder, on Windows 7 it will include your profile folder.
- **All Users / Shared Desktop contents:** Includes desktop files and folders from the *All Users* desktop. As well as the special folders like *Computer*, *Network*, etc, the desktop presents a merged view of two underlying disk folders, which is where files that you drop on the desktop are actually stored. One folder contains your own personal files, and the other contains files that are accessible to all users. This option lets you hide the All Users files, leaving only your own personal files displayed.
- **Computer:** Includes an item for the *Computer* virtual folder (the folder that displays your disk drives and network shares).
- **Collections:** Includes a link to the *File Collections* root folder (the folder that displays all your top-level [File Collections](#)).
- **Network:** Includes an item for the *Network* virtual folder (the folder that displays your local network).
- **Libraries:** Includes an item for the *Libraries* root folder (the folder that displays all your [libraries](#)).
- **Recycle Bin:** Includes a link to the *Recycle Bin* folder.
- **FTP:** Includes an item for the *FTP* root folder (the folder that displays the top-level of your FTP Address Book).
- **Operating system files:** If this option is off, operating system files (those marked with the **H** and **S** attributes) will be hidden from the desktop. The desktop quite often contains two **desktop.ini** files (one in your personal desktop folder and one in the All Users folder) so being able to hide these is quite useful.
- **Control Panel:** Includes a link to the *Control Panel*. Note that in Vista and above, double-clicking this link will open Control Panel in Explorer.

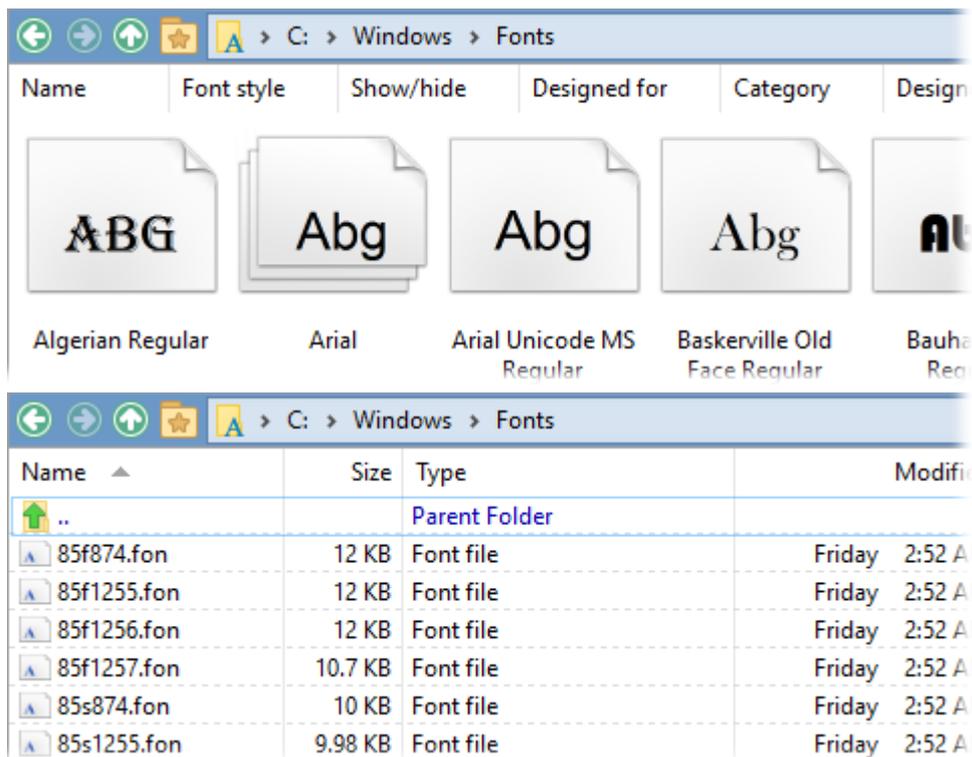
The **Include additional folders when using Windows Search from the Desktop** option controls which folders are searched when you use Windows Search (e.g. via the search field in the top right of the Lister) from the Desktop folder. By default, Opus only searches your personal Desktop folder (**/desktopdir**) and, if configured to appear, the shared desktop folder (**/commondesktopdir**). If this option is turned on the search will include the profile and shared Desktop folders and also the Documents folder, and potentially others. (It is up to Windows exactly which folders are searched, and you'll see similar from Explorer.)

The **Native display of Computer** option similarly lets you enable or disable native display for the *Computer (My Computer)* folder. The options for this are:

- **Include "Files Stored on This Computer"**: This option adds a group to the Computer display which contains links to the document folders on this machine (your own personal folder, and the shared documents folder). This option is provided for compatibility with Windows XP's Explorer, which includes similar entries. Explorer in Vista and Windows 7 does not show these items in the Computer folder but in Opus you can enable them if desired.
- **Show empty disk drives**: If this option is turned off, removable disk drives will be hidden from the list of drives if they are empty.
- **Sort drives by name instead of letter**: By default drives are sorted by drive letter (C:, E:, L:, etc). If you turn this on, they will be sorted by their label.
- **Show drive labels as**: Lets you choose how drive labels are displayed.

You can configure the file display format for both these folders using the [Folder Formats](#) page - for *Desktop*, you would add a **Path Format**, and for *Computer*, use the item in the **System folders** category.

The bottom section of this dialog lets you control how Opus handles virtual folders that also have an underlying "real" disk folder behind them. Windows contains several folders like this - the *Desktop* is one, but others include the *Fonts* folder (usually *C:\Windows\Fonts*) and the *Virtual Machines* folder (usually *C:\Users\<<Your Name>\Virtual Machines*). These folders can be displayed in Opus in two different ways - as the underlying disk folder, in which case you will see all files in that folder, or as a virtual folder, in which case you will see the virtualized view that Windows provides.



The above screenshots show the difference between the virtualized and "real" views of the Fonts folder. You can choose from the following options:

- **Treat all "virtual filesystem folders" as virtual:** All such folders will be displayed in their virtualized forms.
- **Treat all virtual folders as real:** All such folders will be displayed as the real underlying folder. Note that this only applies to virtual folders that actually do have an underlying disk folder - some "purely virtual" folders like Network neighborhood will always be virtual.
- **Treat the following virtual folders as real:** This option lets you specify exactly which folders should be displayed as real. For example, you could set the Fonts folder to always show as real but all other such folders will be displayed as virtual. Use the toolbar buttons to add (+) folders to the list, or remove (-) existing folders.

## Folder Tabs

This category contains options that control how Folder Tabs work.

- [Appearance](#): Options that control the appearance of Folder Tabs.
- [Options](#): General options that affect the behavior of Folder Tabs
- [Tab Groups](#): Lets you define "groups" of multiple tabs (folders) that can be opened at once.

## Folder Tab Appearance

This page contains various options that control the appearance of [Folder Tabs](#).

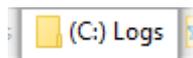
- **Same size tabs:** Normally tabs are sized automatically to fit their label (up to the maximum specified by the option above). With the **Same size** option turned on, tabs will all be the same width. There are two ways this can work:
  - **Automatic size:** Tabs are all the same width, but this width is determined automatically to fit the widest label.
  - **Fixed size:** Tabs will all be the width in pixels that you specify here.
- **Maximum tab size:** This option lets you specify the maximum width of a tab, in pixels. Tabs will never be wider than the width you set here.
- **Minimum tab size:** This option lets you specify the minimum width of a tab, in pixels. Tabs will never be narrower than the width you set here.
- **Reduce tab sizes to fit available space:** As the number of tabs increases the size of each tab will be reduced to avoid the tabs overflowing the available space.

## Folder Tab Options

This page contains various options that control the appearance and behavior of [Folder Tabs](#).

- **Display folder tabs:** Controls when the tab bar is displayed. You can't have zero tabs - there's always at least one tab open, but you can choose whether or not to display the tab bar when there is only one tab.
  - **Always:** Folder tabs will always be displayed, even if there's only one tab open.
  - **When needed:** Tabs will only be displayed when needed, that is, when there is more than one tab open. If you have multiple tabs open, the tab bar will be hidden when the second-last tab is closed.
  - **When needed (balanced):** If the Lister is in dual-display mode, and one side of it has more than one tab open, tabs will be displayed on the other side as well. This is more for aesthetic reasons than functional; it can look strange to have tabs visible on one side but not the other.
- **Tab position:** Controls where the tabs appear in relation to the file display they're attached to. Note that this can be overridden on a per-Lister basis using the [Set TABPOSITION](#) command.
  - **Below:** Folder tabs will be displayed horizontally, below the file display.
  - **Above:** Folder tabs will be above the file display.
  - **Left:** Folder tabs will be displayed vertically, to the left of the file display.
  - **Right:** Folder tabs will be to the right of the file display.
- **Dual display position:** Controls where the tabs appear when the Lister is in [dual-display mode](#). Note that this can be overridden on a per-Lister basis using the [Set TABPOSITION](#) command.
  - **Normal:** Both tab bars appear in the same relative position (as set by the **Tab position** option).
  - **Together:** The tab bars appear together. For example, the top file display's tabs would be below the file display, and the bottom file display's tabs would be above the file display - resulting in the two tab bars being next to each other.

- **Apart:** The tab bars appear apart. For example, the left file display's tabs would be to the left of the file display, and the right file display's tabs would be to the right.
- **Click selected tab:** Controls what happens when you click the already selected tab.
  - **Rename tab:** The tab will go into inline rename mode, letting you assign a custom label to it.
  - **Go to previous:** Clicking the already selected tab will activate the previously selected tab.
  - **Do nothing:** Clicking the tab does nothing.
- **Display drive letter in tab label:** With this option on, the drive letter will be displayed in the tab along with the name of the folder.



- **Display new tab button:** Adds a small + button at the end of the folder tabs, which you can click to quickly open a new tab. The drop-down control lets you choose what folder is displayed in a new tab opened this way:
  - **Current folder:** duplicates the current tab's folder.
  - **Default folder:** This opens the "default folder", which is actually the folder that the [Default Lister](#) opens. To change it you need to change the Default Lister.
  - **Empty tab:** does not automatically read a folder, leaving the new tab empty.
  - **Location:** Opens a specific folder.
- **Double-click tab-strip to open a new tab:** If this option is on, double-clicking an empty spot on the tab bar will open a new tab. You can choose what folder is displayed in a new tab opened this way:
  - **Current folder:** duplicates the current tab's folder.
  - **Default folder:** This opens the "default folder", which is actually the folder that the [Default Lister](#) opens. To change it you need to change the Default Lister.
  - **Empty tab:** does not automatically read a folder, leaving the new tab empty.
  - **Location:** Opens a specific folder.
- **Double-click tabs to close them:** If this option is on, double-clicking a tab with the left button will close it.
- **Folder tab close buttons:** This option causes close buttons to be displayed in each tab (whenever there is more than one tab open). If this is turned off, you can still close tabs by double-clicking them (if the above option is on), using the file display's close button, by clicking them with the middle mouse button (if you have one) or by right-clicking them and choosing **Close Tab** from the context menu. There is also an option to make the close buttons smaller to save space.



- **Lister close button closes active tab:** If multiple tabs are open, clicking the Lister close button (the main close button for the whole window) will close only the active tab instead of the whole Lister. (A similar option also exists for the [File Display border](#) close button, under [File Displays / Border](#).)

- **Lister closes when last tab closes:** If this option is enabled then Listers will close when the last tab within them is closed (similar to how web browsers work).
- **Open new tabs next to the active tab:** Normally new tabs open at the end of currently existing tabs - with this option turned on, new tabs will open immediately to the right of the currently selected tab.
- **Preserve folder tree expansion when switching tabs:** When this option is on, the expansion state of folders in the Folder Tree will be preserved between tabs. That is, when you switch away from a tab, Opus remembers which folders in the folder tree were expanded and which were collapsed, and restores this state when you switch back to that tab.
- **Process file changes in background tabs:** When this option is turned on, tabs that aren't visible still process any file changes that occur in the folder they're displaying. If you turn it off, changes won't be processed for non-visible tabs - instead, the tab will be marked as dirty and automatically reloaded when you switch back to it. Turning off background change processing can enhance performance when you're working with lots of tabs simultaneously.
- **Treat tab label as folder when dragged or right-clicked:** Normally you can drag tabs around, or access their context menu, by clicking anywhere on the tab. If this option is turned on, the tab is split into two parts for the purpose of mouse clicks. The icon acts as the hotspot for the tab, and the label acts as the hotspot for the folder displayed on the tab. So with this option on, you need to click and drag the icon if you want to drag the tab, or right-click the icon if you want to access the tab's context menu. Right-clicking the label will display the context menu for the folder, not the tab, and dragging from the label will represent a drag of the actual folder, which lets you copy the folder or create a shortcut to it using drag and drop.
- **Use popup menu when tabs exceed available space:** There are two options for how tabs are handled when the available space is not enough to show them all at once. With this option off, a small pair of arrows is displayed at the end of the tab bar when the tabs don't all fit, and you can use these arrows to scroll left and right through the open tabs. Note that this option only applies when tabs are displayed horizontally (above or below) - when tabs are displayed vertically, a scrollbar will appear that lets you scroll up and down the tabs if they don't all fit.



When this option is on, a popup menu button is displayed at the end of tab bar instead.



Clicking this button displays a popup menu that lists the tabs that aren't currently displayed. If you turn on the **Show all tabs in menu** option as well, the popup menu will include all currently open tabs, not just the ones that don't fit.

## Tab Groups

Tab Groups are sets of predefined tabs (folders) that can be opened in one operation. For example, you might have a set of work folders for a particular project that you always need quick access to. You could define a tab group that opens all the folders in folder tabs, and then when you're ready to work on that project, you only need to select the group in order to open all those

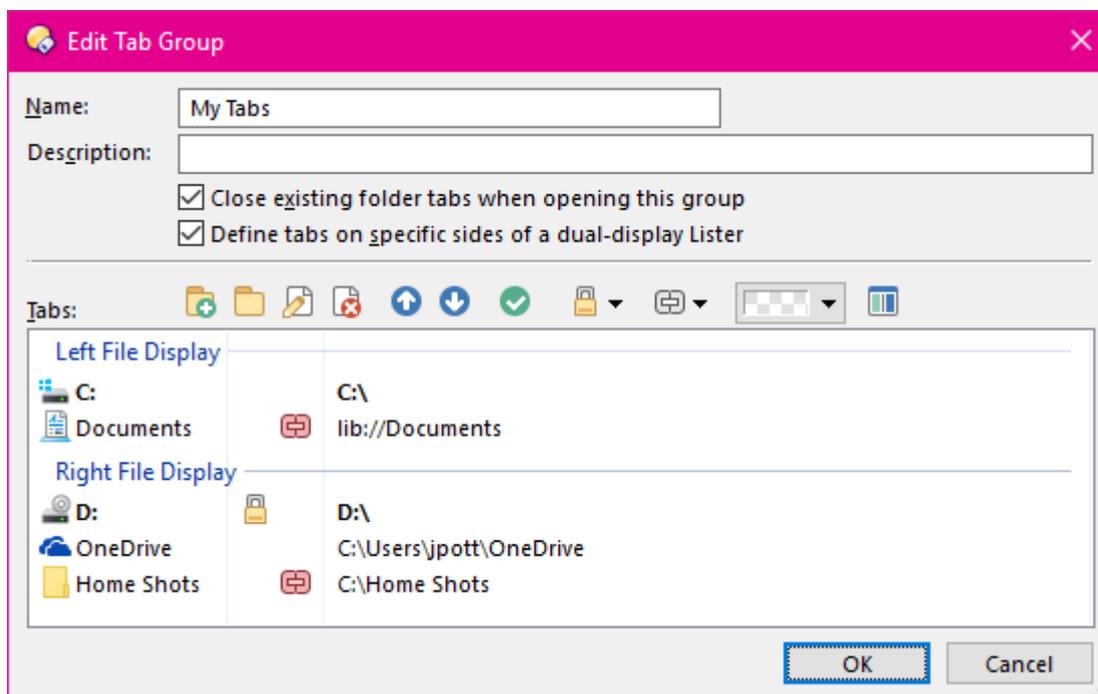
folders at once.



Use the toolbar buttons to configure tab groups. The buttons are  (add a group),  (add a folder - useful for categorizing large numbers of groups),  (duplicate an existing group),  (rename a group),  (edit a group),  (delete a group) and  (mark as hidden, to stop it appearing in tab group lists outside of this Preferences page). If you click on a tab group in the list at the top, the tabs defined in that group will be previewed at the bottom of the window.

To organize tab groups into folders, use the  button to create a folder. You can then create tab groups in that folder, or use drag-and-drop to move existing groups into that folder.

To edit a tab group, select it and click the  button, or double-click it. The *Edit Tab Group* dialog will be displayed.



The **Name** field lets you modify the tab group name, and the **Description** field lets you assign your own description to the group.

Each entry in the **Tabs** list represents a tab, or more specifically, a folder that will be opened in a tab. Use the toolbar buttons above the list to configure the tabs:  (add a folder tab),  (edit an existing folder tab),  (edit the folder format for the tab),  (delete a tab),  (move a tab up the list),  (move a tab down the list),  (set initial tab),  (set tab as locked),  (set tab as linked) and  (change sides).

- You can assign any label you like to a tab - it doesn't have to have the name of the folder displayed on it. Use  to assign a label to a tab (you can also use this button to change the folder for an existing tab). You can use several special "tokens" to insert information in the tab label:

**%P** - full path of the current folder  
**%N** - name of the current folder  
**%R** - drive root of the current folder

- Each tab has a [Folder Format](#) specified for it. Note that this format is saved with the tab itself and so overrides the normal folder format system for the folder - for example, changing the format for a folder through the Folder Formats page will not affect that folder when it is saved in a tab group. Use the  button to edit the format for a tab.
- The arrow buttons let you move tabs up and down the list. The top-to-bottom order of the list represents the left-to-right order of tabs in the file display.
- Each tab group can have one tab that is designated the active tab - this is the tab that will be initially selected when the group is opened. Use  to set the active tab. The tab set as active is displayed in bold text.

When the **Define tabs on specific sides of a dual-display Lister** option is turned on, the tab group can define tabs that open in the left file display separately from those that open in the right. Selecting such a group will automatically put the Lister into dual-display mode if needed. When specific sides is enabled, the  button switches the selected tab from one side to the other.

Folder Tabs can optionally be set to one of several different locked states. A padlock symbol will be displayed to the left of the tab to indicate it is locked. Use the  drop-down on the toolbar to set the lock mode for a tab. The different lock modes are:

- Unlocked:** The tab is not locked, meaning you can freely navigate away from the initial folder.
-  **Locked:** The tab is locked. Attempts to navigate away from the initial folder will cause a new tab to open, leaving the original tab unchanged.
-  **Locked (allow folder changes):** The tab is locked, but you can navigate away from the initial folder. If you switch to another tab and then back again the original folder will be restored.
-  **Locked (reuse unlocked tab):** The tab is locked. Attempts to navigate away from the initial folder will reuse an existing unlocked tab if one exists, otherwise a new tab will be opened.

When the **Define tabs on specific sides** option is on, a tab on one side of the Lister can be linked to a tab on the other side. Linked tabs are indicated, as in the above screenshot, with an icon (). The  drop-down on the toolbar is used to link and unlink tabs. When two tabs are linked,

selecting one in the Lister to make it active automatically activates the linked tab too. You can also link tabs in Navigation Lock mode, using the **Navigation Lock** option in the drop-down. When tabs are linked in Navigation Lock mode, the destination tab will automatically follow the source tab whenever the folder changes. (See the [Tabs](#) page for more information.)

If the **Close existing folder tabs when opening this group** option is enabled, then when the tab group is opened in a file display, all existing tabs will be closed. If this option is off then the tabs defined by the group will be added to any existing tabs.

## Folder Tree

This category contains options that affect the appearance and behavior of the Lister Folder Trees.

- [Appearance](#): Options that affect the appearance of the tree.
- [Contents](#): Options that control which items are displayed in the tree.
- [Options](#): Options that affect the behavior and operation of the tree.
- [Selection Events](#): Lets you override the default behavior for mouse clicks on tree items.

### Folder Tree Appearance

The options at the top of this page affect the appearance of the tree.

- **Show drive labels as**: This lets you choose how disk drive labels and letters are shown under the *Computer* folder. You can choose labels before letters, letters only, or letters before labels.
- **Tree style**: This controls the visual appearance of the folder tree. The **Lines** style is the traditional Windows XP style with + / - buttons next to each folder and dotted lines joining the tree structure. **No Lines** uses the same + / - buttons but does not draw the dotted lines between items. **Visual style** draws the tree with the same style as Explorer, with expansion arrows that fade in and out, and no lines joining items.
- **Full-row selection**: Turn this option on to use full-row selection in the tree, which lets you click anywhere on the row to select an item. With this option off you need to click on an items label to select it.
- **Highlight path to selected folder**: If this option is on the tree will display a "highlight" line that marks the path from the root of the tree to the currently selected folder.
  - **Highlight path to all currently open tabs**: The path to all currently open tabs will be shown instead of just to the active tab. The paths to non-active tabs are faded so as not to obscure the current tab's path.
  - **Only highlight paths when tree is active**: The path highlight will only be shown when the mouse is over the tree or the tree has input focus.
  - **Rounded corners**: The path highlight will be have smooth, rounded corners instead of right-angles.
    - **Shadow around active path**: The line highlighting the path of the active tab will have a shadow drawn around it, making it thicker and easier to see. This is especially useful when displaying lines for all tabs as it ensures that the line for the active tab stands out against all the others. The shadow is drawn using the color configured in the [Display / Colors and Fonts](#) page (the **Tree highlight path to selection: Shadow** color).

- **Use color from tab:** If a tab has a non-standard color (if, for example, it's linked to another tab), the highlight path to that tab will be drawn in the tab's color. Otherwise the highlight path is drawn using the color configured in the [Display / Colors and Fonts](#) page (the **Tree highlight path to selection: Highlight** color).
- **Separate items at the root level:** This option causes root-level items in the Folder Tree (e.g. *Desktop*, *FTP*, *Favorites*, etc) to be separated by blank space, for a less-cluttered visual effect.
- **Show Folder Tree header:** This lets you turn off the tree's "header" (the title bar at the top of the tree).
- **Use configured file display colors for tree items:** If this option is turned on, folders in the tree will be displayed with the same colors they would be shown as in the file display. That is, system folders, compressed folders, labelled folders, etc. will all use the colors defined for them on the [Colors and Fonts](#) and [Labels](#) pages. If this option is turned off all folders will be shown in the same color.

## Folder Tree Contents

The options on this page affect the contents of the tree (which folders are displayed, and in some cases where).

- **Start Folder Tree at:** This drop-down lets you choose the starting folder for the tree. The options are:
  - **Desktop:** This is the normal starting position - at the root of the system namespace, everything else appears below the *Desktop*.
  - **Computer:** The tree will begin at the *Computer* level, without showing the *Desktop* or any of the desktop's other children.
  - **Drives:** This is similar to *Computer*, except displays a cut-down *Computer* item that only lists your disk drives, omitting any other folders under *Computer*.
  - **Current Drive:** This starts the tree at the root of the current drive. For example, if you navigated to *C:\Data\Temp*, the tree would begin at the root of C:.

If you navigate to a location that is normally hidden as a result of this option (e.g. you have selected to start at *Computer*, but you navigate to the *Network* folder which is under *Desktop*), the tree will automatically "re-root" itself in order to make the location you have navigated to visible. When you return to the original location the tree will reset itself to the configured starting location.

- **Creative Cloud Files:** If Adobe Creative Cloud is installed, controls whether the *Creative Cloud Files* folder appears under the *Desktop* folder. (This option will be hidden if Creative Cloud is not detected on the computer.)
- **Empty disk drives:** Displays empty disk drives under the *Computer* folder. If turned off, empty removable drives will be hidden.
- **Favorites:** Displays your configured [Favorite](#) folders in a separate branch of the tree.
- **File Collections:** Displays your [File Collections](#) in the tree.
  - **Display under the Desktop item:** With this option on, the File Collections branch is displayed as a child of the *Desktop*. If this option is off the File Collections branch starts at the root level as a sibling of the *Desktop*.
- **Folder Aliases:** This option displays your configured [Folder Aliases](#) in a separate branch of the tree.

- **All:** All folder aliases, including the internal default aliases, will be shown. With this option turned off only user-defined aliases are shown.
- **FTP Sites:** Displays the contents of your [FTP Address Book](#) in the tree.
  - **Display under the Desktop item:** If this option is on the *FTP* branch is displayed as a child of the *Desktop*. If turned off, the *FTP* branch appears at the root level of the tree.
- **Hidden folders (subject to global filters):** Allows hidden folders to be displayed in the tree, provided they are not filtered out globally. If this option is off, any folders with the **H** attribute set will not be displayed. If this option is on, folders with the **H** attribute will be displayed provided they are not filtered out by the **Global Hide Filter** settings on the [Folder Display](#) Preferences page. The purpose of this option is to allow you to reduce clutter in the tree while still showing hidden folders in the file display.
- **Homegroup:** The *Homegroup* folder on Windows 7 and up.
- **Libraries:** Displays your [Libraries](#) in the tree.
  - **Display under the Desktop item:** Displays the *Libraries* branch as a child of the *Desktop* folder.
  - **Show member folders individually:** Individual member folders are listed in the tree under each library. For example, the **Pictures** library by default contains two members - your personal pictures folder, and the shared public pictures folder. If this option is on, the **Pictures** library will have both these members as separate folders beneath it, with their contents listed under them. If this option is off, the member folders are not shown individually, and the contents of the library are shown merged beneath the library itself.
- **One Drive:** The *OneDrive* folder on Windows 8 and up.
- **Protected operating system folders (subject to global filters):** Allows protected operating system folders to be displayed in the tree, provided they are not filtered out globally. If this option is off, any folders with the **H** and **S** attributes set will not be displayed. If this option is on, folders with the **H** and **S** attributes will be displayed provided they are not filtered out by the **Global Hide Filter** settings on the [Folder Display](#) Preferences page. The purpose of this option is to allow you to reduce clutter in the tree while still showing system folders in the file display.
- **Quick access:** The *Quick access* folder on Windows 10.
- **Recent List:** Displays the [Recent List](#) in a separate branch of the tree.
- **SmartFavorites:** Displays your [Smart Favorites](#) folders in a separate branch of the tree.
- **User Profile folder:** Displays your user profile folder in the tree as a child of the *Desktop* item.
  - **Display at the top of the tree:** Displays the user profile folder as the first item in the tree below the *Desktop* rather than lower down.
- **Virtual (non-filesystem) folders:** If this option is off virtual folders will not be shown in the tree - for example, the *Recycle Bin* would be hidden by turning this option off.
- **ZIP files and other archives:** If this option is on then archive files will be shown in the tree as if they were ordinary folders.

Even if a folder is hidden from the tree using the above items, it will be displayed if you navigate to it or a folder underneath it. For example, if FTP sites are not shown normally in the tree, but you navigate to an FTP site, a temporary entry will be added to the tree as long as you are in the site - it will be removed again when you navigate away.

## Folder Tree Options

These options let you control how the Folder Tree behaves.

- **Automatically expand to current folder:** If this option is on, the tree will expand to track your location as you navigate in the file display. If you turn this option off, the tree won't automatically track your location - you can still navigate with the tree by expanding and selecting folders yourself, but it won't follow you when you navigate by other methods. When this option is on, you can choose how the folders are populated when they expand automatically to show you your current location:
  - **Fully populate contents:** All automatically expanded folders are fully populated - the tree will show the full contents of all folders leading up to your current location.
  - **Populate contents for local devices only:** Only folders on local devices (hard drives, USB drives, etc) will be full populated. Folders on network drives or FTP sites will only display the member folder that forms part of your current location.
  - **No content population:** No automatically expanded folders will be fully populated - you will need to manually populate them by clicking their expansion button to reveal their full contents.

The tree will always show you your current folder (and the path leading to it) so the population options only affect the display of other items in the expanded folders. When you expand an item manually, its contents are always fully populated - so if the tree has partially expanded an item, you can click on its expansion button to fully expand it and reveal the rest of its contents. These options can dramatically improve performance when navigating folders on slow devices like networks - rather than having to read the full contents of all folders leading up to your current location (which can be very slow over a network), the tree will simply display the folders leading to your current path (and since it already knows what those folders are, it doesn't have to read any information from the network to do this).

- **Collapse non-selected branches:** If this option is on, the tree will automatically collapse any expanded branches that aren't needed to show you your current location. This can help keep the tree compact and easier to navigate as you move around the file system.
  - **Unless open in other tabs:** Branches will not be collapsed if they contain a folder that is currently displayed in another folder tab.
- **Expand branches when dragging over the tree:** When you drag files over the tree, this option causes the tree to automatically expand any collapsed folders that you hover over with the mouse. This lets you drag and drop files to sub-folders that aren't necessarily visible when the drag and drop begins. Turn on the **Spring-loaded** option to have folders that automatically expand this way automatically collapse again when the drag is complete.
- **Expand 'My Computer' when tree opens:** This option causes the *Computer* branch of the tree to be automatically expanded when the tree opens.
- **Expand selected branch:** This option causes the tree to automatically expand the branch of the tree that represents your current location. Normally only folders leading up to your current location are expanded (subject to the options above) - with this option on, the current location's branch will also be expanded.
  - **Expand selected branch when changing tabs:** The tree will automatically expand the branch for the current tab when you change which tab is active. Without this option, branches are only expanded when you actually navigate to a new folder in a tab.

- **Expand/collapse all child branches when parent double-clicked:** When you double-click a folder's expand/collapse icon, the first level of its child folders will also be expanded or collapsed.
- **Horizontal scrolling style:** This option lets you control how and when the tree scrolls horizontally. Since Microsoft have trouble making up their minds, Explorer has had different behaviour in this regard with every version of Windows lately, and the options here let you choose from these various behaviors:
  - **Fully automatic scrolling:** This is the style used by Explorer in Vista. There is no horizontal scroll bar shown at all, and horizontal scrolling is fully automatic. The tree will scroll left and right as you move the mouse over it to let you access items that may be scrolled out of the display. This option is not available in Windows XP.
  - **Manual scrolling with automatic adjustment on folder changes:** This is the horizontal scrolling style used by Explorer in Windows XP. The tree displays a horizontal scroll bar (if needed), which lets you scroll manually. The tree will also adjust the horizontal scroll position automatically when you change folder to attempt to display as much of the currently selected folder as possible.
  - **Manual scrolling only:** This option disables all automatic scrolling behavior. A horizontal scroll bar will be shown if needed and you can scroll left and right manually, but the tree will never move by itself.
  - **No scrolling or scrollbar at all:** This is the style used by Explorer in Windows 7. The tree never scrolls horizontally at all - no scroll bar is shown and it doesn't move left or right by itself either. If items are out of view to the right of the tree you need to resize the tree to see them.
  - **Prevent scrolling during drag and drop:** This option prevents the tree from scrolling left or right during a drag and drop operation. It doesn't affect how horizontal scrolling works normally.
- **Open second Folder Tree in dual display mode:** If this option is turned on and you set a Lister to dual-display mode, a second tree for the second file display will be automatically displayed. If this option is turned off you can open the second tree manually if desired.
- **Position selected item in the middle of the tree:** This option causes the tree to automatically scroll vertically to keep the selected item as close to the middle of the display as possible. The scrolling only happens when the tree selection is changed automatically (to follow your navigation), so if you're navigating using the tree itself this option has no effect.
- **Share single tree between dual file displays:** If you have dual file displays but only one tree, this option causes the single tree to be shared by both file displays. Whichever file display is source has "ownership" of the tree, and the tree will reflect the current folder shown in that file display only. When you switch source and destination displays the tree will automatically update to reflect the new location (if any). If this option is turned off, and you have dual displays with only one tree, then the tree is owned by the left/top file display only, and the right display doesn't have one.

## Selection Events

These options let you override the command that runs when a folder in the Folder Tree is selected using the mouse. You can configure commands for both the left and middle mouse buttons (the right mouse button is always used for the context menu), and you can have separate commands for when the **Alt**, **Control** and **Shift** keys are held down.

The command for each event must be a command that would make sense on a button or hotkey. You can use [Opus internal commands](#) or external programs (passing arguments to them using the [standard control codes](#)). If you want to run more than a simple single-line command, you can create a more complicated [User](#) command and then specify the name of the user command for the event.

The default settings for these events are:

- **Left mouse button:**
  - **Normal: Go.** Reads the selected folder into the file display currently associated with that tree.
  - **Alt: Go NEWTAB.** Reads the selected folder into a new tab.
  - **Ctrl: Go OPENINDUAL.** Reads the selected folder into the dual display - the file display **not** currently associated with the tree. If the tree isn't currently in dual display mode it will be put into that state automatically.
  - **Shift:** Nothing defined.
  
- **Middle mouse button:**
  - **Normal: Go NEWTAB.** Reads the selected folder into a new tab.
  - **Alt:** Nothing defined.
  - **Ctrl: Go NEWTAB OPENINDUAL.** Reads the selected folder into a new tab in the other file display.
  - **Shift:** Nothing defined.

So from the above we can see that by default, left-clicking a folder will read the folder as normal, **Alt**-clicking a folder will read it in a new tab, and **Ctrl**-clicking will read it into the dual display. Middle-clicking a folder will read it in a new tab (the same as **Alt**-left-click) and **Ctrl**-middle-click will open it in a new tab in the other display.

Say you wanted it so that **Shift**-clicking a folder would open it in a new Lister automatically. You could do this by setting the **Left mouse button / Shift** event to the command **Go NEW**.

You may also want to change it so that left-clicking a folder that's already open in a tab will switch to that tab rather than reading it into the current tab. The command for this would be **Go TABINDEXISTING**.

## Internet

The Internet category contains options that control how Opus uses the Internet.

- [Email](#): Lets you control how Opus sends email (which can be used to send files via email without needing to use a separate email client)
- [Flickr](#): Configure Opus to upload and download images to your Flickr accounts.

- [Proxy](#): Lets you define proxy settings if needed for Opus to access the internet.
- [Updates](#): Controls the automatic update checker - how often it checks and how it alerts you.

## **Email**

Opus lets you send files via email with the **ZIP and Email Files** command (attached to the **Archive Files** drop-down on the toolbar). This provides a quick way to email someone some files without needing to create a new message in an email program and attach the files to it manually. The settings on this page let you configure how Opus sends the files. The **Send email via** drop-down lets you choose from three different methods; the one you pick will depend on your individual system and network.

### **1. Internal SMTP client**

This is the preferred method; it means Opus uses its own SMTP client to connect to a mail server and send the email directly. To use this method you must have access to an SMTP server that will accept relay messages. For home users, your ISP will usually provide this - corporate users however may not have access to an SMTP server directly. The settings for this option are:

- **Email address**: This is the email address that will be used as the 'from' address for messages sent from within Opus.
- **Name**: This is the (your) name that will be used as the 'from' name for messages sent from within Opus.
- **SMTP Server**: This is the domain name or IP address of the relaying SMTP server to use. You can also specify the **Port** if the default of 25 isn't correct.
- **Use SSL/TLS**: Turn this on if the server requires the use of SSL or TLS encryption. This option is enforced if sending through GMail. You can set **SSL** or **TLS** explicitly, or choose **Automatic** to have Opus determine the correct protocol automatically.
- **Server requires authentication**: If your SMTP server requires authentication to relay messages, turn this option on and enter the **Username** and **Password** supplied by your ISP or network administrator.
- **Maximum simultaneous SMTP threads**: This defines the maximum number of simultaneous outgoing messages - if more messages than this are sent at once the additional messages will be queued.

### **2. mailto: emulation**

This method uses the system registered associations for **mailto:** links to trigger an email message in your default email client. If you don't have an email client that handles mailto: links installed then this method won't work. Outlook is an example of a client that does provide support for mailto: links.

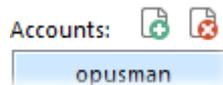
### **3. MAPI client**

This method uses the MAPI system to invoke your default email handler. MAPI is rather out-dated these days and not all email programs support this option.

## Flickr

This page lets you configure the in-built support for [Flickr](#), which is a popular photo sharing service. Opus implements a [synchronization](#) system for Flickr, letting you keep your local photo collection and your Flickr account in sync automatically.

You can configure Opus to work with multiple Flickr accounts. Each account must be authorized through the Flickr service before Opus can access it. To begin the process of authorizing your account, click the **New account** button on the toolbar at the top of this page.



The Accounts list displays the Flickr accounts you have configured through Opus. The toolbar buttons are  (add a new account)  (delete account).

When you add an account, Opus will open your web browser to the Flickr authorization page - from there you must authorize Opus to access your account. When you return to Opus and click the **Complete Authorization** button, Opus will communicate with Flickr to confirm access and download a list of your photosets.

For the selected Flickr account, your User Name and Full Name are displayed to the right. The **Test** button can be used at any time to confirm that Opus still has access to your account.

Also on the right of the page are default upload settings for new photos.

- **Default resizing:** Opus can automatically resize your photos to the selected size when it uploads them.
- **Default privacy:** You can choose the default privacy settings for your photos.

These settings only specify the defaults; whenever you upload photos you can override them.

The section below lets you configure the synchronization system for the selected account.

- **Enable synchronize system:** This option lets you disable synchronization for an account but still keep the entry for it in Preferences. No synchronization will be performed if this option is off.
- **File Collection:** The Flickr synchronization system works through the [File Collections](#) system. A File Collection is created that corresponds to each configured Flickr account, and any photosets or pools are represented via sub-collections. This option lets you choose the parent File Collection that represents the selected account.
- **Default Folder:** When the synchronize causes photos to be downloaded from Flickr, this is the folder they will be stored in. Photos you upload do not have to be kept here (they are managed virtually using File Collections rather than on disk).
- **Photo sets:** This list shows all your photosets as well as pools you are a member of. You can choose to have all photosets and pools synchronized, or turn on the **Only update checked Photo sets** option below to select photosets individually. In this case only checked sets or pools will be synchronized.
- **Identify missing local files on upload:** When uploading photos, this option causes Opus to alert you if photos you have added to the Flickr File Collection can no longer be found on disk. You will be given an

option to locate the files or skip over them. If this option is off, missing photos are silently skipped when uploading.

- **Only update checked Photo sets:** Choose this option in order to select which photosets and pools are synchronized.
- **Update photos that are not in a set:** If this option is off, only photos in sets and pools will be synchronized - so any photos you have added to the account's File Collection (as opposed to its sub-collections) will be ignored.

## **Proxy**

This page lets you configure an HTTP proxy that Opus will use when it needs to access the Internet. This will be used for things like the [update checker](#) and [Flickr](#) photo sharing. This page doesn't control the proxy for FTP access - that is configured through the [FTP Address Book](#).

If **Use a proxy server for HTTP access** is turned on, the options are:

- **Proxy type:** Sets the type of proxy - HTTP, SOCKS/4 and SOCKS/5 proxies are supported.
- **Server:** The IP address or domain name of the proxy server.
- **Port:** The proxy server port.
- **Authentication:** Set this if the proxy requires authentication, and enter the username and password below.

## **Updates**

This page lets you configure the automatic [update checker](#). Opus can be set to look for updates automatically and download them in the background ready for you to install.

- **Check for news and program updates:** This option enables the update checker. You can choose either a weekly or monthly schedule to check for updates.
- **Automatically download new program updates:** If new updates are found, Opus will automatically download them for you. Updates will never be installed automatically, but the files will be downloaded ready for you to install when you're ready.
- **Perform automatic update check silently:** If this option is turned on, Opus will check in the background and only notify you if an update is actually available. If you turn this off, the update checker window will pop-up whenever and automatic update check begins. If this option is on and **Automatically download** is also enabled, you won't be notified until the update has actually been downloaded and is ready to be installed.
- **Don't display the update checker at all:** If this option is turned on as well, the update checker will never open automatically. If an update is found to be available, you will be notified subtly with an icon in the Lister status bar.

If you find the update checker isn't working, you may need to specify a [proxy](#) for Internet access.

## Launching Opus

This page contains options that control how Opus starts up, what it does when it starts up, and how you can open Listers from outside of Opus.

- [Default Lister](#): Contains options that affect the behavior of the [Default Lister](#).
- [Explorer Replacement](#): Lets you enable or disable [Explorer Replacement](#) mode.
- [From the Desktop](#): Lets you open a Lister or run another Opus command when you double-click on the Windows desktop.
- [From the Taskbar icon](#): Lets you open a Lister or run another command when you double-click on the Opus icon in the taskbar (tray).
- [From the Win + E hotkey](#): Lets you change what happens when the Win + E hotkey is pushed.
- [Startup](#): Contains options that control if and how Opus runs automatically on system startup.

## Default Lister Settings

This page contains options that affect the way the [Default Lister](#) settings are used. The Default Lister is the set of settings that define the Lister that is opened whenever you open a new Lister in Opus without specifying a layout. The Default Lister is configured by taking an existing Lister and setting it as the default using the **Set As Default Lister** command in the Settings menu. Please see the [page on this concept](#) for more details.

- **Ignore folder format of Default Lister**: When you set a Lister as the default, the folder format currently used by its file displays is also saved. If you turn this option on, the saved format will be ignored when a new Default Lister is opened. Instead, the format will be set by the usual process of the [Folder Formats](#) system for the folder being displayed in the new Lister.
- **Ignore toolbars of Default Lister**: When you set a Lister as the default, the toolbars it is using are also saved. If you turn this option on, the saved toolbars will be ignored when opening a new Default Lister. Instead, your *Default Toolbar Set* will be used, and can be updated via **Settings / Toolbars / Set As Default Toolbar Set**.
- **Update Default Lister automatically when closing a Lister**: If this option is turned on then whenever you close a Lister, its settings will be saved as the new default. If you open a Lister, modify it in some way and close it again, the Default Lister settings will be updated automatically. You will then get those new default settings back when you open a new Lister.
- **Default Lister positioning**: This controls where new Listers appear on-screen when they are opened using the Default Lister settings. Listers opened via a saved layout are not affected by these options. If your Default Lister is maximized then new Listers will also be maximized and the positioning options mainly just determine which monitor will be used.
  - **In a fixed position relative to the monitor the mouse is on**: The position of the new Lister is based on the Default Lister's position when it was saved. If you have more than one monitor then

the new Lister will open on whichever monitor the mouse is over at the time, with its position relative to that monitor. For example, if you save the Default Lister in the bottom-left corner of your primary monitor, then move your mouse to your secondary monitor and open a new Lister, the new Lister will open in the bottom-left corner of your secondary monitor. The new Lister's position will be adjusted to ensure it does not go off the edge of the monitor it opens on.

- **Always in the same position:** The new Lister always opens in the exact position, and on the same monitor, where the Default Lister was saved. The mouse pointer's position does not affect where the new Lister opens. If the Default Lister's position is partially off-screen or between two monitors, the new Lister will be the same.
- **Always over the mouse pointer:** The new Lister always opens underneath the mouse pointer. For example, if you have two monitors and the mouse is in the top-right corner of your second monitor then the new Lister will open in the same corner, regardless of where the Default Lister was saved. The new Lister's position will be adjusted to ensure it does not go off the edge of the monitor it opens on.

## Explorer Replacement

This page lets you configure the [Explorer Replacement](#) system, which controls if and when Opus will replace Explorer when new folder windows are opened.

- **Don't replace Explorer:** Explorer Replacement mode is disabled. You can still access Opus in many ways, but it will not open in place of Explorer.
- **Replace Explorer for all file system folders:** Opus will replace Explorer for file-system folders, but not for virtual folders. For example, if you had a shortcut to *C:\Files* on your desktop and double-clicked it, Opus would open - but double-clicking the *Recycle Bin* on the desktop would continue to open Explorer.
- **Replace Explorer for all folders:** Opus will replace Explorer for all folders (see note below).
- **Replace Explorer for all but the following folders:** Opus will replace Explorer for all folders except those you add to the list below.



Use the toolbar buttons to add folder to (+) or remove (-) folders from this list. Any folders listed here will continue to open in Explorer when launched in a new window; all others will open in Opus.

- **Open external folders in a new tab:** When Opus intercepts an action that would normally open Explorer, it opens a new Lister instead. If this option is turned on and you already have a Lister open, the folder is opened in a new tab in the existing Lister instead of in a new Lister. The option also affects what happens when archives are opened from outside of Opus, if Opus is set as the system default handler on the [Zip Files](#) page. (If you are using the virtual desktops feature of Windows 10 and above, Opus will normally only look for existing windows on the active virtual desktop. You can change this via the advanced [virtual desktop isolation](#) setting.)
- **Open all Default Lister tabs when opening a new window:** This option controls what Opus does when it intercepts an action which would open a new folder tab but there is no existing window for the tab to open in. If the option is off, Opus will open a window which only displays the folder in question. If the option is on, Opus will open your Default Lister, including any initial folders and folder tabs, and then open an additional tab for the folder in question.

The Explorer Replacement options only affect what happens when some action causes a new folder window to be opened. For example, double-clicking a folder on the desktop, pressing **Windows+E**, or performing an action within a third-party program that would normally cause an Explorer window to open - in these sorts of cases, Explorer Replacement mode will cause an Opus Lister to open instead of an Explorer window.

Opus does not replace the system file open or save dialogs, so these options have no effect on those functions.

In Vista and Windows 7, Opus will not open the *Control Panel* even if **Replace Explorer for all folders** is turned on - the Control Panel (and its various sub-pages) will always open in Explorer.

## ***Launching Opus from the Desktop***

The options here let you launch Opus or run another command by double-clicking on the Windows desktop. This can provide an extremely quick way to access a Lister when you have no windows currently open - simply find an empty area of the desktop (one with no icons on it) and double-click.

- **Disable:** Select this to disable double-click on the desktop.
- **Bring the last active Lister to the front:** If there are one or more Listers already open, a double-click on the desktop will bring to the front the one you used most recently. If no Listers are open a new one is opened using the Default Lister settings. If you are using Windows 10's *virtual desktops* feature, only windows on the currently active desktop will be considered, unless you turn off the advanced [virtual desktop isolation](#) setting.
- **Open the Default Lister:** Opens a new Lister using the settings for the [Default Lister](#).
- **Open a saved Lister layout:** Opens a saved [Lister layout](#). If you turn on the **Close existing Listers** option then any existing Listers will be closed before opening the layout - if this is off, existing Listers will be unaffected.
- **Run a defined User command:** Lets you select a [User command](#) from the drop-down to be run whenever the desktop is double-clicked. This can be used to perform more complex actions including running external programs.

When any of the double-click options are enabled, Opus launches a small helper application called [dopusrt.exe](#) that runs continuously in the background. This is added to the registry to run automatically on startup, and means that double-click on the desktop works even when Opus itself isn't running (**dopusrt.exe** will launch Opus automatically if it's not already running when you double-click).

In order to trap double-clicks on the desktop (which runs in another process to Opus) it's necessary to "inject code" into Explorer - if you have an anti-virus or anti-spyware tool installed you may find it detects this and complains. If you find that double-click on the desktop isn't working check if something is blocking **dopusrt.exe** from hooking into Explorer or from running on startup.

## Launching Opus from the Taskbar Icon

This page lets you configure what happens when you double-click on the icon that Opus adds to the taskbar notification area (also called the "tray" or "systray" icon).



Opus adds the taskbar icon when the option in the [Windows Integration](#) page is turned on. In Windows 7 the icon will be hidden by Windows and moved to the icon overflow area by default, so if you want the icon to always be visible (which makes the double-click options on this page more useful) you will need to tell Windows to show it all the time.

- **Disable:** Select this to disable double-click on the taskbar icon.
- **Bring the last active Lister to the front:** If there are one or more Listers already open, a double-click on the taskbar icon will bring to the front the one you used most recently. If no Listers are open a new one is opened using the Default Lister settings. If you are using Windows 10's *virtual desktops* feature, only windows on the currently active desktop will be considered, unless you turn off the advanced [virtual desktop isolation](#) setting.
- **Open the Default Lister:** Opens a new Lister using the settings for the [Default Lister](#).
- **Open a saved Lister layout:** Opens a saved [Lister layout](#). If you turn on the **Close existing Listers** option then any existing Listers will be closed before opening the layout - if this is off, existing Listers will be unaffected.
- **Run a defined User command:** Lets you select a [User command](#) from the drop-down to be run whenever the taskbar icon is double-clicked. This can be used to perform more complex actions including running external programs.

Users of Windows 7 and greater may prefer to use the [Jump List](#) rather than the tray icon as it offers far more flexibility.

## Launching Opus from the Win + E hotkey

This Preferences page lets you configure what happens when you push the Win + E hotkey (that is, hold down the Windows key and push E).

This page is only shown on Windows 10 and above, and where Explorer Replacement is available (Opus Pro, installed normally, or a portable install with Explorer Replacement enabled in the [portable .ini file](#).)

- On earlier versions of Windows, we do not have as much control over the Win + E hotkey and it will typically cause the *This PC* aka *My Computer* folder to open in the default file manager, which will be Opus if it is set to replace Explorer. If you wish to customize what Win + E does on earlier versions of Windows, try creating a [System-Wide Hotkey](#) which overrides the standard one. (Note that if you set up a hotkey that way on Windows 10, it may override the settings on this Preferences page. Only use one method or the other.)

The Win + E hotkey is what many people are used to using to open a new File Explorer window. When using Opus instead of Explorer, you'll probably want the hotkey to open Opus instead.

The options below let you configure what happens when you push the hotkey:

- **Don't replace Explorer:** The Win + E hotkey will open File Explorer the same as it would have before Directory Opus was installed.
- **Bring the last active Lister to the front:** If there are one or more Listers already open, the Win + E hotkey will activate the one you were using most recently and bring it to the front. If no Listers are open, a new one is opened using the Default Lister settings. If you are using Windows 10's *virtual desktops* feature, only windows on the currently active desktop will be considered, unless you turn off the advanced [virtual desktop isolation](#) setting.
- **Open the Default Lister:** Always opens a new Lister using the settings for the [Default Lister](#).
- **Open a saved Lister layout:** Opens a saved [Lister layout](#). If you turn on the **Close existing Listers** option then any existing Listers will be closed before opening the layout - if this is off, existing Listers will be unaffected.
- **Run a defined User command:** Lets you select a [User command](#) from the drop-down to be run whenever the Win + E hotkey is pushed. This can be used to perform more complex actions including running external programs.

## ***Launching Opus On Startup***

The settings on this page let you set Opus to run automatically when Windows starts up. You can also control what Listers are opened automatically when Opus starts, and whether Opus stays running in the background when no Listers are open.

The **Startup settings** section controls whether Opus runs on start up.

- **Launch Directory Opus automatically on system startup:** When Windows boots, Directory Opus will start running automatically. Having Opus always running means opening a Lister is much quicker, and it

also lets you take advantage of configurable [system-wide hotkeys](#) and [floating toolbars](#). Run-on-startup is implemented by adding a shortcut to the *Startup* program group in the Start menu.

- **Show introductory start window:** If this option is on, Opus will display a splash screen when it starts as long as no Listers are opened automatically. If Listers are opened on startup the splash screen will not be shown. The splash screen provides links that let you easily open a new Lister, access the Preferences system, etc.

Windows 8 and above add a significant delay before launching applications at startup. The **Remove Startup Delay** button lets you remove this delay. Changing this option affects all programs launched via the Start Menu startup folder, not just Opus, and the setting is in the registry rather than the Directory Opus configuration (so you'll need to change it on individual machines if desired). It's a Windows setting, not an Opus one - we just provide a convenient way to change it in Preferences. You can use the **Restore Startup Delay** button to turn it back on again if you've previously removed it.

The **Listers opened automatically when Directory Opus starts** section lets you control what Opus does when it starts. This applies when Opus runs on system startup and also when you run Opus by double-clicking the shortcut icon on the desktop or in the start menu. It doesn't affect what happens if Opus starts because of another action - e.g. double-clicking the desktop uses the settings in the [From the Desktop](#) page, whether Opus is already running or not.

- **Don't open any Listers:** No Listers will be opened. The splash screen will be shown if the Show introductory start window option is turned on.
- **Open the Default Lister:** The Default Lister is opened.
- **Open the Listers that were open when the program was last closed:** When Opus quits it remembers which Listers were opened, and restores them the next time it starts.
  - **Include virtual folders:** Virtual folders (like *Recycle Bin*) are included in the remembered set of Listers. If this is turned off only Listers showing real file-system paths (like *C:\*) are remembered.
- **Open a saved Lister layout:** Opens a saved [Lister layout](#).
- **Run a command:** Lets you enter a command line to be run when Opus starts up. This can be used to perform more complex actions including running external programs.

The **Shutdown Directory Opus when the last Lister closes** option lets you configure Opus to shutdown automatically once you close the last Lister. This makes Opus behave more like a traditional application that you run, use and then exit when finished with. We recommend that you keep Opus running in the background even when you're not using it as it makes it much quicker to open Listers, and means things like [system-wide hotkeys](#) and [floating toolbars](#) can work, but if you don't want to use it like this you can turn this option on.

## Layouts and Styles

This category contains the pages that let you configure the [Lister Layouts](#) and [Styles](#) systems.

- [Layouts](#): Control your saved Lister Layouts.
- [Styles](#): Create and edit Lister Styles.

### Layouts

This page shows a list of your saved [Lister Layouts](#). A Lister Layout is a saved collection of one or more Listers that you can re-open at any time. If you select a layout in this list a tiny preview at the bottom-right of the page gives you an indication of the Listers defined by the layout.



Use the toolbar at the top of the page to edit your layouts. The buttons are (add a new layout), (new folder), (rename layout), (set description for layout), (delete layout), (insert a separator), (move layout up), (move layout down), (sort layouts) and (create shortcut).

- The add button () saves all currently open Listers as a new layout (or over the top of an existing layout if the name matches one already in use).
- You can organise layouts into folders using the new folder button () and by drag & drop.
- The set description button () lets you assign a description to a layout. If you don't assign a description Opus will show a default one based on the settings in the Layout.
- The insert separator () , move up/down (, ) and sort () buttons are used to add separators to, and change the order of items in, the layout list when it is shown in drop-down menus. The layout list can be accessed from the *Settings / Lister Layouts* menu and if the option is on in the [Windows Integration](#) page, by right-clicking the Windows desktop.
- The create shortcut button () creates a shortcut to the selected layout on your desktop.

The options at the bottom of the page can be modified for each layout in the list:

- **Close all existing Listers when loading this layout:** If this option is on, any existing Listers will be closed automatically when this layout is loaded.
- **Hide this layout from layout lists:** This option lets you mark a layout that is hidden when the layout list is displayed. Layouts that are hidden can still be accessed using the **Prefs LAYOUT** command.

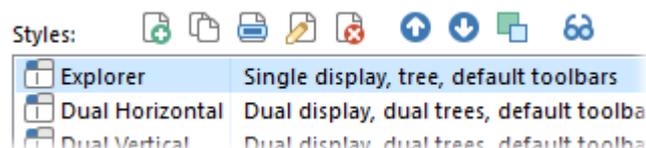
- **Ignore folder formats saved within this layout:** When you save a Lister layout, the folder format currently used by its Listers' file displays is also saved. If you turn this option on, the saved formats will be ignored when a layout is opened. Instead the formats for the Listers that are opened will be set by the usual process of the [Folder Formats](#) system for the folders being displayed.
- **Ignore toolbars saved within this layout:** When you save a Lister layout, the toolbars currently in use are also saved. If you turn this option on, the saved toolbars will be ignored when opening a layout. Instead, your *Default Toolbar Set* will be used, and can be updated via **Settings / Toolbars / Set As Default Toolbar Set**.
- **Open layout relative to the monitor the mouse is currently on:** Normally a saved layout will re-open its Listers in the same location on-screen they were saved in. If you have a multiple monitor system and turn this option on, the Listers will be opened relative to the monitor the mouse is on (so, for instance, a Lister that was saved on monitor #1 would open in the same relative location on monitor #2 if the mouse was on that monitor at the time the layout was opened).

You can open a Lister layout by using the layout list in the *Settings / Lister Layouts* menu or the context menu on the desktop (if the option in [Windows Integration](#) is on). You can also create shortcuts to layouts by dragging the layout from the list on this page and dropping it on the desktop.

There is no way to modify the Listers saved in existing layouts - instead, open the layout, make the changes to the Listers as desired, and then re-save it from this page using the add button.

## Styles

This page lets you create and edit [Lister Styles](#). A style is a pre-defined configuration that can be applied to an existing Lister. For example, you can define a style that opens the viewer pane and the metadata pane, and closes the folder tree all in one operation. You can switch between Lister Styles using the *Lister Configuration* drop-down menu on the toolbar.



The Styles list displays the styles that are currently configured. Use the toolbar buttons above the list to configure them: (add a new style), (duplicate an existing style), (rename a style), (edit a style), (delete a style), (move style up the list), (move style down the list), (sort the list) and (mark style as hidden).

- The add button () creates a new style that you can configure using the options in the bottom half of the page.
- The move up/down (, ) and sort () buttons are used to change the order of items in the list of Styles when it is shown in drop-down menus. You can also display a horizontal list of styles on a toolbar using the **Lister Styles - List** or **Lister Styles - Tabs** commands, and the left-to-right order of these is defined by the

top-to-bottom order of this list. Similarly, the hide (68) button lets you mark a style that is hidden when the style list is displayed. Styles that are hidden can still be accessed using the **Prefs STYLE** command.

Double-click a style, or select it and click the  button, to display the **Edit Lister Style** dialog.

At the top of this dialog, the **Name** field lets you rename the style and the **Description** field lets you assign a description to a style. If you don't assign a description Opus will show a default one based on the settings in the style.

Below these fields are a number of checkboxes and drop-downs that let you define exactly which Lister elements the style affects. If an item's checkbox is turned off, that element in the Lister will be completely unaffected by the style. If the checkbox is on, the setting for that element will be used when the style is applied. The elements that a style can effect are:

- **File Display:** The style can force the Lister into [single or dual-display](#) modes. The drop-down lets you select Single or Dual in either horizontal or vertical layout. You can also choose to set the lister into dual display with the [Navigation Lock](#) automatically activated (represented by the  icons in the drop-down list).
- **Folder Tree:** You can make the style turn the [folder tree](#) off or on. You can also set the style to turn dual folder trees on if the Lister is in dual-display mode.
- **Viewer Pane:** The style can turn the [Viewer Pane](#) off or on in either horizontal or vertical orientations.
- **Metadata Pane:** You can turn the [Metadata Pane](#) either off or on in either horizontal or vertical orientations.
- **Toolbar:** To have the style change which toolbars are displayed, turn this on and select an individual toolbar or a [toolbar set](#).
- **Utility Panel:** The style can turn the [Utility Panel](#) off or on. If turning it on you can choose which mode it opens in - [Find](#), [Synchronize](#) or [Duplicates](#).
- **Status Bar:** The style can turn the Lister's [status bar](#) on or off.
- **Format Lock:** The style can turn the [Format Lock](#) on or off.
- **Filmstrip:** This option puts the Lister into a special mode with the Viewer Pane turned on, and a single scrollable row of thumbnails shown in a single file display.

Below the options for the Lister elements are additional options that let you configure the folder, folder format and tabs that the Style displays. If these options are turned off then the style will only reconfigure the Lister - it won't change what folder the Lister is currently showing - but you can also define a style that reads a new folder, opens a set of tabs, etc.

Note that these options are configured for the **Left File Display** and **Right File Display** separately. If the Lister is not in dual-display mode then only the options on the **Left File Display tab** are used.

- **Tab Group:** Makes the style open a folder tab group. The drop-down displays a list of all tab groups configured on the [Folder Tabs / Tab Groups](#) Preferences page. Selecting a tab group overrides all of the subsequent options.

- **Close existing folder tabs:** This option makes the style close all existing tabs before opening the new tabs defined by the style. If turned off, any tabs defined here will be added to the existing tabs in the file display.
- **View Mode:** The style can change the [view mode](#) in the file display. You can select any of the standard view modes (Details, List, etc) from the drop-down, as well as Auto. Auto is a special mode that means "reset the view mode to what would normally be the default for the folder shown". The [Folder Formats](#) system will be consulted to determine the appropriate view mode for the displayed folder.
- **Format:** The style can define a folder format that is applied either to the current folder, or (if **Folder** , below, is activated) the new folder after it is read.
- **Flat View:** The style can turn [Flat View](#) off, or turn if on in any of its various modes.
- **Folder:** The style can define a new folder to read into the file display when the style is activated.

## Miscellaneous

The Miscellaneous category contains pages that wouldn't fit comfortably in the other categories.

- [Advanced:](#) This is the scariest page in the whole Preferences system. You don't need to look in here. Really.
- [Sounds:](#) Lets you configure sound effects for various events.
- [Windows Integration:](#) Various options that affect how Opus integrates with Windows.

## Advanced Options

This page contains a list of what are considered "advanced" options. They're generally esoteric settings that most users would never need to know about let about want to change.

Each option has a name and a value. The values can be Boolean (true/false), numeric, a string, or a choice from a drop-down list. To edit an option's value, double-click the existing value, or select the option and press the **F2** key. When an option is not set to its default value, it is displayed in bold.

When you select an option a description of it will be shown at the bottom of the page. Items marked with an asterisk are global settings - they affect all users of the computer. You can reset an option to its default value by selecting it and clicking the **Reset** button at the bottom.

<b><u>Category: Behavior</u></b>		
<b>Option</b>	<b>Global?</b>	<b>Description</b>
<b>button_editor_advanced</b>		Always open the <a href="#">Command Editor</a> in <a href="#">Advanced</a> mode, even for simple commands.
<b>compress_collections</b>		Compress <a href="#">File Collections</a> to save disk space. If turned off, File Collections will be saved as uncompressed

		XML files - if you have a lot of files in a collection, this can make the files take up a lot of space.
<b>fayt_firstchar_repeat</b>		If the first letter typed is pressed repeatedly in the <a href="#">Find-As-You-Type</a> field, scroll to next file beginning with that character. For example, typing <i>E-E-E-E</i> would progressively scroll to each file beginning with an E, rather than searching for a file that literally begins with EEEE.
<b>find_extension_func</b>		On Windows XP, Opus registers a <a href="#">Search Handler</a> that lets you access the Opus Find function from the Start Menu. You can use this option to configure the command that Opus runs in response to the search handler. If not specified, the default <a href="#">Find</a> command is used.
<b>find_unique_collections</b>		Don't let simultaneous <a href="#">Find</a> operations use the same collection for results. The Find function defaults to presenting its results in the <i>Find Results File Collection</i> . If you have two or more Find operations running in parallel, this option causes the name of the output File Collection for the second and subsequent tasks to be modified to be unique, so that the results are not intermingled.
<b>flatview_folder_filters</b>		Enable filtering out of sub-folders in <a href="#">Flat View</a> . By default the quick filter (i.e. the filter set by the <a href="#">Filter Bar</a> ) is not applied to folders when the file display is in Flat View. The reason for this is that when a folder is filtered from the list, all its contents are automatically filtered as well, and this may lead to undesired results. The Filter Bar has a checkbox option displayed when the file display is in Flat View mode that lets you turn folder filtering on or off, and you can use this option to control the default state of that checkbox (that is, if you normally do want the quick view filter to apply to folders as well as files in Flat View, turn this option on.)
<b>go_up_always_back</b>		The <a href="#">Go UP</a> command normally re-reads the parent folder, even if you had previously visited it and it is in the history list, and you can add the <b>BACK</b> argument (i.e. <b>Go UP BACK</b> ) to cause Opus to use the history if possible (preserving file selections) when going to the parent folder. If you turn on this option then <b>Go UP</b> will always behave as if it had the <b>BACK</b> argument specified as well.
<b>help_interface</b>		Controls whether the built-in help (accessed from the <b>Help</b> menu or by pressing <b>F1</b> ) is shown in the <i>HTML Help</i> viewer, or in your web browser. By default help is shown in your web browser using a local http server. If you want to change to the <i>HTML Help</i> viewer, select <b>CHM</b> for this option. When using http a

		default port is chosen; if you want to specify a port number to use you can enter it here.
<b>image_locate_services</b>		Lets you configure the image location services used by the <a href="#">Image LOCATE</a> command. Each line consists of the keyword used to refer to the service, an equals sign, and a URL that will be opened in your default web browser. The insert codes <b>%lat%</b> and <b>%lon%</b> are used to pass through the latitude and longitude of the image.
<b>layout_string</b>		This option lets you define the name that is given to <a href="#">Lister Layouts</a> that you drop on your desktop from the <a href="#">Layouts</a> page in Preferences. You must specify a string that contains the code <b>%1</b> - this code is replaced with the name of the layout in the generated shortcut. If nothing is specified for this option, the default is <i>Layout '%1'</i> .
<b>manual_sort_names</b>		Lets you define additional <a href="#">manual sorting</a> orders. Opus can remember more than one manual sorting order and you can switch between your named orders easily. For example, you might have a manual sort order that you use at work and another that you use at home. You must enter each name on a line by itself. If you don't specify any names there will only be one, unnamed, manual sort order available.
<b>open_container_in_tabs</b>		This option is only used in <a href="#">File Collections</a> , when you right-click with more than one item selected, and the items come from multiple <a href="#">virtual filesystems</a> . In that instance only, Opus shows a cut-down context menu with two hard-coded entries at the top of it, one of which is an <i>Open Containing Folder</i> command. This option causes that command to open the items' containing folders in new <a href="#">tabs</a> . Note that in all other cases, the commands displayed at the top of the context menu for collection items are configured through the <a href="#">File Types</a> system.
<b>powermode_singledrag</b>		<a href="#">Power mode</a> file displays have persistent selection by default - when you click on an unselected item, it selects it but does not automatically deselect any other selected items. Normally when you click on an item and drag it out, all other selected files are also dragged as well. When this option is on, if you click on a single file and drag it to the left or right, only that file will be part of the drag - any other selected files will not be.
<b>rename_default_focus</b>		Selects which field in the <a href="#">Rename</a> dialog gets input focus by default - the <i>Old Name</i> or <i>New Name</i> field. Rename scripts can override this setting.
<b>save_button_editor_pos</b>		Save the default position of <a href="#">Command Editor</a> dialogs. If you turn this option on Opus will remember the position of the Command Editor when it is closed, and

		use that saved position the next time one is opened. If turned off, the Command Editor will appear centered over its parent window.
<b>scroll_lock</b>		If the <b>Scroll Lock</b> key is turned on the cursor keys will scroll the file display without changing the focus item.
<b>search_warn_nonindexed</b>		When true, if you use Windows Search on a folder which is not indexed, and the search takes more than a few seconds, Opus will display a banner warning you that the search may be slow because the folder is not indexed. If off, the warning will not be displayed. (This has no effect on Opus's built-in Find Files functionality. Find Files is independent from Windows Search indexing and does not display similar banners.)
<b>setwallpaper_file</b>		The <a href="#">Properties SETWALLPAPER</a> command can be used to set a selected image file as the desktop wallpaper. By default the selected file is copied to your documents folder, but you can override the location by specifying a full path and filename for this option. The filename specified must end in either <b>.bmp</b> or (in Vista and above) <b>.jpg</b> .
<b>show_release_history</b>		When turned on, the Release History page in the help file will open automatically the first time Directory Opus is used after installing a new version. This helps you learn about new features, fixes and other changes in the new version. Turn the option off to prevent the release history page from appearing automatically.
<b>single_click_rename</b>		If you have <a href="#">single-click mode</a> enabled, and the file display is in <a href="#">Details mode</a> with full-row selection enabled in <a href="#">Preferences</a> , it is normally not possible to initiate <a href="#">inline rename</a> using the mouse, because a click anywhere on the line would open the item. With this option enabled you can however initiate inline rename using the mouse - once the entry is selected by hovering, you can click anywhere on the line <i>other</i> than the filename to begin rename.
<b>slow_dblick_rename</b>		Set this to <b>False</b> to disable the standard functionality that lets you enter inline rename mode via a "slow double-click" (clicking an already selected file or folder outside of the double-click time). If slow double-click rename is disabled you can still use the <b>F2</b> key to enter inline rename mode.
<b>status_metadata_trigger</b>		Controls when the status bar, if displaying selected or total media duration, may trigger calculation of media durations (time lengths) if nothing else in the file display has triggered it already.  Since calculating the duration of media files involves opening and parsing data from inside each file

	<p>individually, it can take some time and tie up the disk or network which the files reside on, particularly if there are a lot of files to process.</p> <p>If durations have already been calculated for one of the columns in the file display (e.g. the <b>Duration</b> column), the setting is moot since the status bar will re-use those calculations. The setting only changes whether the status bar on its own can trigger the calculations.</p> <p>If the status bar definition does not include any of the <a href="#">media duration codes</a> -- i.e. <code>{smp3}</code> or <code>{tmp3}</code> -- then the setting is also moot, since the status bar won't trigger calculation of data it is not interested in.</p> <p>The default setting is <b>Always, except Flat View and Collections</b>, which means the status bar can trigger duration calculations everywhere, except when <a href="#">Flat View</a> is enabled or when in <a href="#">File Collections</a>. Flat View and Collections often involve a large number of files, making them a worst case. The other options are <b>Always</b> and <b>Never</b>.</p>
<b>styles_update_previous</b>	<p>If you have made changes to a Lister's appearance (by either opening or closing user interface elements manually, or via a <a href="#">Lister Style</a>), the <a href="#">Prefs STYLE</a> command lets you return the Lister to its initial appearance. If you turn this option on then what is considered to be the "previous" style is updated whenever a new Lister Style is chosen (and so <b>Prefs STYLE=^prev</b> would from then on reset the Lister to that style rather than its initial layout).</p>
<b>tab_click_nofocus</b>	<p>Clicking on a <a href="#">folder tab</a> to bring it to the front normally activates that file display and makes it the source. If this option is on then clicking a tab in the destination file display would leave the source/destination status unchanged.</p>
<b>virtual_desktop_isolation</b>	<p>Windows 10 and above. Changes how Opus interacts with virtual desktops. When set to true, parts of Opus will be more inclined to ignore windows which are not on the current, active virtual desktop.</p> <p>You would usually want this set to true if you use virtual desktops to organize <i>activities</i>, with multiple programs collaborating on each desktop and Opus existing on more than one desktop at once. For example, you may have a Graphics Editing desktop with Opus and Photoshop on it, and a Source Code Editing desktop with Opus and Visual Studio. When</p>

on one desktop, if you double-click a folder, you want it to open in a window on the same desktop, and not switch to another one.

On the other hand, you would usually want to set this to false if you use virtual desktops to organize *programs*, with a File Management desktop that has all your Opus windows on it. In that case, if you double-click a folder on another desktop, you want it to open in one of the existing windows on the desktop dedicated to Opus.

More specifically, setting the option to true makes changes like these:

- Layouts set to close existing listers when they open will only close listers on the active virtual desktop.

The "*Save All Listers*" command only saves windows from the current desktop.

Explorer Replacement (and things like the *Go EXISTINGLISTER* command), if set to reuse existing windows or open new tabs in them, will tend to open a new window if it can't find an existing one on the current desktop. (There are some exceptions, where Windows itself decides which lister to activate for a folder. Commands and settings which refer to the "*Last Active Lister*" also tend to remain unchanged.)

"*Reuse existing viewer window*" only considers viewer windows on the current desktop.

"*Close All Viewers*" from the viewer's system menu only closes viewers on the current desktop.

**Set LISTERCMD=showall, minimizeall, and toggleminimizeall** only act on the current desktop's windows.

There are also some commands which always behave one way or the other, unaffected by this setting. For example, *Set LISTERCMD=tileh* only tiles windows on the current desktop, ignoring windows on other desktops. Some commands have additional switches to tell them if they should be restricted to the current

		desktop or not (e.g. <i>Close ALLLISTERS CURRENTDESKTOP</i> ).
<b>wheel_acceleration</b>		Opus normally applies a small amount of acceleration when scrolling in file displays using the mouse wheel, but you can disable it using this option if desired.
<b>wordbreak_char_names</b>		This lets you specify one or more characters that will be used as "word break" characters in edit fields when editing filenames (e.g. in inline rename). Word break characters affect where the cursor stops when you press the <b>Ctrl+Left</b> or <b>Ctrl+Right</b> keys. A space is always treated as a word break.
<b>wordbreak_char_paths</b>		This lets you configure additional word break characters in edit fields when editing paths (e.g. in the <a href="#">Breadcrumbs path field</a> ). The standard path delimiters (:/) are always treated as a word break.

**Category: Compatibility**

<b>Option</b>	<b>Global?</b>	<b>Description</b>
<b>convert_descriptions</b>		Convert old description files to the <i>description</i> format. Unless you are migrating from an extremely old version of Directory Opus you should never need to set this.
<b>def_func_cd_sourcedir</b>		Use the current source directory as the default current directory when launching programs from buttons that don't specify a current directory. The default for such functions is to set the current directory to the <i>C:\Windows\System32</i> directory for security reasons.
<b>dll_dir_security</b>		<p>This option, enabled by default, prevents Opus from loading DLLs from the current working directory. This reduces the risk of "binary planting" exploits which can trick your computer into running untrusted software when you open things like photos or music from folders in which someone has hidden a malicious DLL.</p> <p>Turning this option off will reduce your security but may be needed if it causes problems with poorly written shell extensions or other third party software which is installed on your computer. (Such software is typically only tested with Windows Explorer, which does not traditionally use the more secure mode which Opus uses by default.)</p>

		<p>This option only affects Opus and, potentially, the software you launch from Opus. It is not a system-wide setting (although there is one if you want to be even more secure; search the web for <code>CWDIllegalInDllSearch</code> for more information).</p> <p>This option cannot be modified when running from a portable USB install.</p>
<b>dragdrop_async</b>		<p>When files are dropped on Opus from another program, Opus will usually handle them in the background so that its window remains responsive. If the program you dragged from supports the Windows API for asynchronous drag &amp; drop then this should never be a problem; however, a large number of programs fail to support the API.</p> <p>When <b>dragdrop_async</b> is on, Opus will attempt to handle <i>all</i> drops in the background, bending the rules if it has to, unless it detects a drop from a program known to have problems with this. If the option is off, Opus will be more strict and will only handle drops in the background for programs that explicitly support it.</p> <p>If you run into problems when dragging files to Opus from certain programs, try turning this option off to see if it improves compatibility. The downside is that you may sometimes have to wait for drops to complete before you can continue using the window you dropped on.</p>
<b>function_default_async</b>		<p>By default, a function that contains a single command will run <a href="#">asynchronously</a>, and functions that contain two or more commands will run <a href="#">synchronously</a>. This option lets you override this behaviour and make all functions (single or multiple commands) run asynchronously by default.</p>
<b>global_explorer_replacement</b>	Global	<p><a href="#">Explorer Replacement</a> mode is normally a per-user setting, but if you turn this option on then the Explorer Replacement mode setting will be global for all users of the machine. This may improve the ability of Opus to intercept some calls to Explorer, but normally you would leave this turned off.</p>
<b>mp3_custom_comments</b>		<p>The MP3 ID3v2 tag used for comments is not rigidly defined, and different third-party tools often use their own form of this tag. You can use this option to make Opus write COMM tags with the specified descriptions, to provide compatibility with these tools when using the <a href="#">Metadata pane</a>.</p>

		For example, MediaMonkey labels its COMM tag as <b>Songs-DB_Custom1</b> , so you would put this string into the <b>mp3_custom_comments</b> value to make Opus MP3 comments compatible with MediaMonkey. If you use multiple tools you can add multiple comment descriptions, one on each line, and Opus will write a separate COMM tag for each.
<b>no_async_icons</b>		File types for which icons should not be retrieved asynchronously. Normally Opus retrieves file icons on a background thread for greater performance, but we have found occasionally that the icons for some file types can not be generated successfully in the background. Multiple file extensions should be separated with a semi-colon. Note that the following extensions are automatically blocked for background icon extraction: <b>.msg;.oft;.xnk</b>
<b>regex_style</b>		Set the style of <a href="#">Regular Expressions</a> used by Opus. The default is <i>TRI ECMAScript</i> , but you can change this to revert to the original regular expression style supported by older versions of Opus. You would really only want to do this if you had a large number of saved <a href="#">Rename</a> presets that for some reason don't work with the new flavor of regular expression.
<b>zip_large_file_support</b>		Enable support for Large Zip files. Large Zip files support individual items, and archives, that are more than 4 GB in size, and more than 65536 individual items in an archive. Large Zip support is enabled by default but if you need to maintain compatibility with legacy archives you can turn this option off. Opus will always read Large Zip files even if this option is turned off - it merely controls the type of archive that Opus creates.

**Category: Cosmetic**

<b>Option</b>	<b>Global?</b>	<b>Description</b>
<b>cd_thumb_coverart_file</b>		Folders containing this file always display as CD albums in <a href="#">thumbnails mode</a> , regardless of other content. The default is <b>coverart.jpg</b> .
<b>collection_icon_default</b>		Custom default icon for individual file collections which don't match any of the other types of collections.

	<p>This setting changes the default icon. You can also change a particular collection's icon via its properties dialog.</p> <p>You can specify the path to a standalone .ICO file (e.g. <b>C:\MyIcon.ico</b>), or specify icons within .EXE and .DLL files using numeric IDs to select a particular icon (e.g. <b>/system/imageres.dll,101</b>).</p>
<b>collection_icon_find</b>	<p>Similar to <b>collection_icon_default</b>, but specific to collections generated for Find Results and Search Queries.</p>
<b>collection_icon_flickr</b>	<p>Similar to <b>collection_icon_default</b>, but specific to collections generated for Flickr Synchronization.</p>
<b>collection_icon_marked</b>	<p>Similar to <b>collection_icon_default</b>, but specific to Marked Pictures collections generated by marking images in the viewer.</p>
<b>collections_icon</b>	<p>Custom icon for the <a href="#">File Collections</a> root folder.</p> <p>This only affects the actual "File Collections" root folder. The other settings above cover individual collections of various types.</p> <p>You can specify the path to a standalone .ICO file (e.g. <b>C:\MyIcon.ico</b>), or specify icons within .EXE and .DLL files using numeric IDs to select a particular icon (e.g. <b>/system/imageres.dll,101</b>).</p>
<b>context_menu_icon_set</b>	<p>Some context menus contain items which Opus generates automatically, and where you do not have an opportunity to change which size or style of icons are used, other than by moving an icon set to the top of the list and affecting <i>all</i> toolbar and menu icons. Since you may wish to use a smaller icon set in context menus than everywhere else, this setting allows you to specify another set to use instead of the normal one. Opus will use this setting when generating context menu items for adding to and extracting from archives. Note that this only applies when the menu is shown inside of Opus; the similar menu in Explorer does not use icon sets in the first place.</p> <p>The name should match the icon set's XML "name" attribute, <i>not</i> its "display_name". For example, if the icon set is defined with <code>&lt;iconset name="flat_small"&gt; ... &lt;display_name&gt;Small Icon Set (Flat)&lt;/display_name&gt;</code> then you should type "flat_small" into the setting, and not "Small Icon Set (Flat)".</p>

<b>gloss_and_gradients</b>	<p>Enable gloss and gradient effects in various user interface elements. Turn this off if you prefer a flatter, more boring appearance.</p> <p>When set to <i>Automatic</i>, a flat and simple look is chosen if on Windows 8 or above, and a shiny look with gradients is chosen on Windows 7 and below.</p>
<b>grid_lines_first_last</b>	<p>Changes how horizontal grid lines are drawn in the file display. This is primarily to help visually separate the file display column header from the files and folders below it, since some Windows versions draw the header without a bottom edge and with the same background color as everything below it.</p> <p>If set to <b>First</b> or <b>Both</b> with solid grid lines, the alternating pattern begins on the first item instead of the second. If set to <b>First</b> or <b>Both</b> with thin grid lines, an extra grid line is drawn before the very first item, just under the column header (but only if visual styles are used for the file display, and line spacing is 0 or 1).</p> <p>Setting this option to <b>Last</b> or <b>Both</b> with thin grid lines adds an additional grid line after the very last item, instead of skipping it. The <b>Last</b> option no effect on solid grid lines.</p> <p>In all cases, the setting has no effect on items under group headers, since group headers have their own lines already.</p>
<b>jobsbar_no_arrow</b>	<p>Turn this on to prevent the animated arrow from appearing when a new item is added to the <a href="#">Jobs Bar</a>.</p>
<b>no_contextmenu_fix</b>	<p>When the "Office 2003" style is selected for <a href="#">toolbars</a>, Opus has to perform some magic to make owner-draw third party context menu extensions look good. If you find that Opus' magic isn't quite up to par, you can disable it with this option.</p>
<b>no_folder_cd_thumbs</b>	<p>Disable automatic rendering of CD album thumbnails. If Opus detects that a folder contains mostly music files, and a thumbnail has been provided for the folder, it will normally render the thumbnail as if it were a CD cover. If you set this option this automatic behaviour is disabled, although you can still force a folder to appear as a CD cover in thumbnails mode by placing in it a file called <i>coverart.jpg</i> (or an alternate filename specified by the <b>cd_thumb_coverart_file</b> option).</p>

<b>progress_smoothing</b>		Smooth progress bar updates on Vista and above. Turn off for lag-free progress bars which jump immediately when updated.
<b>resize_grips</b>		By default Listers do not have visible resize grips, for improved cosmetics, especially in dual-horizontal with inline status bars. Grips can be turned back on using this option if you so desire.
<b>thumb_48x48_icons</b>		<p>Enables the use of 48x48 icons for thumbnails and tiles. This has trade-offs:</p> <ul style="list-style-type: none"> <li>• If the setting is off, icons which have neither 256x256 nor 32x32 versions will look bad.</li> <li>• If the setting is on, icons which have neither 256x256 nor 48x48 versions will look bad.</li> </ul> <p>32x32 icons are far more common than 48x48 icons, so the setting is off by default.</p> <p>(The exact icon sizes vary with system DPI settings. The sizes used here are for the standard DPI.)</p> <p>Each icon can contain various versions of itself in different sizes. Regardless of this setting, Opus always prefers the high-quality 256x256 versions of icons which have them. For icons which do not have 256x256 versions, Opus requests a 32x32 version if the setting is off and a 48x48 version if the setting is on.</p> <p>When an icon contains neither of the requested sizes, one of its other sizes is selected and scaled to fit. This is done by Windows itself and the scaling is not done to a high quality, with resized icons becoming blurry and messy. Windows will not say which sizes of an icon are available without scaling and will not indicate when an icon had to be scaled to fit a requested size, so the quality of the result is determined by the requested size and some icons will look bad either way.</p> <p>(This option affects the icons for things like .EXE files. It does not affect actual .ICO files because Opus is able to generate thumbnails for them without these problems and compromises.)</p>

**Category: Filesystem**

<b>Option</b>	<b>Global?</b>	<b>Description</b>
---------------	----------------	--------------------

<b>copy_buffer_size</b>	<p>Copy buffer size. Units can be KB, MB or GB. If no units are specified, defaults to KB.</p> <p>This specifies the amount of data that Opus will read or write at once when <a href="#">copying files</a>. Although it shouldn't matter, some USB devices or networks appear to be sensitive to the size of the copy buffer. If you find that copies are much slower or less reliable than you would expect, try increasing or decreasing the buffer size.</p> <p>The <a href="#">Copy</a> command's <b>BUFSIZE</b> argument can be used to override this setting on individual buttons, hotkeys, etc.</p> <p>This buffer is in addition to any buffering provided by the filesystem, hardware, and so on; it is not connected to the non-buffered IO mode controlled by the <b>copy_nonbufferio_threshold</b> setting.</p>
<b>copy_nonbufferio_threshold</b>	<p>File size threshold above which file copies to or from local devices will be performed in non-buffered mode, where the filesystem buffers provided by Windows are bypassed. Units can be KB, MB or GB. If no units are specified, defaults to MB.</p> <p>For very large files, copying in non-buffered mode can increase the memory efficiency, copy speed and UI responsiveness. On the other hand, non-buffered mode may slow things down for smaller files or certain devices. In rare cases, non-buffered mode may not work at all (e.g. if you have a device which mis-reports its sector size).</p> <p>Set the value to 0 (zero) to disable non-buffered mode. For compatibility reasons, it is disabled by default. If you wish to enable it, we recommend 1 MB as a good starting value.</p> <p>The <a href="#">Copy</a> command's <b>NONBUFIO</b> argument can be used to override this setting on individual buttons, hotkeys, etc.</p> <p>(Non-buffered mode is not supported on Windows XP, so you'll only see this option on newer versions of Windows.)</p>
<b>dos_automap_unc_paths</b>	<p>Automatically create drive letter mappings for UNC paths in <a href="#">MS-DOS batch commands</a>. This lets</p>

		you run DOS batch scripts on network drives directly from a UNC path.
<b>move_netshare_semantics</b>		Special handling for PGP Netshare when moving folders. PGP Netshare behaves differently when folders are moved compared to files - turn this option on if you are finding that moving folders to Netshare-protected locations is failing.
<b>network_errors</b>	Global	Specify error codes that indicate network authentication errors. Some network filesystems return undocumented error codes in the case of a username/password authentication error - if you are finding that you simply get an error dialog when attempting to connect to a network drive, you should note the error code and add it to this option. When Opus receives that error code it will treat it as an authentication failure and prompt you for credentials to access the resource. You can specify multiple error codes by separating them with semi-colons (e.g. <b>50;1003;1245</b> ).
<b>no_copy_creation_time</b>		Disable copying of file creation time. Normally Opus copies all three timestamps (creation, last written and last accessed) when copying files (unless the <i>Preserve the timestamps of copied files</i> option is turned off on the <a href="#">File Operations / Copy Attributes</a> page in Preferences). If you turn this option on then Opus will not copy the source file's creation time.
<b>no_copy_dir_dates</b>		Disable copying of folder timestamps. Timestamps will only be preserved for files when copying; folders that are created as part of a copy operation will have the current time and date.
<b>remember_net_paths</b>		Opus normally remembers the previous path used in a system File Open or Save dialog (for example, when you use the <a href="#">Preferences Backup or Restore</a> function), but for performance reason it does not remember network paths. Turn this option on if you want Opus to remember network paths as well.

**Category: FTP**

Option	Global?	Description
<b>ftp_copy_buffer_size</b>		FTP copy buffer size (in bytes). This defines the size of the buffer used when transferring files via FTP. You should not normally need to change this.
<b>ftp_dblclk_cache_time</b>		Time (in minutes) to cache double-clicked files from <a href="#">FTP</a> sites. When you double-click a file to open it on an FTP site, Opus downloads the file and saves it to a temporary

		folder. If you double-click the same file again within the specified time, the already-downloaded file will be used to save downloading the file again. This is set to 0 by default, meaning files are not automatically cached for double-click.
<b>ftp_do_not_cache</b>		Normally the <a href="#">FTP</a> system caches the directory listings of remote sites to improve performance. Turn this option on if you want to force remote directories to be refreshed every time you change folder on an FTP site.
<b>ftp_no_case_sensitive</b>		FTP sites are normally case sensitive (e.g. a file called <i>TEST.TXT</i> would be different from a file called <i>test.txt</i> ). Set this option to disable case sensitivity for FTP sites.
<b>ftp_no_pasv_change</b>		If Opus detects a non-routable address has been specified as the result of a passive (PASV) FTP connection, it will try to correct this to the site's routable address. If you turn this option on then Opus will attempt to use the non-routable address as specified, which will only be successful if the FTP server is on the same network as the client machine.
<b>ftp_ssl_verbose</b>		Causes the FTP log to display extended output during SSL connections.

### Category: Image Formats

Option	Global?	Description
<b>amiga_icon_borders</b>		Display borders around Amiga icons in thumbnails mode.
<b>amiga_icon_palette</b>		Palette to use when displaying legacy Amiga icons.
<b>clipboard_image_paste</b>		<p>If you have an image in your clipboard (e.g. via the PrtScn key) and paste into a folder, Opus will save the clipboard image into a file. This setting changes the image format which will be used: JPG, PNG, BMP or GIF.</p> <p>You can also set your Ctrl-V hotkey to run <a href="#">Clipboard PASTE AS=ask</a> to have Opus prompt you for the image format (and filename) when you paste an image into a folder. In that case, this setting determines the format that is selected by default.</p>
<b>clipboard_image_paste_dpi</b>		When turned on, pasting clipboard image data to a file (by pressing <b>Ctrl+V</b> in a Lister) will automatically scale it to compensate for the system DPI.
<b>psd_image_preference</b>		Photoshop PSD files can have up to three embedded images: A small (164x164) thumbnail with very lossy compression and no opacity/alpha data; a full-size, flattened preview image, stored in a lossless

	<p>format, which can include opacity/alpha data; and finally the full, layered image data in Photoshop's internal format. (The layered data is not decoded by Opus, and few things other than Photoshop itself will render it.)</p> <p>By default, Opus uses the small thumbnail image for thumbnails and the full-size preview image for the viewer. The setting allows you to override this so that one image or the other is used for both thumbnails and the viewer.</p> <p>If you want Opus to show PSD thumbnails larger than 164x164, or with opacity/alpha data, select the option to use the preview image for both the viewer and thumbnails.</p> <p>However, some PSD files do not have valid preview images, so there is also the option of never using the preview images and always using the small thumbnails, even in the viewer. Whether a PSD contains a flat preview image depends on how it was saved from Photoshop. The "maximize compatibility" or similar option should be turned on to include a preview image. In Photoshop CC 2015.5.0, the option is under Preferences / File Handling. In some cases, where there is no 'real' preview image there will be a black and white placeholder with text, written by Photoshop, telling you there is no preview image. Unfortunately, some versions of Photoshop write a corrupt placeholder image in some cases. (The data is well-formed, but the image it represents is not. In either case, it is difficult for Opus to detect if a 'real' preview image is there, or just a placeholder, as both are simply different pixel data encoded into exactly the same part of the file as a real preview image.) If your workflow cannot be changed and results in PSD files with missing or corrupt preview images, you can tell Opus to always use the small thumbnails, even in the viewer, so you can at least get a rough idea of what each file looks like.</p>
<b>tiff_assume_alpha</b>	<p>Assume the 4th channel in TIFF images is (non-multiplied) alpha in files which do not specify its meaning. With this option turned off a 32-bit TIFF image must specify that it contains a valid alpha channel for Opus to treat it as so.</p>
<b>tiff_max_doc_metadata</b>	<p>Maximum size (in megabytes) of TIFF images that Opus will attempt to extract document metadata</p>

		from. This prevents problems with very large TIFFs, for example those saved by Photoshop.
<b>use_color_management</b>		Use color management when loading images. If enabled Opus will check to see if an image file contains a color profile, and if so will use the ICC file you specify (or the default sRGB profile) it to render the image more accurately. Currently this is only used for JPEG and PNG images.
<b>viewer_disable_internal</b>		Allows you to disable built-in image formats in the viewer and preview pane. Does not affect viewer plugins (which can already be disabled directly); in fact, the purpose is to allow you to override internal viewers with plugins and third-party components in situations where they cannot do so themselves (generally, Preview Handlers and ActiveX controls which don't know anything about Opus). For example, you may wish to divert the TIFF viewer to a third-party ActiveX control which handles multi-page TIFFs. The setting is a string listing the image types you wish to disable. Setting it to <b>tiff</b> would disable the internal TIFF viewer. Setting it to <b>JPG,PNG</b> would disable the internal JPG and PNG viewers.

**Category: Information Display**

<b>Option</b>	<b>Global?</b>	<b>Description</b>
<b>custom_date_format</b>		Opus normally uses the date format defined by the system (or the current locale) when displaying dates (e.g. in date columns in the Lister). This setting allows you to override it and specify your own custom format.  See the <a href="#">Codes for date and time</a> page for information on date and time formats.
<b>custom_date_format_long</b>		Similar to <b>custom_date_format</b> but used in places where a long (more verbose) date format is used. The long date format is not used in many places, so <b>custom_date_format</b> is more likely what you want if you are not sure.
<b>custom_time_format</b>		Opus normally uses the time format defined by the system (or the current locale) when displaying times (e.g. in date/time columns in the Lister). This setting allows you to override it and specify your own custom format.

	See the <a href="#">Codes for date and time</a> page for information on date and time formats.
<b>desc_show_info</b>	<p>For each file, the <b>Description</b> column normally shows:</p> <ul style="list-style-type: none"> <li>• A metadata summary. (e.g. Picture type and dimensions, Audio codec and bitrate.)</li> <li>• The user-defined description, if any. (This may be set via the <b>Properties, Description</b> command on the default toolbars. Some file types also allow user-defined descriptions/comments within their tags.)</li> <li>• The target path, if the file is a shortcut or link.</li> </ul> <p>This option lets you remove the user-defined descriptions and/or targets from the Description column. For example, if you have added the separate <b>User description</b> and <b>Target</b> columns then you may want only the metadata summary to be left in the <b>Description</b> column.</p>
<b>display_folder_extensions</b>	If this option is enabled folders are treated as having extensions (suffixes) for the purpose of display only (i.e. in the <i>Name</i> and <i>Extension</i> columns).
<b>file_size_units</b>	Specifies the units to use when displaying file sizes and disk space. The traditional form is to use binary units (2 <sup>10</sup> -based) with "decimal" prefixes (1 KB = 1024 bytes). You can also choose to use decimal (10 <sup>3</sup> -based) units (1 KB = 1000 bytes) or binary units (1 KiB = 1024 bytes). See <a href="#">this Wikipedia page</a> for more information.
<b>graphs_separate_files_dirs</b>	Bar graphs that appear in the file display (relative size, relative age) are calculated separately for files and folders. If you turn this option off, they are calculated for all items.
<b>group_column_maxwidth</b>	Specifies the maximum width of the <i>Group</i> column. When file display columns are set to auto-size, this option prevents the <i>Group</i> column from growing larger than the specified number of pixels.
<b>image_res_units</b>	Units to use for columns relating to image resolution ( <i>Resolution (X)</i> , <i>Resolution (Y)</i> ). If not set, each image file itself specifies the units, and so you may have a mix of inches and centimetres in any one directory - setting this option overrides the images and always displays consistent units.
<b>image_size_units</b>	Units to use for columns relating to image size ( <i>Physical Width</i> , <i>Physical Height</i> , <i>Physical Size</i> ). If not set, each image file itself specifies the units,

		and so you may have a mix of inches and centimetres in any one directory - setting this option overrides the images and always displays consistent units.
<b>multipart_extensions</b>		A "multipart file extension" is a file extension that consists of more than one part separated by dots. For example, in the Unix world <b>.tar.gz</b> files are quite common. In many cases (renaming, sorting, etc.) it makes sense to treat this whole string as the file extension, rather than simply <b>.gz</b> as is standard on Windows. Opus does this automatically for several archive formats, and the <b>multipart_extensions</b> option lets you add your own custom extension as well. Enter one extension per line.
<b>name_group_high_pri_chars</b>		When <a href="#">grouping</a> the file list by filename, files beginning with these characters will be put in a "high-priority" group at the top of the list. Normally files beginning with a non-alphabetical character or number will be placed in the <i>Unspecified</i> group, but you could use this option to, for example, place files beginning with <b>!</b> or <b>#</b> in a separate group.
<b>win7_show_sharing_overlays</b>		Windows 7 removed the icon overlay (📁) for shared folders, but many people like this feature and so Opus substitutes its own. If you would prefer not to see these overlays on Windows 7, turn this option off. Prior to Windows 7, the sharing overlay comes from Windows itself and this option has no effect. This option will also have no effect if overlays are turned off completely via the <i>Show shortcut arrows and other icon overlays</i> option on the <a href="#">Folder Display Preferences</a> page.

**Category: Limits**

Option	Global?	Description
<b>context_menu_max_files</b>		Because generating context menus for large numbers of files can take quite a while, Opus displays a confirmation message when you right-click with more than a defined number of files selected. The limit defaults to 1000 but you can change this or set it to <b>0</b> for no confirmation at all.

<b>max_folder_thumb_time</b>		Specifies the maximum time (in milliseconds) that Opus will spend when generating the thumbnail for a folder.
<b>max_md5_file_size</b>		<p>Specifies the maximum file size (in kilobytes) that Opus will calculate the MD5 or SHA-1 hash for when the <b>MD5 Checksum</b> or <b>SHA-1 Checksum</b> columns are displayed in a Lister. If a file is larger than this size its checksum can still be generated manually with the <a href="#">GetSizes MD5</a> or <a href="#">GetSizes SHA</a> commands.</p> <p><b>Special values:</b></p> <ul style="list-style-type: none"> <li>• <b>0</b> - Ignore file sizes and always automatically calculate file hashes.</li> <li>• <b>1</b> - Ignore file sizes and never automatically calculate file hashes.</li> </ul>
<b>max_thumbnail_mem_size</b>		The maximum memory (in megabytes) per file display for in-memory thumbnails. Opus will discard out-of-view thumbnails to keep the memory usage for each file display's thumbnails below the specified size. If set to 0 there is no limit.
<b>max_thumbnail_size</b>		The maximum pixel size (width and height) that Opus will generate thumbnails at. Larger thumbnails require more memory. This controls the maximum size thumbnail that be selected on the <a href="#">Thumbnails</a> page in Preferences, and also via the <a href="#">Thumbnails Size</a> slider.
<b>zip_dbclk_cache_time</b>		Time (in minutes) to cache double-clicked files from within ZIP files. When you double-click a file to open it from a ZIP file, Opus extracts the file (and possibly others, subject to the Auto-extract options on the <a href="#">Archive Options</a> Preferences page), and saves it to a temporary folder. If you double-click the same file again within the specified time, the already-extracted file will be used to save extracting the file again. This is set to 0 by default, meaning files are not automatically cached for double-click.

### Category: Troubleshooting

Option	Global ?	Description
<b>allow_context_menus</b>	Global	List of context menu extensions that Opus will ignore errors from. Normally Opus blocks context menu extensions that cause an exception - you can override this by adding the context menu's GUID to this list. The GUID

		for each context menu extension must be listed on a separate line. You can use the <b>context_menu_debug</b> option to discover a context menu's GUID.
<b>clipboard_change_delay</b>		Delay (in milliseconds) before processing clipboard change events. You might want to increase this if certain software (e.g. Microsoft Office) has problems modifying the clipboard while Opus is running.
<b>collection_change_delay</b>		Delay (in milliseconds) before processing external delete/rename operations in <a href="#">File Collections</a> . You might want to increase this if you find that files are disappearing from your file collections when you edit them in other programs (e.g. Word).
<b>context_menu_debug</b>		Display debug output for context menu extensions. See the <a href="#">FAQ</a> on debugging context menu problems for more information. Users of Directory Opus Light can set the registry value <b>HKEY_CURRENT_USER\SOFTWARE\GPSSoftware\Directory Opus\ContextMenuDebug</b> (DWORD) = 1 as an alternative to this option.
<b>ignore_context_menus</b>		List of context menu extensions that Opus will block. If Opus detects that a context menu extension causes an exception it will add it to this list automatically, but you can add your own entries here to block menus that Opus isn't able to trap (or that you just don't like). The GUID for each context menu extension must be listed on a separate line. You can use the <b>context_menu_debug</b> option to discover a context menu's GUID.
<b>mtp_enable</b>		This option can be used to disable the internal support for MTP (portable) devices. When turned off, devices will be accessed by a hosted Windows Explorer view.
<b>no_external_change_notify</b>		Don't monitor for external file changes. This lets you disable the detection of file changes that occur outside of Opus - only file operations that Opus itself performs will be noticed and reflected in the Lister.
<b>notify_debug</b>		Display debug output for file notification. If you are having problems with Opus not noticing file changes that happen outside of Opus, tech support may ask you to turn this on to gather debugging information. See the <a href="#">FAQ</a> for more information.
<b>notify_max_time</b>	Global	The maximum amount of time, in milliseconds, each file display will spend processing change notifications from the filesystem before considering other inputs and events.  Defaults to 50 milliseconds. In rare situations, you may need to raise this from its default value if events are being generated faster than they are consumed. You can also specify 0 (zero) to process change events indefinitely, although you would probably only want to do so as a test, not as a permanent setting. If this is set to zero, or set too

		<p>high, file displays could become less responsive to user input when a lot of filesystem events are being generated.</p> <p>This is a global setting. If you change it for one user on a machine then it will affect all other users, as is most likely required. On a multi-user system, if the setting is changed by one user, the others will not see the change until they restart Opus.</p>
<b>notify_min_items</b>	Global	The minimum number of events to process before checking the <b>notify_max_time</b> time limit. See the description of <b>notify_max_time</b> , above, for situations where you may wish to raise this, and how the setting behaves for multiple users.
<b>script_output_level</b>		This lets you adjust the type of output that is shown in the <i>Script Output</i> log ( <a href="#">Utility Panel</a> / <i>Other Logs</i> ).
<b>shellchange_debug</b>		Display debug output for shell file notification. If you are having problems with Opus not noticing file or folder changes that happen outside of Opus, tech support may ask you to turn this on to gather debugging information. See the <a href="#">FAQ</a> for more information.
<b>sync_debug</b>		This should normally be left off, but you may be asked to turn it on to debug decisions made by the <a href="#">Synchronize tool</a> . If you use the Synchronize tool while this option is on, a file with debugging information will be created on your desktop.

## Sounds

The Sounds page lets you assign sound files to be played whenever certain events occur in Opus. Sound events can be enabled or disabled globally with the **Enable Sound Events** checkbox at the top of the page. When enabled globally, individual sound events can also be enabled or disabled with their own checkboxes.

To assign a sound to an event, select the event in question and then use the **Browse** button at the bottom of the page to choose the **.wav** file you want to use. You can preview the sound using the play / stop controls on the right.

The events you can assign sounds to are:

- **Close a Lister:** Played whenever a Lister is closed (except on shutdown).
- **Close Folder Tab:** Played whenever a folder tab is closed.
- **Dock a Floating Toolbar to the screen edge:** If you drag a [floating toolbar](#) to the edge of the screen to dock it, this sound will play.
- **Dock a Lister with another one:** When Lister docking is enabled in the [File Display Border](#) Preferences page, this sound will play whenever you dock two Listers together.
- **Find-As-You-Type No Match:** When typing into the [FAYT](#) field in find mode, this sound plays if the string you type can't be matched to a file.
- **FTP copy failure:** Played if a copy to or from an FTP site fails.
- **FTP copy success:** Played when a copy to or from an FTP site succeeds.
- **FTP error:** Played if an FTP error occurs.
- **FTP login failure:** Played if an FTP login attempt fails.
- **FTP login success:** Played when an FTP login succeeds.
- **FTP lose connection:** Played if an existing FTP connection times out or is otherwise lost.
- **FTP timeout:** Played whenever an FTP operation times out.
- **Menu Command:** Played whenever a command is chosen from a drop-down toolbar menu.
- **Menu Pop-up:** Played whenever a pop-up menu opens.
- **Navigation Click:** Played whenever you click on a hyperlink-style control in a dialog.
- **Open a new Lister:** Played whenever a new Lister is opened (except on startup).
- **Open New Folder Tab:** Played whenever a new folder tab is opened.
- **Opus Shutdown:** Played when Opus shuts down.
- **Opus Startup:** Played when Opus starts up.
- **Read/change folder:** Played when you navigate to a new folder in a file display.
- **Switch Folder Tab:** Played when you change from one folder tab to another.
- **Undock a Floating Toolbar from the screen edge:** If you drag a previously-docked floating toolbar away from the edge of the screen to undock it, this sound will play.
- **Undock a Lister:** When Lister docking is enabled in the [File Display Border](#) Preferences page, this sound will play whenever you undock (split) a dual display Lister.

## Windows Integration

The options on this page affect how Opus integrates with certain aspects of Windows or Explorer.

- **Add File Collections icon to the Desktop:** If this option is enabled Opus will place an icon on your desktop that represents the main [File Collections](#) folder; double-clicking it will open a Lister showing your File Collections.
- **Add File Collections list to the Send To menu:** This option causes Opus to add links for your [File Collections](#) to the *Send To* menu (the menu that is displayed when you right-click a file and select *Send to*

from its context menu). In Windows XP the File Collections will be displayed in a sub-menu and all sub-collections are also displayed, but technical changes in Vista and Windows 7 means that File Collections can only be displayed at the top-level of the *Send to* menu.

- **Add layout and other items to Desktop context menu:** If this option is on then Opus will add several commands to your desktop context menu (the menu that is displayed if you right-click on an empty area of the desktop), including a list of all your configured [Lister Layouts](#). This can provide a very quick way of opening a new Lister or a saved layout if you don't currently have any Listers open. You can configure the order your layouts are displayed in, break up the list with separators, and optionally hide some from the display using the [Lister Layouts](#) page.
  - **Display layouts in a sub-menu:** If the above option is on then this option causes the list of layouts to be displayed in a sub-menu rather than on the desktop context menu itself.
- **Add icon to the Taskbar Status Area:** This option causes Opus to add an icon () to the taskbar notification area (also called the "tray", "system tray" or "systray"). This icon lets you access Opus quickly by double-clicking it (the behavior of which can be modified on the [Launching Opus from the Taskbar Icon](#) page) or by right-clicking the icon to display its context menu (which can be configured from [Customize / Context Menus](#)).

In Windows 7 and above the icon will be hidden by Windows and moved to the icon overflow area by default, so if you want the icon to always be visible (which makes it much more useful) you will need to tell Windows to show it all the time. Users of Windows 7 and greater may prefer to use the [Jump List](#) rather than the tray icon as it offers far more flexibility.

- **Add 'Open in Directory Opus' item to folder context menus:** If this item is enabled, Opus will add an *Open in Directory Opus* command to the context menu for folders, letting you open them in an Opus Lister via the context menu. This option is only useful when [Explorer Replacement](#) mode is turned off - when Explorer Replacement is on, this command is added anyway and can't be disabled.
- **Add Preferences and Customize icons to the system Control Panel:** If you enable this option Opus will add icons to the Windows Control Panel that let you access the [Preferences](#) and [Customize](#) dialogs even if Opus isn't running.
- **Hide Windows items on file context menus (shift overrides):** If this option is on then any context menu items that come from the system (i.e. most of them) will be hidden by default when you right-click on a file or folder in Opus. The only items that will be displayed are those defined through Opus itself (e.g. *Cut*, *Copy*, *Paste*, *Delete*, *Rename*, *Properties*). This option is useful if you want to tidy up your context menus (as most context menus are generally quite messy once you have a lot of third-party software installed). Using Opus it's possible to hide all Windows items by default and then selectively add them back wherever you want (including on a sub-menu). Please see the [tip on the Resource Centre](#) for more information about doing that. If this option is on, you can force the display of the full context menu by holding the Shift key down when right-clicking the item.
- **Make Directory Opus the default handler for FTP sites:** Turning on this option will make Opus the default handler for FTP sites; when you click on an FTP link in a web browser, or a shortcut to an FTP site, it should open in an Opus Lister automatically.

## Toolbars

This category contains options that let you control the appearance and behavior of toolbars and menus.

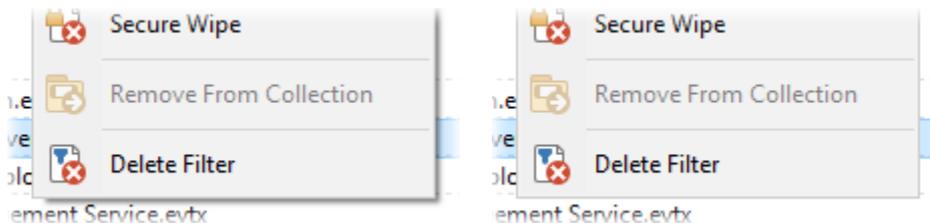
- [Appearance](#): Options that affect the appearance of toolbars.
- [Icons](#): Lets you change the icons used by toolbar buttons.
- [Options](#): Options that control aspects of toolbar behavior.

- [Scripts](#): Lets you manage script add-ins which extend or modify Opus's functionality.
- [Toolbar Sets](#): Lets you manage Toolbar Sets for switching between multiple toolbars.

## Toolbar Appearance

This page contains options that affect some aspects of the appearance of toolbars and menus.

- **Display drop shadow under menus:** This options lets you control whether drop-down (or pop-up) menus have a shadow underneath them or not.



- **Display toolbar top and bottom borders:** This option controls whether toolbars have a top and bottom border or not.



- **Scale Toolbar button images:** This option lets you control whether images you use for toolbar buttons will be scaled to the "standard" size if they are larger than the size set for the toolbar. The standard Opus toolbar image sizes are 22x22 for small images and 32x32 for large images.
- **Use Office 2003-style for toolbars:** If this option is enabled, Opus will draw toolbars and menus in a style inspired by Office 2003. If turned off, toolbars will be drawn in the "normal style" for your OS version.



If Office 2003-style toolbars are enabled, you can configure the colors used when they are "hot" or "active":

- **Use custom button highlight colors:** If this is enabled you can completely configure the colors for all elements and states of the toolbar. If disabled, most colors are generated automatically from the options below.
- **Highlight base color:** This specifies the base highlight color - the actual highlight is rendered using a gradient derived from this color.
- **Menu gradients:** This specifies the gradient colors used to fill the left-hand edge ("gutter") of drop-down menus.
- **Horizontal button spacing:** You can use this option to increase the horizontal spacing between toolbar buttons.
- **Vertical button spacing:** You can use this option to increase the vertical spacing between toolbar buttons (it also affects the padding at the top and bottom of toolbars).

You can configure the background color or images of toolbars from the [Colors and Fonts](#) and [Images](#) pages.

## Toolbar Icons

This page displays a list of the icon sets you have installed. Icon Sets are collections of icons that can be used for toolbars and menus. Opus comes with one set built-in and you can add as many additional sets as you like.

You can use the controls on this page to change the order icon sets are used in. Toolbar buttons and menus that use icons usually refer to the icon by name (for example, the **Copy Files** command would use the icon called "copy"), and icon sets specify the name of all the icons they contain. When Opus needs to display an icon it starts at the first icon set in the list and looks for an icon with the desired name. If that set doesn't contain the icon it moves on to the next set, and so on. So it's generally possible to change the icons in use by your toolbars simply by changing the order that Opus looks through the icon sets for the desired icon.



The toolbar buttons at the top of the page let you manipulate the icon sets:  (move set up in the order of precedence),  (move set down),  (import a set from disk),  (delete a set) and  (refresh the list of sets).

The higher up the list a set is, the sooner it is checked for a given icon. In the above screen shot, the *Default Icon Set* for Opus 10 is at the top, and so the default toolbars will be using this - but if you selected the *Directory Opus 9 Icon Set* and moved it to the top, then the default toolbars would switch back to using Opus 9 icons.

The bottom part of this page displays a preview of all the icons contained in the currently selected icon set.

You can download additional icon sets from the [Resource Centre](#). If you want to create your own icon sets, the **Save Icon Set Template** command in the **File** menu can get you started. It produces a template of the standard icon names - you will need to provide your own images however! See the [Icon Sets](#) page in the Reference section for more information on creating your own sets.

## Toolbar Options

This page contains options that affect the behavior of toolbars and menus.

- **Allow floating Toolbars to be docked with the edge of the screen:** Floating toolbars (those that aren't embedded in a Lister) can optionally be docked with an edge of the screen to turn them into "app-bars", kind of like the task bar. If this option is on you can dock toolbars by dragging them near one of the edges of the screen. Holding the **Shift** key down when dragging will override the docking process, which lets you position a toolbar close to the edge of the screen without docking it.
- **Alt-Click to edit Toolbar buttons:** If this option is on, you can enter [Customize](#) mode and bring up the [Function Editor](#) for a button in one single action, by holding the **Alt** key down and clicking the button with the mouse. Having this option on means you can't use the **Alt** key with any buttons that use qualifier keys to change their behavior. For example, the items in the **Go** menu use the **Go USEQUALKEYS** command to implement four different behaviors depending on the state of the qualifier keys (read folder normally without any keys down, open a new Lister with **Shift** down, open in dual-display mode with **Control** down, and open a new tab with **Alt** down). If you turn this option on then **Alt**-click will edit the button rather than run the function attached to it.
  - **Minimize Customize dialog:** If you have the above option turned on then this setting will minimize the Customize dialog automatically, so that only the Function Editor for the button you clicked on is shown. If you click **OK** or **Cancel** in the function editor, and you have not either edited another button or opened the minimize Customize dialog, the Customize dialog will automatically close when the function editor closes. This lets you make quick changes to existing buttons without having the Customize dialog appear at all.
- **Animate menus:** If this option is turned on, drop-down and pop-up menus are animated as they open. If turned off, they will appear immediately.
  - **Use system setting:** When Animate menus is turned on, this option causes Opus to use the current Windows setting for drop-down menu animation. When turned off, Opus always uses a fade effect for drop-down menus.
- **Display popup help text:** This causes tooltips to be displayed when the mouse hovers over toolbar buttons or items in drop-down menus. The tooltips display the configured description for the button in question - if the button has [multiple mouse button functions](#) defined, these will be listed separately within the tooltip.
  - **Show shortcut keys in help text:** When tooltips are displayed for toolbar buttons, this option causes the button's hotkey (if any) to also be displayed in the tooltip text.

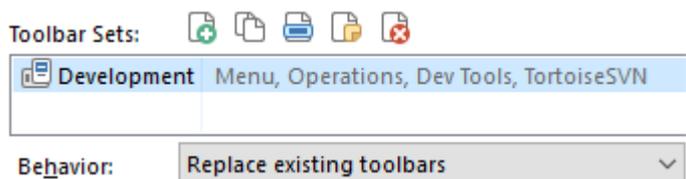


- **Lock Toolbars:** If this option is turned on, toolbars in the Lister will be locked to prevent them from being dragged (repositioned) or accidentally floated. You can also lock or unlock them from the toolbar context menu (right-click on an empty area of any toolbar).
- **Save state of floating Toolbars automatically on exit:** Normally you have to use the **Save Floating Toolbars** command in the [Customize / Toolbars](#) page (on the File menu) to save the position and state of toolbars you have floated. If you don't save the state then the next time Opus runs, toolbars you floated, or floating toolbars you repositioned in the previous session will not be remembered. If you turn on the **Save state** option then when Opus shuts down it will automatically remember which toolbars are floating and where you have positioned them.
- **Simulate middle mouse click with control + left click:** Opus lets you assign [three completely separate functions](#) to toolbar buttons that can be accessed with the left, right and middle mouse buttons. If your mouse doesn't have a middle button, you can use this option to still take advantage of this functionality - holding the **Control** key down when you left-click a toolbar button will be treated as a middle-click instead.

- **Slide auto-hide floating Toolbars when hiding and revealing:** When a floating toolbar is docked to the edge of the screen it can be set to auto-hide - it will move itself off-screen so as to take up as little screen real estate as possible, and reappear automatically when you move the mouse over it. If this option is on, the toolbar will use a slide effect when it moves in and out of view, instead of appearing or disappearing immediately.
  - **Reveal delay:** Specifies a delay before the toolbar starts to appear. For example, if this is set to 250 milliseconds (250 ms) then the toolbar will only start to appear if the mouse is over it for a quarter of a second, helping to avoid triggering it accidentally when moving the mouse nearby.
  - **Hide delay:** Specifies a delay before the toolbar starts to hide. For example, if this is set to 500 milliseconds (500 ms) then the toolbar will not start to hide until half a second after the mouse has left it.
  - **Reveal time:** Specifies how quickly the toolbar slides when appearing. For example, if this is set to 100 milliseconds (100 ms) then the toolbar will slide in quickly, taking only one tenth of a second to do so.
  - **Hide time:** Specifies how quickly the toolbar slides when hiding. For example, if this is set to 2000 milliseconds (2000 ms) then the toolbar will slide away very slowly, taking two seconds to do so.

## Toolbar Sets

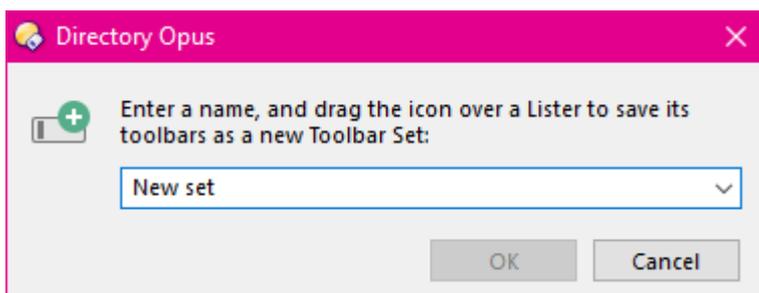
This page displays a list of any toolbar sets you have created.



Each set displays the toolbars it contains. You can rename sets and assign custom descriptions using the toolbar icons. For each set you can also use the **Behavior** drop-down to assign a default behavior - when the set is loaded in a Lister the default behavior will be used unless overridden on the command line. The behaviors you can choose from are:

- **Add toolbars to existing toolbars** - any toolbars in the set that aren't already turned on are turned on when the set is loaded. Existing toolbars are unaffected.
- **Replace existing toolbars** - all currently open toolbars are closed and replaced with the toolbars in the set.
- **Toggle existing toolbars** - if all the toolbars in the set are turned on, they are removed and replaced with the toolbars from the default set. Otherwise, any toolbars in the set that aren't currently on are turned on. This option lets you quickly toggle between the default set and another toolbar set.

Toolbar sets are created using the toolbars in an existing Lister as a template. To add a new toolbar set, first pick a Lister and open or close the appropriate toolbars. Then click the **New** button (📄+). The following dialog will open:

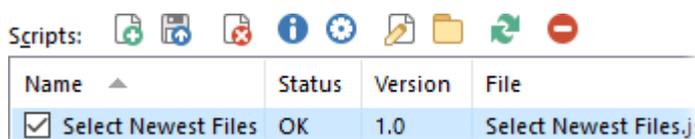


To create the set, enter a name for it, and then click on the  icon and drag it out of the dialog and over the Lister whose current toolbars you want to save into the set. You can also create a set from the Lister itself by right-clicking a toolbar and choosing the *Toolbar Sets / Save Toolbar Set* command from the context menu.

You can duplicate an existing set using the **Duplicate** button () . Use the **Rename** button () to rename a set, the **Description** button () to assign your own description, and the **Delete** button () to delete a set.

## Scripts

The **Scripts** page displays a list of all your installed [script add-ins](#).



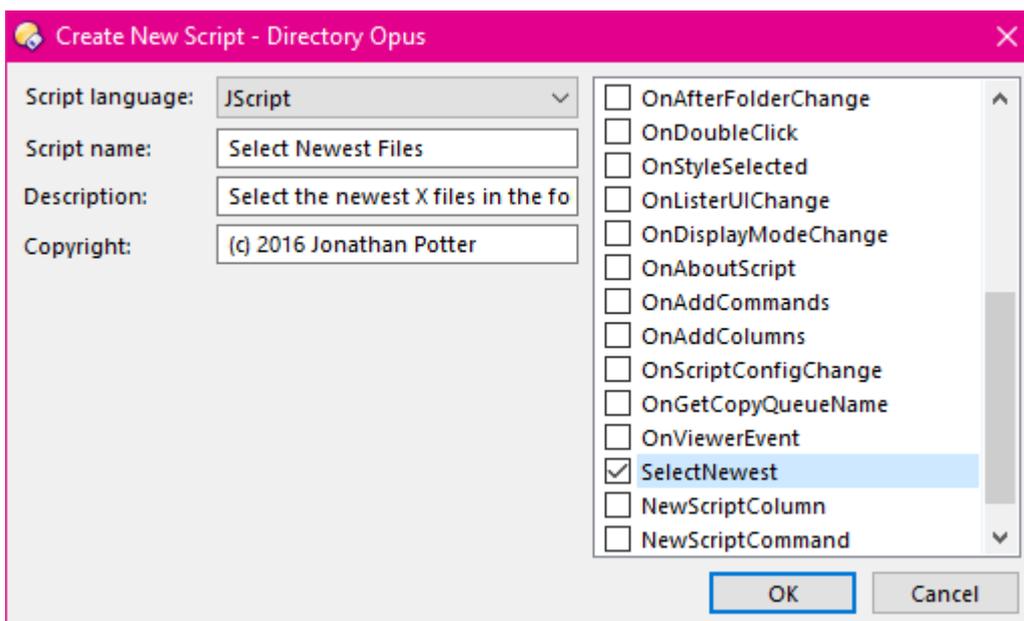
You can configure which columns the script list displays by right-clicking the column header. Any columns that aren't turned on are displayed at the bottom of the dialog when you select a script from the list. The checkbox next to each script name lets you disable the script without deleting it.

The toolbar buttons at the top allow you to control your installed scripts:

-  **New Script**: Displays the *Create New Script* dialog, which makes it easy to start a new script. See below for more information.
-  **Import**: Import a script that someone has shared with you. You can also install new script files by dragging and dropping them onto the list

-  **Delete:** Delete a script. If the script is part of a script package, the whole package will be deleted.
-  **About:** If a script provides About information (via the [OnAboutScript](#) event), clicking the *About* button will display it.
-  **Configure:** If a script defines configuration options, clicking the *Configure* button will display a dialog letting you make changes to the script's configuration.
-  **Edit:** If a script is a standalone file (rather than a script package), clicking the *Edit* button will open it in your default text editor.
-  **Open Folder:** Opens the script folder in a new tab (the `/dopusdata/Script AddIns` folder).
-  **Refresh:** Refresh the list of scripts.
-  **Disable All Scripts:** This option lets you disable all script add-ins at once - individual enable/disable states will be preserved. You need to click **Apply** to make this change.

If you click the New Script button (  ) the *Create New Script* dialog is displayed, which makes it easy to build a new script



Currently this dialog lets you create a template for a *JScript* or *VBScript* script.

Select the desired language, and enter a name, optional description and copyright string. Then use the checkboxes in the list to select the events that you want the script to create functions for.

The last two events in the list, *NewScriptColumn* and *NewScriptCommand*, lets you create a script that adds internal columns and commands. When you turn this option on it will activate and let you enter a name for the column or command. Press return to accept the new name, and another *NewScriptColumn* or *NewScriptCommand* entry will be added to the list. In this way you can easily create a template for a script that [adds multiple internal commands](#).

## Viewer

This category contains options that affect the Opus image viewer - both the [Standalone Viewer](#) and the [Viewer Pane](#).

- **Appearance:** Contains options that affect the appearance of the standalone image viewer.
- **Behavior:** Contains options that affect the behavior of the standalone image viewer.
- **Mouse Buttons:** Contains options that affects how your mouse buttons behave in the standalone image viewer.
- **Viewer Pane:** Contains options that modify the behaviour of the Viewer Pane.
- **Plugins:** View and configure your installed viewer plugins (third-party components that can add additional image formats to Opus).

### Viewer Appearance

These options affect the [Standalone Image viewer](#) - the Opus viewer that opens in a separate window. Options that affect the [Viewer Pane](#) are configured on a [separate page](#).

- **Auto-size viewer window:** Normally the viewer will remember the size and position of its window from one use to the next. If you turn this option on then when the viewer opens it will automatically resize itself; the options available for this are:
  - **To fit every picture:** The viewer will resize to fit the first picture, and resize itself for every subsequent picture as well.
  - **To fit the first picture:** The viewer will resize itself to fit the first picture only.
  - **Full screen:** The viewer will always open in full-screen mode.
- **Minimum window width:** If the viewer is auto-sized, this option lets you specify a minimum width below which it won't go. You can save this from a existing viewer using the **View / Save Minimum Width** command in the menu.
- **Center viewer window:** This causes the viewer window to be centred on the screen. If this is turned off you can position the viewer where you like and it will remember its position in future.
- **Default gamma correction:** Use this option if you need to adjust the gamma of images for proper color reproduction on your monitor. This sets a default gamma adjustment but you can adjust the display on-the-fly from within the viewer.
- **Frame picture:** This option causes a frame (with shadowing) to be drawn around the image displayed in the viewer.
- **Hide scrollbars:** If this option is turned on the viewer will not display scrollbars even if the image is too large to fit in the display. You can pan around an image with the cursor keys, and using the mouse (depending on the options set below). Note that some viewer plugins may not support this option.
- **Show status icons:** If the currently viewed image has any [status icons](#) assigned these will be displayed in the status bar (or top-left corner of the viewer window if the status bar is turned off).

- **Show thumbnails in Marked Pictures pane:** If this option is on, thumbnails of each [marked image](#) will be shown in the Marked Pictures pane in the viewer.
  - **Show in icon mode:** Thumbnails will be shown when the Marked Pictures pane is in icon mode.
  - **Thumbnail size:** Lets you configure how big the thumbnails in the Marked Pictures pane will be.
- **Use EXIF information to auto-rotate pictures:** Most modern digital cameras contain an orientation sensor that records into the image the orientation of the camera when the picture was taken. If this option is turned on Opus will read the orientation information from the image and automatically rotate the displayed picture so that it appears the right way up. The actual image file on disk is not modified, only the display in the viewer.
- **Background color:** This lets you define the background color used by the viewer. If an image doesn't completely fill the viewer's window it will be framed by this color. You can turn on the **Auto** option to have Opus try to pick a background color that matches the current image.
- **Toolbar:** This lets you select a different toolbar to use in the viewer.
- **Viewer title bar:** These options let you configure the text shown in the viewer's title bar.
  - **Display full path:** Turn this option on to display the full path to each file in the title bar of the viewer window. If turned off, only the filename will be shown.
  - **Custom title:** This option lets you completely configure the string used in the title bar. If you turn this option on, the text you provide in the *Custom title* field will be used to generate the title string.

You can use several special "tokens" in the title string to insert various pieces of information:

- %P - full path of the currently viewed image
- %N - name of the current displayed image
- %R - drive root of the current image
- %E - displays \* if the image's metadata has been modified and not saved
- %I - current image's index (number) in the list of images
- %O - total number of images in the list
- %W - width of the current image
- %H - height of the current image
- %D - depth of the current image (bits per pixel)
- %M - current image's dimensions
- %S - file size on disk
- %F - folder name
- %C - collection name if current image is [marked](#)
- %L - any [labels](#) assigned to the current image
- %T - complete original title (useful for simply adding a prefix or suffix to the title)
- %% - insert a literal % character

## Viewer Behavior

These options affect the [Standalone Image viewer](#) - the Opus viewer that opens in a separate window. Options that affect the [Viewer Pane](#) are configured on a [separate page](#).

- **Accelerated scrolling while dragging:** If this option is on, grabbing the image with the mouse will move the image faster, such that you only have to move the mouse a small amount to scroll the entire image. If it's turned off, you'll get 1:1 movement where the part of the image you drag will stay under the mouse. Holding **Ctrl** while dragging will also switch modes, in case you sometimes need both.

- **Check for marked pictures when viewer is opened:** If this is turned on, the viewer will check if any images in the folder have already been [marked](#) in a previous session.
- **Display Marked Pictures pane when a picture is marked:** When you [mark](#) an image in the viewer, the Marked Pictures pane will open automatically. If this option is turned off you can still open the pane manually.
- **Generate Next/Previous list:** If this option is on, when you double-click on an image in a Lister to view it, Opus automatically builds a list of all the other images in that folder as well. This lets you move to the next or previous image in the folder, and to run a slideshow for all images in the folder, just by double-clicking one image. If turned off, only the image you double-clicked on will be added to the picture list. This doesn't affect what happens if you select one or more images and run the **Show** command on them - in that case, only the selected images will be in the picture list.
- **Reset scroll position for each picture:** If this option is turned on then whenever you move to the next or previous image, Opus will reset the scroll position to the top/left instead of leaving it in the current position.
- **Reset zoom level for each picture:** If this option is turned on then whenever you move to the next or previous image, Opus will reset the zoom level if you changed it while viewing the current image. You can choose from *Fit To Page*, *Grow To Page* or *Original Size*.
- **Reuse existing viewer window:** If this option is turned on Opus will re-use an existing viewer window when you double-click on a new file, instead of opening a new one. (If you are using the virtual desktops feature of Windows 10 and above, Opus will normally only look for existing windows on the active virtual desktop. You can change this via the advanced [virtual desktop isolation](#) setting.)
- **Save marked images to a file collection:** By default, Opus records images you [mark](#) to a file collection. This option lets you configure the name of that collection. The default, **Marked Pictures\%F** uses **%F** to insert the name of the current folder. You can also use **%D** to insert the current date. If you turn this option off, the viewer will use [checkbox mode](#) in the parent Lister to indicate images that you mark.
  - **Ask for collection name:** The viewer will prompt you to enter a name for the collection when you begin a new marking session.
  - **Open collection when viewer is closed:** After you mark some images and close the viewer, the collection will be automatically opened in a new tab.
- **Wrap-around picture lists:** When the viewer's picture list contains more than one image, this option causes the viewer to wrap around to the first image in the list when you try to advance past the last image (and the same option affects slideshows).

The options listed under **Standalone Viewer Slideshow Settings** affect the Slideshow function of the standalone [Image Viewer](#).

- **Automatic slideshow:** If this is on, the viewer will go into slideshow mode whenever it opens with more than one image in the picture list.
- **Randomize slideshow:** This option randomizes the order of pictures in the list when displaying a slideshow.
- **Slideshow speed:** This lets you control the speed in seconds between each image. You can use fractions of a second if desired.

## Mouse Buttons

These options affect the [Standalone Image viewer](#) - the Opus viewer that opens in a separate window. Options that affect the [Viewer Pane](#) are configured on a [separate page](#).

- **Left mouse button:** This lets you control what the left mouse button does when you click it in on the image. The options you can choose from are:
  - **Select clipboard region:** Lets you click and drag to select a region of the image to copy to the clipboard (or to crop to). Holding the **Shift** key changes the behavior to *Scroll image*.
  - **Scroll image:** Lets you pan around an image that's larger than the viewer window by click and drag. Holding the **Shift** key changes the behavior to *Select clipboard region*.
  - **Expand/scroll image:** If the image is displayed at a reduced size (e.g. the zoom level is set to *Fit to Page*) then clicking and holding the mouse button will expand the image to full size.
  - **Advance to next image:** Advances the viewer to the next image in the picture list.
  - **Toggle Full Screen mode:** Clicking the button will toggle full-screen mode on or off.
  - **Close Viewer:** Clicking the mouse button will close the viewer window.
  - **Toggle Slideshow:** Clicking the button will toggle slideshow mode on or off.
  - **Run a command:** Clicking the button will run the specified Opus command in the context of the viewer.
  - **Script event:** Clicking the button will trigger any script add-ins that implement the [OnViewerEvent](#) script event.

Under **Left mouse button** there is also the **Click left/right edges to go to previous/next picture** option. When enabled, clicking the left and right edges of the window will advance to the next or previous image irrespective of the actual setting for the mouse button. You can control the percentage of the window width that Opus considers to be "the edge" (the default is 20%).

- **Left double-click:** This lets you control what the left button does when you **double-click** it on the image; the options are mostly the same as for the **Left mouse button** setting except you can't choose **Select clipboard region** or **Scroll image** for double-click.
- **Middle mouse button:** This lets you define what the middle mouse button does when you click it on the image; the options are the same as for the **Left mouse button** setting.
- **Mouse wheel:** This lets you control what the mouse wheel does when you turn it over the viewer window. Note that the viewer ignores this setting when viewing some file types. For example, when viewing a text file the wheel always scrolls up and down. The options are:
  - **Scroll image:** The mouse wheel will scroll the current image up and down.
  - **Advance to next image:** Scrolling the mouse wheel over the viewer will have the effect of moving forwards and backwards through the current picture list.
  - **Zoom:** The mouse wheel will zoom in and out of the image. If one of the other options is selected you can still zoom with the wheel by holding down the **Control** key.

Under **Mouse wheel** there is also the **Accumulate wheel movements** option, which is enabled when **Advance to next image** is selected. This option causes wheel movements to accumulate and several quick wheel movements can move you several places forward or back in the list of images without loading each image in between. (On mice with 'smooth' wheels, this could make it easy to accidentally skip images by turning the wheel too much).

## Viewer Pane

This page contains options that affect the [Lister Viewer Pane](#). Changes you make here don't affect the [standalone image viewer](#) - there are separate pages for those options.

- **Accelerated scrolling while dragging:** If this option is on, grabbing the image with the mouse will move the image faster, such that you only have to move the mouse a small amount to scroll the entire image. If it's turned off, you'll get 1:1 movement where the part of the image you drag will stay under the mouse. Holding **Ctrl** while dragging will also switch modes, in case you sometimes need both.
- **Automatically refresh image when file changes:** If this option is turned on then Opus will monitor the currently viewed file, and refresh the display of it if it changes.
- **Automatically select next file when viewed image is deleted:** If the image you are currently viewing is deleted, Opus will automatically select the next file in the file display. This lets you quickly weed out a large collection of pictures using only the keyboard - click on the first file to view, press **Cursor Down** if you want to keep it, or **Delete** if you want to delete or (in which case Opus will automatically move on to the next file).
- **Display shell thumbnails:** If nothing else (except the hex viewer) can display a file, ask the Windows shell to generate a thumbnail for it and display that in the viewer pane. Explorer's viewer pane also does this and it is useful for file formats that have thumbnail providers but no viewers. Shell thumbnails and icons are unavailable within archives and VFS plugins. Changes to the setting affect the next file loaded by the viewer pane.
- **Display shell icons:** Similar to the **display shell thumbnails** option, except it displays the file's icon instead of its thumbnail. Thumbnails are given priority if both options are enabled.
- **Frame picture:** This option causes a frame (with shadowing) to be drawn around the image displayed in the viewer.
- **Gamma-correct pictures:** Use this option if you need to adjust the gamma of images for proper color reproduction on your monitor.
- **Hide scrollbars:** If this option is turned on the viewer will not display scrollbars even if the image is too large to fit in the display. You can pan around an image with the cursor keys, and using the mouse (depending on the option set below). Note that some viewer plugins may not support this option.
- **Scroll with left mouse button:** This option modifies the default behavior of the left mouse button when you click on the image. If this option is turned on you can click the left mouse button on the image and drag it to pan around the display, and if you hold the **Shift** key down when you click, you can drag an area of the image to select it for copying to the clipboard. With this option turned off, these behaviors are reversed - you would need to hold the **Shift** key down to pan the image.
  - **Expand and scroll:** If the image is displayed at a reduced size (e.g. the zoom level is set to *Fit to Page*) then clicking and holding the mouse button will expand the image to full size.
  - **Expand and scroll (Left double-click):** The expand/scroll mode will be triggered on a double-click instead of a single click.
- **Show control bar:** This causes a toolbar to be displayed at the bottom of the viewer panel, which contains buttons for commonly used functions.



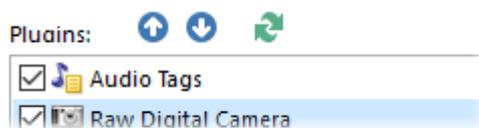
From left to right, the buttons are **Previous File** (⏪), **Next File** (⏩), **Rotate Left** (↶), **Rotate Right** (↷), **Zoom In** (+), **Zoom Out** (-), **Original Size** (✖), **Fit To Page** (📐), **Grow To Page** (📏), **Hex View** (🔍), **Slideshow** (🎞️), **Full Screen** (🖥️), **Print** (🖨️) and **Settings** (⚙️).

- **Use EXIF information to auto-rotate pictures:** Most modern digital cameras contain an orientation sensor that records into the image the orientation of the camera when the picture was taken. If this option is turned on Opus will read the orientation information from the image and automatically rotate the displayed picture so that it appears the right way up. The actual image file on disk is not modified, only the display in the viewer.
- **Picture background color:** This lets you define the background color used by the viewer. If an image doesn't completely fill the viewer's window it will be framed by this color. You can turn on the **Auto** option to have Opus try to pick a background color that matches the current image.

## Viewer Plugins

Viewer Plugins are third-party extensions to Opus that extended the ability of Opus to view, convert, and access metadata in images and other file types.

This page shows you a list of all your currently installed plugins. If plugins offer a configuration dialog of their own, you can access it from here. You can also change the order of precedence for plugins, which is useful if you have two plugins that can handle the same type of file.



The plugin list displays your installed plugins. The toolbar buttons can be used to change the order (⬆️ - move up, and ⬇️ - move down) and refresh the list to detect newly added plugins (🔄). Plugins must be installed in the [/home/Viewers](#) folder.

You can also turn plugins off using the checkboxes next to their names; this lets you turn off plugins for file types you don't use which may speed up the process when viewing files.



When you select a plugin in the list it expands to show some information about that plugin - normally a description of the plugin and a list of the file extensions that it handles. A plugin may also display an **About** button to display more information about the plugin, and a **Configure** button which will let you configure the plugin itself. These two functions are provided by the plugin itself so the options provided and behavior of the configuration dialogs may be different to those of Opus.

When you view an image in Opus that Opus doesn't handle itself, Opus will move through the plugin list from top-to-bottom to try to find a plugin that can handle the file. If you have two plugins installed that can handle the same type of file you can use the up and down buttons to change the order of precedence and so control which plugin is used to handle the file.

Some plugins are a special type known as *catch-all* - these are plugins that by definition can handle (in some way or another) any type of file. For example, the supplied **Text** plugin can ultimately handle any type of file because it displays a raw hex-view of any file that isn't plain text. Catch-all plugins can not be moved up from the bottom of the list as they must be checked last after other, more specific, plugins have been tried.

## Zip and Other Archives

This category contains options relating to Opus's handling of archive files. Opus provides built-in support for Zip files, and the supplied 7-Zip plugin provides support for many other common archive formats. You can add additional plugins to extend the archive handling capabilities even further.

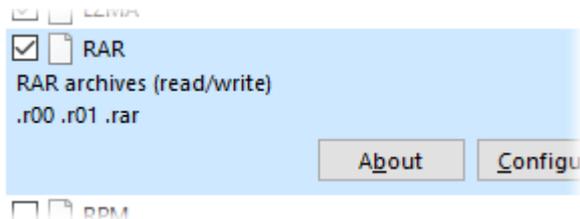
- [Archive and VFS Plugins](#): View and configure your installed VFS plugins (third-party components that can add additional archiving formats to Opus).
- [Archive Context Menu](#): Control which archive-related options Opus will display in the context menu for files and folders.
- [Archive Options](#): General options relating to archive files.
- [Zip Files](#): Options specifically relating to the built-in Zip support.

### **Archive and VFS Plugins**

VFS Plugins are third-party extensions to Opus that extended the ability of Opus to list, extract and create archive files. **VFS** is an acronym for Virtual File System and refers to the fact that plugins can also implement whole file-systems (based on a URL-formatted path like *coll://* for

[File Collections](#)). However by far the most common use for plugins is to add support for archive files to Opus.

This page shows you a list of all archive types handled by your currently installed plugins. The refresh button at the top of the list (🔄) lets you refresh the list to detect newly added plugins. Plugins must be installed in the [/home/VFSPlugins](#) folder.



Each individual archive type can be turned on or off using the checkbox button. A plugin may also display an **About** button to display more information about the plugin, and a **Configure** button which will let you configure the plugin itself. These two functions are provided by the plugin itself so the options provided and behavior of the configuration dialogs may be different to those of Opus.

Note that support for Zip files is built-in to Opus, and so there is no entry for Zip in the Plugins list. Instead you can configure Zip options on a [separate page](#).

When a plugin provides a list of supported archive formats to Opus, those formats are automatically added to the members of the **Archives File Type Group**, which means you should be able to access them like folders (by double-clicking to display their contents) automatically.

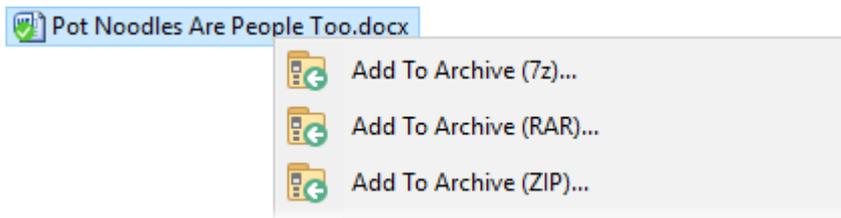
## Archive Context Menu

Opus can add items to the context menu for files and folders that lets you access its archiving capabilities. This page lets you choose whether such items are added, and which ones.

Use the **Enable Archive context menu** option at the top of the page to enable or disable the context menu support. The list of options below that is grouped into five sections:

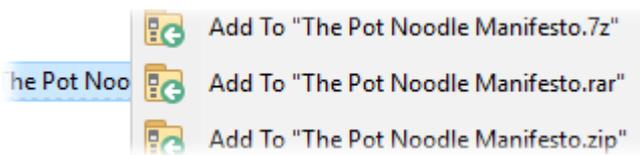
- **Add to Archive**

The options in this section will add context menu items that invoke the [Add to Archive](#) dialog box for the selected files. The Add to Archive dialog lets you choose the archiving format to use, so you really only need one option from this section selected. The format you select here will be the default when the dialog box opens, so you can turn on multiple formats in order to have a quick way of invoking the dialog with the format already chosen.



- **Add to Named Archive**

The options in this section will add context menu items that immediately create archives in the specified format. The name of the new archive will be based on the first selected item. You won't have any way to control archiving options using these commands - the archives are created immediately.



- **Archive and Email**

These options add context menu commands that archive the selected files and send them automatically as an email (using the settings on the [Email](#) page).

- **Extract**

These options add context menu commands that let you extract archive contents.

- **Extract from archives:** When you right-click on a file in a format that Opus recognizes (or drag and drop such a file with the right mouse button), Opus will add context menu items to let you extract the archive contents to the current folder.
- **Convert to Self-Extracting Archive:** When you right-click on a Zip file, Opus will add a command that lets you convert the file to a [self-extracting archive](#). This is only available for Zip archives.

- **Options**

This section contains options that affect the archive context menus.

- **Add single folders as sub-folders of new archives:** Changes what happens when you right-click a single folder and choose one of the **Add to Named Archive** context menu commands. When on, the folder itself will be included in the archive, with the folder's contents below it. (e.g. *Folder.zip/Folder/File.txt*) When off, the *contents* of the folder, but not the folder itself, are added to the archive. (e.g. *Folder.zip/File.txt*) When zipping multiple folders, or when creating archives via something other than the **Add to Named Archive** context menu commands, any selected folders are always included at the top level of the archive. In particular, note that the **Add to Archive** context menu command is not affected by this option.
- **Cascade context menu items:** If this option is turned on, the archive commands that Opus adds to context menus will be displayed in a sub-menu instead of on the top level of the menu.
- **Display context menus in Explorer as well as Opus:** If this is turned on then the context menu items added through this page will also be shown when you right-click on files in Explorer. If turned off, the context menus will only be shown in context menus within Opus.
- **Display icons in context menus:** Adds icons to each context menu item.

## Archive Options

This page contains options that apply to archive handling in general, irrespective of the archive format.

- **Auto-extract archive contents on double-click:** Turn this on to have Opus automatically extract the full contents of the archive to a temporary folder when you double-click on a file within the archive. This is most useful for archives containing installers - if you double-click on the enclosed *Setup.exe* program you would normally want all the other files of the archive extracted as well in order for the installer to run successfully. You can optionally use the drop-down to have Auto-extract only take effect when the specified qualifier key is held down.
  - **Extensions:** This specifies the file extensions that Auto-extract works on. This is specified as a list of file extensions separated by semi-colons. Normally you wouldn't need to set this to anything other than **.exe** for executable files.
  - **Prompt before extracting:** Before the contents of the archive are automatically extracted, Opus will prompt you for confirmation.

## Zip File Options

Support for Zip files is built-in to Opus, and so options affecting Zip files are found on this page rather than on the [Archive Plugins](#) page.

- **Enable internal Opus Zip support:** Turn this on to enable Zip file support in Opus. If this is turned off then the built-in Zip support is disabled completely; Opus would not be able to handle Zip files unless a third-party plugin were installed to do so.
- **Make Opus the system default handler for Zip files:** If this is turned on then Opus will register itself as the default handler for **.zip** files. Double-clicking a Zip file in Explorer or other program will cause an Opus Lister to open showing the contents of the archive.
- **Ask for encryption/compression settings when copying into Zip files:** If this is turned on and you copy files to an existing Zip archive, Opus will prompt you for encryption and compression settings on the fly.
- **Open Zip files as read-only by default:** If this is turned on, when you navigate to an existing Zip file it will automatically be treated as read-only by Opus. You will not be able to make changes to the contents of the Zip file without turning off [read-only mode](#).
- **Use temporary file when copying to Zip files:** If this is turned on and you add files to an existing Zip archive, Opus makes a temporary copy of the Zip file before modifying it. This is safer as if something goes wrong (power failure, etc) the contents of the original Zip file will be unaffected. However it can slow down the process of adding files to archives.
- **Use temporary folder when copying to Zip files on removable drives:** Normally when copying to Zip files on a removable drive (e.g. a USB drive) Opus will first write the archive to the system temporary folder, and then copy it to the destination. Turn this option off if you want to write data to removable drives directly.
- **Set archive date to date of newest file within it:** If this is turned on, the "last modified" timestamp of the Zip archive will be automatically set to the time of the newest (most recent) file within the archive. The timestamp is updated whenever files are added to or removed from the archive.
- **Compression level:** This lets you set the default compression level when adding files to Zip files. If you are using the [Add to Archive](#) dialog you can override this at the time. There are six compression levels available, ranging from *Store* (which does no compression at all and so is the fastest) to *Best* (which produces the highest level of compression but takes longer to archive).

- **Enhanced compression level:** This activates an [enhanced compression algorithm](#) that may not be backwards compatible with some Zip tools - if you are sending Zip files to other people you should make sure they can decompress such archives.
- **Zip Extensions:** This lets you specify the file extensions that Opus will treat as Zip files. Many common file formats are actually Zip files "in disguise", for example [.jar](#) files are readable with Zip tools.

Zip Extensions:

This is specified as a semicolon-separated list of file extensions.

- **Hide from Tree:** This option is used to specify which file extensions (from the Zip Extensions list) are not to be displayed in the tree. If you turn on the option in [Folder Tree / Contents](#) to display archives in the tree, this lets you stop certain Zip formats from appearing there. For example, **.exe** files can be treated as Zip files if they are self-extracting archives, but you probably don't want every **.exe** file on your machine appearing in the folder tree.
- **Packed column:** When you are viewing the contents of a Zip file in a file display you can add several Zip-specific columns to the file display, including the **Packed** column. This column displays the compressed or "packed" size of files within the archive. You can use this option to change the units the **Packed** column displays file sizes in (*bytes*, *KB* or *auto* - which means Opus chooses the most appropriate unit automatically).

# Customize Mode

The Customize system is used to configure the toolbars, menus and hotkeys in Directory Opus. All the toolbars in Opus are configurable - you can edit the supplied ones or create your own. Toolbars can also be *floated* - opened as independent windows not tied to a Lister - for example, to be used as program launchers. Using Opus you can also configure *hotkeys* - combinations of keystrokes that can activate Opus commands or launch external programs.

One upshot of there being no "fixed" toolbars in Opus is that when this help file refers to accessing commands on such-and-such a toolbar or in such-and-such a menu, it refers only to the default toolbars and menus - if you've edited the defaults or turned those toolbars off then obviously those instructions will be less helpful.

Customize works as a "modal" system. When you enter Customize mode, all toolbars switch to "edit" mode and normal Lister operations are suspended. The idea is that you enter Customize mode, make your desired edits, then leave Customize mode and try out your changes.

There are a number of ways to enter Customize mode - the most common are:

- Select **Customize Toolbars** from the **Settings** menu in a Lister
- Right-click on an empty area on a toolbar and choose **Customize** from the context menu
- Click on the window icon (the top-left icon in the title bar) in a Lister and choose **Customize** from the menu. This is a handy thing to remember as it always lets you get to Customize even if you have turned off all your toolbars or accidentally deleted the Customize command from your menu.
- If the appropriate options are turned on in the [Windows Integration](#) Preferences page, you can access Customize mode from the Windows control panel or by right-clicking an empty area of the desktop

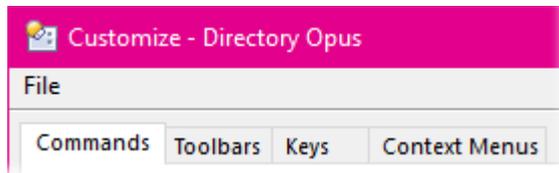
Opus comes with four pre-defined toolbars ([the default toolbar set](#)). These toolbars are treated slightly differently to others - you can still edit them and turn them off, but you can't delete them and you can't float them outside the Lister. If at any time you want to return to the default toolbars, there are a number of ways to do it:

- If one of the default toolbars are currently open and you just want to reset it to its defaults, you can right-click an empty area on the toolbar and choose **Factory Reset this Toolbar** from the context menu. Only the selected toolbar will be reset.
- If you want to reset all toolbars to the defaults, right-click an empty area on any toolbar, and from the Toolbars sub-menu in the context menu, choose **Factory Reset Toolbars**.
- You can also choose the **Factory Reset Toolbars** command from the Toolbars sub-menu of the default **Settings** menu (unless you've edited it!).
- Finally you can reset the individual default toolbars to their original settings from the [Toolbars](#) page of the Customize dialog.

When you reset the default toolbars, any changes you may have made to those toolbars will be lost, but any other toolbars you have created will be unaffected.

# The Customize Dialog

The Customize dialog appears when you enter [Customize](#) mode. It is divided into a number of tabbed pages.



- [Commands](#): Displays a list of commands that you can easily add to toolbars by drag and drop.
- [Toolbars](#): Displays a list of your toolbars, lets you turn them on or off, and create and edit new ones.
- [Keys](#): Displays a list of all configured hotkeys and lets you edit them and create new ones.
- [Context Menus](#): This is where various context menus can be edited. The context menus for files and folders are not edited through here - instead they are controlled through the [File Types](#) system. The context menus here are for user interface elements like the column header in a details mode file display, the taskbar icon and so on.

At the top of the Customize dialog, the **File** menu contains a number of commands:

- **Import**: The behavior of this command depends on the currently selected page in the Customize dialog.
- **Export**: Same for this; what is actually exported depends on the current page.
- **Save Floating Toolbars**: This command saves the state and position of any floating toolbars that are currently displayed. For example, if you want to set up a floating toolbar to act as a program launcher, you would float it, position it as desired, and then choose this command to make Opus remember this for the future. If you turn on the **Save state of floating Toolbars automatically on exit** option on the [Toolbar Options](#) Preferences page, your floating toolbars will be remembered automatically when Opus shuts down.
- **Undo Changes**: This command depends on the current selection in the current page. For example, on the Toolbars page it lets you restore the currently selected toolbar, undoing any changes to it since Customize mode was entered.
- **Undo All Changes**: This command will undo all changes to everything it is possible to undo since Customize mode was entered. Clicking the **Cancel** button on the Customize dialog has the same effect.

## Commands

The Commands page displays a list of commands that you can easily add to toolbars by drag and drop. The commands are divided into a number of categories that loosely groups similar commands (often commands are grouped based on where they appear in [the default toolbars](#)).

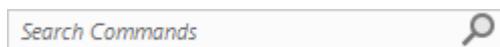
- [Edit](#): These are commands that normally appear in the Edit menu (copy, cut, paste, undo, etc.)

- [File](#): This category contains commands that normally appear in the File menu (new Lister, new tab, etc.)
- [File Commands](#): These are commands that act on files and folders - normally found on the default Operations toolbar (copy file, create folder, delete, etc.)
- [Go](#): These are mostly commands relating to navigation, including FTP, favorites, etc. This category also contains a number of [field commands](#) that add location fields to the toolbar.
- [Help](#): This category contains commands that normally appear on the Help menu (about, help, etc.)
- [Miscellaneous](#): Miscellaneous commands that don't fit in any other category.
- [New](#): This category contains commands that let you create new, empty toolbar buttons and other elements.
- [Settings](#): This contains commands relating to configuration (Preferences, Customize, etc.)
- [Script Commands](#): This contains any commands that have been added by a [Script add-in](#).
- [Tools](#): Commands that normally appear in the Tools menu (find, synchronize, etc.)
- [User-defined Commands](#): This category lists any [User commands](#) that you have created.
- [View](#): Contains commands that relate to the view and appearance of file displays (view mode, flat view, etc.)

Opus functions are built from a set of [internal commands](#) (like a simple scripting language), and each command takes multiple parameters ("arguments") that control its behavior. While this can be very flexible it does require delving into the bowels of Opus somewhat, and so many common functions have been supplied in the list on this page as pre-defined commands.

The pre-defined commands have user-friendly names, and if you click on them a description of the command is displayed to the right of the page. To add one of the commands from this list to your toolbars, simply drag it from the list and drop it where you want it to go. When dropped on a toolbar they resolve to their underlying command and arguments in the button that's created, so if you [edit](#) one of these buttons after dropping it on a toolbar you can see what the "real" command is.

At the bottom of the Customize dialog (when it's showing the Commands page) is a field that lets you filter the commands list.



When you enter one or more keywords in the field and press the **Enter** key the list will be filtered to only show those commands whose name or description contains the keyword(s). This can make it very easy to find a pre-defined command when you remember its name but not what category it is in. To clear the filter either press **Escape** when the cursor is in the field, or click the little **X** button at the right-hand end of the field.

The category list on the Commands tab works the same as the similar list on the [Preferences](#) dialog. You can use the keyboard to navigate it: **Cursor Up / Down** to move through the list, **Cursor Right / Left** to expand or collapse a category, **\*** to expand all categories and **-twice** to collapse all categories.

The **Import User Command** and **Export User Command** commands in the *File* menu for this page let you import and export [user-defined commands](#).

### ***Edit Category (Pre-defined commands)***

The pre-defined commands in this category are usually found in and around the default Edit menu.

- **Copy:** Copies selected files and folders to the clipboard.
- **Cut:** Cuts selected items to the clipboard.
- **Invert Selection:** Inverts the selection of all items in the current file display.
- **Paste:** Paste the clipboard contents to the current folder. As well as files and folders you can also paste text (a `.txt` file is created) and images (an image file is created as a JPEG by default, and you can change the format using the `clipboard_image_paste` option on the [Miscellaneous / Advanced](#) page in Preferences).
- **Paste Shortcut:** Pastes a shortcut to the files and folders currently on the clipboard.
- **Read-Only:** Toggle [read-only](#) mode on or off when inside a Zip archive.
- **Reselect Files:** Reselects the files and folders that were used by the previously executed command.
- **Select Advanced:** Access the [Advanced Selection](#) dialog.
- **Select All:** Select all files and folders in the current file display.
- **Select Field:** This places a [file selection field](#) on the toolbar.
- **Select None:** Deselect all files and folders in the current display.
- **Select Wildcard:** Access the [Simple Wildcard Selection](#) dialog.
- **Undo:** Undo the last operation (if possible).
- **Undo List:** [Dynamic button](#) that displays a list of previous operations that can be undone.
- **Undo Log:** Display the [Undo Log](#).
- **Undo Menu:** Adds a pop-up menu item that displays the undo list.

### ***File Category (Pre-defined commands)***

This category contains commands that usually appear on the default File menu.

- **Close Lister:** Closes the current Lister, but leaves Opus running (depending on the state of the **Shutdown Directory Opus when the last Lister closes** option in [Preferences / Launching Opus / Startup](#)).
- **Exit Program:** Closes all Listers and exits Directory Opus.
- **Open New Lister:** Opens a new Lister window.
- **Open New Tab:** Creates a new tab in the current file display.

## **File Commands (Pre-defined commands)**

This category contains pre-defined commands relating to file operations. Many of these commands appear on the [default Operations toolbar](#). Most of these commands use the Opus concept of [source and destination](#) folders.

- **Add To Archive:** Displays the [Add To Archive](#) dialog to create an archive from the selected files and folders.
- **Administrator Mode:** Turns on [Administrator Mode](#) for the current Lister (except on Windows XP).
- **Copy As:** [Copy](#) selected files and folders to the destination, giving them new names.
- **Copy File:** [Copy](#) selected files and folders to the destination folder.
- **Copy Filter:** Turn the recursive [Copy Filter](#) on and off for the current Lister.
- **Create Folder:** Create a [new folder](#) in the current file display.
- **Delete:** [Delete](#) selected files and folders.
- **Delete Filter:** Turn the recursive [Delete Filter](#) on or off for the current Lister.
- **Duplicate:** Duplicate (make copies of in the same location) selected files and folders.
- **Edit Metadata:** Edit the [metadata](#) for selected files.
- **Email Files:** Send selected files as email attachments (files are sent as-is, not zipped first).
- **Extract:** Extract the contents of selected archive files.
- **File Filter:** Turn on or off both the [Copy and Delete Filters](#).
- **Get Sizes:** [Calculate the sizes](#) for selected folders (if no folders are selected, all folders in the current list will be scanned).
- **Join:** [Join](#) selected files together.
- **Move As:** [Move](#) selected files and folders to the destination, giving them new names.
- **Move File:** [Move](#) selected files and folders to the destination folder.
- **New Archive:** Create a [new archive file](#).
- **Play:** [Play](#) selected sound files using the Opus sound player.
- **Print:** Print selected files.
- **Properties:** Display the properties dialog for selected files and folders.
- **Remove From Collection:** Remove selected items from a [file collection](#) (only works within collections). If you use the **Delete** command on items within collections, the actual files will be deleted.
- **Rename:** [Rename](#) selected files and folders (this defaults to the [Simple Rename](#) dialog).
- **Secure Wipe:** [Securely delete](#) selected files and folders.
- **Set Attributes:** [Change the attributes](#) and timestamps of selected files and folders.
- **Set Description:** Assign a [description](#) to selected files and folders.
- **Set Label:** Assign a [label](#) to selected files and folders.
- **Shortcut:** Create a shortcut to selected files and folders in the destination.
- **Show:** Display selected image files using the Opus [image viewer](#).
- **Slideshow:** Run a slideshow of images in the current folder (if any images are selected only they will be shown, otherwise all images in the current folder will be displayed).
- **Split:** [Split](#) selected file into multiple smaller parts.

- **Update All:** Copy selected files [if they have changed](#) (have more recent timestamps) or don't exist at all in the destination.
- **Update Existing:** Copy selected files [if they have changed](#) (have more recent timestamps) than those in the destination. Only files that already exist in the destination will be copied.
- **ZIP And Email Files:** Zip selected files and send the resulting archive as an email attachment.

## Go Category (*Pre-defined commands*)

This category contains pre-defined commands to do with navigating (changing folders) in Listers. Many of these commands are [dynamic buttons](#) (buttons that outside of Customize mode expand to multiple buttons that display various dynamic items) and [field buttons](#) (buttons that outside of Customize mode expand to fields or other controls).

- **Compatibility Files:** This button switches to the current folder's [compatibility folder](#) (if it has one) and back again.
- **Disconnect Network Drive:** Disconnect a mapped network drive.
- **Drive Buttons:** A dynamic button that displays a list of your available disk drives.
- **Drive List:** A drop-down list of your available disk drives.
- **Favorites Add:** Add the current folder to your [Favorites](#) list.
- **Favorites Add (Dialog):** Displays a display to add the current folder to Favorites.
- **Favorites Edit:** Edit the Favorites list (opens Preferences dialog to the [appropriate page](#)).
- **Favorites List:** A dynamic button that displays a list of your favorite folders.
- **Favorites Menu:** A drop-down menu that displays a list of your favorite folders.
- **Favorites Menu (Dual Display):** Same as **Favorites Menu** except folders selected from this list will be opened in the dual-display.
- **Favorites Smart:** A dynamic button that displays a list of your [SmartFavorites](#).
- **Favorites Smart Menu:** A drop-down menu that displays your SmartFavorites.
- **FTP Address Book:** A dynamic button that displays a list of the sites in your [FTP Address Book](#).
- **FTP Address Menu:** A drop-down menu that displays your FTP sites.
- **FTP Quick Connect:** Displays the [FTP Quick Connect](#) dialog to connect to an FTP site.
- **Go Back:** Go back to the previous folder.
- **Go Back (Drop-down List):** Go back to the previous folder, with an attached drop-down list of all previous folders.
- **Go Forward:** Go forward to the next folder.
- **Go Forward (Drop-down List):** Go forward, with an attached drop-down list of all subsequent folders.
- **Go To Computer:** Go to the *Computer* folder.
- **Go To Documents Folder:** Go to your personal documents folder.
- **Go To File Collections:** Go to the root [File Collections](#) folder.
- **Go To Libraries:** Go to the root *Libraries* folder.
- **Go To Network:** Go to the system *Network* folder.

- **Go To Root:** Go to the root of the current drive.
- **Go To The Desktop:** Go to the *Desktop*.
- **Go To The Recycle Bin:** Go to the system *Recycle Bin* folder.
- **Go To The Start Menu:** Go to the folder containing your start menu program links.
- **Go To The Startup Folder:** Go to the folder containing your startup programs (the *Startup* group within the *Start Menu*).
- **Go Up:** Go up to the parent folder.
- **Map Network Drive:** Map a network location to a drive letter.
- **Navigation Lock:** Turn [Navigation Lock](#) on or off in the current Lister.
- **Path Field:** A field button that displays a very basic field for editing the current location.
- **Path Field (Breadcrumbs):** A field button that displays a [breadcrumbs field for the current location](#).
- **Path Field (Favorites List):** A field button that displays a location field with a drop-down list of your Favorite folders attached.
- **Path Field (Folder Tree):** A field button that displays a location field with a drop-down folder tree attached.
- **Path Field (Recent List):** A field button that displays a location field with a drop-down list of your recent folders.
- **Recent List:** A dynamic button that displays a list of your recent folders.
- **Recent List Clear:** Clear the recent folder list.
- **Recent Menu:** A drop-down menu that displays a list of your recent folders.
- **Swap Source/Destination:** Swap the current [source and destination](#) file displays.

### ***Help Category (Pre-defined commands)***

This category contains commands that are usually displayed in the default Help menu.

- **About:** Displays information about the program including the current version number.
- **Help:** Opens this help file.
- **Licence Manager:** Displays the [Licence Manager](#) where you can install and manage your program licence (including requesting a free evaluation).

### ***Miscellaneous Category (Pre-defined commands)***

This category contains miscellaneous commands that don't fit into any other category. Currently all the commands in this category are special [dynamic buttons](#) known as *marker* buttons. They are used to mark the place on a toolbar or in a menu where commands that come from the system

(known as namespace-specific commands) should be added. Some folders add toolbar buttons that are specific to the folder. For example, the *Virtual Machines* folder adds buttons to create a new virtual machine. In Opus, these buttons will be displayed by the marker buttons.

The types of marker buttons are:

- **All Menus:** This will display all namespace-specific commands with the one command.
- **Edit Menu:** Displays namespace-specific commands intended for the Edit menu.
- **File Context Menu:** Displays the Windows-provided context menu items for the selected file or folder. This is used in the File menu by default to display context-sensitive commands for selected files and folders.
- **File Menu:** Displays namespace-specific commands intended for the File menu.
- **Help Menu:** Displays namespace-specific commands intended for the Help menu.
- **Lister Context Menu:** Displays the Windows-provided context menu items for the current folder (not selected folder, but the current location). This is used in the Folder Context menu (the menu you get when you right-click on an empty area of the file display) to display context-sensitive commands for the current folder.
- **Other Menu:** Displays namespace-specific menu items that aren't provided by the other categories.
- **Toolbar:** Displays namespace-specific toolbar buttons.
- **Tools Menu:** Displays namespace-specific commands intended for the Tools menu.
- **View Menu:** Displays namespace-specific commands intended for the View menu.

### ***New Category (Pre-defined commands)***

The items in this category are not actually commands; instead they provide a way to add new (empty) elements to your toolbars via drag and drop.

- **New Button:** Creates a new button with no function defined. Use this if you want to create a button from scratch instead of starting with a pre-defined command.
- **New Menu:** Creates a new [drop-down menu](#). A menu has a label and image but no function of its own; instead, it contains buttons and sub-menus.
- **New Menu Button:** Creates a new [menu button](#), which is a combination of a button (with a function) and a menu (a drop-down).
- **Spacer:** Creates a spacer, which is used to add padding or right-justification to toolbars.

### ***Settings Category (Pre-defined commands)***

This category contains pre-defined commands relating to configuration tasks. Some of these commands are [dynamic buttons](#) (special buttons that expand to display a list of items).

- **Backup and Restore Settings:** Access the Preferences [Backup and Restore](#) dialog.
- **Customize:** Enter [Customize](#) mode (displays the Customize dialog).

- **Enable Sounds:** Turn [sound effects](#) on or off.
- **File Types:** Open the [File Types](#) editor, that lets you edit file context menus, tooltips, icons and more.
- **FTP Add To Address Book:** Add the current FTP site to the address book (only works when connected to an [FTP site](#)).
- **FTP ASCII Transfer Mode:** Set the transfer mode of the current FTP site to ASCII.
- **FTP Auto Transfer Mode:** Set the transfer mode of the current FTP site to Automatic.
- **FTP Binary Transfer Mode:** Set the transfer mode of the current FTP site to Binary.
- **FTP Edit Address Book:** Open the [FTP Address Book](#) dialog.
- **Full-Row Selection:** Turn full-row selection on and off in the current file display (when in details or power mode).
- **Lister Themes:** Open the [Lister Themes](#) dialog.
- **Preferences:** Open the [Preferences](#) dialog.
- **Save All Listers:** Save all Listers as a new [Lister layout](#).
- **Save Single Lister:** Save the current Lister to a Lister layout (ignores any other open Listers).
- **Saved Layouts - Drop-Down:** Adds a drop-down dynamic list of your saved Lister layouts.
- **Saved Layouts - Edit:** Displays the [Lister Layouts](#) Preferences page.
- **Saved Layouts - List:** A dynamic button that displays a list of your saved Lister layouts.
- **Set As Default Lister:** Set the current Lister as the [Default Lister](#).
- **Tab Group List:** A dynamic button that displays a list of your [tab groups](#).
- **Toolbar List:** A dynamic button that displays a list of your toolbars that you turn them on or off.
- **VFS Plugin List:** A dynamic button that displays a list of your installed [VFS plugins](#) (archive formats).
- **Viewer Plugin List:** A dynamic button that displays a list of your installed [viewer plugins](#).

### ***Tools Category (Pre-defined commands)***

The commands in this category are mostly functions that you might find in the default Tools menu.

- **Change Default Printer:** A [dynamic button](#) that expands to show a list of your installed printers, letting you quickly change the default printer.
- **CLI:** Displays the Opus [CLI](#) window which lets you enter commands interactively.
- **Command Field:** A [field button](#) that displays a text field that lets you enter commands directly.
- **Command Prompt Here:** Opens a DOS prompt window with the current directory set to the current folder.
- **Convert Images:** Access the [image conversion](#) tool.
- **Duplicate Files:** Search your computer for [duplicate files](#).
- **Find:** Access the Opus [Find tool](#) to search for files or folders.
- **Flickr Synchronize:** Access the [Flickr synchronization](#) system.
- **Print Folder:** [Print the contents](#) of the current folder (to a printer, disk file or the clipboard).

- **Search Field:** A field button that displays a quick search field that uses [Windows Search](#) for fast indexed searches.
- **Synchronize:** [Synchronize](#) files and folders.

## ***User-defined Commands***

User-defined commands are pre-defined commands that you create yourself. Effectively it's like creating a toolbar button with your own function on it, but instead of the button living on a toolbar, it lives in this command list and other buttons or menus can refer to it by name.

See the page on [User-defined commands](#) in the [Creating your own buttons](#) section of the manual for more details.

## ***View Category (Pre-defined commands)***

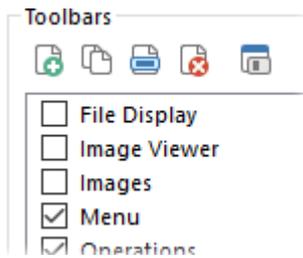
The View category contains commands relating to the view mode and other options affecting the current file display.

- **Check-Box Mode:** Toggle [check-box mode](#) on or off in the current file display.
- **Current Content Type:** A [field button](#) that adds a drop-down control that both displays and lets you change the current [Content Type](#) format in use in the file display.
- **Dual File Display:** Toggle from single to dual file display and back again.
- **File Filter Field:** A [field button](#) that gives you a field for [filtering the file display](#) (unlike the more specific fields below, this field can filter both files and folders and has many more options).
- **Flat View (Grouped):** Puts the current file display into [Flat View](#) (Grouped) mode.
- **Flat View (Mixed No Folders):** Puts the current file display into [Flat View](#) (Mixed No Folders) mode.
- **Flat View (Mixed):** Puts the current file display into [Flat View](#) (Mixed) mode.
- **Flat View Off:** Turns [Flat View](#) mode off in the current file display.
- **Flat View Toggle:** Toggle [Flat View](#) on or off in the current file display. You can edit the button and change the default mode that it toggles into.
- **Folder Formats List:** A dynamic button that displays a list of your favorite [folder formats](#).
- **Folder Formats Menu:** A drop-down menu that displays a list of your favorite folder formats.
- **Folder Options:** Opens the [Folder Options](#) dialog for the current file display.
- **Folder Options List:** Opens the Folder Options dialog, and has an attached drop-down menu that displays a list of your favorite folder formats
- **Folder Properties:** Displays the properties dialog for the current folder.
- **Folder Tree:** Toggles the [folder tree](#) on or off.
- **Format Lock:** Toggles the [format lock](#) on or off (if you don't have it displayed in the status bar you can use this command to access this feature).

- **FTP Site Properties:** Displays the Site Properties dialog for the current FTP site (only works when connected to an FTP site).
- **Hide Files Field:** A field button that lets you edit the current Hide Files pattern for the current folder format.
- **Hide Folders Field:** A field button that lets you edit the current Hide Folders pattern for the current folder format.
- **Lister Styles - Drop-Down:** A drop-down button that gives you a list of your [Lister styles](#).
- **Lister Styles - List:** A dynamic button that produces a list of your Lister styles.
- **Lister Styles - Tabs:** A field button that turns into a tab control letting you quickly switch between your Lister styles.
- **Metadata Pane:** Toggle the [Metadata Pane](#) on or off.
- **Refresh:** Refresh the current file display.
- **Refresh All:** Refresh both file displays and trees.
- **Refresh Both:** Refresh both file displays.
- **Refresh Tree:** Refresh the folder tree.
- **Show Files Field:** A field button that lets you edit the current Show Files pattern for the current folder format.
- **Show Folders Field:** A field button that lets you edit the current Show Folders pattern for the current folder format.
- **Status Bar:** Toggle the [status bar](#) on or off in the current Lister.
- **Thumbnail Size:** A field button that displays a slider control whenever the file display is in thumbnails mode - it lets you resize the thumbnails in real-time.
- **View As Large Icons:** Set the current file display into large icons mode.
- **View As Small Icons:** Set the current file display into small icons mode.
- **View In Details Mode:** Set the current file display into details mode.
- **View In List Mode:** Set the current file display into list mode.
- **View In Power Mode:** Set the current file display into power mode.
- **View In Thumbnails Mode:** Set the current file display into thumbnails mode.
- **View In Tiles Mode:** Set the current file display into tiles mode.
- **View Mode Cycle:** Cycle through display modes in the current file display. If you edit the function you can specify if any view modes are excluded from the cycle.
- **Viewer Pane:** Toggle the [Viewer Pane](#) on or off.

## Toolbars

The Toolbars page displays a list of all your toolbars, and lets you create new ones, delete existing ones, and turn them on or off. You can also change various appearance settings for individual toolbars from this page.



The checkboxes indicate whether a toolbar is currently turned on or off. Use the toolbar buttons to manipulate the toolbars:  (create a new toolbar),  (duplicate an existing toolbar),  (rename toolbar),  (delete toolbar) and  (float toolbar). Note that the [default toolbars](#) cannot be deleted or renamed. The **Reset to Defaults** command only applies when one of the default toolbars is selected.

When you select a toolbar from the list (and it is turned on), you are able to edit a number of settings on the right-hand side of the page. These are grouped into two sections.

The **Background** section controls the background color and image of individual toolbars. By default toolbars are set to use the *Standard Toolbar Image*, which makes it easy to change the image for all toolbars at once (you only have to change one setting) and means that [Lister Themes](#) can also change the background image of toolbars. However you can make a toolbar use any background image that you have added to the list on the Preferences [Display / Images](#) page. The options in this section are:

- **Color:** Turn this on if you want to specify a solid background color for the toolbar. If this is turned off the default color from Preferences will be used.
- **Image:** Turn this on if you want to select a background image from the drop-down. To use an image on a toolbar you must first add it to the list of [Images](#) in Preferences. You can also specify whether the image is stretched, tiled or shared across the Lister. If **Image is inherited by submenus** is turned on then any drop-down menus from the toolbar (and any of their child menus) will display the same background image as the parent toolbar.

The **Images & Labels** section lets you override the image and label settings of individual buttons on the toolbar. Toolbar buttons can define their own image and label state, as well as image size, but the options in this section let you override them and instantly turn all labels or images on or off for the whole toolbar.

- **Image Size:** Specify the image size (*small* or *large*) for buttons on the toolbar. If you select *default* then the buttons' own settings will be used.
- **Image State:** Specify the image state (*on* or *off*) for all buttons on the toolbar, or select *The Import and Export commands in the File menu for this page let you export the selected toolbar so you can share it with other people, or import toolbars you have received from others.* to use each individual button's own settings.
- **Label State:** Specify the label state (*on* or *off*) for all buttons on the toolbar, or select *default* to use the buttons' own settings.
- **Label Color:** Turn this option on to specify the label color for buttons on the toolbar. The color specified on the [Colors and Fonts](#) page of Preferences will be used if turned off.

- **Font:** Turn this option on if you want to specify a font for button labels. If turned off the font configured on the [Colors and Fonts](#) page of Preferences will be used instead.

If the **Always enable this toolbar's keys in Listers** option is turned on, any hotkeys defined within the toolbar will be active in Listers whether or not the toolbar itself is currently visible. This is used for the two main default toolbars (*Menu* and *Operations*), so that even if you turn them off, standard keys like **Ctrl+A** (for *Select All*) will still function.

From this page you can float a toolbar (other than one of the default toolbars) by selecting it from the list and clicking the *Float Toolbar* button (). See the [Controlling Floating Toolbars](#) page for information on controlling the appearance and behavior of floating toolbars.

Double-clicking a toolbar in the list will cause any instances of that toolbar to briefly flash. This can be a handy way to find where a toolbar is located. Any instances of that toolbar that are floating are also displayed and will flash briefly to identify them.

The **Import Toolbar** and **Export Toolbar** commands in the *File* menu for this page let you export the selected toolbar so you can share it with other people, or import toolbars you have received from others. The **Factory Reset** command lets you reset any of the default toolbars to their factory settings - you can also do this by right-clicking the toolbar itself (in a Lister) to access its context menu.

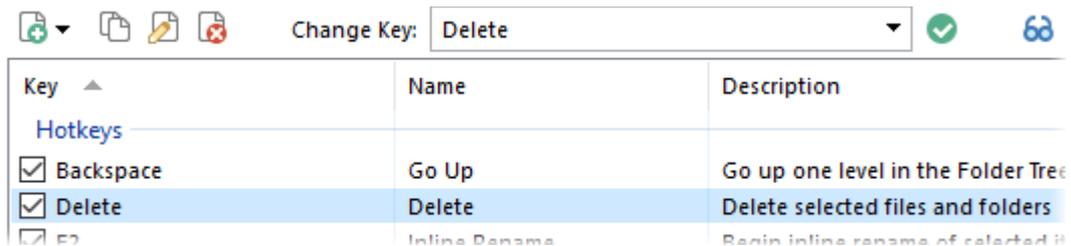
## Keys

The Keys page lets you configure "hotkeys" (combinations of keys) that can be used to run commands or launch programs. Opus lets you assign one or more keys to a toolbar button or menu command, and pressing the key has the same effect as clicking the button. You can also create "hotkey-only" functions, which only exist as key assignments and aren't linked to a toolbar.

This page shows a list of all the defined hotkeys, grouped into the following categories:

- **Hotkeys:** These are functions that exist purely on this page as a hotkey function. They are active only inside a Lister.
- **Toolbars:** These are hotkeys assigned to a toolbar button. They are active only inside a Lister.
- **Favorite Folders:** These are hotkeys assigned to [Favorite](#) folders. They are active only inside a Lister.
- **Hotkeys (System-wide):** These are hotkey-only functions that are active throughout Windows - they work whether Opus is active or not.

- **Floating Toolbars (System-wide):** These are hotkeys assigned to a toolbar button in a [floating toolbars](#). They are active globally, but the toolbar must have its **Enable Hotkeys** option turned on to enable them.
- **Tray Menu (System-wide):** These are any hotkeys assigned to functions in the [taskbar icon context menu](#). They are active throughout Windows.
- **Image Viewer (Keys):** These are hotkeys that are only active in the [standalone image viewer](#).
- **Image Viewer (Toolbar):** These are hotkeys assigned to a toolbar button in the currently designated toolbar for the [standalone image viewer](#).



The checkboxes next to each key in the list let you temporarily disable a hotkey without having to delete it.

Use the toolbar at the top of the page to manipulate the hotkeys. The buttons are (create new hotkey), (duplicate an existing key), (edit the selected key) and (delete the selected key).

The (Locate toolbar) button is active whenever a hotkey in a **Toolbars** category is selected - clicking it will highlight the toolbar button that the hotkey comes from (it will also open any sub-menus needed to make the button visible).

You can only delete items in the **Hotkeys** category through this page - toolbar buttons can only be deleted from the toolbar itself. *Favorites*-type keys can't actually be edited from this page at all - they are shown in the list as a convenience, but you have to use the [Favorites](#) page in Preferences to edit them.

Clicking the button to create a new hotkey displays the standard [Command Editor](#) with the addition of the **System-wide Hotkey** option, which you can turn on to make the new hotkey global.

Clicking the button will display the standard [Command Editor](#) for the selected key, but if all you want to change is the key itself (and not the function), you can use the **Change Key** field at

the top of the page. Select the key you want to change, click in the change key field, press the new key combination and then use the  button to accept the change.

The **Change Key** field, as well as the hotkey field in the Command Editor (and the filter field at the bottom of this page) are all special controls designed to make it easy to enter a key combination. See the page on [Using the Hotkey Control](#) for more information.

You can also define keys that use the **Windows** key. Windows itself defines many hotkeys using the **Windows** key (e.g. **Windows+E** opens an Explorer window), but Opus lets you override these for global hotkeys. To override one of the standard Windows hotkeys, create a new hotkey, turn on the **System-wide Hotkey** option, and then use the key field to enter the key combination as you would any other. (Note: How well this works depends on the version of Windows, since hotkeys involving the **Windows** key may be intercepted by the operating system at a low level. If you are on Windows 10 or above and just want to change what the **Windows + E** hotkey does, [Preferences / Launching Opus / From the Win + E hotkey](#) may be easier and more reliable.)

At the bottom of the page is a filter field that lets you filter the hotkey list. This filter has two modes. The default mode is to show hotkeys that use a specified key combination. For example, click in the field, press the **A** key and then click the filter button to the right to show all functions that use the **A** key in some way. Click the red X button to clear the filter and re-display all the keys.



Click the keyboard icon () to switch the filter to the other mode, in which it filters based on the name, function and description of the hotkey. For example, in this mode typing "rename" would filter the list to show all hotkeys related to renaming functions.

The **Export Keys** command in the *File* menu for this page let you export your hotkeys list in three different formats (use the **Save as type** drop-down in the save dialog to choose what type to use).

- *Importable Hotkey Format* is a format that you can use to export your hotkeys in order to re-import them into Directory Opus (e.g. on another computer, or to share your keys with a friend). Selecting this format actually exports the functions as well as the key information. When this format is selected only your *Hotkeys* and *Media Keys* are exported.

- *Comma-Separated List* exports a list of all the hotkeys along with their names and descriptions (but not the functions they run) as a `.csv` file. For example, you could use this if you wanted to make yourself a printable key reference. You could import this file into Excel or Word and perform further manipulation on it before printing. In this mode, all types of hotkeys are exported.
- *Text Format* exports a list of all your hotkeys as a plain text file. Like the *Comma-Separated List* option, all types of hotkeys are exported in this mode.

The **Import Keys** command in the *File* menu lets you import a previously-exported *Importable Hotkey Format* file. You will be given the choice of merging the keys in the imported file with your existing ones, or replacing your existing hotkeys completely.

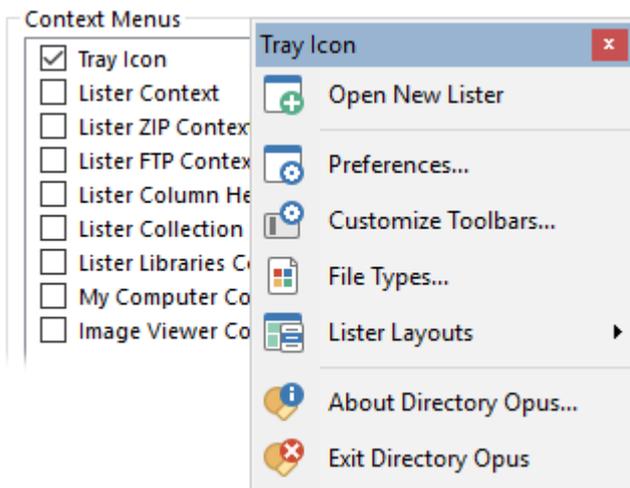
## Context Menus

The Context Menus page is very similar to the [Toolbars](#) page. It lets you configure various context menus (right-button menus). The menus that can be configured through this page are:

- **Tray Icon:** The context menu for the icon that Opus places in the taskbar notification area (a.k.a. "tray icon", "systray", etc). You can turn this icon on from the [Windows Integration](#) Preferences page, and then right-click it (in Windows 7 you can also left-click it) to display this configurable menu. In Windows 7 the icon will be hidden by Windows and moved to the icon overflow area by default, so if you want the icon to always be visible (which makes it much more useful) you will need to tell Windows to show it all the time.
- **Lister Context:** This is the context menu shown for a normal folder (a folder in the normal file system like `C:/Users`) when you right-click on the background of the file display (i.e. on an empty area, not on a file).
- **Lister ZIP Context:** Similar to **Lister Context** except this menu is shown when you are inside a Zip archive.
- **Lister FTP Context:** Similar to **Lister Context** except this menu is shown when you are connected to an FTP site.
- **Lister Column Header:** This is the context menu shown when you right-click the column header in a details or power mode file display.
- **Lister Collection Context:** Similar to **Lister Context** except this menu is shown when you have navigated to a [file collection](#).
- **Lister Libraries Context:** As above, this is the menu shown when you right-click on an empty area of the file display when viewing a [library](#).
- **My Computer Context:** This menu is shown when you are viewing the [native view](#) of the *Computer* folder and you right-click an empty area of the file display.

For the most part the six folder context menu types will be the same, but they do let you add functions that are only relevant to one particular type of folder (e.g. the FTP folder context menu has commands for site properties and sending a raw command which don't make sense anywhere other than on an FTP site).

Functions launched from the **Lister Column Header** can discover which column was right-clicked on using the `%headeritem%` environment variable.



To make changes to a menu, click its checkbox in the list to reveal it. The menu will be displayed in a popup window - kind of like a floating toolbar - and it's this window where you can edit the buttons on the menu. Once you have finished editing the menu you can click the close button in the popup window's title, or click the checkbox again (or just close the Customize dialog). You can display more than one of the menus at once while editing, if for example you want to drag and drop buttons from one to the other.

On the right-hand side of the dialog page are the same **Background** and **Images & Labels** settings that are found on the [Toolbars](#) tab. Select a menu from the list to make changes to these settings.

You can reset the context menus to their default settings using the command in the **File** menu.



# Creating your own buttons

One of the unique features of Opus is that the toolbars are completely configurable. All toolbar (and menu) buttons are built up from one or more commands, consisting of either:

- The [internal commands](#), which give access to the internal features of Opus. All the internal commands have a set of [arguments](#) that can be used to modify their behavior to produce different results (e.g. to move a file instead of copy it).
- External commands, which let you run third-party programs from within Opus. Using [special codes](#) you can pass information like the names of selected files to external programs, integrating them more fully into your Opus configuration.

Toolbar buttons can also be written as a script using an ActiveX scripting language like VBScript or JScript. See the page on [Scripting](#) for more detail on script buttons.

A few points which may not be immediately apparent:

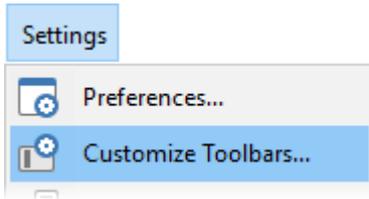
- Toolbar buttons and drop-down menus are the same thing. A drop-down menu is really just vertical toolbar.
- All the toolbar buttons and menus that come with Opus can be changed. You can edit the supplied buttons (move them around, delete them, modify their function, etc), and add your own buttons (either to the standard toolbars, or to toolbars that you create). You can even turn off the standard toolbars altogether.
- If you ever want to return to the default toolbars, just right-click any toolbar and choose the Reset to Defaults command.

And the most important thing to remember about Opus is:

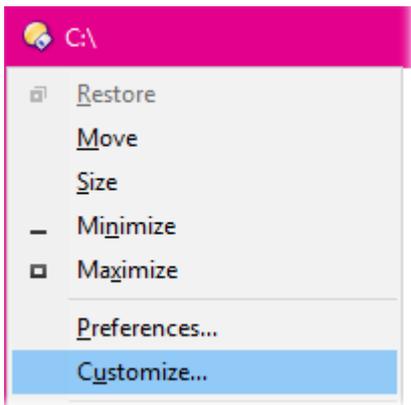
*If you don't like something, you can probably change it!*

## Editing the Toolbar

The first step to editing the toolbar and making your own buttons (or modifying existing ones) is to go into [Customize](#) mode. The easiest way to do this is to select the **Customize Toolbars** command from the **Settings** menu.



*A useful tip to remember:* if you ever find that you can't get to the **Customize Toolbars** command (because you have deleted it from the menu, or turned off that toolbar) you can always get to it from the Lister's window menu (click on the icon in the top-left corner).



Another way to get to Customize mode, particularly if you want to edit a specific button, is to turn on the **Alt-Click to edit Toolbar buttons** option on the [Toolbars / Options](#) page in Preferences. With this option on, holding the **Alt** key and left-clicking a toolbar button will set Opus into Customize mode and display the command editor for the button you clicked on.

However you get into Customize mode, once you are there all toolbars become editable. *Another useful tip to remember:* you can still run a toolbar button from Customize mode, by **Shift**-clicking it.



The above image shows the default toolbars in Customize mode. The only visible change you'll notice when you go into Customize mode is that the various fields have been replaced by placeholders. In the example above, the **Search** field is resizable (indicated by the grip symbols on its right edge) - you can click and drag it by the right edge to resize it. The **Location** and **Spacer** fields do not have resizing grips, which indicates they are set to *full width* mode. See the [Field Buttons](#) page for more information about field buttons.

You'll also notice that, in Customize mode, the buttons on the bottom toolbar don't quite fit in the space available. This is because a previously hidden button - the **Toolbar Marker** button in this case - becomes visible in Customize mode. See the [Dynamic Buttons](#) page for more information about this type of button.

When buttons don't fit in the available space in Customize mode, two small arrows appear on the right edge of the toolbar. You can use these arrows to scroll the toolbar and access the items that are out-of-view.

Toolbar editing is largely mouse-based. A list of common toolbar-editing actions is below. Remember that you need to be in Customize mode to make any changes to toolbars.

- **Edit an existing button**

To edit an existing button, double-click it. Alternatively, right-click it and choose **Edit** from the context menu. In either case, the [Command Editor](#) dialog will open to let you make changes to the appearance and function of the button.

To edit a function in a drop-down menu, left-click the menu to open it, and then edit the button within it as normal. In Customize mode, a drop-down menu will remain open until you click it again to close it (or until another drop-down menu on the same toolbar is opened).

- **Add a new pre-defined button**

To add a pre-defined button (one with a command already assigned to it), locate the desired command on the **Commands** tab of the Customize dialog, then drag and drop it to the toolbar. The button will be added at the location you drop it. You can also add the button to a drop-down menu by dragging over the menu - after a short period of time the menu will pop open, and you can then drop the button into it.

- **Add a new, empty button**

To add a brand new button (one with no function defined) to a toolbar, do any of the following:

- From the Customize dialog's **Commands** tab, expand the **New** category, then drag **New Button** out of the list and drop it on the toolbar. This will add the button at the location you dropped it.
- Right-click the empty space at the end of a toolbar and choose **New > New Button** from the context menu. This will add the button to the end of the toolbar.
- Right-click an existing button and choose **Insert New > New Button** to insert the new button immediately after the one you clicked on.

Once the new button is added, you can edit it (by double-click or right-click as described above).

- **Add a button that runs an external program**

To add a button that launches an external program, you can simply drag the program's icon (or a shortcut to the program) and drop it on the toolbar. For example, you could drag a shortcut from the Windows Start menu, or a program icon from the C:\Program

Files directory.

The [Launch Options](#) dialog will appear when you drop a program on the toolbar, allowing you to choose how the program is run. For example, you may want to run the program on its own, or to pass selected files to it. If the **Launch Options** dialog has been configured to always use particular settings without re-appearing, you can force it to appear by holding the **Ctrl** key as you drop a program on the toolbar.

- **Add a button that takes you to a folder**

Drag a folder (or a shortcut to a folder) and drop it on the toolbar to create a button that will navigate to that folder.

- **Duplicate an existing button**

To make a copy of an existing toolbar button, you can either:

- Drag the button, and while holding the **Ctrl** key down, drop it in the new location.
- Right-click the button and choose **Copy** from the context menu, and then right-click either the empty space on the toolbar or an existing button, and choose **Paste** from the context menu. If you right-click an existing button then the copy will be inserted immediately after it. If you right-click the toolbar's empty space then the copy will be added to the end of the toolbar.

You can copy an existing button to the same toolbar or to another toolbar. You can copy a button into a drop-down menu by dragging the button over the menu and hovering there for a short time - the menu will pop open, letting you drag the button into it. Make sure you keep the **Ctrl** key held down until the mouse button is released.

- **Move an existing button**

To move a toolbar button from one location to another (on the same toolbar or a different toolbar), you can either:

- Drag the button and drop it in the new location.
- Right-click the button and choose **Cut** from the context menu, and then right-click either the empty space on the toolbar or an existing button, and choose **Paste** from the context menu. If you right-click an existing button then the cut button will be moved to the location immediately after it. If you right-click the toolbar's empty space then the cut button will be moved to the end of the toolbar.

You can move a toolbar button into a drop-down menu by dragging the button over the menu and hovering there for a short time - the menu will pop open, letting you drag the button into it. You can also move into or out of a drop-down menu using copy and paste - just click the menu to open it, and then right-click the button within it as normal.

- **Delete a button**

To delete a toolbar button you can either:

- Drag the button from the toolbar and drop it on the Customize dialog.
- Right-click the button and choose **Delete** from the context menu.

- **Add a new, empty sub-menu (or menu button)**

To add a brand new sub-menu (within which you can then add further items) to a toolbar, do any of the following:

- From the Customize dialog's **Commands** tab, expand the **New** category, then drag **New Menu** (or **New Menu Button**) out of the list and drop it on the toolbar. This will add the menu at the location you dropped it.
- Right-click the empty space at the end of a toolbar and choose **New > New Menu** (or **New Menu Button**) from the context menu. This will add the button to the end of the toolbar.
- Right-click an existing button or menu and choose **Insert New > New Button** (or **New Menu Button**) to insert the new menu immediately after the one you clicked on.

Once the new menu is added, you can edit it (by double-click or right-click as described above), and if you open it with a left-click you can add items inside of it in the same way as the top-level toolbar.

See [Drop-down Buttons and Menus](#) to learn about the different sub-menu types.

- **Insert a separator**

To insert a separator between two buttons you can either:

- Drag the button a small distance (a few pixels) to the right. If you drag the button too far you will move it - you only need to drag it a very small distance to insert a separator.
- Right-click the button and turn on the **Begin a Group** option in the context menu.

In either case, the separator will be added between the button you dragged or right-clicked on, and the one immediately before it (to the left).

- **Remove a separator**

To remove a separator from between two buttons you can either:

- Drag the button (the one to the right of the separator) a few pixels to the left.
- Right-click the button and turn off the **Begin a Group** option in the context menu.

- **Put a gap between buttons**

You can use the **Spacer** field to put a gap between buttons.

- On the [Commands](#) tab of the Customize dialog, locate **Spacer** underneath the **New** category, and drag it to your toolbar. Alternatively, right-click an existing button and choose **Insert New > Spacer**, or right-click the toolbar's empty space and choose **New > Spacer**.
- You can adjust the size of the spacer manually, or right-click it and turn on the Full-width option to make it automatically right-justify any subsequent toolbar buttons. See the section on [Field Buttons](#) for more information.

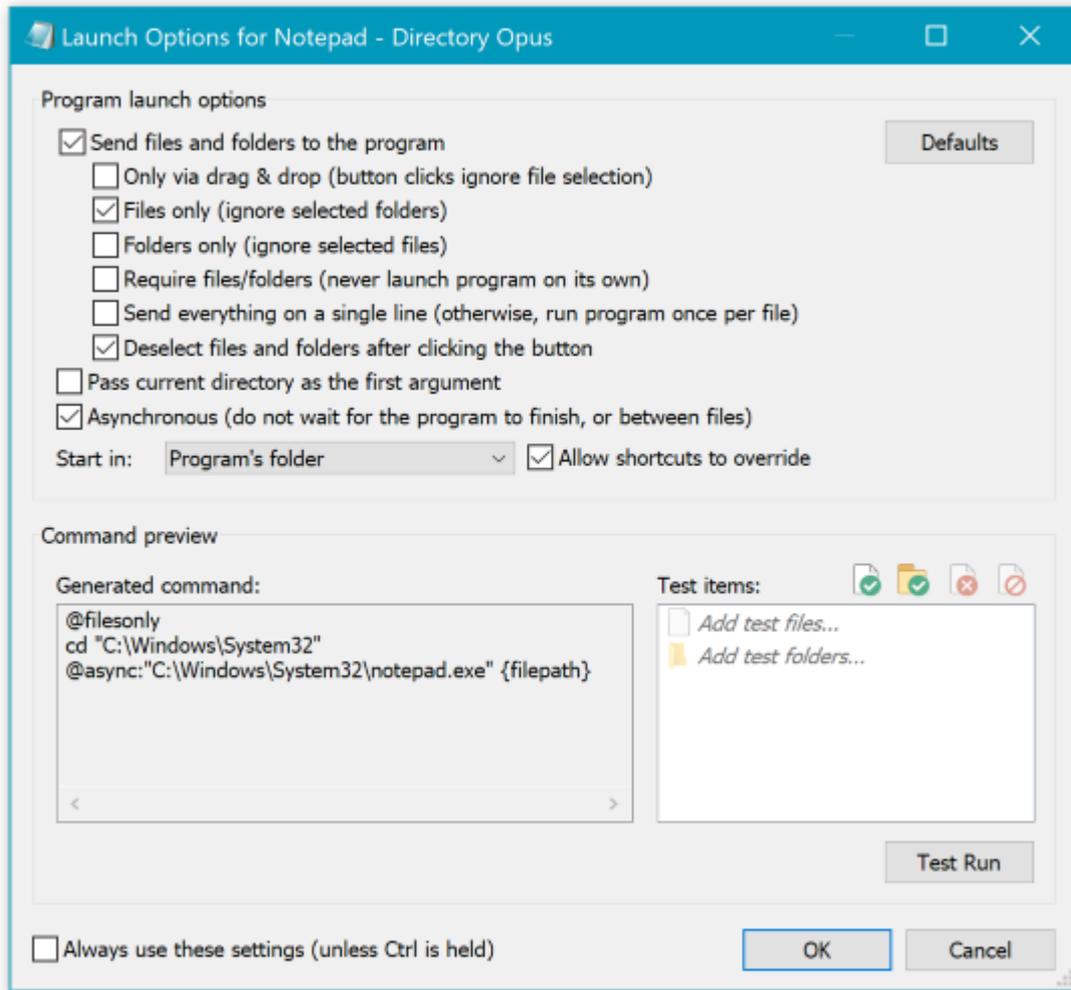
All buttons (and the toolbars themselves) have a [context menu](#) that you can access by right-clicking when in Customize mode.

### ***Launch Options dialog***

The Launch Options dialog is displayed when you drag a program (or shortcut to a program) to a toolbar while in [Customize](#) mode, to create a button that runs the program. It allows you to choose how the program will be launched when the button is clicked. For example, you may want to run the program on its own, using a toolbar as a simple launcher, or you may want to pass selected files to the program so the button is a way to edit files in a particular program.

The Launch Options dialog does not affect how Directory Opus itself is launched. For that, please see [Preferences / Launching Opus](#)

If you are editing a toolbar and drop a .exe file (or shortcut to one) on it, you will see a dialog like this:



If you *don't* see the dialog, it could be for one of a few reasons:

- You aren't in [Customize](#) mode.

You need to be in Customize mode to edit toolbars. Outside of Customize mode, drag & drop on to toolbars allows you to send files to buttons, rather than to create new buttons.

- You had previously chosen **Always use these settings** in the Launch Options dialog.

Hold **Ctrl** while dropping the program on the toolbar to override it and make the dialog appear.

- You've dropped something which isn't an executable (.exe file) or a shortcut to one.
- You've dropped an MSI shortcut or similar, which runs a program but only indirectly.

MSI shortcuts do extra work before launching programs, such as checking if they are installed and up to date, and will be left as-is. You can usually find the main .exe for a program and drop it on the toolbar instead.

The Launch Options dialog provides the options we think are most likely to be needed when creating buttons for programs, and will build the button for you without requiring intimate knowledge of how Opus commands and buttons work. The dialog does not cover every possible situation and is not intended to replace the full [Command Editor](#). The toolbar buttons generated by the dialog are the same as any others and can be edited as usual afterwards, should you need to change any details.

The dialog also provides a simple interface for testing the options you have chosen work with the particular program you are making a button for. Each program may be different and require different things. For example, some editors only let you specify one file at a time, and need you to run them again for a second file; other editors allow you to specify all the files at once to open them in a tabbed view. Editors which support tabbed views may still only allow one thing to be sent to them at a time, but often work by checking if they are already running when run again, and passing the new file to the existing instance of the program so it opens a new tab. Opus has no way to know what a given program needs, and you would normally look into the program's documentation to find out, but you can also experiment with different things in the Launch Options dialog to see what works.

Finally, the dialog has an **Always use these settings** checkbox which lets you choose particular settings and then always use them when dropping new programs on a toolbar. If you use toolbars as launchers and edit them a lot, this may save you some time.

The dialog's main options are as follows:

- **Defaults**

Clicking the Defaults button will reset the options at the top of the dialog to the defaults.

- **Send files and folders to the program**

Turn this on if you want the button to be able to launch the program and pass it files or folders to view, edit or do something else with at the same time. Turn this off if you want the button to simply launch the program on its own, and ignore any selected files or folders.

For brevity, we mostly talk about files below but the same generally applies to folders, unless using the options to restrict things to only one or the other.

- **Only via drag and drop**

Turn this on if you want clicks on the button to ignore selected files and only launch the program on its own, but want to retain the ability to pass files to the program by dropping

them on the button.

- **Files only**

Turn this on if you want the program to only process selected files and ignore selected folders when it is clicked. Note that this does not affect drag & drop, which does not filter what is dropped on the button when passing it to the program.

- **Folders only**

The same as the **Files only** option, except button clicks ignore files and only process folders. Drag & drop ignores this as well.

- **Require files/folders**

Turn this on if the button should only do something when files are passed to it, and you never want to run the program on its own if the button is clicked with nothing selected.

- **Send everything on a single line**

If multiple files are selected when you click the button, there are two possibilities:

If this option is on, the program will be run only once, with all of the selected files specified on a single command line, one after the other.

If this option is off, the program will be run once per selected file, and only given one file at a time on each command line.

Sending all files at once is usually better, if the program supports it, but many programs do not support it.

- **Deselect files and folders after clicking the button**

After the button completes, any files or folders which were passed to the program will be deselected if this option is on, and left selected if it is off.

(Unless [Preferences / File Operations / Options / Deselect files used in functions](#) is off, in which case the option has no effect and deselection is never automatic.)

- **Pass current directory as the first argument**

You may wish to run some programs against the current folder, rather than any selected files or folders below it. Or you may wish to pass the current folder as an argument, followed by the paths to any selected files or folders. If you want either of those, turn this

option on.

- **Asynchronous**

If this option is on, Opus will not wait for the program to finish. If you are launching the program on its own or with just one file, or with all files on a single line, then this option does not matter much. But if you are sending multiple files to the program by running it once for each file, this option can have a large effect.

When the option is on, Opus will not wait for the program and will run multiple instances of the program in parallel. Depending on the program, this may result in multiple windows appearing at once, one for each file, or it might result in one window appearing for all of the files. (Or, if the program is not written very well, a mixture of both.)

Note that the order the windows or files open may seem random because the different instances of the program all compete with each other; an instance which starts slightly later may overtake a slightly earlier one.

When the option is off, Opus will run the program for one file and then wait for it to exit before running it for the next file. This may also result in a progress dialog appearing. (You can add **@noprogess** to the command to stop this by editing the button after creating it.)

Note that Opus only waits for the instances of the program that Opus runs. If the program was already running before you clicked the button, running it again with a new file may mean a new instance starts, notices the existing instance, passes the file to it and then exits. The original instance is still running and now has the file, but the instance Opus ran has exited, and Opus will continue to the next file. This is usually a desirable thing, if you're just using the button to send files to a program and not do anything else with the files afterwards.

- **Start in**

This option specifies the *current directory* from which the program starts. In the full button editor you can specify any folder path at all, but in this dialog you have just the following options:

- **Current folder (least secure):** The folder the Lister is currently in -- usually the same folder the selected files are in -- is used as the current directory.

Some programs require this mode, particularly when a file is not self-contained and depends on other files in the same folder. Programs generally should not require this, since they are given the full path to each file and can work out where to look for companion files themselves, but not all programs are well written.

You should not use this mode unless the program you are running needs it, because it is less secure. It opens you to a type of attack called *DLL planting*

where many programs are easily tricked into loading and running code from DLLs placed in the same folder as the files you ask them to open. This is not a large risk if you know where the files came from, but could pose a risk with archives from the Internet. For example, if you download and extract a zip archive full of cat photos, you might not notice a strange DLL lurking among the JPEGs, and running a program with that DLL in the current directory may cause the program to run the code inside the DLL, which could then do absolutely anything.

- **Program's folder (default):** The location of the program's .exe file is used as the current directory when running the program.

This is the default option and is generally secure, unless the program itself is in a folder which you do not control (but in that case, you can't trust the program itself).

Some programs require this mode because they look for their other components via the current directory instead of relative to where their .exe file is. Programs generally should not require this, but some are not well written, and this flaw affects quite a lot of software.

- **System32 folder:** The C:\Windows\System32 folder (or equivalent location) is used as the current directory.

This is the most secure option in terms of avoiding *DLL planting*, since if a program tries to load a Windows DLL from the current directory it usually get the right one instead of an imposter.

Not all programs will work with this mode, however, and it is not usually more secure than using the program's own folder as the current directory.

Note that if the program tries to create random files in, or does something destructive to the current folder, then you absolutely do not want to run it via System32, as it could trash your system. Of course, Windows UAC will usually protect you from this, unless you have turned it off or are running the program elevated.

The **Allow shortcuts to override** checkbox, if on, means Opus will use the *start in* directory specified by a shortcut when a shortcut to a program is dropped on the toolbar. This is usually a good idea since if a shortcut specifies the program needs to run out of a particular folder it usually has a good reason. The checkbox has no direct effect when dropping the program itself, or when dropping a shortcut which does not specify a *start in* folder, but remember the Launch Options dialog is for configuring the defaults for new buttons as well as setting up individual buttons.

- **Always use these settings**

The dialog always remembers the last settings you used the next time it appears, but you can turn on this option to make it use those settings automatically without appearing at all. If you do that, you can hold down **Ctrl** while dropping a program to turn the option off and make the dialog appear again.

### Command preview and testing the button

- The **Command preview** part of the dialog lets you see and test the command which the options above have generated.

Testing can be important because different programs need different options, and the only way to tell is to try them or read the other program's manual.

- The **Generated command** field is read-only and purely to give you a better idea of what the options do and to help you get used to the way Opus commands look, should you want to get into more detailed button editing.
- The **Test items** list lets you add any files or folders to it to use when testing the command.

You can use the buttons above the list to add things to it, or drag & drop things into it. Once an item is added, you can double-click it to edit it via a file chooser, or single-click it (once already selected) to edit its path using the keyboard.

The list remembers the list of test files between uses, and the rightmost button above it can be used to quickly clear the whole list.

Note that the selected item in this list does not affect the button. You can select items in the list to remove or edit them, but that is all selection is for here. Where we talk about *selected files* above, we only mean files selected in a Lister's folder tab when the button is run normally.

- The **Test Run** button will run the program based on the current settings, as if the finished button had been clicked.

Files and folders from the **Test items** list will be passed to the program, if appropriate.

If the **Test Run** button is disabled, it means you have chosen **Require files/folders** without adding any to the Test items list, or something similar.

### Completing the new button

- If you click **OK**, the new button will be configured based on the options you have specified.

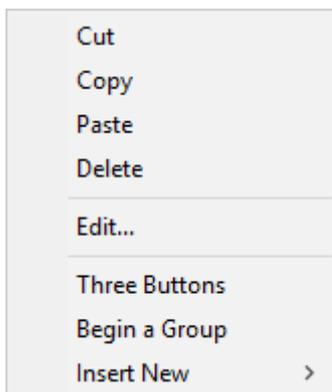
If **Always use these settings** (discussed above in detail) is on, the settings will then be used next time you drop a program on a toolbar, unless you hold down **Ctrl**.

- If you click **Cancel** instead, the new button will be deleted.
- Whichever option you choose, you will still be in Customize mode and the changes to the toolbar not saved to disk until you make the ultimate choice of clicking OK Customize dialog, or Cancel to revert any toolbar changes made since you entered Customize mode.
- If, while the Launch Options dialog is still open, you close the toolbar (or the window it is part of) or click OK in the Customize dialog, the button will be kept but in a default state, without any of your changes. If you do that by accident, the easiest thing is to delete the button and drag the program to the toolbar again.

### **Toolbar Context Menus**

When in Customize mode, all toolbar buttons have a context menu that you can access by right-clicking them. The commands on this context menu will vary based on the type of button.

For a "regular" button, the context menu looks like this:



**Cut:** Cut the button to the clipboard - when pasted to another location on the toolbar, the button will be moved.

**Copy:** Copy the button to the clipboard - you can paste a copy of it to the same or another toolbar.

**Paste:** Paste the button from the clipboard - inserts the button to the left of the button you right-clicked on.

**Delete:** Delete the button.

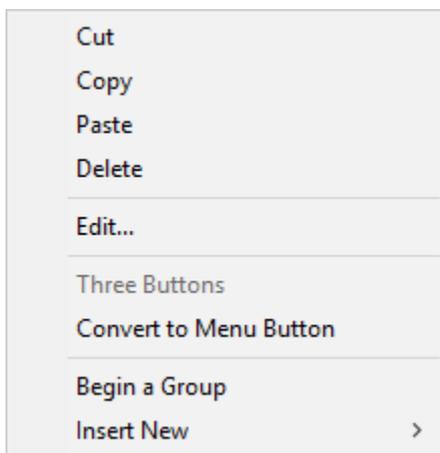
**Edit:** Edit the button.

**Three Buttons:** Turn a simple button into a [multiple-function button](#) (a "three button button").

**Begin a Group:** Insert a separator to the left of this button.

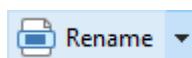
**Insert New:** Insert a new button, drop-down menu or spacer to the left of this button.

The context menu for a *drop-down menu* looks almost the same.



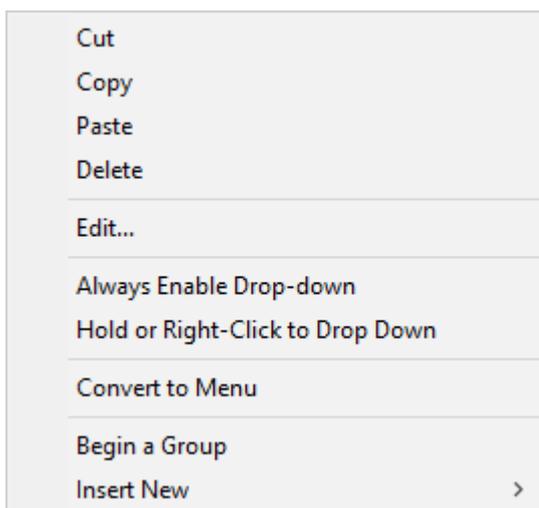
The **Three Buttons** option is unavailable (a drop-down menu can't be converted to a multiple-function button), and there is a new command:

**Convert to Menu Button:** This command turns a drop-down menu (a button whose sole purpose is to drop-down a menu of other buttons) into a drop-down menu button. A menu button combines a regular button, that you can assign a function to, with a drop-down menu. The main part of the button runs the assigned command, and a smaller button to the right that displays an arrow glyph is used to pop open the menu.



See the [Drop-down Buttons and Menus](#) topic for more details.

Again, the context menu for a *drop-down menu button* is very similar, with only one additional command.



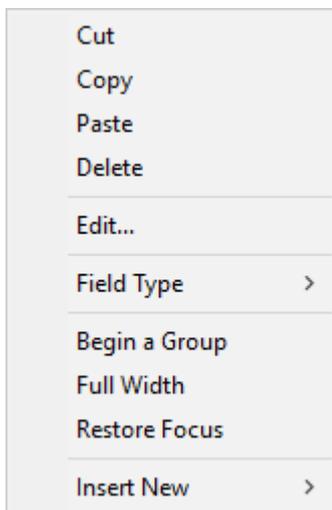
**Always Enable Drop-down:** This option makes the drop-down part of a menu button always available, even if the "button part" is disabled.

**Hold or Right-Click to Drop Down:** This option removes the visible drop-down arrow. The drop-down menu can be accessed either by right-clicking on the button, or by left-clicking and holding the mouse button down until the menu appears.

The **Go Up** button () in the default File Display toolbar makes use of both of these options. The **Go Up** button is disabled when you are at the Desktop level (because you can't go up any further from there). Without the **Always enable drop-down** option turned on, this would make it impossible for you to

access the commands in the drop-down menu attached to the **Up** button.

The context menu for a *field button* looks like this:



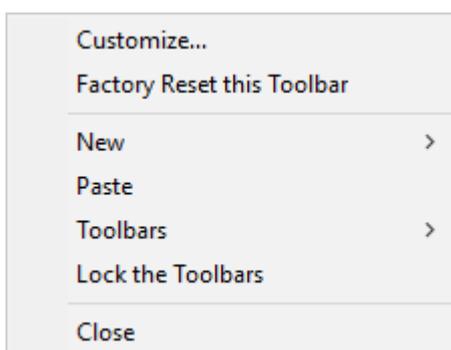
**Field Type:** Lets you change the type of the field button. Field buttons can only be created by dragging the appropriate item from the Commands tab of the Customize dialog (or by duplicating an existing field), but once the button exists you can change it between any of the available types from the context menu.

**Full Width:** Sets the field to *full width* mode. In this mode, it will expand (when Opus isn't in Customize mode) to occupy all the available space in the toolbar. If a field isn't in full width mode you can resize it to any size you like.

**Restore Focus:** If this option is enabled, pressing the **Enter** key in the field will return focus automatically to the file display.

See the [Field Buttons](#) topic for more information about field buttons.

Finally, the context menu for the *toolbar* itself (displayed when you right-click an empty area of the toolbar) looks like this:



**Customize:** Brings the **Customize** dialog to the front if it has been hidden behind another window.

**Factory Reset this Toolbar:** Only available on the [default toolbars](#), this resets the toolbar to its initial settings. Any changes you have made to that toolbar will be lost.

**New:** Add a new button, menu, menu button or spacer to the toolbar. The new button appears at the end of the toolbar.

**Paste:** When you have copied or cut a button to the clipboard, this command lets you paste it to the end of the toolbar.

**Toolbars:** This sub-menu displays a list of all your toolbars, letting you turn them on or off. This sub-menu also contains a **Factory Reset Toolbars** command - selecting this will reset **all** of the [default toolbars](#) to their initial settings. Any changes you have made to the default toolbars will be lost. Any toolbars you have created yourself will be turned off, but otherwise won't be affected.

**Lock the Toolbars:** Removes the dragging grips from the edge of the toolbars, preventing you from moving them around.

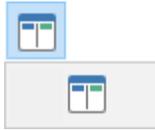
**Close:** Close the toolbar.

## ***Multiple Function Buttons***

A multiple-function button is a button that has up to three distinct functions attached to it (it's therefore also called a "three-button button").

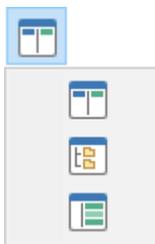
- The first, or primary, function is executed by **left**-clicking the button as normal
- The second function is executed by **right**-clicking the button
- The third function is executed by **middle**-clicking the button. If you don't have a middle mouse button, the **Simulate middle mouse click with control + left-click** option on the [Toolbars / Options](#) page in Preferences can help you make full use of this function.

To make a multiple-function button, you must start with a regular, single-function button. See the [Editing the Toolbar](#) page for information on how to create a new button if desired. In Customize mode, right-click the button and select the **Three Buttons** option from its context menu. This immediately turns the single button into a multiple-function button, and the button's original function becomes the new *primary* function.



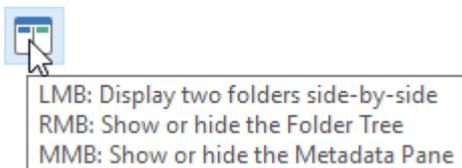
In Customize mode, a multiple-function button behaves very much like a [drop-down menu](#) (other than that the number of functions in the menu is limited to three). The image above shows what you would see initially if you turn the **Dual Display** button on the standard toolbar into a multiple-function button. Effectively, the original button becomes a drop-down menu, and the original function is duplicated and becomes the first button in the new menu.

You can now add one or two additional buttons to the drop-down menu, using the techniques described on the [Editing the Toolbar](#) page.



Here we have dragged the **Folder Tree** button from the Customize dialog and dropped it in the menu - it has become the button's second function. Similarly, the **Metadata Pane** button has been added as the button's third function. The multiple-function button is now "full" - attempts to add additional functions to the menu will fail.

When you leave Customize mode, the toolbar button will appear the same as it did before - and indeed, left-clicking it will perform the same action as before (to turn [dual display](#) mode on or off). It's only when you hover the mouse over the button to reveal the tooltip that the difference becomes apparent.



The tooltip makes it obvious that this button has multiple functions attached to it. The button associated with each function is indicated by the **LMB** (left mouse button), **RMB** (right mouse button) and **MMB** (middle mouse button) prefixes. So in this example, a left-click on the button

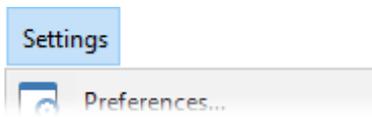
will turn dual display on or off, a right-click on the button will turn the folder tree on or off, and a middle-click on the button will turn the metadata pane on or off.

## Drop-down Buttons and Menus

A drop-down button is a special type of button that displays a menu. A menu can contain one or more buttons, and these can themselves be drop-down buttons. This lets you create cascading menus. There is no limit to how many levels of drop-down menus you can create.

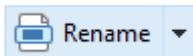
There are two different types of drop-down button:

- Menu: A drop-down menu is a button that displays a menu, and nothing more.

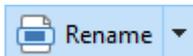


Clicking the button makes the menu appear.

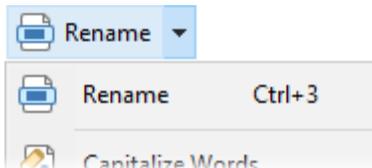
- Menu Button: A drop-down menu button is like a regular button combined with a menu.



A menu button has two distinct parts. The larger part - labeled Rename in the above example - acts like a regular button. It can have a function defined for it, and clicking the button part runs the function.



The smaller part - containing the arrow glyph - displays the drop-down menu. Clicking the arrow part of the button drops the menu down.



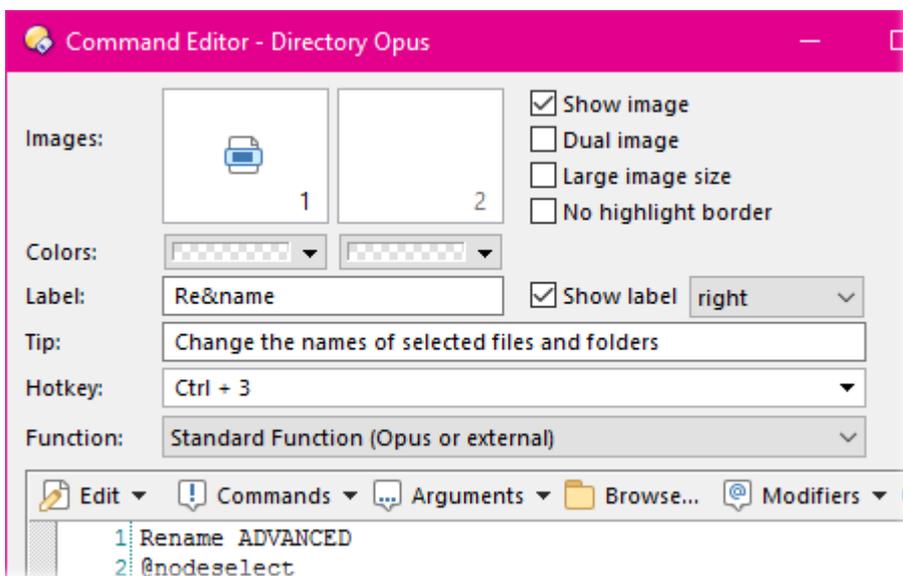
Drop-down buttons are edited in Customize mode like any others - please see the [Editing the Toolbar](#) page for a complete description of how to edit buttons on the toolbar. There are a few minor differences when dealing with drop-down buttons:

- You can convert from one type of drop-down button to the other, and back again. Right-click the menu or menu button, and choose the **Convert** command from the context menu. When you convert a menu to a

menu button, the first button from the menu becomes the "button" part of the menu button. Going the other way, the "button" part of the menu button becomes the first button in the menu.

- To edit a drop-down button, you must right-click the button and choose the **Edit** command from the context menu - double-clicking the button won't work.
- To edit the buttons inside the menu (for either type of drop-down button), pop open the menu first by clicking the button once. Clicking the button again will close the menu.

Drop-down buttons on the top-level of a toolbar respond to an ampersand (&) character in their label. This can be used to mark a letter of the button's label as a type of hotkey - pressing **Alt** plus that letter will pop-open the menu, which lets you access its contents using the keyboard.



This screenshot shows the definition for the **Rename** menu button on the default toolbar. You can see that the button's label is actually set to *Re&name* - even though on the toolbar it displays as *Rename*. The 'n' in the label has been marked by the ampersand, and this lets you press **Alt+N** to pop open the drop-down attached to the **Rename** button. The button also has a hotkey defined (**Ctrl+3**) - this applies to the "button" part of the menu button. Pressing **Ctrl+3** runs the button's command (**Rename ADVANCED**) - the same as if you click the "button" part and not the drop-down.

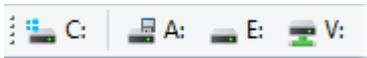
If you want to use a literal ampersand in a button's label, you must use a double **&** (e.g. *Backup && Restore*).

## Dynamic Buttons

Dynamic buttons are a special type of button in Opus toolbars that behave differently in and out of Customize mode. In Customize mode they appear like any other button, and can be edited as normal - but in normal operation, the button itself disappears and is replaced by a dynamic list of items. For example, the Drive Buttons command on the supplied Drives toolbar is a normal button in Customize mode:



But once the Customize dialog has been closed, we can see that the two buttons have disappeared, to be replaced with one button for each of the drives in the system:

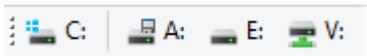


When out of Customize mode the generated buttons behave like normal ones, but the advantage is that you only need a single button to generate them - instead of having to define a button for each drive.

Dynamic buttons are also used for things like displaying your Favorite folders, your FTP sites, and Lister [layouts](#) and [styles](#).

## Drive Buttons Configuration

The **Go DRIVEBUTTONS** internal command produces a dynamic button that expands to show you a row of buttons representing the disk drives in your system. You can use these buttons to navigate from one drive to another.



By default, the command will produce a list of all disk drives in your system, but you can use the various optional parameters for the **DRIVEBUTTONS** argument to control which drives are shown, and also how they appear. You can also combine the command with some of the other [Go command](#) arguments to control how the generated buttons behave.

## 1. Controlling which drives are shown

The following parameters control which drives are shown in the generated list:

- **fixed**: Display fixed drives (hard disks).
- **network**: Display network drives.
- **cdrom**: Display CD/DVD drives.
- **removable**: Display removable drives (e.g. USB flash drives).
- **ramdisk**: Display RAM drives.
- **offline**: Only show offline (disconnected) network drives.
- **online**: Only show online (connected) network drives.
- **hideempty**: Hide removable drives that are empty (those that have no media or disk inserted).
- **+<letters>**: Only display the specified drives. For example, **+def** would only list drive letters D, E and F in the drop-down.
- **-<letters>**: Do not display the specified drives. For example, **-gz** would not display drives G or Z.

To use multiple parameters, separate them with a comma. For example, to display only removable and CD/DVD drives, excluding drive P, the command would be:

**Go DRIVEBUTTONS=removable,cdrom,-P**

To display only non-empty removable drives, the command would be:

**Go DRIVEBUTTONS=hideempty,removable**

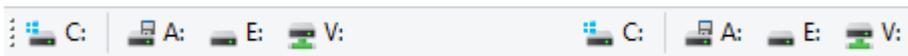
## 2. Controlling which file display the buttons apply to

In a dual-display Lister the generated drive buttons normally apply to the current source file display. You can change this by combining the **DRIVEBUTTONS** argument with other arguments to the [Go](#) command that affect which file display is used:

- **NEW**: The drive will be opened in a new Lister.
- **NEWTAB**: The drive will be opened in a new tab. You can combine this with the **OPENINDEST**, etc. arguments to control which file display the tab is opened in.

- **OPENINDEST**: The drive will be read into the destination file display. If the current Lister is not in dual-display mode, then could mean the folder is read into another Lister altogether.
- **OPENINDUAL**: The drive will be read into the destination file display in a dual-display Lister. The difference between this argument and **OPENINDEST** is that this will force a single-display Lister into dual-display mode if it is not in that mode already. The default layout (horizontal or vertical) will be used in this case.
- **OPENINLEFT**: Reads the drive into the left-hand (or top) file display in a dual-display Lister. In a single-display Lister, this argument has no effect (the drive will be read into the single display as normal).
- **OPENINRIGHT**: Reads the drive into the right-hand (or bottom) file display in a dual-display Lister. If the current Lister is not already in dual-display mode it will be set to that mode automatically. The default layout (horizontal or vertical) will be used in this case.
- **USEQUALKEYS**: Activates pre-configured behaviour for the main qualifier keys - holding the **Control** key will open the drive in the dual-display, **Shift** will open it in a new Lister and **Alt** will open it in a new tab.

Many people use these arguments to define two separate lists of drive buttons, one that always applies to the left-hand file display, and one for the right. For example, a toolbar with the following commands on it would list all fixed (internal) drives first, followed by all other drives, separately for both the left and right file displays. The buttons always apply to the left and right displays, irrespective of the current source and destination.



```
Go DRIVEBUTTONS=fixed OPENINLEFT
Go DRIVEBUTTONS=cdrom,network,removable,ramdisk OPENINLEFT
Go DRIVEBUTTONS=fixed OPENINRIGHT
Go DRIVEBUTTONS=cdrom,network,removable,ramdisk OPENINRIGHT
```

Note the use of a Spacer between the second and third buttons to cause the right-hand drive buttons to be right-aligned.

### 3. Controlling how a drive is opened when the button is clicked

The optional **multifunc** parameter for the **DRIVEBUTTONS** argument causes the generated drive buttons to be [multiple function buttons](#) (three-button buttons). Clicking them with the left mouse button will act as if **OPENINLEFT** were set, the right button will act as if **OPENINRIGHT** were set, and the middle mouse button will act as if **NEW** were set.

You can also use **multifunctabs** which is similar to **multifunc**, except the left and right mouse button functions will open a new tab on the appropriate side of the Lister.

For example,

## Go DRIVEBUTTONS=multifunctabs,removable

Additionally, the following [Go](#) command arguments can be used in conjunction with **DRIVEBUTTONS** to modify the behaviour of the generated buttons.

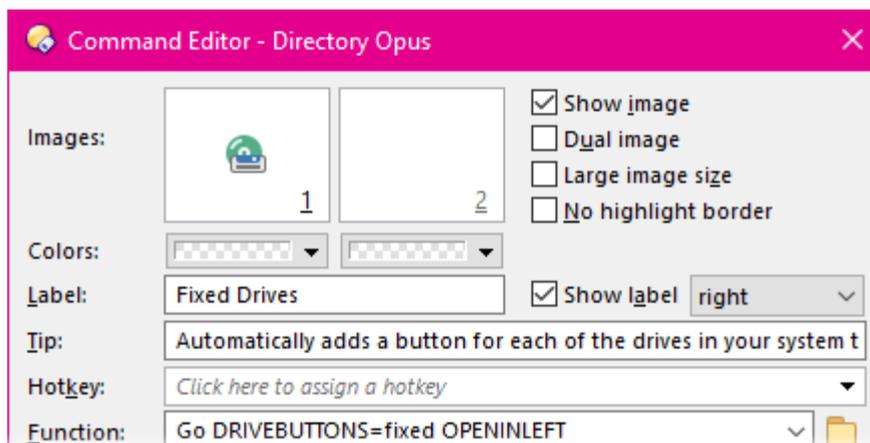
- **CURDIR**: The generated drive buttons will use "current directory" mode. This mode makes Opus navigate to the most recently accessed folder on the specified drive, rather than simply the root of the drive. (Exception: If you select the drive you are already on, you will be taken to the root.) Opus will remember the "current directory" for each drive in your system, even from one session to the next. The drive letter button representing the "current drive" will be highlighted. In this way you can click around from one drive to another remembering the previously used folder on each drive.

For example, **Go DRIVEBUTTONS=hideempty CURDIR**.

- **KEYARGS**: The **KEYARGS** argument can also be used in conjunction with **DRIVEBUTTONS** - see the description of the internal [Go](#) command for a discussion of this argument.

## 4. Controlling the appearance of the buttons

The generated drive buttons will inherit the appearance settings of the **Go DRIVEBUTTONS** button that generates them. Therefore, if you wish to control whether drive buttons display images, labels, have colors, etc, you must edit the parent button.



The drive buttons generated by the button shown above will display images (icons for each drive), and a label to the right of each icon. If, for example, you did not want to show the drive icons, you would turn off the **Show image** option.

If drive buttons are configured to show only icons and no labels, drive buttons will draw small drive letters over the icons themselves.



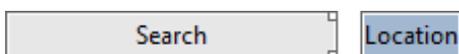
You can override this to force it on or off if you want.

You can use the following optional parameters for the **DRIVEBUTTONS** argument to control the appearance of the generated buttons:

- **iconletterson**: Force the display of generated drive letters (as shown above).
- **iconlettersoff**: Disable the automatic display of generated drive letters.
- **labels**: Displays the label of each drive. By default only each drive's letter is shown. For example, *Local Disc (C:)* instead of *C:*.
- **lettersbeforelabels**: When showing both drive letters and labels, the letters will be displayed first. Without this letters are shown following the labels. For example, *C: (Local Disc)* instead of *Local Disc (C:)*.
- **noletters**: When used with labels, prevents the display of the drive letters. For example, *Local Disc* instead of *C: (Local Disc)*.

## Field Buttons

Field buttons are another special type of button. They are used to place field controls (for example, the location fields) on your toolbars. In Customize mode they appear as a simple frame which you can resize by clicking and dragging the border.



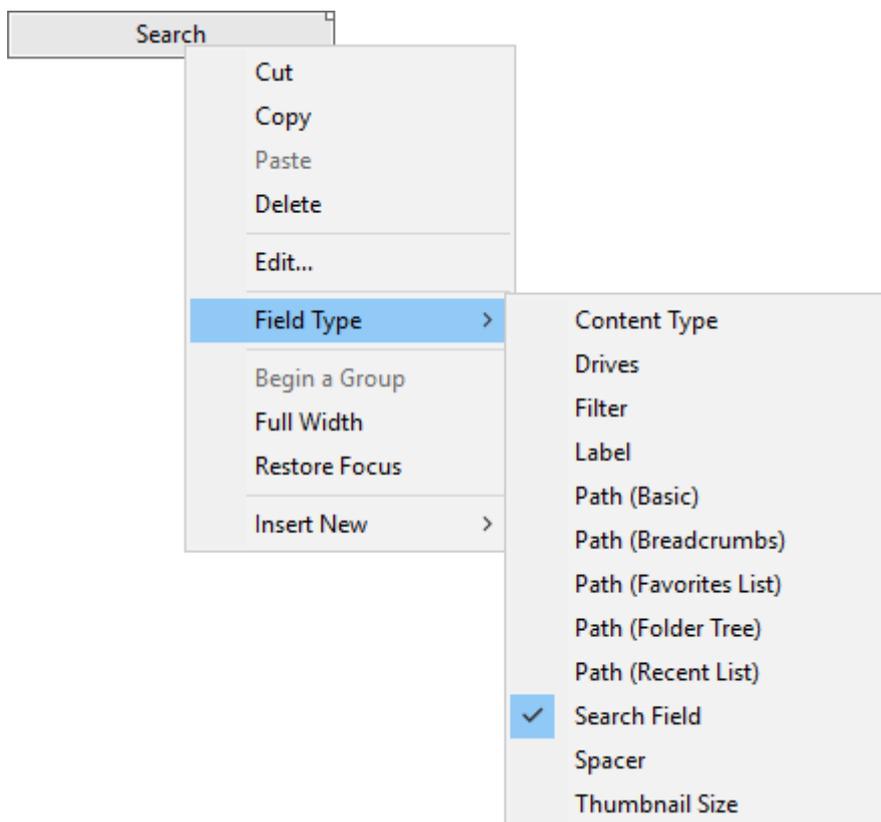
The above image shows the search field from the default Menu toolbar and the location field from the default File Display Border toolbar. You can see that the search field has grab handles at the right-hand edge, which let you resize the field. The location field does not, because it is set to *Full Width* mode, meaning it will expand to fill all available space on the toolbar.

Once we leave Customize mode you can see that the frames revert to fully functional fields:



Note that the compatibility files button disappeared when leaving Customize mode; this is because when the screenshot was taken the Lister was not currently displaying a folder that has compatibility files, and so the button was automatically hidden. You'll find that many buttons on Opus toolbars can change their state (checked or unchecked, enabled or disabled, hidden or shown) dynamically. If we had navigated to *C:\Program Files* before taking the screenshot the compatibility files button would have been visible.

Field buttons are the only type of button that you can't create manually (by starting with a new button). To add a field button to a toolbar you have to find its entry in the list on the **Commands** page and drag it to the toolbar. Once it is on the toolbar, however, you can change what type of field it is as well as configure certain properties by right-clicking it when in Customize mode.



The context menu for a field button lets you configure two options that only apply to fields:

- **Full Width:** If this option is enabled, the field will use all the available space on the toolbar (that is, any space that's not used by another button). If two or more fields on the same toolbar have this flag set, they will share the available space.

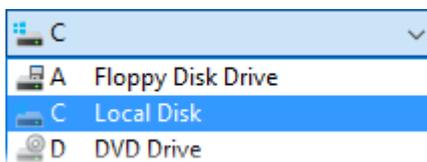
- **Restore Focus:** If this option is enabled, pressing the **Enter** key in the field will return focus automatically to the file display.

The **Field Type** sub-menu lets you switch between the various types of field:

- **Content Type:** If folder [content type detection](#) is enabled, displays the current content type format in use. You can also use the drop-down list to instantly switch to one of the defined content type formats.



- **Drives:** The [Drive List](#) field is drop-down list of your disk drives. Selecting a drive from the list navigates the source file display to that location.

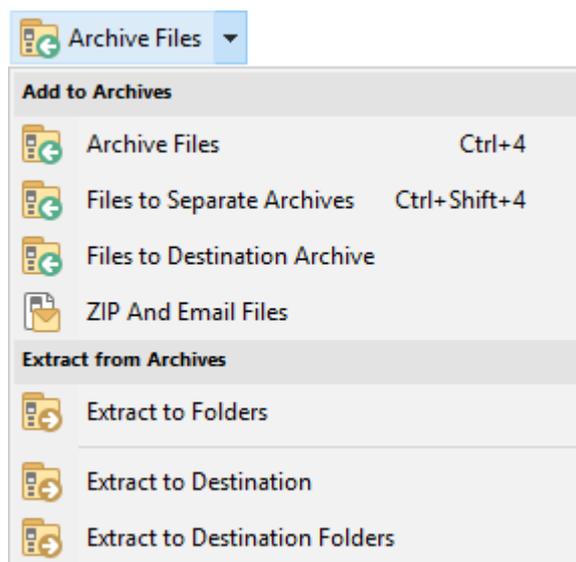


- **Filter:** The [filter field](#) lets you filter the items visible in the current folder by entering a [wildcard pattern](#).



- **Label:** A label field has no function other than to display a static label. You can use it to provide a label for other controls or just to display a static text string on a toolbar.

Label fields in drop-down menus are drawn in a bold font and left-aligned to stand out visually from the other items in the menu.



- **Path (Basic):** The basic path (location) field is a minimalist text field that displays the current path and lets you type a new path to navigate in the Lister. By default this field controls the current source file display, but you can [configure](#) it to specify the display it is linked to.

C:\

- **Path (Breadcrumbs):** The [breadcrumbs](#) path (location) field is a powerful navigation tool that provides instant access to prior locations in the current path. This field has [lots of configurable options](#) that can modify its appearance and behaviour.

This PC > C: > Local >

- **Path (Favorites List):** The favorites list path field combines the basic path field with a drop-down menu of your [favorite folders](#), which you can navigate to by selecting them from the drop-down. By default this field controls the current source file display, but you can [configure](#) it to specify the display it is linked to.

C:\

- **Path (Folder Tree):** The folder tree path field combines the basic path field with a drop-down list containing the folder tree hierarchy leading to the current location. This lets you navigate to previous folders in the tree from the drop-down. By default this field controls the current source file display, but you can [configure](#) it to specify the display it is linked to.

C:\

- **Path (Recent List):** The recent list path field combines the basic path field with a drop-down menu of your [recently visited folders](#). By default this field controls the current source file display, but you can [configure](#) it to specify the display it is linked to.

C:\

- **Search Field:** The search field lets you search the current folder using the indexed [Windows Search](#) system.

Search C:\

- **Spacer:** A spacer field has no function other than to take up room on the toolbar - it's used to right-justify toolbar elements.
- **Thumbnail Size:** The thumbnail size field lets you dynamically resize thumbnails in the current folder. It will be hidden when the Lister is not in thumbnail mode.

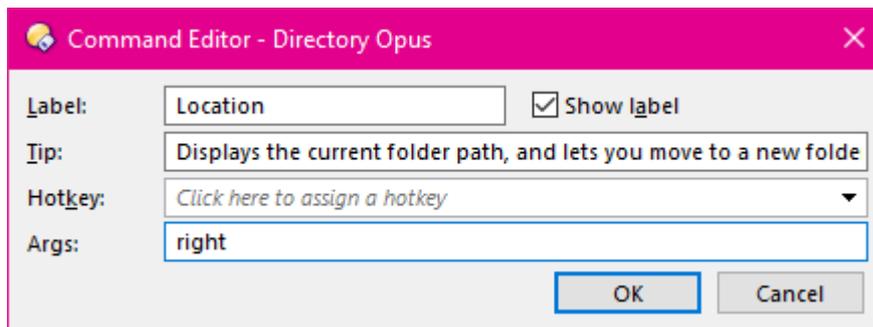


By default the slider will set the thumbnail size using the aspect ratio configured in Preferences, however if you edit the underlying command and add the keyword **preserve** to the *Args* field, it will preserve the current aspect ratio (this would only be useful if the aspect ratio has been changed manually using the **Set THUMBNAILSIZE** command).

See the [Editing The Toolbar](#) section for more information on how to edit toolbar fields.

## Path Field Configuration

The various types of [path fields](#) (basic, breadcrumbs, favorites, folder tree, recent) normally operate on the current [source file display](#) - in a dual-display Lister, the path shown in the field will update as you switch the source from left to right. If desired, these fields can all be configured to control a specific file display. For example, this allows you to have a dual-display Lister with two separate path fields, one for the left and one for the right. You can configure this behaviour by editing the button definition of the field in [Customize](#) mode.



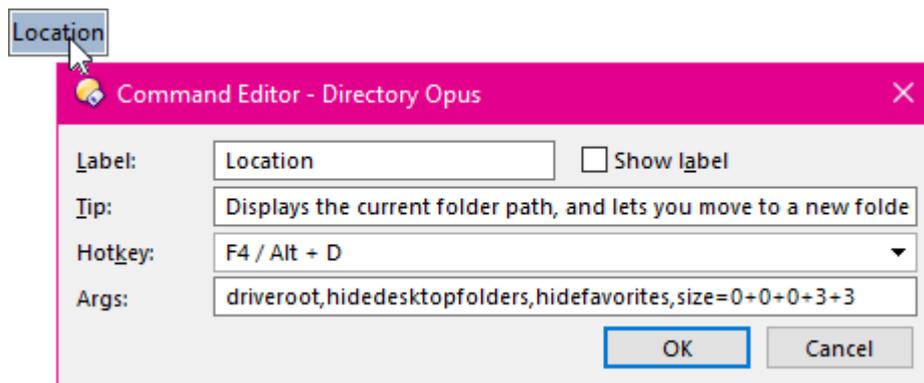
Select **Customize** from the **Settings** menu, and then right-click on the path field (it will have reverted to a simple frame - see the discussion on [Field Buttons](#) for more information on this) and choose **Edit**. The **Args** field in the command editor dialog is used to provide various keywords that define the behaviour of the field. The available keywords are:

- **left**: The path field will always control the left (or top) file display.
- **right**: The path field will always control the right (or bottom) file display.
- **dest**: The path field will always control the destination file display.
- **focus**: If the path field is configured to control a file display other than the source, adding the **focus** keyword as well will cause the focus to be set to that file display automatically (i.e. that file display will become the new source whenever the folder is changed).

Please note that the [breadcrumbs path field](#) has [lots of other configurable options](#) as well as the above keywords.

## Breadcrumbs Configuration

You can control the behaviour and appearance of [breadcrumbs location fields](#) by editing the button definition of the field in [Customize](#) mode.



Select **Customize** from the **Settings** menu, and then right-click on the breadcrumbs location field (it will have reverted to a simple frame - see the discussion on [Field Buttons](#) for more information on this) and choose **Edit**. The **Args** field in the command editor dialog is used to provide various comma-separated keywords that define the behaviour and appearance of the breadcrumbs field. The available keywords are:

- **dragnoreself**: Blocks drag & drop from the path field to itself. Turn this on to avoid accidents when dragging out of the path field and releasing the mouse button prematurely. Note that, even if this is on, you can still hover over the arrows between path components and drop into the menus that pop-up, should you need to. Whether or not this is on, dragging from one path field to another is always enabled, as is dragging from the file display or other elements to the path field.
- **dragsafetyoff**: Makes it possible to copy or move folders by dragging them from the breadcrumbs field without having to hold a qualifier key. Without this, the drag & drop action defaults to creating shortcuts to folders unless you explicitly override it by holding **Ctrl** to copy or **Shift** to move. The default mode is safer if you primarily drag from breadcrumbs to the folder tab bar to open new tabs for parent folders; if you accidentally drop the folder on the way then it will create a harmless shortcut rather than copy or move the whole parent folder structure.
- **drivelabel**: Displays the drive label along with the drive letter (e.g. *System (C:)* instead of *C:*).
- **driveroot**: Removes the *Computer* branch from the field. So for example, instead of the breadcrumbs trail *Desktop -> Computer -> C: -> Program Files*, you would get *Desktop -> C: -> Program Files*.
- **editend**: Moves the initial cursor position to the end of the path string, with none of it selected, when you edit the path. This makes it easier to add a subdirectory or modify the last component of the path. If this is not used, the whole path string will be selected when you start editing the path, making it easier to type a completely new path over the top of it. Whichever you use, you can always push **Ctrl+A** to select the whole path or **End** to go to the end.
- **hidedesktopfolders**: Any folders on the desktop will be omitted from the *Desktop* branch drop-down.
- **hideemptydrives**: Hides empty disk drives from the *Computer* branch drop-down. This overrides the **Show Empty disk drives** option on the [Folder Tree / Contents](#) page in Preferences.

- **hidefavorites:** Does not display your favorite folders in the *Desktop* branch drop-down. This overrides the **Show Favorites** option on the [Folder Tree / Contents](#) page in Preferences.
- **hidehidden:** Forces hidden folders (ones with the **H** attribute but not the **S** attribute) not to be included in the branch drop-down menus. This partially overrides the **Show hidden and system folders** option on the [Folder Tree / Contents](#) Preferences page and the **Global Hide Filter** settings on the [Folder Display](#) Preferences page.
- **hidesystem:** Forces protected operating system folders (ones with both the **H** and **S** attributes set) not to be included in the branch drop-down menus. This partially overrides the **Show hidden and system folders** option on the [Folder Tree / Contents](#) Preferences page and the **Global Hide Filter** settings on the [Folder Display](#) Preferences page.
- **hotkeymenu:** If there is a hotkey assigned to the Breadcrumbs control, pressing it will open the *Desktop* branch drop-down instead of giving focus to the edit field.
- **mycomputericon:** Displays the *Computer* branch as an icon instead of a label.
- **mycomputerfull:** Enable the display of non-drive folders in the drop-down for the *Computer* folder.
- **noghostpath:** Prevents the field from preserving "ghost" crumbs of your previous location when navigating up the folder tree.
- **noitalicghosts:** Displays ghost crumbs in a normal font rather than in italics.
- **noarchives:** Prevents archives being included in the drop-down menus as if they were folders.
- **nopreservepath:** Specify this keyword if you wish to keep ghost paths when going up a level but prevent them from remaining when you switch into a new directory that has the same folders below it.
- **noselectprevious:** Overrides the "Select previous folder when going up" Preferences option, acting as if it was always off when navigating to the parent of the current folder. (The folder may still be selected for other reasons. e.g. If you go into a folder using double-click and then go back to a cached view of its parent, the view will remember that the folder was selected by the double-click itself.)
- **notheme:** Modifies the look & feel of the breadcrumbs path field. Also slightly affects its height. The breadcrumbs themselves will be drawn without using the current system theme (Windows visual style), and the field background will not fade in and out when active or inactive. As an alternative to the **notheme** keyword, if you instead specify the **Toolbar and menu defaults / Field background** color under [Preferences / Display / Colors and Fonts](#), that will override much of the breadcrumb aesthetics while keeping certain elements of the visual style and re-coloring them according to the other color choices on the same Preferences page. That is often preferable to using **notheme** if you dislike the standard look & feel. If the **Field background** color is not specified, and **notheme** is in use, then the colors used to draw the field come from the **Office 2003-style** settings (a legacy option nowadays) on the [Toolbars / Appearance](#) Preferences page.
- **selectprevious:** Overrides the "Select previous folder when going up" Preferences option, acting as if it was always on when navigating to the parent of the current folder. In other words, if you click part of the path to go up a level, the folder you were in before clicking will be selected.
- **showemptydrives:** Shows empty disk drives in the *Computer* branch drop-down. This overrides the **Show Empty disk drives** option on the [Folder Tree / Contents](#) page in Preferences.

- **showfavorites:** Displays your favorite folders in the *Desktop* branch drop-down. This overrides the **Show Favorites** option on the [Folder Tree / Contents](#) page in Preferences.
- **showhidden:** Forces hidden folders (ones with the **H** attribute but not the **S** attribute) to be included in the branch drop-down menus. This partially overrides the **Show hidden and system folders** option on the [Folder Tree / Contents](#) Preferences page and the **Global Hide Filter** settings on the [Folder Display](#) Preferences page.
- **showsystem:** Forces protected operating system folders (ones with both the **H** and **S** attributes set) to be included in the branch drop-down menus. This partially overrides the **Show hidden and system folders** option on the [Folder Tree / Contents](#) Preferences page and the **Global Hide Filter** settings on the [Folder Display](#) Preferences page.
- **size:** This lets you configure the size of various elements of the breadcrumbs field. This keyword accepts up to five numeric values, each separated with a + sign.

The full template for this keyword is **size=<arrow width>+<folder left>+<folder right>+<text left>+<text right>**.

<arrow width> is the width of the drop-down button in each path part. <folder left> and <folder right> control the left and right spacing of the *Desktop* button (and the *Computer* branch if **mycomputericon** is specified), and <text left> and <text right> specify the spacing of the text buttons.

The default values if this keyword is not provided are **size=13+1+4+4+4**. You can specify **0** for a value to indicate the default setting.

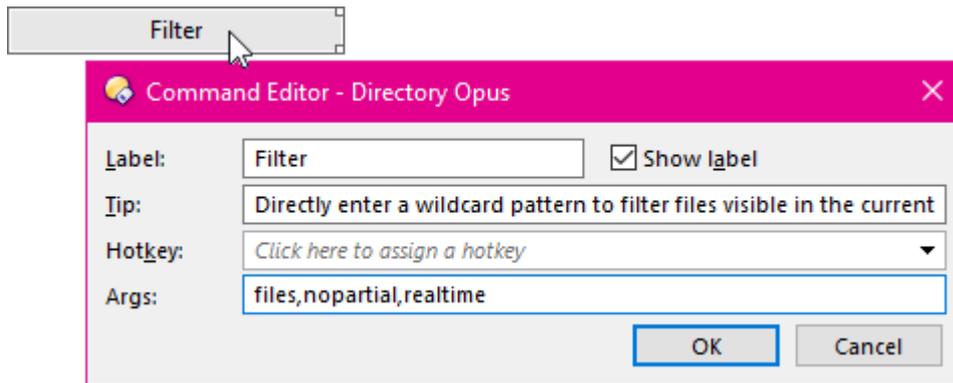
The supplied sizes will be automatically scaled to compensate for your system DPI settings. If you want to disable DPI scaling for a size, specify it as a negative value.

The breadcrumbs field also supports the [standard path field](#) keywords:

- **left:** The breadcrumbs field will always control the left (or top) file display.
- **right:** The breadcrumbs field will always control the right (or bottom) file display.
- **dest:** The breadcrumbs field will always control the destination file display.
- **focus:** If the breadcrumbs field is configured to control a file display other than the source, adding the **focus** keyword as well will cause the focus to be set to that file display automatically (i.e. that file display will become the new source whenever the folder is changed).

## Filter Field Configuration

Changes you make to the behaviour of a [filter field](#) via its drop-down menu are not permanent - if you close the Lister and open a new one, the field will be reset to its default settings. You can change these defaults by editing the filter field in [Customize](#) mode.

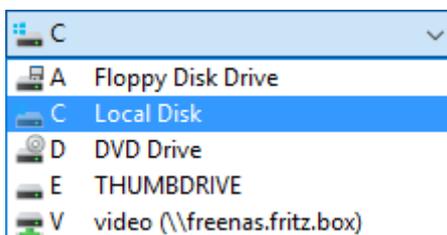


Select **Customize** from the **Settings** menu, and then right-click on the filter field (it will have reverted to a simple frame - see the discussion on [Field Buttons](#) for more information on this) and choose **Edit**. The **Args** field in the command editor dialog is used to provide various comma-separated keywords that define the default behaviour of the filter field. The available keywords are:

- **files**: Set the field to edit the Folder Options **Filename** filter (affects the **Show Filter** by default).
- **folders**: Set the field to edit the Folder Options **Folders** filter (affects the **Show Filter** by default).
- **both**: Set the field to edit both the Folder Options **Filename** and **Folders** filters.
- **hide**: In conjunction with any of the above three arguments, this will make the filter affect the Folder Options **Hide Filter** instead of the **Show Filter**.
- **nopartial**: Disables the **Partial Match** option.
- **norealtime**: Disables the **Filter in Real Time** option.
- **noautocontent**: Disables the **Auto-content** option.

### Drive List Configuration

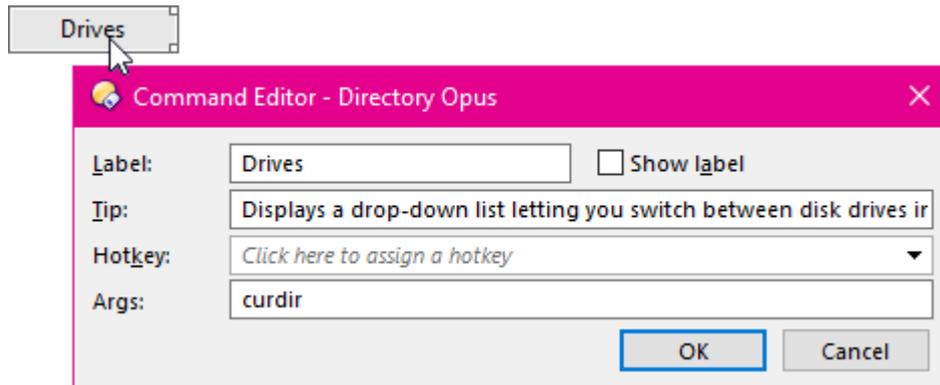
The **Drive List** field gives you a drop-down list of your drives directly on your toolbar. You can use this drop-down to switch from one drive to another.



When you create the **Drive List** field by dragging it to your toolbar, it is automatically assigned the **curdir** argument, which makes it operate in "current directory" mode. In this mode, Opus

will remember the last used directory on each drive, and if you pick a new drive through the drop-down you will be taken to the most recently used location. The alternative mode will simply read the root folder of the selected drive, with no "current directory" concept. You can select this behaviour by editing the field in [Customize](#) mode.

If you change the argument to **curdir,rootmode** then the control will work as a combination of both modes; selecting a new drive will take you to the current directory on that drive, but selecting it again (i.e. the drive you are already on) will take you to the root of the drive.



Select **Customize** from the **Settings** menu, and then right-click on the filter field (it will have reverted to a simple frame - see the discussion on [Field Buttons](#) for more information on this) and choose **Edit**, then remove the **curdir** keyword from the **Args** field and click **OK**.

In a dual-display Lister, the Drive List field normally applies to the current source file display. You can change this with the following argument keywords:

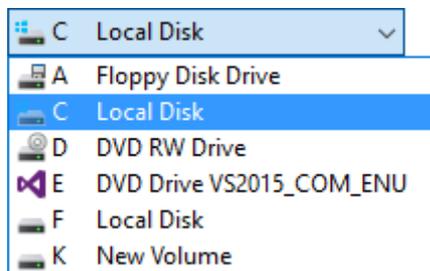
- **left:** The Drive List field will always apply to the left-hand file display of a dual-display Lister, rather than the source.
- **right:** The Drive List field will always apply to the right-hand file display, rather than the source.
- **dest:** The Drive List field will always apply to the destination file display, rather than the source.
- **nofocus:** Prevents the file display affected by the Drive List field from taking the focus (i.e. the source / destination state will remain unchanged).

The drop-down list normally shows all drives in your system, but you can also configure which drives or types of drive are displayed with the following arguments:

- **fixed:** Display fixed drives (hard disks).
- **network:** Display network drives.
- **cdrom:** Display CD/DVD drives.

- **removable**: Display removable drives (e.g. USB flash drives).
- **ramdisk**: Display RAM drives.
- **offline**: Only show offline (disconnected) network drives.
- **online**: Only show online (connected) network drives.
- **hideempty**: Hide removable drives that are empty (those that have no media or disk inserted).
- **+<letters>**: Only display the specified drives. For example, +def would only list drive letters D, E and F in the drop-down.
- **-<letters>**: Do not display the specified drives. For example, -gz would not display drives G or Z.

By default the drop-down list displays drive labels but the drop-down control itself doesn't (as shown at the top of this page). If you add the keyword **labels** to the button's **Args** field, the drive label will be shown in the drop-down control as well.



You can combine multiple argument keywords in the **Args** field by comma-separating them. For example:

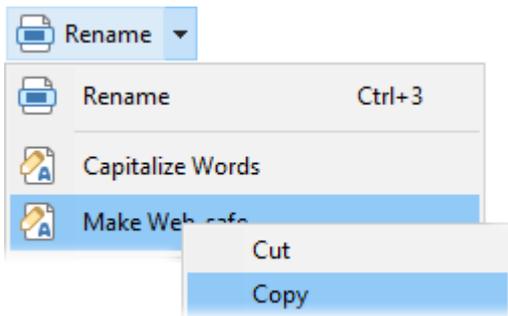
**left,fixed,cdrom,-e,hideempty** - control the left file display, only display fixed and CD/DVD drives, do NOT show drive E, and hide any empty drives.

**curdir,right** - control the right file display, and keep track of the current directory for each drive.

### ***Sharing functions with others***

Opus makes it easy to share buttons and toolbars with your friends and other Opus users. For example, you might have developed a drop-down menu containing lots of useful rename commands - if you want to share this with someone, the easiest way is to use the clipboard.

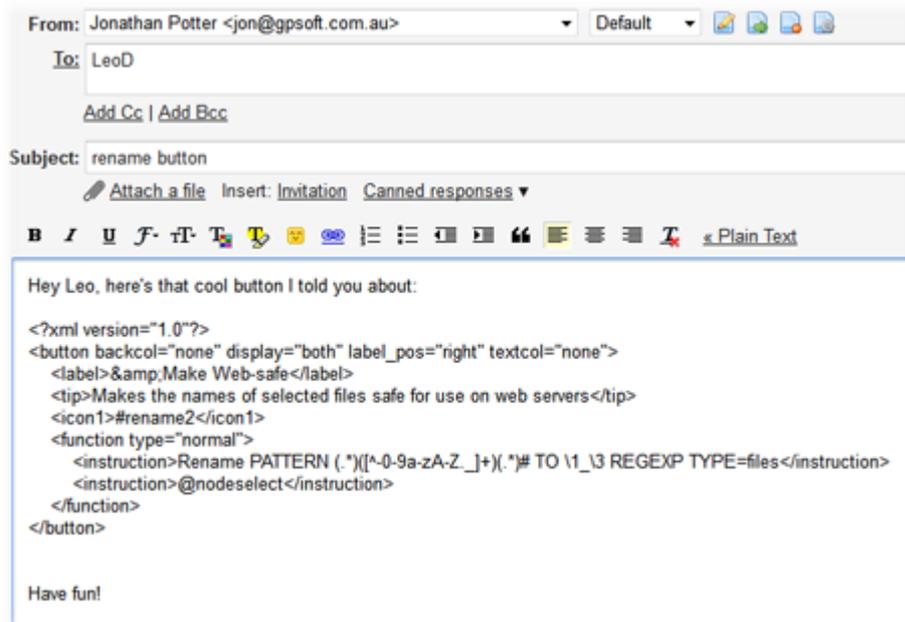
In [Customize](#) mode, right-click on the button or the menu that you want to share, and choose **Copy** from the context menu.



At this point, you can exit Customize mode. The button or menu you copied remains on the clipboard, and you now need to get it into a form that you can give to someone else. There are two easy ways to do this:

- Use a text editor, like Notepad, to paste the clipboard contents into a new document (or, say, into an email that you're writing to someone)
- Paste the clipboard contents directly into a Lister (**Edit / Paste**) to create a new text file

Either way, this will convert the definition of the button into a text-based XML format that can be easily shared with someone.



When your friend gets the text-based button code, all they have to do to add it to their own toolbar is:

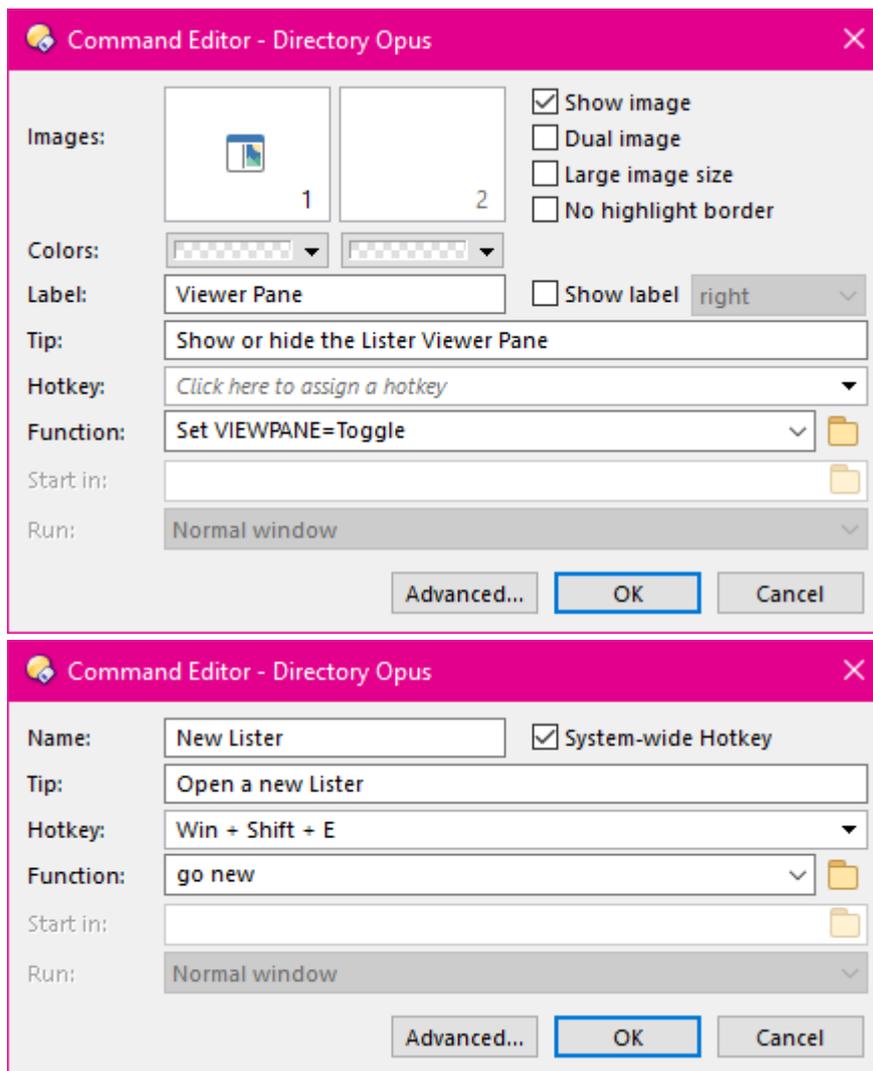
- Copy the button definition to the clipboard (by selecting all the text from the `<?xml version="1.0"?>` line down to the final `</button>` line and copying it to the clipboard).

- Set Opus into [Customize](#) mode.
- Paste the new button to their toolbar (right-click on the toolbar and choose **Paste** from the context menu - see [Editing the Toolbar](#) for more information).

You can copy an individual button, or a whole menu including all its child buttons (and any child menus it may have as well). You will often find people sharing toolbars and buttons on the Opus Resource Centre - see the [Buttons & Scripts](#) forum for lots of examples!

## Command Editor

The Command Editor is the main route to creating your own [buttons](#), [hotkeys](#), [User commands](#) and other functions in Opus. There are several variants of the command editor dialog used throughout the program, although they all have the same basic design.



This screenshot shows two variations of the command editor. The one on the left is the editor for a standard toolbar button - the function being edited is the **Viewer Pane** command from the default toolbar. The one on the right is the editor for the **New Lister** hotkey.

- **Images:** When the command is displayed in a toolbar or a drop-down menu, the image settings will be used to display an image for the command. Hotkeys do not have images as they are never physically displayed on screen.
  - **Images:** The two **Images** fields let you define up to two images for a button. The first image is the primary image; it is the one most often displayed. If the **Dual image** option is enabled, the second image field lets you define a secondary image that's displayed when the mouse is over the button.
  - **Show image:** Turn this on to enable the button's images. This may be overridden by settings defined for the toolbar - see below for more details.
  - **Dual image:** Turn this on to enable the secondary image when the mouse is over the button.
  - **Large image size:** Displays large images instead of small. This may be overridden by settings defined for the toolbar - see below for more details.
  - **No highlight border:** The button won't have a border (frame) when the mouse is over it.

- **Colors:** You can specify a text and background color for the button that will override the normal colors for the toolbar. Hotkeys do not have color fields.
- **Label:** When the **Show label** option is turned on, the button's label will be displayed in the toolbar or menu. If the button has an image as well you can choose the label's position relative to the image - for example, selecting *right* will position the label to the right of the image. This may be overridden by settings defined for the toolbar - see below for more details.

If a button's label has an ampersand (&) character before a letter, that letter will be marked as a type of hotkey for that button. When the button is on the top-level of a toolbar, pressing **Alt** plus that letter activates the button. For example, the **File** menu's label is actual set to *&File* - meaning you can press **Alt+F** to open the File menu. If you want to use a literal ampersand character in the label, you must use a double **&** instead (e.g. *Backup && Restore*).

- **Tip:** This is a description of the command that's displayed in a pop-up tooltip when you hover the mouse over the button. You can insert a line break in the tooltip (and therefore split it to multiple lines) using the `\n` sequence.
- **Hotkey:** This lets you assign a hotkey to the button. Pressing the assigned key will activate the button's function as if you had clicked it. The hotkey field is a special control designed to make it easy to enter a key combination. See the page on [Using the Hotkey Control](#) for more information.

There are two types of hotkeys; *local* and *global*. A hotkey you define for a button in a Lister toolbar or menu is a local hotkey, that works only when a Lister is the active window. Global hotkeys work anywhere in the system (as long as Opus is running). By definition, hotkeys defined for buttons in floating toolbars are global. The command editor for a hotkey function (that is, a function that exists solely as a hotkey, rather than a hotkey assigned to a toolbar button) lets you select the local or global state using the **System-wide Hotkey** option. See the [Hotkeys](#) page for more information on defining hotkeys.

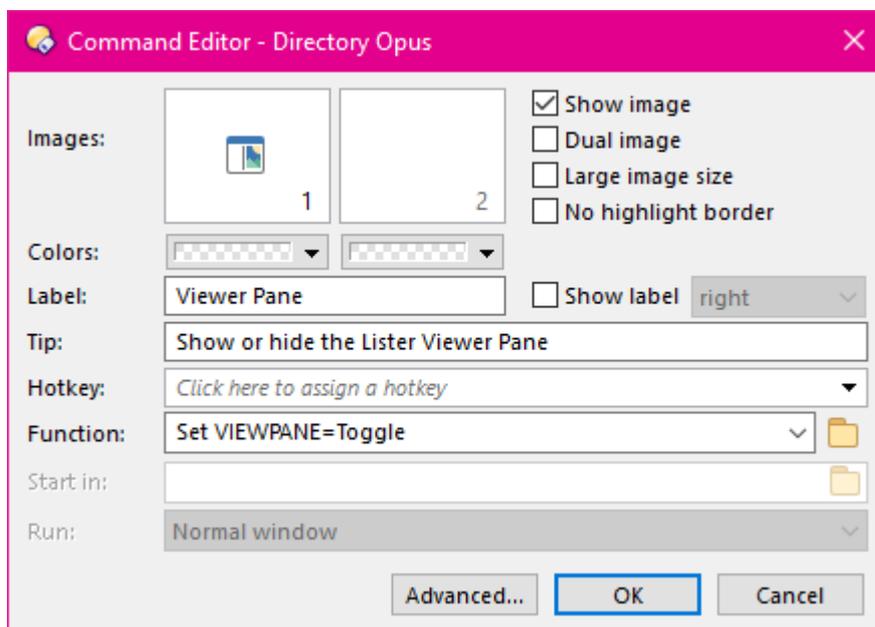
If you assign a hotkey to a button that opens a drop-down menu, pressing the hotkey will pop the menu open.

- **Function:** This is the actual function that the button will run when it's activated. The screenshot above shows the dialog in *simple* mode, where the function can only be a single line - there is also an *advanced* mode which lets you have multiple-line functions. The function can run an internal Opus command or an external program. See the pages on both [simple](#) and [advanced](#) mode for more information on defining the function.
- **Start in:** When a command runs an external program, this field specifies the folder that the program will start in (in other words, the new program's current directory). This field isn't used for internal Opus commands.
- **Run:** When running an external program this lets you specify how the program's window is to appear. You can choose from *Normal Window* (the program itself defines the window size), *Minimized* (the program will open minimized to the taskbar), *Maximized* (it will open full screen) and *Hidden* (it won't show a window at all). Not all programs will respect these settings. You need to be careful when using *Hidden* as it can result in the program running without any visible user interface - you should only use it when you know that's definitely what you want. It's most useful when launching a DOS script that runs and quickly exits, as it lets you hide the quick "flash" of the DOS command prompt.

Some of the options in the command editor can be overridden by the settings for the toolbar the button is in - the **Show image**, **Large image size** and **Show label** options can all be overridden by the options in the **Images & Labels** setting on the [Toolbars](#) tab in the [Customize](#) dialog.

The **Advanced** button at the bottom lets you switch from [simple](#) to [advanced](#) mode.

## Simple Command Editor



In the simple mode of the [Command Editor](#) (as shown above), the **Function** field is only a single line. In this mode you can specify the function (what the button actually does when you click it) in three different ways:

- Using the drop-down attached to the field, you can select a pre-defined command (as shown on the [Commands Tab](#) in the Customize dialog). When you select a pre-defined command from the drop-down, the function field will be populated with the underlying Opus internal command and arguments. For example, selecting **Viewer Pane** from the drop-down will fill the field in with **Set VIEWPANE=Toggle** as shown above.
- You can directly enter the name of an [Opus internal command](#), with its optional arguments.
- You can select an external program to run using the Browse button (📁). You can append various [external control codes](#) to the external program's command line in order to [pass filenames of selected files](#) to the program.

Function:

This would launch **Notepad**, passing it the name of the first selected file in the file display.

When you select an external program to run, the **Start in** and **Run** fields will be enabled.

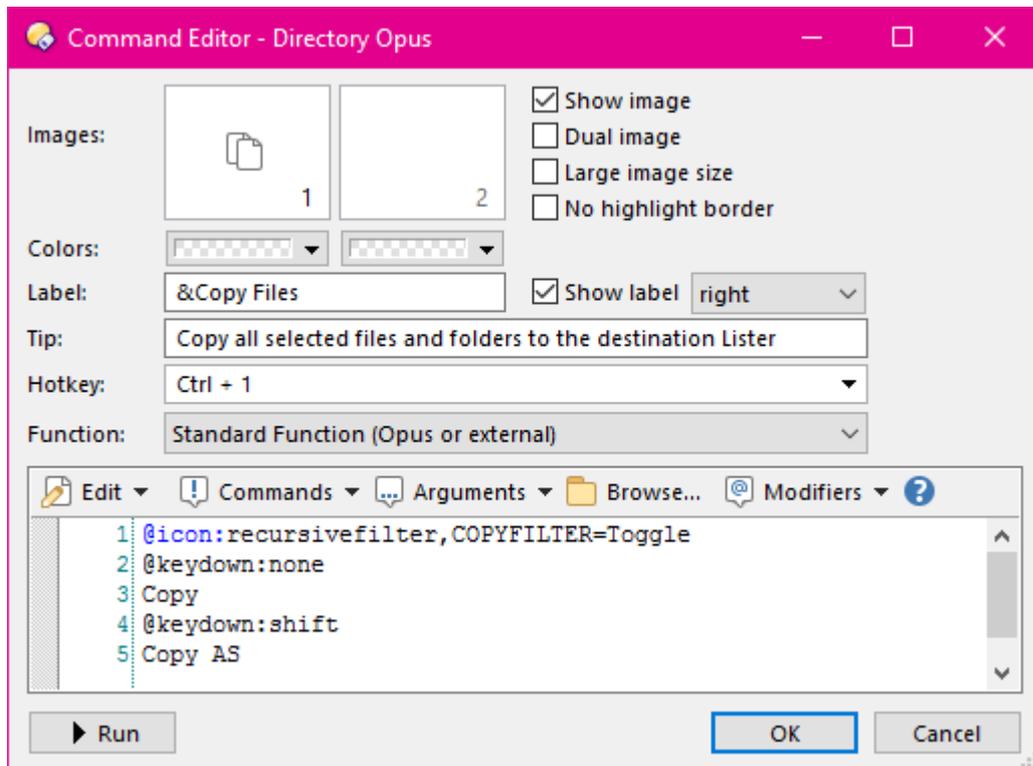
- **Start in:** When a command runs an external program, this field specifies the folder that the program will start in (in other words, the new program's current directory).
- **Run:** When running an external program this lets you specify how the program's window is to appear. You can choose from *Normal Window* (the program itself defines the window size), *Minimized* (the program will open minimized to the taskbar), *Maximized* (it will open full screen) and *Hidden* (it won't show a window at all). Not all programs will respect these settings. You need to be careful when using *Hidden* as it can result in the program running without any visible user interface - you should only use it when you know that's

definitely what you want. It's most useful when launching a DOS script that runs and quickly exits, as it lets you hide the quick "flash" of the DOS command prompt.

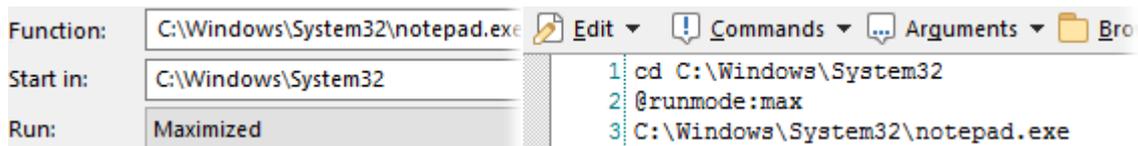
Click the **Advanced** button to switch to [advanced mode](#).

## Advanced Command Editor

The [simple mode](#) only lets you define functions that consist of a single line. From simple mode, click the **Advanced** button to switch to advanced mode, which lets you create more complex functions. The command editor will switch to advanced mode automatically if the button you are editing already contains a function that extends over more than one line. In the advanced mode the single line **Function** field is replaced by a multiple-line text box that lets you create more complicated, multiple-line functions.



This screenshot shows the command editor when it is in advanced mode (the function shown is from **Copy Files** button on the default toolbar). You can see that the single-line **Function** field has been replaced with a multiple-line field. There is a new drop-down which lets you select the type of the function (*Standard* or *MS-DOS Batch*). Also, the **Start in** and **Run** fields have disappeared - if they had been specified in simple mode, they will be replaced by equivalent commands in the multi-line function definition. (Not to be confused with the **Run** button, which is discussed below.)



In these screenshots you can see that the **Start in** field has been converted to a **cd** command (to set the **current directory**) and the **Run** setting has been converted to a **@runmode** modifier. There are a number of command modifiers that you can use for commands in advanced mode. Another one is shown above: in the **Copy File** function, the **@keydown** modifier is used to change the button's function based on whether the **Shift** key is held down or not. See the [Command modifiers](#) page for more information on modifiers.

The **Function** drop-down in advanced mode lets you select from three types of functions:

- **Standard Function:** This is used for most commands, and is the option you will most often select. Both internal Opus functions and the launching of regular external Windows programs work in this mode.
- [MS-DOS Batch Function:](#) This is used to run MS-DOS type programs (including **.bat** scripts). It's intended for programs that don't open a window of their own, but instead are designed to print text to a DOS prompt.
- [Script Function:](#) This type is used to define a button written using an ActiveX scripting language like VBScript or JScript. Script buttons normally begin with a **@script** directive to specify the script language, which is then followed by the actual script code. See the page on [Scripting](#) for more details on scripts.
- **WSL Script Function:** This option is available if you have WSL (Windows Subsystem for Linux) installed from the Windows Store under Windows 10. It lets you run a WSL (Bash) script from within Opus.

The multi-line text field is where you enter the instructions that make up the command. The toolbar along the top of the field contains a number of drop-downs that can help you build up the command:

- **Edit:** This dropdown contains a number of text editor-like functions for the edit field; clipboard operations, search and replace, etc.
- **Commands:** Displays a list of the [internal commands](#) (including any [user-defined commands](#)). Selecting one will insert the command into the function.
- **Arguments:** This is a context-sensitive drop-down that displays the arguments (if any) for the command that begins the line the cursor is on. So for instance, if you had selected **Copy** from the **Commands** drop-down, the **Arguments** drop-down would then show you the arguments for the **Copy** command. Again, selecting arguments from the drop-down menu inserts them into the body of the function. The **Arguments** drop-down also displays a list of the [external control codes](#) (which you can use with both internal commands and external programs to pass things like the names of selected files through to the command).
- **Browse:** This button displays a standard file browser dialog letting you locate an external program to run. The full path to the program will be inserted into the function.

- **Modifiers:** This displays a drop-down list (with descriptions) of the various [command modifiers](#) that you can use to change the behavior of commands.

Lines beginning with `//` are ignored, allowing you to put comments in your commands. The `//` must be at the very start of the line, with no spaces or anything else before it, for the line to become a comment.

The **Run** button at the bottom of the dialog allows you to instantly run your command or script to test it out without having to close the button editor. You can use the **F5** hotkey to do the same thing. (When working on something complex, it's still a good idea to save your work occasionally, just in case! Saving only happens when you click OK in the button editor and then OK again in the Customize dialog to exit Customize mode.)

Buttons run from the editor will be run against the source folder tab in the lister which launched the editor. If your button requires selected files, you will probably find it useful to add the [@nodeselect](#) modifier on a line near the top of the command to prevent the selection being cleared each time you test the button. Depending on the type of button, you may or may not want to leave the modifier there once you are finished. When writing a script, you can do a similar thing using the [Command](#) object's **deselect** property, and the default script you get when entering script mode shows you an example of how to use it.

Test-running a script will also cause an output panel to open at the bottom of the editor where any error messages or strings passed to the [DOpus.Output](#) method will appear. These will also appear in the Script Log if you have it turned on in the lister.

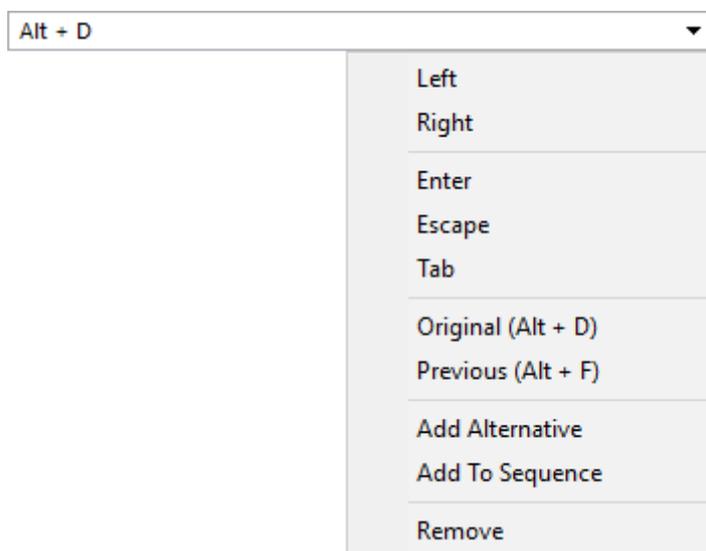
### ***Using the Hotkey Control***

The [Command Editor](#) and the Keys page in the [Customize](#) dialog both use a special field, the *Hotkey Control*, to make entering hotkey sequences easier.



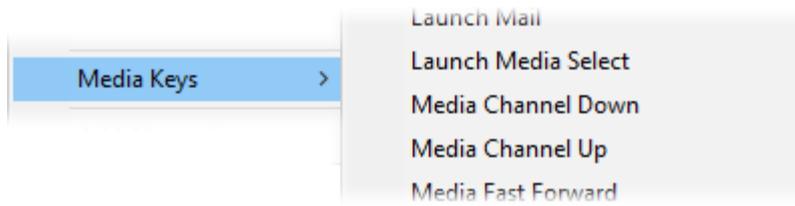
To assign a simple key press, simply click in it to activate it like a normal string field, and then press the desired key combination. For example, to assign **Alt + D** to a function, click in the field and press the **Alt** and **D** keys together.

If you click the drop-down arrow with the mouse, a menu is displayed with several commands relating to the hotkey.



- **Left / Right:** In a button located in the [File Display Toolbar](#) (as shown above), the **Left** and **Right** items let you assign a key that applies only to the left or right file display. See below for an example of how to use this.
- **Enter / Escape / Tab:** Allows you to assign a hotkey to those keys.
- **Original / Previous:** If you have changed the key, these two commands let you revert to the previous setting, and the original key assigned when the dialog first opened.
- **Add Alternative:** This command lets you add an alternative key sequence to the same command. For example, the default [location field](#) can be activated by pressing either **Alt + D** or the **F4** key. See below for an example of how to use this.
- **Add To Sequence:** This command lets you add an additional key to the current key sequence. You can use this to create hotkeys where you have to press two or more keys in sequence (rather than at the same time) to activate the function.
- **Remove:** This removes (clears out) the hotkey from the control.

When editing a hotkey function (as opposed to a toolbar button's hotkey), the drop-down will also let you select from a sub-menu of *Media keys*.



These are special input events that aren't associated with the traditional keys on the keyboard. For example, you could use this to assign a function to the *Play* button on your keyboard, or the *Back* button on your mouse.

## Assigning alternative key sequences to the same command

In the above screenshot, **Alt + D** is the only current key assignment. Selecting the **Add Alternative** command displays a + sign indicating that the next key press will add an alternative sequence.



If you then press another key, it is added as the beginning of an alternative sequence. For example, if you press **F4**:



Pressing either **Alt + D** or **F4** would now activate the function. You can add as many alternate key sequences as you like. When a control with alternate key sequences is active, it displays a separate drop-down listing each alternate sequence separately.



The sequence shown in the main edit field (**F4** in the above image) is the *current sequence*, and is the only one that can be edited. For example, if you pressed **F3** at this point, **F4** would be replaced by **F3** but the **Alt + D** sequence would be unaffected. If you want to edit a sequence other than the current one you can remove it by clicking its **Remove** link and add it back again.

## Creating multiple key sequences

Depending on how much you use hotkeys, you may find yourself running out of keys on the keyboard to assign functions to. Multiple key sequences can let you create additional hotkeys that require two or more keys to be pressed in sequence to activate the function, which can dramatically increase the number of hotkey functions you can define.

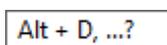
The procedure for adding a key to the sequence is similar to adding an alternative. Select the **Add To Sequence** command, which will display a **&** sign indicating that the next key press will add to the current sequence.



Press the key you want to add to the sequence, and the control will update to show the new assignment. For example, if you press **1** at this point:



To activate this hotkey you would now need to press **Alt + D** followed by **1**. Different hotkeys can use the same keys up until the last one in the sequence. For example, you could therefore create multiple hotkeys that all start with **Alt + D**, followed by a different number (e.g. **Alt + D** followed by **1** could run one function, whereas **Alt + D** followed by **2** could run another). You can add as many keys as you like to a sequence. In the Lister, when you press a key that matches the start of a multi-key sequence, Opus displays a small popup in the bottom-left corner as a visual cue that you need to press another key to run a function.



## Differentiating the left and right file display toolbars

The [File Display Toolbar](#) is handled slightly differently to all other toolbars, because in [dual-display mode](#) there are two copies of it. Normally, pressing a hotkey bound to a button in the file display toolbar will activate the function in the source file display. However, you can assign

hotkeys that will specifically activate the function in the left (or top) or the right (or bottom) file display, irrespective of which is the source. For example, you could activate the left file display's location field by pressing **Shift + F4**, and the right file display's location field by pressing **Ctrl + F4**.

Start with the first key sequence that you want to assign:

Then from the drop-down menu select the **Left** command. This will bind that key sequence to the left file display.

Then, follow the steps above to add an alternative key sequence for the right file display. Select the **Add Alternative** command from the drop-down:

 +

And press the key for the right file display:

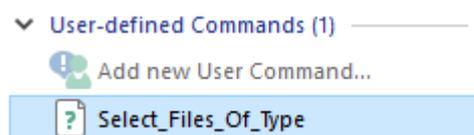
  
 [Remove](#)

Finally, select the **Right** command from the drop-down menu to bind the new sequence to the right file display.

## User-defined Commands

User-defined commands are pre-defined commands that you create yourself. Effectively it's like creating a toolbar button with your own function on it, but instead of the button living on a toolbar, it lives in this command list and other buttons or menus can refer to it by name. User-defined commands are created via the **Commands** tab of the **Customize** dialog, using the *Add new User Command* option in the *User-defined Commands* category.

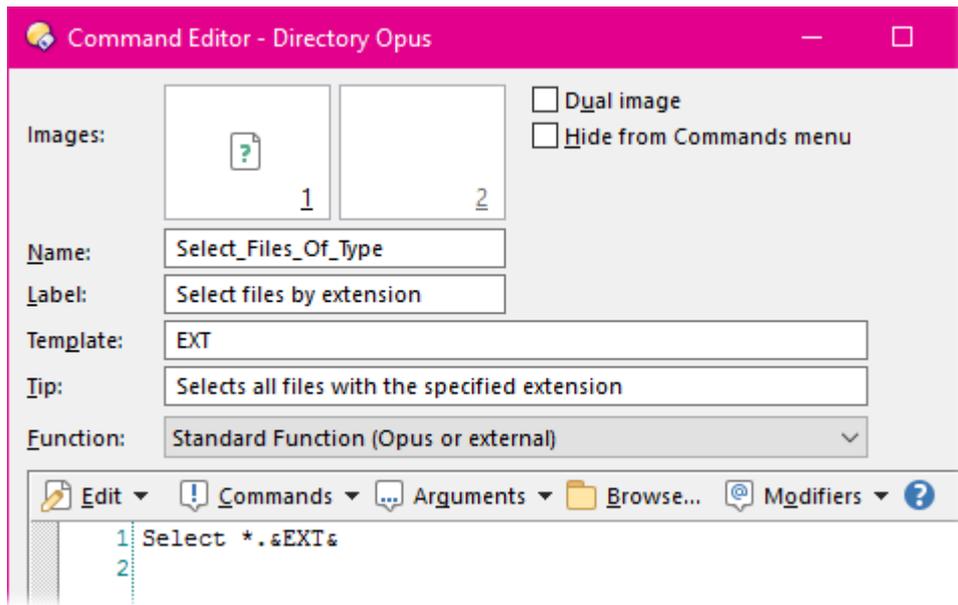
When you create a user command you give it a name (which must be unique and not clash with any of the Opus commands), and other buttons can then run your user command using its name. User commands can also accept parameters (arguments) on the command line, which lets you pass through parameters to external programs or scripts that your user command might invoke.



To create a new user command, double-click the *Add new User Command* item in the list, or right-click it or the *User-defined Commands* category header and choose **New** from the context menu. The context menu is the main way to perform other actions on user commands. If you right-click an existing user command, the commands in the context menu are:

- **Edit:** Edit the definition of the user command. You can also open the editor by double-clicking the command.
- **Duplicate:** Make a duplicate of the user command.
- **Copy:** Copy the command to the clipboard. You can paste it over an existing command (to replace its definition) or onto the *Add new* item to make a new copy of the command.
- **Export:** Export the command to a text file. This lets you share user commands with others. To import a user command use the Import command in the File menu.
- **Delete:** Delete the command.

The screenshot above shows an example of a user command in the list. As you can see, the name of the command is **Select\_Files\_Of\_Type** (command names cannot contain spaces; any spaces are automatically converted to an underscore). Double-clicking that command to open the editor reveals the following configuration:



The user command editor is based on the standard command editor so we won't document fully how to use it here - instead see the [Command Editor](#) documentation for a full description. The principle elements of the user command are:

- **Images:** Defining an image for the user command sets its default image - the one that is used when it's dragged from the Customize dialog to a toolbar. This can be changed in the button after it's created.
- **Name:** The name of the user command. This is the name that other functions must use to invoke your command. If you change this and you have functions already set up to use your command, they'll stop working unless you update them with the new name.
- **Label:** A string that defines the default label for the button created when the command is dragged to a toolbar. This can be changed in the button after it's created.
- **Template:** This specifies the command line argument template for this command, if any is desired. See below for a discussion on the template.
- **Tip:** A string that defines the command's description. This is shown in the Customize editor when your command is selected, and is also used as the default tooltip when you drag the command to a toolbar to create a button.
- **Function:** Defines the function that this user command runs when it's invoked.
- **Hide from Command List:** If this option is selected, the user command will not be displayed in the drop-down command list shown in the command editor (the one displayed when you click the **Commands** drop-down in the above screenshot). This lets you create user commands that you can still use in buttons and hotkeys, but they won't clutter up the command list.

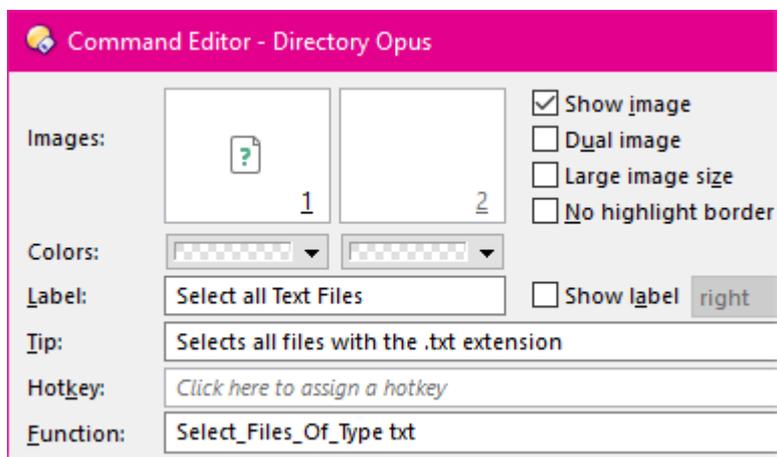
For Windows 7 users, the [Jump List](#) Preferences page lets you select individual user-defined commands that can be added to the Jump List menu. The **Label** and **Image** defined for the command will be shown in the Jump List.

When I created this user command I wanted a function that would select all files with a given file extension; therefore I needed to use the template field to specify a command line argument that is passed through to the command. This uses the [same syntax](#) as for internal Opus commands. In

the above example, we only want a single string value. The argument has been given the name **EXT** in the template field. This is not the *value* of the argument (that doesn't exist until something uses this user command), but the *name* of the argument.

The function for the user command, **Select \*.&EXT&**, calls the internal **Select** command, passing it a wildcard string that is built from the supplied **EXT** parameter. The argument name is supplied in the function definition surrounded by ampersands (**&**) which indicates to Opus that the value of that argument is to be inserted in the command line.

So how do we actually use this user command? Well, because it requires a command line argument to do anything, we need to put it on a button or menu (or hotkey) and then edit the function to supply the desired extension. The following button definition would do this; it will select all **.txt** files in the current file display.



As you can see, we have edited the label and tooltip (description) to one more suited to the actual button we created. The function defined for this button calls our new **Select\_Files\_Of\_Type** user command, passing it the argument **txt**. The user command takes the argument and passes it on the command line as described above to the **Select** command, which then performs the desired action by selecting all files that end with a **.txt** extension.

In the above example, **EXT** is a string argument and so the value supplied is passed through unchanged to the user command, but for Boolean options the behaviour is different. For example, consider the following user command:

```
Name:      ExampleCmd
Template:   EXAMPLE1/S,EXAMPLE2/O
Function:   C:\DummyProgram.exe &EXAMPLE1& &EXAMPLE2&
```

The **ExampleCmd** user command will run the *C:\DummyProgram.exe* program when it's invoked. Its template has two Boolean options, **EXAMPLE1** (a straight switch argument, either

on or off), and **EXAMPLE2** (an option switch argument, which can be off, on or have a string value supplied). By default, a switch argument will insert the value **1** when it is set, and **0** when it isn't. For example, the following use of the user command would produce the following command line for *DummyProgram*:

```
Command: ExampleCmd EXAMPLE1
Runs:    C:\DummyProgram.exe 1 0
```

However, you can use the **&..&** insert to specify the actual strings that are passed through for that argument. For example:

```
Function: C:\DummyProgram.exe &EXAMPLE1:yes:no& &EXAMPLE2:one:two&
Command: ExampleCmd EXAMPLE2
Runs:    C:\DummyProgram.exe no one
```

For option switch (**/O**) arguments this only applies if the argument has been used as a switch, and not to provide a string value - if a string value is supplied instead, it is passed through unchanged. For example:

```
Command: ExampleCmd EXAMPLE1 EXAMPLE2=test
Runs:    C:\DummyProgram.exe yes test
```

Your template can also specify a list of values for an option switch which will then be shown in the drop-down list in the [advanced command editor](#), making it easy to pick the value for the argument. If you use this, you can also specify a default value, which will be passed through if a value is not given for the option. Consider the following template and the two example uses:

```
Name:      ExampleCmd
Template:  EXAMPLE/O[<default>,one,two,three]
Function:  C:\DummyProgram.exe &EXAMPLE&
```

```
Command: ExampleCmd EXAMPLE
Runs:    C:\DummyProgram.exe default
```

```
Command: ExampleCmd EXAMPLE=two
Runs:    C:\DummyProgram.exe two
```

## Synchronous and Asynchronous functions

External programs launched from toolbar buttons and hotkeys can be either synchronous or asynchronous.

- Synchronous in this context means that:
  - Functions that contain multiple commands will run those commands one at a time. Opus will wait for each command to finish before running the next.
  - When a command makes use of selected files, and more than one file is selected, the files will be processed one at a time.
- Asynchronous is the reverse:
  - A function that contains multiple commands will run the commands simultaneously (or at least, will not wait for one command to finish before running the next).
  - When multiple files are selected, Opus will not wait for the first file to be processed before moving on to the next.

As a simple example of the difference, take a command that runs Notepad and passes it the name of the selected item:

**notepad.exe {f}**

If only one file were selected when you ran this command, Notepad would open showing the selected file, and that would be that. If, however, three files were selected when you ran the command, the behaviour would vary:

- If run synchronously, Notepad would open showing the first file. When you close that instance of Notepad, another one would immediately open showing the second file. Closing the second Notepad window would open a third instance, showing the third file.
- If run asynchronously, you would get three copies of Notepad opening simultaneously, each one showing a different file.

By default, a function that contains a single command will run asynchronously, and functions that contain two or more commands will run synchronously. You can override this in several ways:

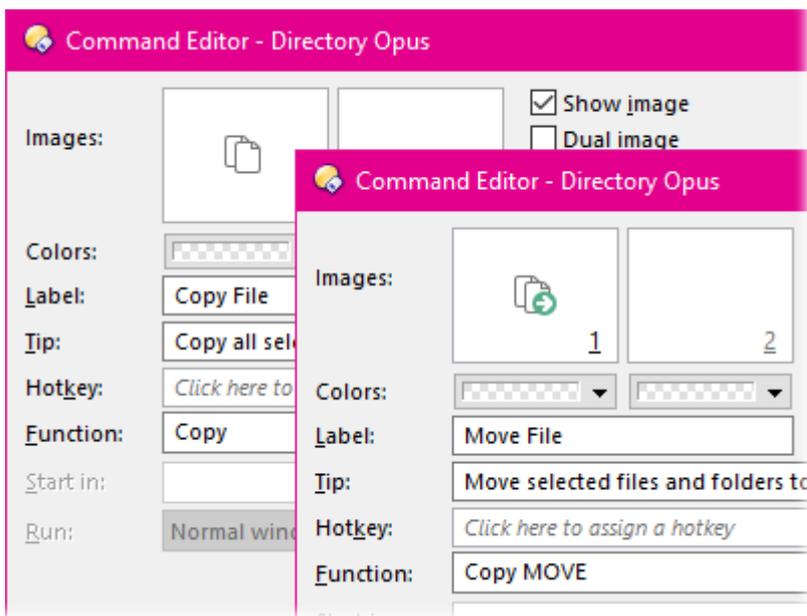
- Using the [@asynccommand modifier](#), you can force a command to run asynchronously.
- Using the [@synccommand modifier](#), you can force a command to run synchronously.
- You can set the **function\_default\_async** flag on the [Miscellaneous / Advanced](#) page in Preferences to *True* to make all functions (single or multiple commands) run asynchronously by default.

Internal commands always run synchronously - Opus will always wait for an internal command in a function to finish before moving on to the next instruction. If you want to run an internal command asynchronously for some reason, you can run it using the [DOpusRT](#) command (i.e. **dopusrt /cmd**) - it then behaves as an external program, and will respect the **@sync** and **@async** modifiers.

## Internal Command Arguments

The internal command set consists of around thirty commands, each of which can accept one or more arguments that modify its behaviour.

As an example, the internal **Copy** command's default behaviour is to copy selected files from the source to the destination folder. If you modify its behaviour by specifying the **MOVE** argument, the command will move selected files rather than copy them.



There are three main types of arguments:

- *Switch arguments* act as "on or off" options. For these arguments, the mere presence of the argument name on the command line is enough to activate the behaviour associated with that argument.

The **MOVE** argument to the **Copy** command is an example of a switch argument. So to change a copy button to a move button, the command would be **Copy MOVE**.

- *Value arguments* are arguments that supply a value to the command. The argument name by itself has no effect - instead, it is used to provide a user-defined value for that argument.

For example, the **COPYATTR** argument to the **Copy** command can accept two values - **yes** or **no**. Specifying the command **Copy COPYATTR** would not be sufficient; instead, you would need to specify **Copy COPYATTR=yes** if you wanted to force the copying of file attributes.

- *Optional arguments* combine the behaviour of switch and value arguments. An optional argument can appear by itself, to activate the default behaviour of that argument, or it can be used to provide a value when appropriate.

For example, the command **Copy ADDTOARCHIVE** defaults to creating a Zip archive. In this instance, **ADDTOARCHIVE** is acting like a switch. If you want to override the default behaviour, and create a 7-Zip archive instead, the command would be **Copy ADDTOARCHIVE=.7z**.

Each command has what's called a *template* - a full listing of all of the arguments the command accepts. The template marks each argument with one or more qualifiers to indicate what type of argument it is. You never actually type these qualifiers - they are not part of the argument name, they are merely a clue as to the type of the argument. For example, let's look at the beginning of the template for the **Copy** command.

```
ADDTOARCHIVE=ADDTOZIP/O[<default>,fullpaths,nofullpaths],
ARCHIVE=ZIP/O[<all>,single,keepfolder], AS/O, BUFSIZE/K/N,
BURNCD/S, CLEARREADONLY/K[yes,no]
```

This is not the whole template, just the first few arguments as an example. Let's take them one at a time:

- **ADDTOARCHIVE** is an optional argument (indicated by the **/O** qualifier). For historical reasons, some arguments have synonyms - **ADDTOZIP** is a synonym for this argument. You can use either term interchangeably, although this help file will always refer to the primary argument name. As the template indicates, **ADDTOARCHIVE** can be used by itself, as a switch, and it can also accept a value that modifies its behaviour. The two keywords that it can accept are **fullpaths** and **nofullpaths**. If you read the full description of the [Copy](#) command you'll see that you can also provide an archive type to control the default archive format created by this command. Values for this argument can be combined by separating them with a comma - for example, **Copy ADDTOARCHIVE .zip,fullpaths**.
- **ARCHIVE** is also an optional argument, and also has a synonym (**ZIP**).
- **AS** is an optional argument, but no specific keywords are shown in the template. Instead, the value for this argument (if provided) is completely user-defined. It is used to provide a new name or wildcard pattern when copying files. When used without a value (**Copy AS**), it acts as a switch causing Opus to prompt you for a new name for each file. When used with a value (e.g. **Copy AS \*.bak**) it lets you provide the new name on the command line, and Opus will not prompt for it.
- **BUFSIZE** is a value argument (indicated by the **/K** qualifier - "K" for "keyword"). The argument name means nothing by itself - it must be followed by a value to have any effect. In this instance, the argument is used to override the default buffer size when copying files. The **/N** qualifier indicates that the value provided must be a number. For example, **Copy BUFSIZE 128000**.
- **BURNCD** is a switch argument (indicated by the **/S** qualifier). This argument is used to initiate the burning of a CD or DVD using the Windows staged disc burning system. The argument takes no value, simply specifying the keyword is enough to activate its behaviour. For example, **Copy BURNCD**.

- **CLEARREADONLY** is a value argument, and the template indicates that it only accepts one of two specific keywords - **yes** or **no**. In this particular instance, the argument is used to override the default setting of a Preferences option. If the argument value is given as **yes** (i.e. **Copy CLEARREADONLY=yes**), read-only attributes are cleared when copying files from a CD, and if given as **no**, the read-only attributes are not cleared. If the argument is not provided at all, the current setting of the Preferences flag is used instead.

The qualifiers that you will see in the command templates are as follows. Remember that you **never** type the qualifiers when using arguments - they are merely a clue as to the argument type.

Qualifier	Type	Description
/S	Switch	Indicates a switch argument (a Boolean option that can either be on or off).
/K	Keyword	Indicates a value argument (a value must be provided following the argument keyword).
/O	Optional	Indicates an optional argument (can be used either by itself, as a switch, or with a following value).
/N	Numeric	The value of the argument must be a number.
/M	Multiple	The argument can accept multiple values (e.g. a list of files; see below).
/R	Raw	The argument accepts a "raw" value. For these arguments, the rest of the command line following the argument name is taken as the value.  Arguments of this type are the only ones that do not require quotes around values which contain spaces.

Some commands also have an argument that accepts a value, but is not marked with either /K or /O. For example, the **FILE** argument for the **Copy** command lets you specify the file or files to copy - and so clearly it takes a value, but it is not marked as such with the /K qualifier. This is a special type of argument - the *default argument*. In these cases, the use of the argument name itself is optional - you can provide it if desired, but you can also leave it out. For example, **Copy FILE C:\foo.txt TO D:\** is equivalent to **Copy C:\foo.txt TO D:\**. A command can have at most one default argument.

At this point we should discuss the issue of command parsing, spaces, and values. When Opus parses a command line, it uses the template for the command in question to identify the various arguments that you have provided. Take the following example command:

## Copy AS My Zip File.zip

The intention of this command is clearly to use the **AS** argument to provide a new name (*My Zip File.zip*) for the copied file. The problem that this command line introduces for the command parser is that **ZIP** is also a valid argument for the **Copy** command. The embedded spaces in the filename will confuse the command parser - it isn't able to tell that you intended the value of the **AS** argument to be "My Zip File.zip", and instead it will see the following command:

## Copy AS My ZIP File.zip

Opus will read the command as containing two arguments, **AS** (with a value of "My") and **ZIP** (with a value of "File.zip"). Obviously interpreting the command in this way means it will not behave as intended. To avoid this confusion, whenever you provide a value that contains an embedded space, you **must** enclose it in quotation marks. The correct form of the command above is:

## Copy AS "My Zip File.zip"

This is unambiguous as far as the command parser is concerned - it will see only one argument in the command (**AS**) and its value will be "My Zip File.zip", as intended.

An equals sign is generally optional when providing a value for an argument. That is, **Copy AS "My Zip File.zip"** is equivalent to **Copy AS="My Zip File.zip"**. There are two cases however where one form or the other must be used:

- When the value you are providing is also another argument for the command, you **must** use the equals sign. For example, **Copy CREATEFOLDER sendmail** would not behave as intended (copying selected files into a new folder called "sendmail") because **SENDMAIL** is also an argument for the command. Instead, you must provide the equals sign, as in **Copy CREATEFOLDER=sendmail**.
- When you are providing multiple values for a **/M** argument, the equals sign must not be used.

Some arguments accept multiple values. For example, the **FILE** argument to the **Copy** command is marked as **/M**, indicating that you can provide one or more values for the argument. When you do provide multiple values, each value must be separated by spaces. For example:

**Copy FILE D:\data\pic\*.jpg "J:\image files\pic\*.jpg" TO "E:\image store"**

This command has two values for the **FILE** argument - **D:\data\pic\*.jpg** and **"J:\image files\pic\*.jpg"** (note the quotation marks surrounding the second value - as described above, values containing embedded spaces must be quoted).

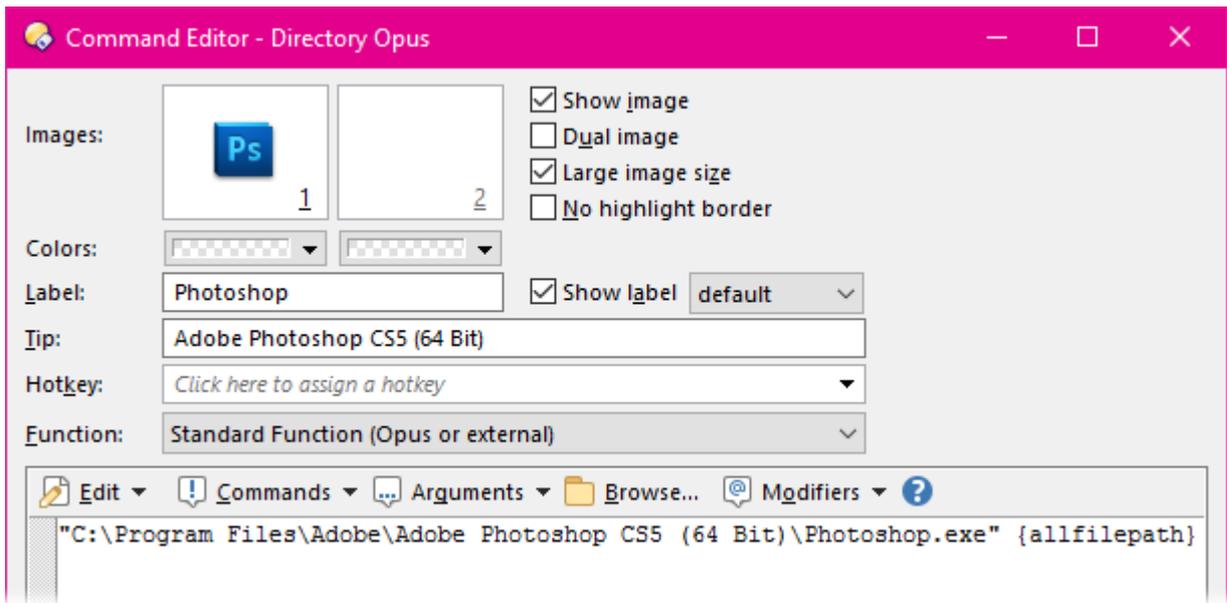
Just to confuse things further, some value arguments can accept multiple keywords as their value. You should not confuse these with /M arguments, even though it would be easy to do! For example, the **Set VIEW** command is used to change view mode (e.g. **Set VIEW=details**), but it can also be used to toggle between two different view modes (e.g. **Set VIEW=details,power**). In this case, the single argument (**VIEW**) is taking a single value (**details,power**) and it is the command itself that interprets this as multiple keywords. **VIEW** is not marked as /M and therefore the command format **Set VIEW details power** would not work.

The distinction is subtle and really you don't need to worry about it - each argument is documented as to what sort of values it can accept. It's just worth being aware of the difference.

See the documentation for the [individual commands](#) for a full list of their arguments, as well as a description and example of how each argument is used. In many cases multiple arguments can be given at once and the command documentation also describes when and where this would be appropriate.

## Passing files to external programs

Using Opus as a simple program launcher is nice, but even more useful is the ability to pass the names of files through to programs that you launch. For example, you could configure a button to automatically run Photoshop, passing the names of selected files on the command line. If you then select six image files and click your Photoshop button, they would all open in Photoshop automatically.



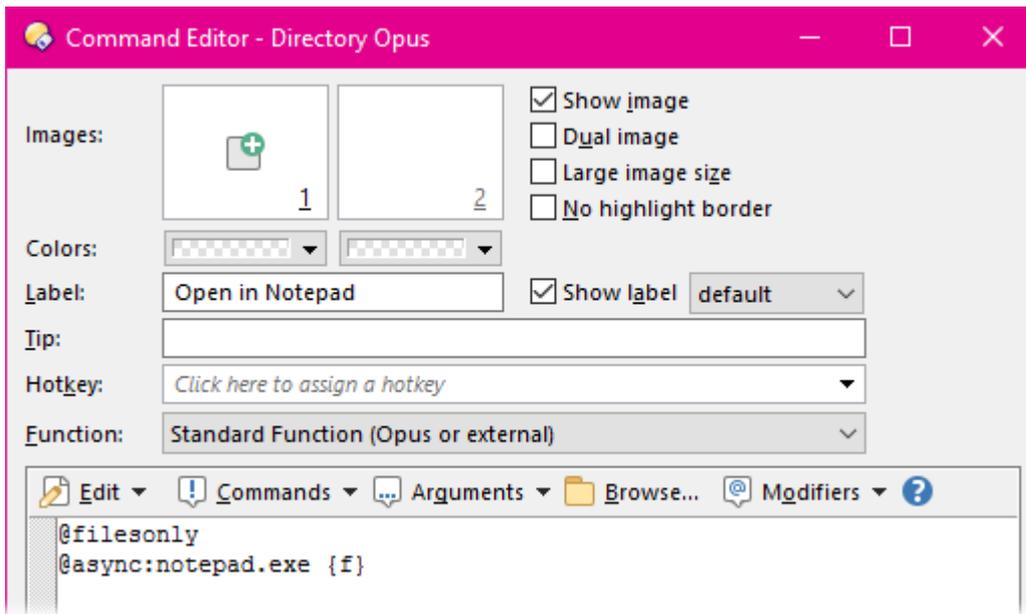
In the screenshot above, **{allfilepath}** is a special code responsible for passing the names of all selected files on the command line when the program is launched. With this command on a toolbar, you can select one or more image files and click the button to have them loaded into Photoshop - alternatively, you can drag-and-drop the files directly to the button.

Opus lets you pass several different types of information through to external programs, including the names of selected files, the name of the current source folder, strings representing the current date and time, and more. See the [External control codes](#) section for a full list.

## Command modifiers

Opus supports various command modifiers that can be used in toolbar buttons and hotkeys to modify the behaviour of the function. A command modifier is not a command itself - for the most part you can think of them as options that control how the function is executed.

Each modifier is given on a separate line - therefore, to use command modifiers in a function, you must edit the function in the [advanced command editor](#) (as the simple editor does not support multiple-line functions). Some modifiers (like **@async**) affect an individual command in the function, and are given on the same line as the command - others affect the whole function, and are supplied on a line by themselves.



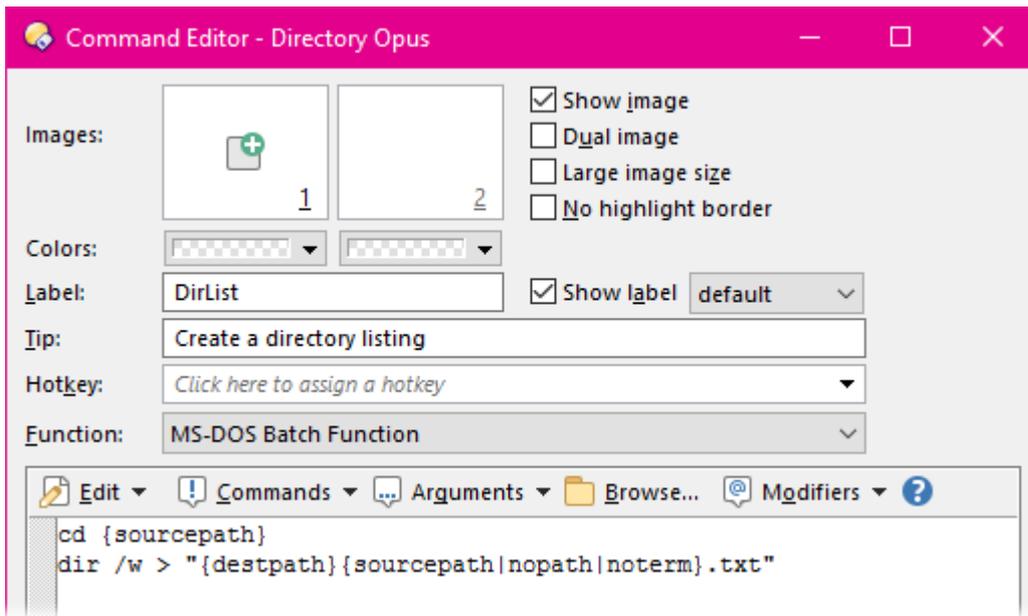
This screenshot is an example of a simple function that makes use of two command modifiers:

- The **@filesonly** modifier causes the function to only use selected files - any currently selected folders will be ignored. The `{f}` [external control code](#) means each filename will be passed in turn to Notepad.
- The **@async** modifier causes the command following it, **notepad.exe** in this instance, to run asynchronously. If multiple files are selected, an instance of notepad.exe will be executed for each file, without waiting for the previous instance to return.

See the [Command modifier reference](#) page for a full list of the available command modifiers.

## MS-DOS Batch commands

An Opus button or hotkey function can be set to *MS-DOS Batch Function* mode. A function in this mode is run as if it were a **.bat** batch file, and it can use MS-DOS commands, branching and logic instructions, as well as run external programs that are only designed to be run from the command line. You can also include regular Opus internal commands in the batch function (note that this can cause confusion when attempting to use external commands like **Set** or **Copy** that clash with internal commands - use the **@externalonly** modifier described below to fix this).



The above screenshot shows an example of a simple MS-DOS batch function. This uses the MS-DOS **dir** instruction to produce a directory listing of the current source folder. Using the > redirection symbol, the directory listing will be saved to a file in the destination path named after the current source folder.

Breaking down the function,

- The type is set to **MS-DOS Batch Function** using the **Function** drop-down. This is necessary to use the **dir** instruction - **dir** isn't a real program or command, it's an instruction that's only understood by the MS-DOS command interpreter.
- The first line of the command, **cd {sourcepath}**, sets the current directory of the batch script to the current source folder. This is the folder that **dir** will produce a listing of.
- The second command invokes the **dir** instruction (the **/w** flag turns on *wide* mode). The > character redirects the output of the command, and the combination of [external control codes](#) following the > supply the output filename.
  - **{destpath}** - the path of the current destination directory
  - **{sourcepath|nopath|noterm}** - the current source directory, without the path or trailing termination (i.e. the name of the directory)
  - **.txt** - the file extension for the output file.

When you run an MS-DOS batch function, Opus actually creates a temporary batch file on disk, and writes your instructions to it before invoking it. If you were to look at the batch file created by the above function, you would see something similar to the following:

```
@echo off
chcp 1252 > nul
C:
cd "\Users\Jon\Documents"
dir /w > "C:\Test\Documents.txt"
```

These instructions are written automatically to the **.bat** file, based on the parameters and definition of the function.

- **@echo off** is a standard MS-DOS instruction that prevents each command from being echoed to the prompt window as the batch file is run
- **chcp 1252 > nul** initialises the code page of the command prompt (1252 is the standard Windows code page).
- **C:** sets the current drive of the command prompt - this is generated automatically from the **cd {sourcepath}** instruction in the command definition.
- **cd "\Users\Jon\Documents"** sets the current directory on the current drive - again, this is generated automatically from the **cd {sourcepath}** instruction.
- **dir /w > "C:\Test\Documents.txt"** is responsible for producing the actual directory listing. In this instance, the source folder was called *Documents*, and the destination folder was *C:\Test* - so the name of the output file generated by **{destpath}{sourcepath|nopath|noterm}.txt** was *C:\Test\Documents.txt*.

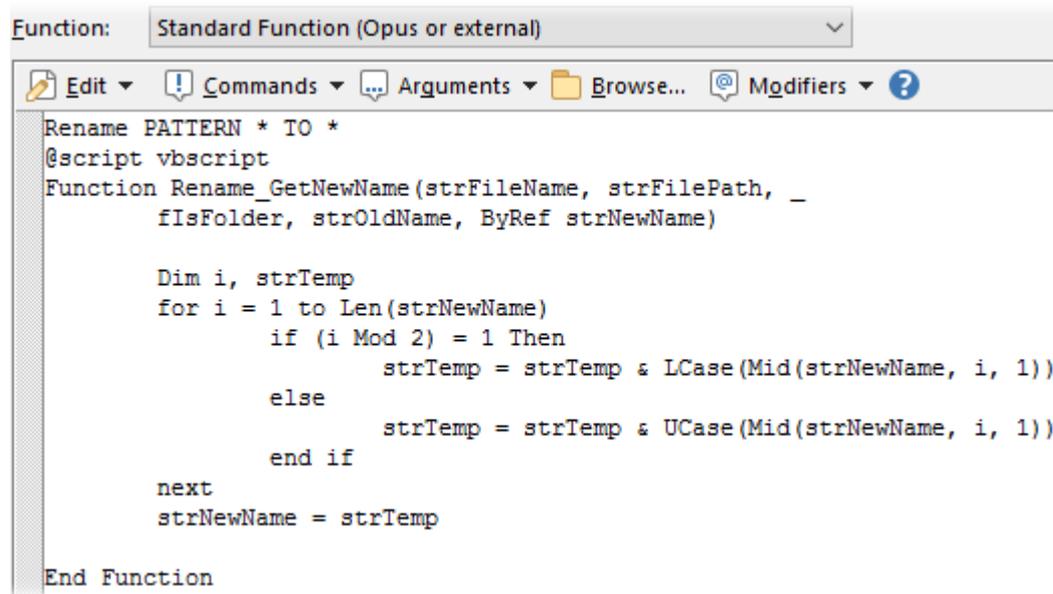
Several of the [command modifiers](#) relate specifically to MS-DOS batch functions:

- **@leavedoswindowopen** can be used to force the DOS prompt window to remain open once the function has finished (letting you see any output or error messages).
- **@codepage** can be used to change the code page of the DOS prompt from the default of *1252*.
- **@externalonly** tells Opus to ignore any command names that might coincide with internal commands (e.g. Set or Copy). Normally internal commands override external ones.
- **@nocall** is used to invoke a batch file and not return control to the parent function.
- **@runbatch** and **@norunbatch** control how external programs and internal commands interact in an MS-DOS batch function
- **@runmode:hide** can be used to prevent the brief flash of the DOS prompt window.

See the [command modifier reference](#) for a full description of command modifiers.

## Embedding Rename Scripts

[Rename scripts](#) can be embedded directly in a button or hotkey, which lets you rename files using a script without having to display the [Advanced Rename](#) dialog first.



```
Function: Standard Function (Opus or external)
Edit Commands Arguments Browse... Modifiers ?
Rename PATTERN * TO *
@script vbscript
Function Rename_GetNewName(strFileName, strFilePath, _
    fIsFolder, strOldName, ByRef strNewName)

    Dim i, strTemp
    for i = 1 to Len(strNewName)
        if (i Mod 2) = 1 Then
            strTemp = strTemp & LCase(Mid(strNewName, i, 1))
        else
            strTemp = strTemp & UCase(Mid(strNewName, i, 1))
        end if
    next
    strNewName = strTemp

End Function
```

This image depicts an example of an embedded rename script. The first line of such a button must call the [Rename](#) command. This command actually invokes the Rename function - and you can use any of the arguments for the **Rename** command to perform "pre-processing" before the script is invoked. For example, you could use the command **Rename CASE=allwords** to capitalize words in the filename first, before the script is invoked. In the above screenshot the **Rename** command will not actually modify the filename itself - it will be passed through verbatim to the script.

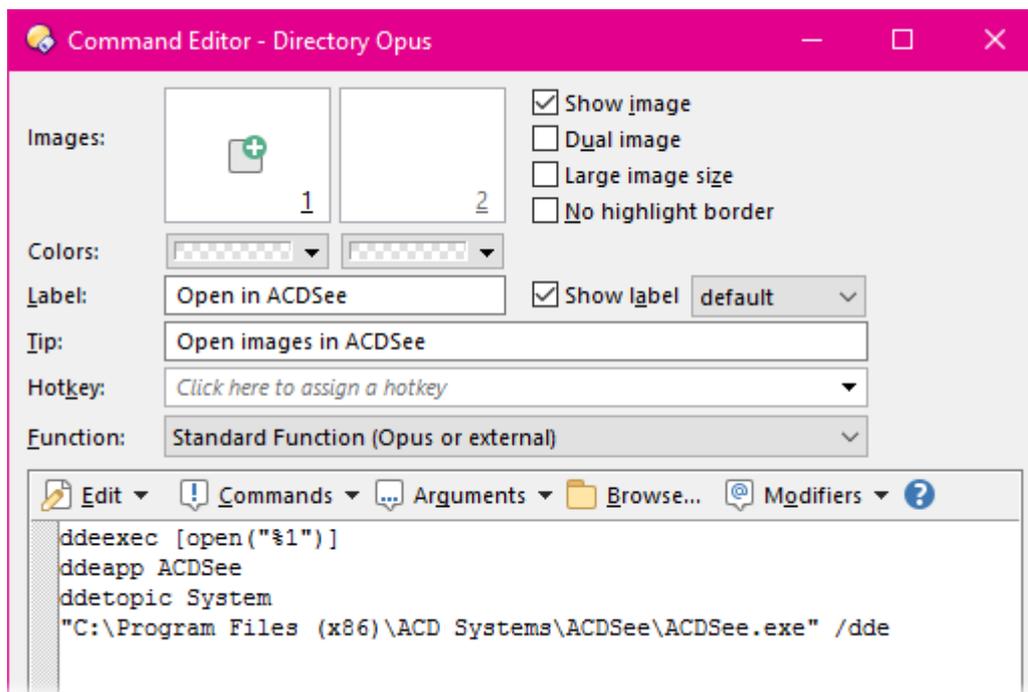
Below the **Rename** command, the **@script command modifier** is used to introduce the script. This tells Opus which language the script is written in - **vbscript** is shown in the above screenshot, but you can specify **jscript** to use JavaScript, and other languages are also supported if you have them installed.

Any text in the function definition below the **@script** line defines the script itself. See the [Rename Scripts](#) page for more information on writing rename scripts.

## DDE Functions

DDE ([Dynamic Data Exchange](#)) is an antiquated system for passing data between different programs on a Windows system. These days, it's not used much - although perhaps surprisingly, [Explorer Replacement](#) mode is actually implemented via DDE (for historical reasons more than anything).

Nonetheless, you may have an old program that supports a DDE interface, and should the need arise, it's possible to configure a button or hotkey to send what's called a DDE message to another application.



Above is an example command that establishes a DDE conversation and sends a command to another program. As this technique requires multiple command lines, you must create the command in the [Advanced Command Editor](#). The four lines in the above function are:

- **ddeexec**: The **ddeexec** instruction specifies the "message" that is to be sent to the other program. This will be defined by that program - the above screenshot is only an example. The **%1** [external control code](#) is used to include the name of the selected file in the message.
- **ddeapp**: The **ddeapp** instruction specifies the "application name" - this is the name that the other program looks for when DDE messages are sent.
- **ddetopic**: The **ddetopic** instruction specifies the "message topic" - again, this is application specific.
- The final line provides the actual command line used to launch the other program. This will normally involve passing some sort of command line flag telling the other program you want it to run in DDE mode - as in the above screenshot, with the **/dde** parameter. Again though, this is entirely dependent on the application in question.

## Embedded functions

There are several internal Opus commands that can cause a new Lister or tab to be opened - the most commonly used of these is **Go NEW** and **Go NEWTAB**. Sometimes, you might want a button or hotkey that opens a new Lister (or tab), and then executes further commands in the context of that new element. This can be achieved quite simply by embedding a function that will be passed to the new element once it has been created.

The embedded function must be surrounded with square brackets - for example, the following command would open a new Lister and put it into thumbnails mode automatically:

```
Go /mypictures NEW
[Set VIEW=thumbnails]
```

More complicated multiple line functions can also be embedded, for example:

```
Go NEW
[
Go /mypictures
Set VIEW=thumbnails
]
```

Additionally, functions can be embedded in a **Go FINDTITLE** command. These functions will be run in **all** existing Listers that match the supplied string. If no matching Listers are found, by default the embedded command will not be run at all - you can use the **RUNEMBEDDEDIFNOTFOUND** argument to cause the function to be run in the current Lister in this situation.

You can also embed functions in several [dynamic button](#) commands - for example, **Go DRIVEBUTTONS**. This lets you define a command that is added to each of the buttons generated by the function. For example,

```
Go DRIVEBUTTONS OPENINLEFT
[
```

```
Set FOCUS=Left  
]
```

This command would produce drive buttons that open their folders in the left-hand file display, and automatically set the focus to that file display at the same time.

The full list of commands that currently support embedded functions are:

- **Favorites** (dynamic button command)
- **Go NEW**
- **Go NEWTAB**
- **Go FINDTITLE**
- **Go DRIVEBUTTONS** (dynamic button command)
- **Go FTPSITELIST** (dynamic button command)
- **Prefs LAYOUT**
- **Properties SETLABEL** (dynamic button command)
- **Recent** (dynamic button command)
- **Show**



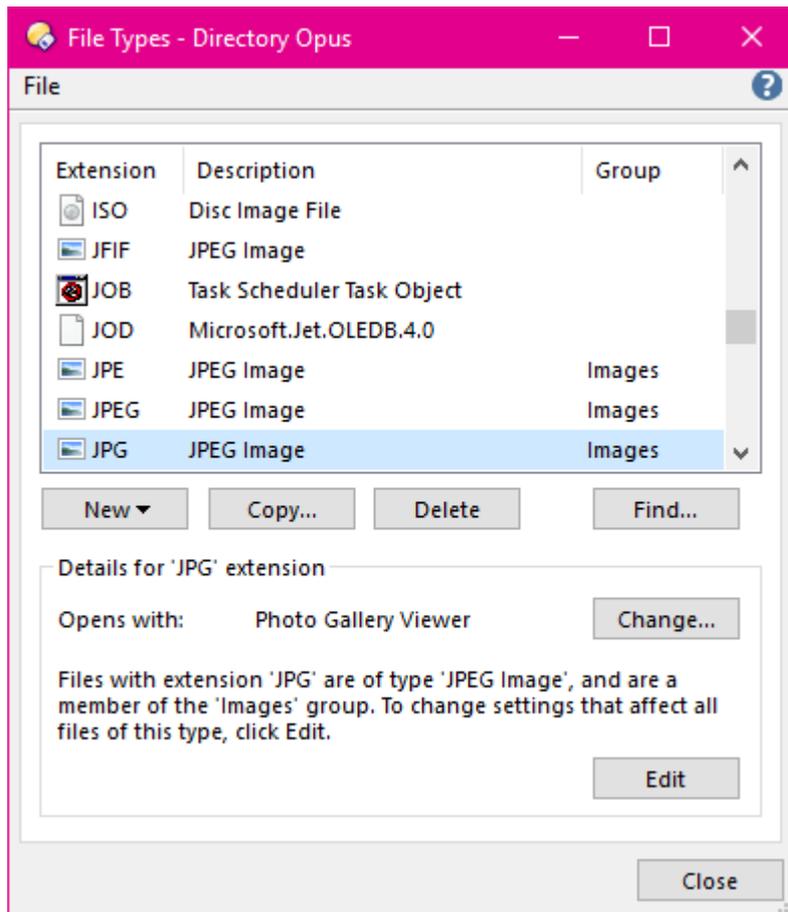
# File Types

File types are the set of registered types that are used to group files by file extension, throughout the system. For example, all files that end with **.txt** are said to be of type **Text Document**. The type of a file is displayed (by default) in the **Type** column in a Lister. Opus lets you configure a number of different settings on a per file-type basis. Actions such as double-clicking a file, or the display of tooltips for a file, can be different for different types of file.

Opus also extends the file types concept with the addition of [File Type Groups](#), which let you define a group of file types and then configure things for the group as a whole. So instead of having to add the same context menu item manually for each type of image file, you can add it to the **Images** group and it will automatically appear for all file types in that group. Groups are also used by the [Content Types](#) system; for each group you can define a folder format (sort order, view mode, etc) which is then automatically applied whenever you navigate to a folder containing mostly files that belong to that group (so for example, the display can automatically switch to thumbnails mode when you navigate to a folder containing mostly images).

The things that can be configured through the file types system are:

- [Standard file type parameters](#) like icon, description, the MIME type (if any) and the extensions assigned to the file type.
- [Actions](#): What happens when certain actions are performed on the file. Actions include standard context menu commands like explore, find, print, etc.
- [Events](#): Lets you define what happens when certain mouse-based events occur on the file. For example, double-clicking a file can do something different based on whether the Shift key is held down or not. You can also define a command that is run when a particular type of file is dragged and dropped.
- [Context Menu](#): You can add commands to the context menu (right-click menu) for files or folders.
- [Drop Menu](#): You can also add commands to the drop menu (the menu you get if you drag a file with the right mouse button and drop it).
- [Info Tip](#): You can define the appearance of info tips (the tool tip that's displayed when the mouse hovers over a file) on a per file-type basis. For example, image files can display a thumbnail in their popup tooltip.
- [Tiles Mode](#): You can also define how files will appear (what information they display) when the file display is in tiles mode.



The File Types dialogs are accessed via **Settings / File Types** in the default menus.

The main File Types dialog displays a list of all registered file extensions, their description, and the group they are assigned to if any. In the above screenshot you can see that **.JPE**, **.JPEG** and **.JPG** are all part of the **JPEG Image** file type, and are also assigned to the **Images** group. You can change the group that a file type is assigned to by moving the mouse over the **Group** column - a drop-down control will appear that lets you pick the group.

The **New** drop-down is used to create a new file type or group. The **Copy** button lets you duplicate an existing file type, and the **Delete** button lets you delete it.

You can locate file types using the **Find** function, which lets you search by extension, descriptions and other properties. You can also quickly jump to a file extension by activating the file type list and typing the extension in.

The area at the bottom of the dialog displays information for the currently selected file type including its "default handler" (*Opens with: Directory Opus Viewer* in the above screenshot). The default handler is the program that will (normally) be used to open the file when it's double-

clicked - you can change this, and also edit the items shown on the [Open With](#) menu for a file type, using the **Change** button.

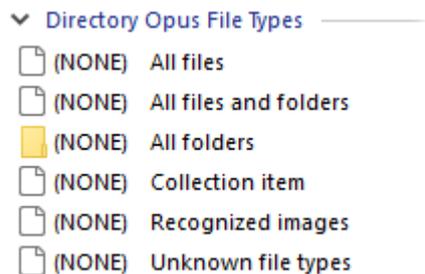
One thing to be aware of is that the file types system in Opus is shared with the rest of the system. For any given file type there are a number of different classes of settings:

- **Global:** The definition of file extensions and types, information set for file types (icons, descriptions, Opens with, etc) will all be used by Windows and reflected by other programs in the system. Changing these in Opus will also change these parameters throughout the system.
- **Mixed:** Some things you change in Opus can affect either Opus only, or the whole system, on a case-by-case basis. For example, you can add context menus that appear in Explorer as well as Opus, but you can also add context menus that only appear in Opus.
- **Opus-only:** Some things Opus lets you change only affect the behavior in Opus. For example, the **Events** tab in the file type editor which let you override the behavior when a file is double-clicked only affects Opus. So it's quite possible to configure a file to open with one program when double-clicked in Explorer and another when double-clicked in Opus.

Use the **Edit** button at the bottom of the dialog to display the [File Type Editor](#) for the currently selected file type.

# Directory Opus File Types

As well as the registered file type extensions and [groups](#), there are a number of pseudo-file types that Opus defines. These are listed at the top of the [File Types](#) dialog.

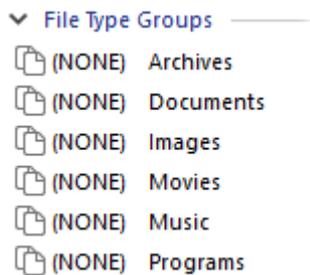


Changes that you make to these file types will affect general classes of file rather than specific types. They are:

- **All files:** All files belong to this class. Changes that you make to this will affect all types of file (but not folders).
- **All files and folders:** All files and folders belong to this class - changes you make to this will affect all files and folders throughout the system. The "standard" context menu commands like **Cut/Copy/Paste** are implemented through this class, as those are context menu commands that are relevant for any file or folder.
- **All folders:** All folders belong to this class. Changes that you make to this will affect all folders but have no effect on files.
- **Collection item:** This class affects items that are stored in Opus [File Collections](#). The normal use of this class is to add context menu items like **Remove from Collection**.
- **Recognized images:** This class contains all image types that Opus recognizes. It's similar but not identical to the **Images** group. Although the **Images** group defaults to containing most image formats Opus understands, you can add any file extensions you like to it - whereas the **Recognized images** class automatically contains all recognized image formats and no others.
- **Unknown file types:** Any unregistered file types belong to this class. For example, if you have a file called **myfile.bumblebee** and **.bumblebee** isn't a registered file extension, files with that extension would automatically belong to this class.

# File Type Groups

File type groups let you create pseudo file-types that contain multiple file extensions. Changes that you make to a file type group (like adding a context menu) are automatically reflected for all files that belong to that group. For example, the **Images** group by default adds a context menu command called **Convert Image** which invokes the [Image Conversion](#) tool. This command will be displayed on the context menu for all files that belong to the **Images** group.



You can add your own groups, and a number of groups are defined by default:

- **Archives:** Represents the archive formats that Opus supports (**.zip**, **.7z**, etc). If you enable or disable support for individual formats through the [Archive and VFS Plugins](#) Preferences page, their file extensions will automatically be added to or removed from this group.
- **Documents:** Contains various common document formats (**.doc**, **.xls**, **.mdb**, etc.)
- **Images:** Contains various common image formats (**.bmp**, **.gif**, **.jpg**, etc.)
- **Movies:** Contains common movie file extensions (**.avi**, **.wmv**, **.mpg**, etc.)
- **Music:** Contains various common music formats (**.mp3**, **.wav**, **.wma**, etc.)
- **Programs:** Contains executable file formats like **.exe** and **.dll**.

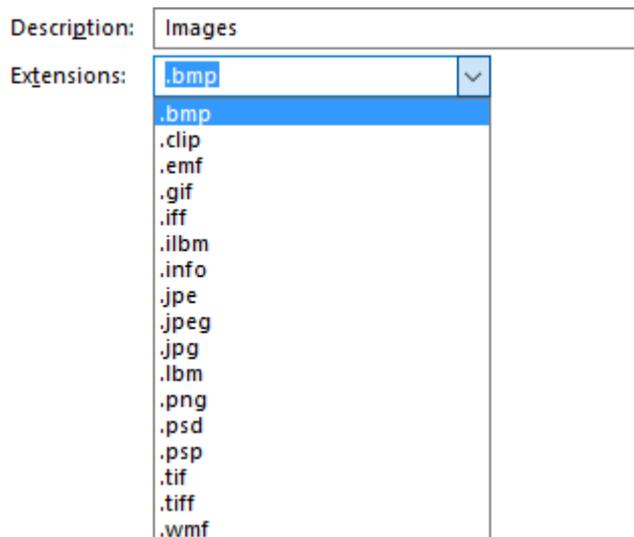
You can change the file extensions assigned to groups by [editing](#) them like normal file types. At any time you can reset one of the default groups by right-clicking its entry in the list and choosing the **Reset to Defaults** command from the context menu. You can also reset all of the default groups at once by right-clicking the **File Type Groups** header in the list.

The concept of groups is similar to that of file types themselves; they both refer to one or more file extensions. The difference (apart from groups being an Opus-only concept) is that a group contains file extensions for related classes of file whereas file types contain file extensions for the same type of file.

To explain briefly, consider the **JPEG Image** file type that is created by default on a Windows machine.

	JFIF	JPEG Image
	JPE	JPEG Image
	JPEG	JPEG Image
	JPG	JPEG Image

The **JPEG Image** file type has a number of file extensions assigned to it (**.jpe**, **.jpeg**, **.jpg** and **.jif**) - but they all represent JPEG-format images. On the other hand, the **Images** group contains file extensions for multiple types of image file.

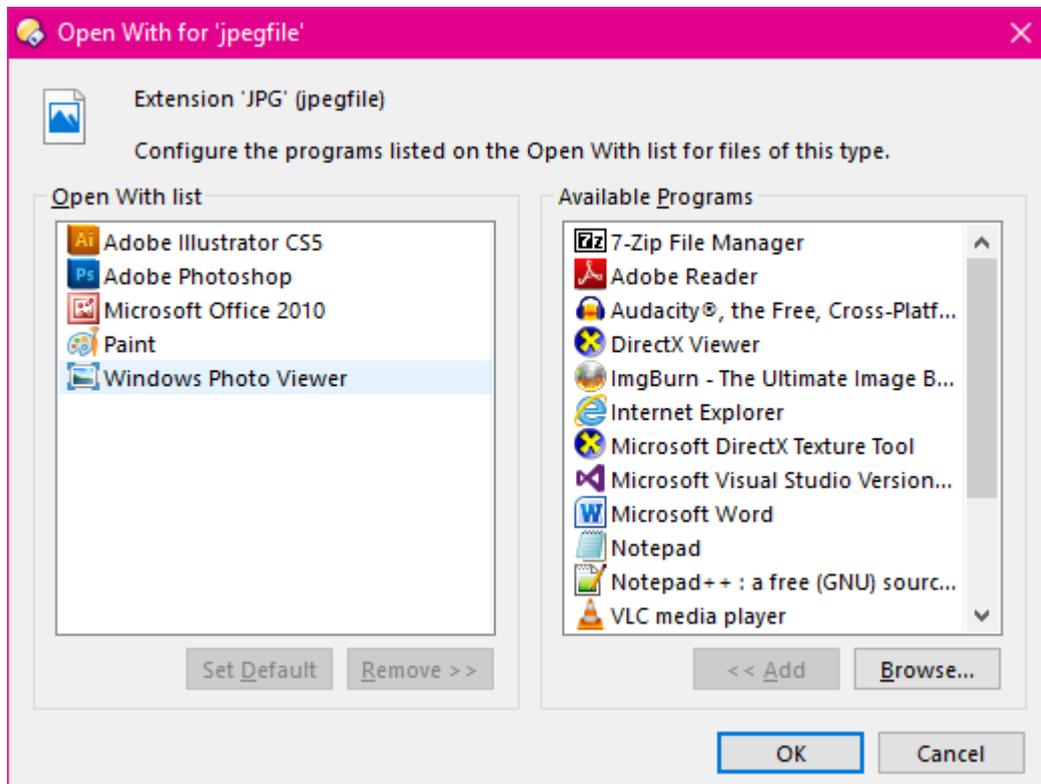


The **Images** group contains the JPEG file extensions, but it also contains file extensions for other formats of image file (**.bmp**, **.gif**, etc).

Groups are also used by the [Content Types](#) system; for each group you can define a folder format (sort order, view mode, etc) which is then automatically applied whenever you navigate to a folder containing mostly files that belong to that group (so for example, the display can automatically switch to thumbnails mode when you navigate to a folder containing mostly images).

## The Open With editor

The Open With editor for a file type lets you edit the programs that are displayed on its *Open With* list - the sub-menu of the same name that appears on a file's context menu. This setting applies to both Opus and Explorer.



To access the Open With editor, select the file type in the [File Types](#) dialog and then click the **Change** button at the bottom.

The Open With editor contains two lists of programs. The list on the right defines the programs that are currently shown in the file's *Open With* menu. The list on the left is a list of all "available programs". These are programs that are "known" to Windows - they have been used to open files in the past. If the program you're looking for isn't on that list, click the **Browse** button to locate it by hand.

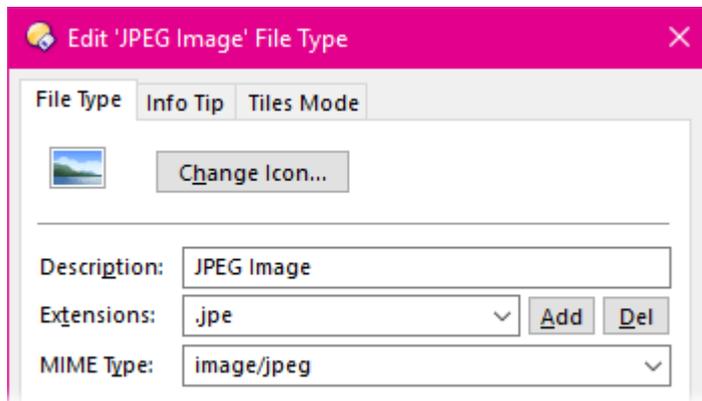
To add a program to the *Open With* list, simply select it in the right-hand list and click the **Add** button to move it to the left. Similarly, to remove a program from the *Open With* list, select it in the list on the left and click **Remove** to move it back to the list on the right.

You can also choose the default program to open files of this type with by selecting the item in the *Open With* list and clicking the **Set Default** button.



# File Type Editor

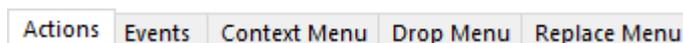
The File Type Editor dialog is used to make changes to individual file types or [file type groups](#). It is accessed by double-clicking on a file type in the list on the [File Types](#) dialog, or selecting the file type and clicking the **Edit** button at the bottom of that dialog.



The top portion of the file type editor lets you define the basic properties for the file type.

- **Change Icon:** This lets you change the icon that is used to represent files of this type. This is a global setting (it affects the whole system, not just Opus). Because Windows manages file icons itself, File type groups and the special Opus file types do not have configurable icons.
- **Description:** This is the description of the file type or group. For file types, this changes the string displayed in the Type column for files of this type. For groups, this specifies the name of the group.
- **Extensions:** This drop-down list lets you edit the file extensions assigned to the file type or group. To add a new extension, type it into the field and then click the **Add** button. To remove an extension, select it from the drop-down list and click the **Del** button. For file types this is a global setting (it affects the whole system).
- **MIME Type:** This lets you edit the MIME association for the file type, which lets you associate file extensions with Internet [MIME](#) types. This is a global setting.

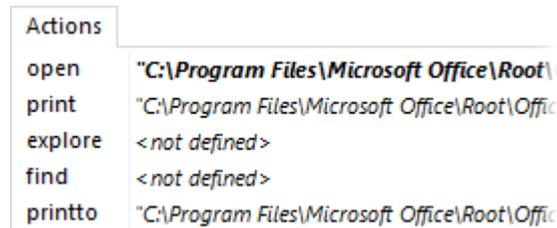
The tabs at the top of the file type editor let you access the [Info Tip](#) and [Tiles Mode](#) settings for the file type.



Below the **MIME Type** field is another set of tabs that let you access the [Actions](#), [Events](#), [Context Menu](#), [Drop Menu](#) and [Replace Menu](#) settings for the file type.

## Actions

The **Actions** tab in the [file type editor](#) lets you configure the behavior of various "actions" - standard commands (verbs) that the system defines and uses for files of various types. The settings on this tab are global - they affect the behavior of the file type throughout the system. [File type groups](#) and most of the special [Directory Opus File Types](#) don't have an actions tab.



Actions	
open	"C:\Program Files\Microsoft Office\Root\
print	"C:\Program Files\Microsoft Office\Root\Offic
explore	<not defined>
find	<not defined>
printto	"C:\Program Files\Microsoft Office\Root\Offic

This screenshot shows the **Actions** tab for a **.doc** file (Word document). The available actions are:

- **explore**: This action only makes sense for folders, it defines the function for the "explore" verb. Normally this is used when you right-click on a folder and choose **Explore** from the context menu.
- **find**: This action is also only generally used for folders, it defines the function for the "find" verb.
- **open**: This is *usually* the default action for a file type - when you double-click on a file, the "open" verb is the one most often invoked. It's also generally what will happen if you right-click on a file or folder and choose **Open** from the context menu.
- **print**: The "print" verb is usually invoked when you right-click on a file and choose **Print** from the context menu. It lets default print handlers be defined for different file types.
- **printto**: This is a variant of the "print" verb that's used when printing to a specific printer.

Of the above actions, **explore**, **open** and **print** are the ones most often used. When an action is displayed in the context menu it can have a label that doesn't necessarily have to correspond with the name of the verb. If such a label is defined then it is displayed in the actions list rather than the name of the verb. This has actually happened in the above screenshot although it's not immediately obvious - the label for the "open" action has been set to Open (with a capital letter).

At the bottom of the actions tab, the **Edit** button lets you edit the selected action (or you can just double-click it), the **Delete** button lets you delete it (clear out the definition), and the **Set Default** button lets you set the default action for a file type (what happens when you double-click it). Normally you wouldn't want to change this from **open** - in the above screenshot, the **open** action is shown in bold to indicate it is the default action.

You can also right-click on the items in the actions list to display a context menu for the item. This context menu lets you use **Copy** and **Paste** to copy the definition from one action to another.

Editing an action displays the **Edit** action dialog. Because actions are global settings (used by Windows as well as Opus), this is not a full-blown [function editor](#) dialog, and you can't configure actions to use internal Opus commands. Instead, two types of function can be defined for an action.

The screenshot shows the 'Edit Action' dialog with the following fields:

- Action:** open
- Type:** Run an application (selected from a dropdown menu)
- Application:** C:\Windows\System32\rundll32.exe "C:\Program Files\Windows\

The first type (selected from the **Type** drop-down) is *Run an application*. This lets you define an action that simply runs an executable program. The name of the file can be passed to the program using the [%1 control code](#). For example, the command line for the open action for .jpg files is:

```
C:\Windows\System32\rundll32.exe "C:\Program Files\Windows Photo Viewer\PhotoViewer.dll", ImageView_Fullscreen %1
```

The **Action** field lets you configure the label that's displayed in the context menu for this action.

The screenshot shows the 'Edit Action' dialog with the following fields:

- Action:** &Print
- Type:** DDE command (selected from a dropdown menu)
- Application:** "C:\Program Files\Microsoft Office\Root\Office16\WINWORD.EXE"
- DDE Message:** [REM\_DDE\_Minimize][FileOpen("%1")[t=IsDocumentD
- AppName:** WinWord
- Topic:** System
- Not Running:** [FileOpen("%1")][FilePrint 0][FileExit 2]

The second type of action is *DDE command*. This uses the (rather antiquated) [DDE system](#) to send a command to an application that may or may not be already running. The above image shows the **print** action for Word documents. You can see that the label for the action has been set to **&Print** - in context menus, this would result in the command being displayed as **Print** with an underscore under the **P** (to represent the key that can be pressed for the function).

It's beyond the scope of this document to explain how DDE works unfortunately, but luckily it's extremely unlikely you'll ever actually need to use this system. The parameters that can be specified for a DDE function are:

- **Application:** The command line that's used to start the application if it's not already running.
- **DDE Message:** The message that's sent to the application to make it take the desired action on the file.

- **AppName:** The DDE name of the application.
- **Topic:** The DDE topic that's used to initiate the DDE conversation.
- **Not Running:** An alternate message that's used if the application wasn't already running and needed to be launched.

## Events

The **Events** tab lets you configure functions for file types that are initiated by various mouse events - double-click and drag and drop events. For example, you can configure Opus to run a different program for files when they're double-clicked and the **Shift** key is held down, or when they're double-clicked with the middle mouse button (if you have one). It's also possible to use this system to have different double-click behavior to Explorer - so that double-clicking a file in Opus can behave quite differently to double-clicking it in Explorer.

Events	
Drag-and-drop	<not defined>
Drag-and-drop + Alt	<not defined>
Drag-and-drop + Ctrl	Image CONVERT=png
Drag-and-drop + Shift	<not defined>
Left double-click	Show
Left double-click + Alt	<not defined>
Left double-click + Ctrl	"C:\Program Files\Adobe\Adobe ...
Left double-click + Shift	<not defined>
Middle double-click	C:\Windows\System32\rundll32...
Middle double-click + Alt	<not defined>
Middle double-click + Ctrl	<not defined>
Middle double-click + Shift	<not defined>

The above screenshot shows an example of some events that have been configured for the **Images file type group**. The events that you can configure are:

- **Drag-and-drop:** What happens when you drag and drop a file to another folder.
- **Left double-click:** What happens when you double-click a file with the left mouse button.
- **Middle double-click:** What happens when you double-click a file with the middle mouse button.

These three events can also be configured separately depending on whether the **Alt**, **Ctrl** or **Shift** keys are held down. There are no right button double-click events as the right button is used for [Context Menus](#). You can also add items to the drag and drop context menu (the menu displayed when you drag a file with the right mouse button) on the [Drop Menu](#) tab.

Whenever one of the above events occurs, Opus searches your configured file types looking for a configured action. File types are searched in the following order:

1. The specific file type for that file (e.g. for a **.jpg** file this might be the **JPEG Image** file type)
2. The file type group that contains that file extension (e.g. **Images**)
3. The **Recognized images** file type if the file is a recognized image file
4. The **All files** file type (or for a folder, **All folders**)
5. The **All files and folders** file type.

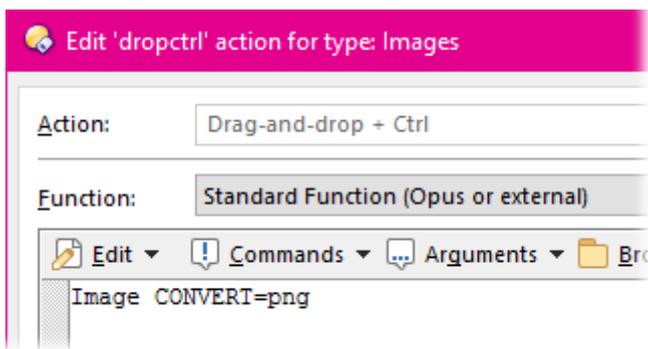
The first, most specific, file type found that has a function defined for that event is the one used.

To edit the event for a file type, select the event from the list and click the **Edit** button at the bottom of the page (or double-click the event in the list). The event editor is a variant of the standard command editor, so please see the [Command Editor](#) page for instructions on how to define a command.

You can also right-click on the items in the events list to display a context menu for the event. This context menu lets you use **Copy** and **Paste** to copy the definition from one event to another.

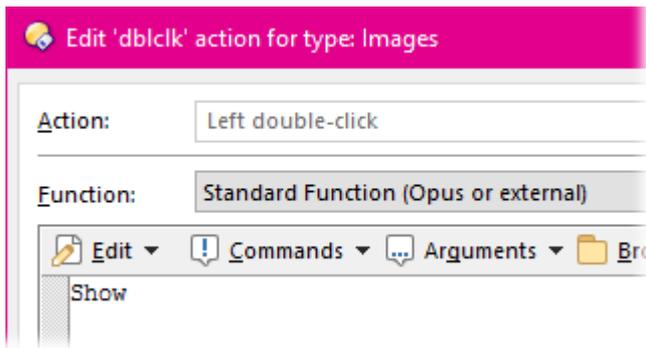
The events system can be very useful; with a bit of careful thought and configuration it's possibly to really streamline your workflow. To take the above screenshot as an example, four events have been configured for the **Images** group (meaning they will act on any file type added to that group).

1. The **Drag-and-drop + Ctrl** event has been configured to convert images to PNG format when you hold the **Ctrl** key and drop them in a folder.



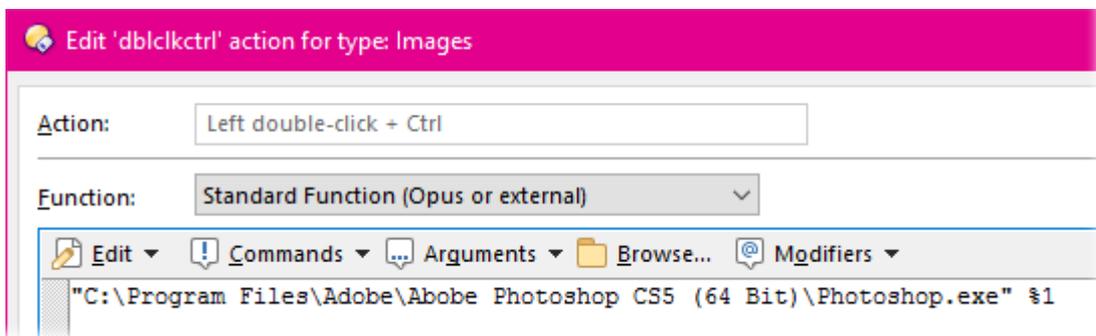
The function used for this is **Image CONVERT=png**. This invokes the [Image Conversion](#) function and specifies an output format of PNG. Because the output format has been specified on the command line, the image conversion dialog will not appear - instead the dropped file will be immediately saved to the target folder in PNG format.

2. The **Left double-click** event has been configured to open the file in the internal Opus viewer when the file is double-clicked.



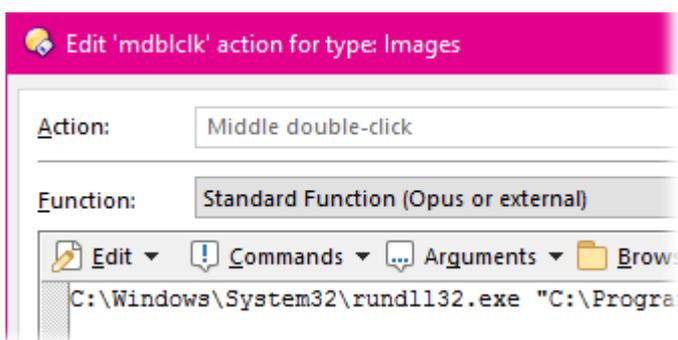
This event calls the internal [Show](#) command to view the image using the [standalone image viewer](#). This lets you double-click images in Opus to preview them quickly in its own viewer, but doesn't have any effect on what happens when you double-click an image file outside of Opus.

3. The **Left double-click + Ctrl** event has been configured to open the file in Photoshop when the file is double-clicked and the **Ctrl** key is held down.



This event runs the external function **"C:\Program Files\Adobe\Adobe Photoshop CS5 (64 Bit)\Photoshop.exe" %1** to open the file in Photoshop. The **%1** [control code](#) is used to pass the filename to **Photoshop.exe**.

4. Finally, the **Middle double-click** event has been configured to open the file in the standard Windows image viewer when the file is double-clicked with the middle mouse button.



This event runs the external function **C:\Windows\System32\rundll32.exe "C:\Program Files\Windows Photo Viewer\PhotoViewer.dll", ImageView\_Fullscreen %1** which invokes the Windows photo viewer. The name of the file is passed to the application using the **%1** [control code](#).

## Context Menu

The Context Menu tab lets you add commands to the context menu for files and folders. For example, Opus uses this system to display the "standard" context menu items of *Cut / Copy / Paste* etc. via the **All files and folders** file type.

Context Menu	
Cut	<i>clipboard cu</i>
Copy	<i>clipboard co</i>
Paste	<i>clipboard pc</i>
Create Shortcut	<i>copy makesh</i>
Delete	<i>delete shift r</i>

When you right-click a file or folder to display its context menu, Opus searches the system registry and its own file type settings for the commands to display on the menu. The context menu is built from **all** the file types that match the file you have clicked on. So for example, the context menu for a **.jpg** file would include commands from the **JPEG Image** file type, the **Images file type group**, the [Recognized images](#) file type, the [All files](#) file type, and the [All files and folders](#) file type.

You can add two types context menu items using this page of the [file type editor](#):

- **Global:** Context menu items that are saved in the system registry, and will be displayed on the context menu in Explorer and other programs as well as in Opus. These commands must invoke external executable programs.
- **Opus-only:** Context menu items that are only displayed on context menus in Opus - they will not appear in Explorer. These commands can use internal Opus commands as well as invoking external programs.

Context menu commands that come from the system but aren't provided as "static verbs" in the registry (e.g. they use *context menu extension handlers*) can't be configured through this system. It is possible (with some fiddling) to control the display of these context menu items as well - see this [FAQ](#) for more information.

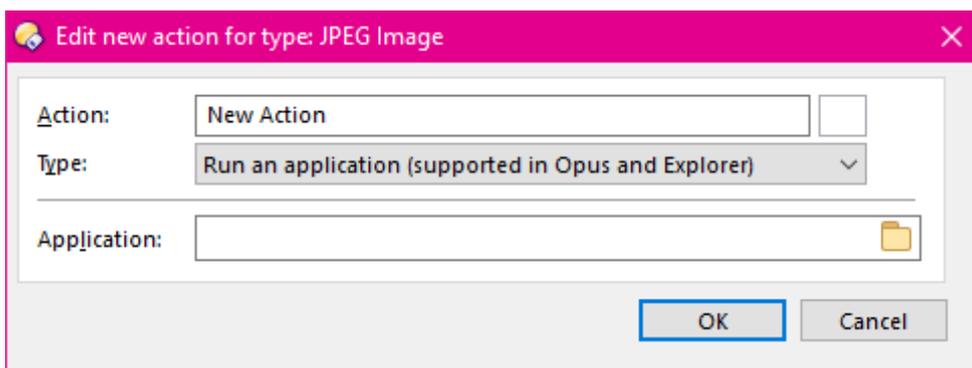
The **New** button at the bottom of this page lets you add a new context menu item to the file type. The **Edit** button lets you edit an existing one (or just double-click it in the list) and the **Delete** button lets you delete it. The **Set Default** button lets you set a *global* (but not *Opus-only*) context menu item as the default action for the file type - the command that will normally be run when you double-click the file.

You can also right-click on the items in the context menu list to display the context menu context menu (heh, sorry!). This context menu lets you use **Copy** and **Paste** to copy the definition from one command to another. There is also the **Begin a group** option which lets you place separators between context menu items. When you right-click on an item and choose **Begin a group**, a separator will appear above it (as you can see above the **Create Shortcut** command in the above screenshot).

You can use drag and drop to reorder the context menu items in the list, or to add or remove separators (by dragging the item a small distance and dropping it on itself). If you have two [File Type editor](#) dialogs open at once you can also drag and drop context menu definitions from one editor to another, to copy commands from one file type to another.

---

When you add a new context menu item to a system file type, you need to choose whether it is going to be *global* or *Opus-only*.



This is accomplished using the **Type** drop-down. This drop-down initially has four options:

- **Run an application:** Defines a context menu command that runs an external program. This will establish the item as global - it will show in the context menu in both Opus and Explorer.
- **DDE command:** Defines a command that communicates with an external program using DDE. This will also define the item as global - it will work in both Opus and Explorer.
- **Run an Opus function:** This will establish the item as Opus-only. Commands of this type can use [Opus internal commands](#) as well as launch external programs. It will only be shown in context menus inside of Opus.
- **Sub-menu:** This is a special type (explained below) that lets you create sub-menus in context menus. This is also Opus-only.

Once a function has been defined as **Run an application** or **DDE command**, you can't turn it into a **Run an Opus function**-type item; and vice versa. If you click **OK** to save the new context menu item, and then select it and click **Edit** you will see that the **Type** field has disappeared (in

the case of Opus-only functions), or now only contains options for **Run an application** and **DDE commands** (for global menu items).

If you add a context menu item to a [file type group](#) the first two type options aren't available - as groups only work inside of Opus it's not possible to add global menu items to them.

For all context menu item types, the **Action** field defines the label that is shown for the command in the context menu. The small box to the right of the **Action** field lets you specify an icon that's also shown in the context menu to the left of the label. The icon is only displayed when the context menu is opened in Opus - it's not supported by Explorer.

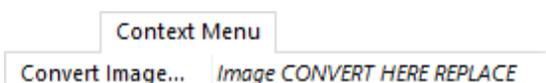
The **Application** field for a **Run an application**-type context menu command is where you define the path to the external program and any arguments you are passing it. For example, a command that opens the selected file in Notepad might look like this:

Action:	<input type="text" value="Open in Notepad"/>	
Type:	<input type="text" value="Run an application (supported in Opus and Explorer)"/>	
Application:	<input type="text" value="C:\Windows\System32\notepad.exe %1"/>	

The **Application** command passes the selected filename to Notepad.exe using the [%1control code](#).

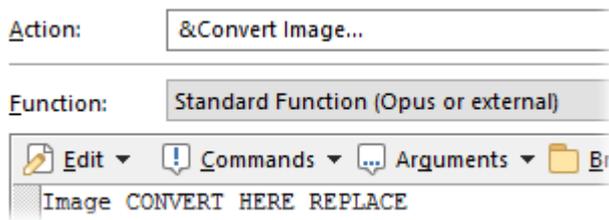
**DDE Commands** are also supported in both Opus and Explorer. Although it's pretty unlikely you will ever need to define a DDE command these days, you can find more information about these in the [help for the Actions tab](#).

The **Run an Opus function** type uses a variant of the standard [command editor](#) to define a function that can use both Opus [internal commands](#) and external programs. You can see an example of this in the context menu for the default **Images** file type group:



This screenshot shows that a context menu item to invoke the [Image Conversion](#) function has been defined for this file type. Because it uses an internal Opus command ([Image](#)) this context menu won't appear in Explorer, only in Opus. Actually because this context menu is on the

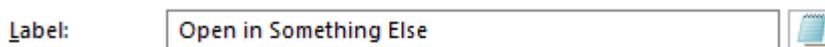
**Images** file type group it has to be an Opus-only command anyway, but it could also appear on the context menu for a system file type like **JPEG Images**.



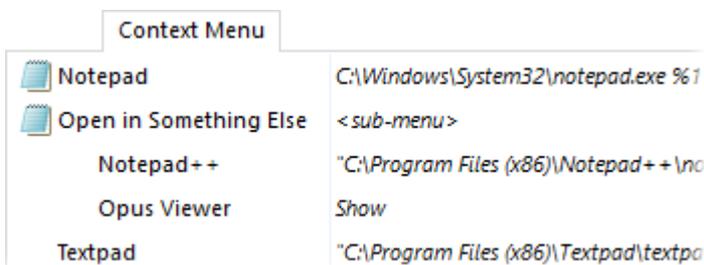
You can see that there is no **Type** drop-down, as the type of the item can't be changed from Opus-only. The standard [command editor](#) controls let you define the function. The label of the context menu item (**&Convert Image...**) is given in the **Action** field - the ampersand (&) in the label specifies which letter of the label is to be underlined in the context menu.

The fourth option in the **Type** drop-down when creating a new menu item is **Sub-menu**. This lets you create sub-menus in context menus - you can move groups of commands off the main context menu and into a sub-menu to keep related commands together, or to keep the main context menu tidy. Sub-menus only work when context menus are displayed in Opus - they won't appear in Explorer. If a global context menu item is added to a sub-menu in Opus, it will appear on the main context menu in Explorer.

The only options for **Sub-menu** are **Label** and icon (to set the icon click the small box to the right of the **Label** field).

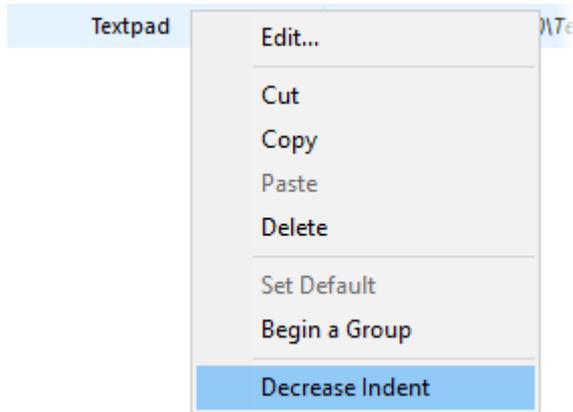


In the context menu list, items that are in a sub-menu are shown as indented.



In the above screenshot, the **Notepad** item will appear on the main context-menu, followed by a sub-menu called **Open in Something Else**. Inside that sub-menu are two items, **Notepad++** and **Opus Viewer**. Following the sub-menu (on the main menu) is another item, **Textpad**.

All context menu items that appear below a sub-menu in the list will automatically be placed in that sub-menu. To specify the item that marks a return to the top-level menu, you must right-click on it in the list and choose the **Decrease Indent** command. In the above screenshot, this command was run on the **Textpad** entry, which decreased its indent level and moved it back to the main menu.

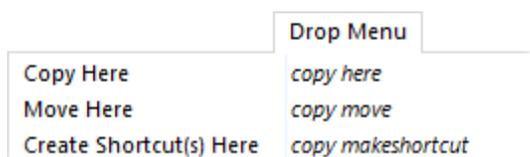


Sub-menus can be nested (you can have a sub-menu inside a sub-menu inside a sub-menu, and so on).

If you right-click on the first item below a sub-menu in the context menu editor, an option called **Button** will also be available. If you turn this option on, the sub-menu will act like a "menu button" - that is, you'll be able to click the sub-menu itself to run the first command within it, as well as popping the menu open to access its other commands.

## Drop Menu

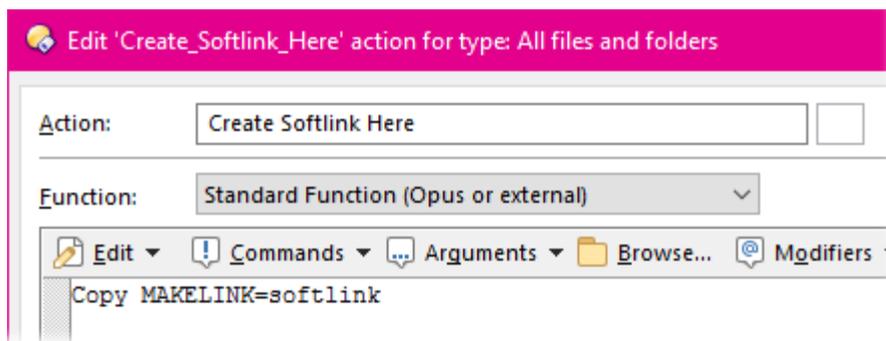
The Drop Menu tab lets you add commands to the drag and drop menu for files and folders. This is the menu that's displayed when you drag a file with the right mouse button and drop it. For example, Opus uses this system to display the "standard" drop menu items of *Copy Here*, *Move Here* and *Create Shortcut(s) Here*, via the **All files and folders** file type.



Configuration of the drop menu for file types is very similar to that of the Context Menu for file

types - please see the [Context Menu](#) page for a more thorough description of the context menu configuration process. The main difference between the drop menu and the context menu is that the drop menu only supports *Opus-only* context menu items - that is, only **Run an Opus function** or **Sub-menu** items can be added to the drop menu.

As an example of one use for the drop menu, you could add a command to automatically make a softlink (Vista and above only) to a file or folder when you drag it with the right button and drop it somewhere. [Locate](#) the **All files and folders** file type and edit it, and then on the **Drop Menu** tab click the **New** button to add a new menu item.

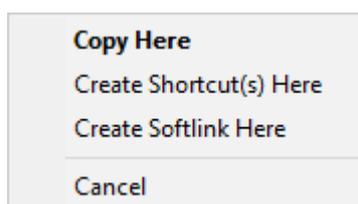


For the **Action** we have specified *Create Softlink Here* - this is the label that will be displayed in the drop menu. The **Type** is set to **Run an Opus function**, and the **Function** definition uses following command:

```
Copy MAKELINK=softlink HERE
```

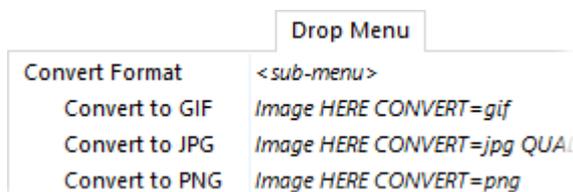
This uses the internal [Copy](#) command with arguments necessary to create a softlink in the target folder.

If you click **OK** to save the new drop menu command, and then drag and drop a file with the right mouse button, you should see your new command on the drop menu:

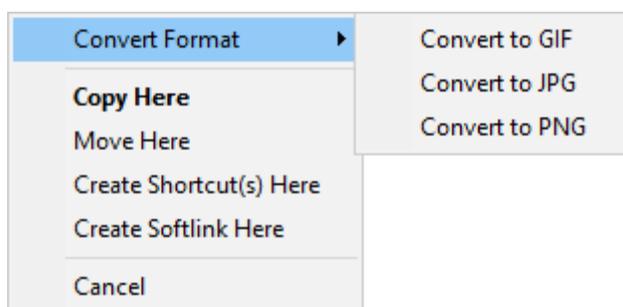


Creating a softlink requires Administrator privileges so don't be alarmed if you get a UAC prompt as a result of trying this command.

You could also adapt the PNG conversion example shown for the [Events](#) page for use in the drop menu. For example:



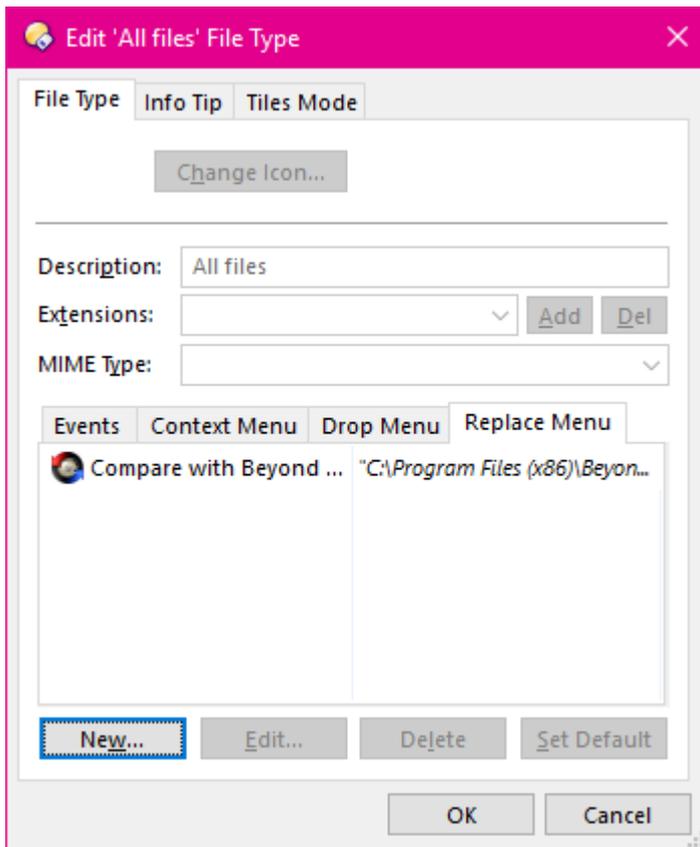
Adding these commands (in a sub-menu) to the drop menu for the **Images** [file type group](#) would give you access to a set of convenient image conversion commands simply by dragging an image with the right mouse button and dropping it in the target folder.



You can see that as well as the new **Convert Format** menu, the drop menu also shows the **Create Softlink Here** command that we created above. Even though they were added to different file types (**Convert Format** was added to the **Images** group, whereas **Create Softlink Here** was added to the **All files and folders** file type) they are both displayed, because both file types applied to the file we dragged. If you dragged a non-image file, the **Convert Format** menu would not be shown.

## Replace Menu

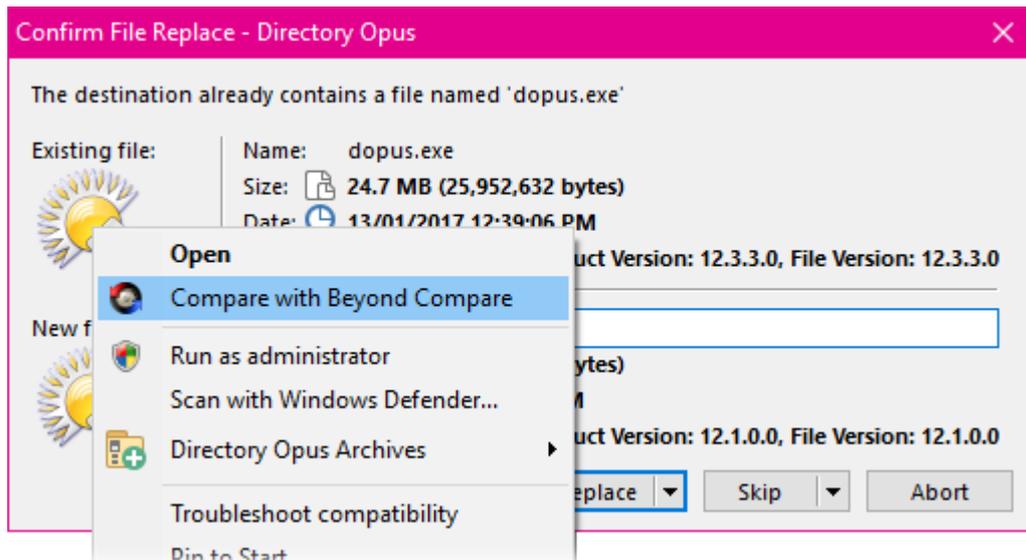
The **Replace Menu** tab lets you add commands to the context menu displayed when you right-click on a file icon in the [Confirm File Replace](#) dialog (displayed when copying a file over one that already exists). Using this you can define commands that, for example, let you compare the old and new files using an external comparison tool.



For example, a command to compare the two files using Beyond Compare might look like this:

```
"C:\Program Files (x86)\Beyond Compare 4\BCompare.exe"  
{filepath$} {filepathdest$}
```

Then, in the Replace dialog, clicking either of the file thumbnails shows this command at the top of the context menu.



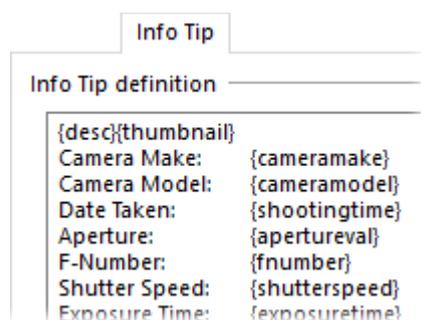
## Info Tip

The **Info Tip** page in the [file type editor](#) lets you define what is shown on the info tip (the popup tooltip) that's displayed when the mouse hovers over files of this type.

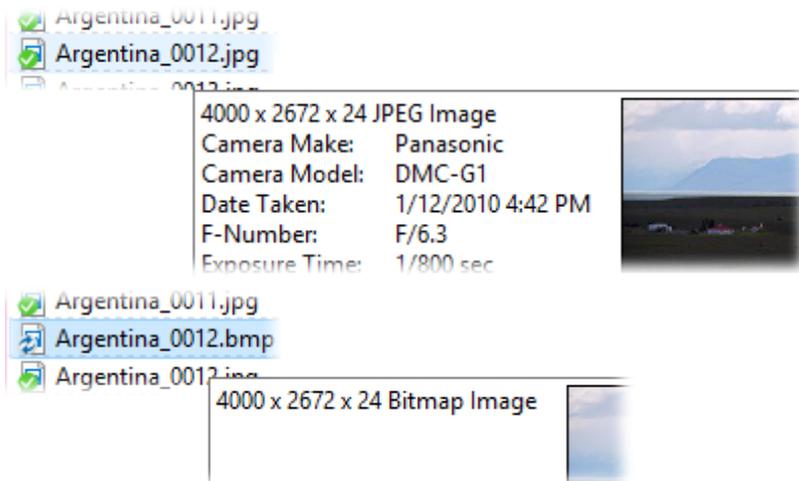
When Opus looks for an info tip to display for a file, it checks file types in the following order:

1. The specific file type for that file (e.g. for a **.jpg** file this might be the **JPEG Image** file type)
2. The file type group that contains that file extension (e.g. **Images**)
3. The **Recognized images** file type if the file is a recognized image file
4. The **All files** file type (or for a folder, **All folders**)
5. The **All files and folders** file type.

The first, most specific, info tip found is the one used. So if you want to assign an info tip to all members of a group you can edit the file type group definition, and then override it on a per file-type basis if needed.



The info tip definition (the image above illustrates the default info tip for the **Images** group) uses various `{..}` codes to insert information about the file into the info tip. Each line in the edit field corresponds to a line in the info tip. If a line in the definition uses a code that isn't valid for the specified file (for example, a **.bmp** file doesn't support EXIF metadata in it, and so fields like `{cameramake}` would be empty), the whole line is omitted from the info tip. You can have multiple codes on the one line and in that case, the line is only omitted if all codes on the line are empty.



You can see that for the **.jpg** file in the above screenshot, all the EXIF information defined in the info tip is shown, but the info tip for the **.bmp** file simply displays the top line (`{desc}` produces the *800 x 599 x 24 Bitmap Image* description and `{thumbnail}` displays the image's thumbnail).

The keywords used in info tips are the same as used by the [Rename](#) function when [renaming files using metadata](#), and the [Set](#) command when adding and removing columns to the file display. See the [Keywords for Columns](#) page for a full list of supported keywords. There are several special keywords that are specific to the info tip definition:

- `{foldersize}`: This code applies to the info tips for folders, and its use will cause Opus to calculate the total size of the folder when its info tip is displayed. This lets you display the size of a folder by simply hovering over it. You can add the **noprefix** keyword to suppress the default *Size:* prefix (for example, `{foldersize:noprefix}`).
- `{foldercontents}`: This code also applies to folders; it will result in Opus displaying the names of the first few files and sub-folders contained in the folder.

By default, both files and folders are shown. You can use the **files** and **dirs** keywords to limit the contents to just one or the other. For example: `{foldercontents:files}`

By default, *"Folders:"* and *"Files:"* prefixes are added before each list, respectively. You can use the **noprefix** keyword to suppress this. For example: `{foldercontents:files,noprefix}`

By default, each file or folder is displayed on a separate line for easy reading. You can use the **singleline** keyword to compact everything into a single line (one line for folders, another for files).

In multi-line mode, each line begins with " " (four spaces) by default. In single-line mode, each item is separated by ", ". You can use the **indent** keyword to change both of these. The **indent** keyword must be the last parameter, since it uses everything up to the end of the string. For example: **{foldercontents:indent=--> }** or **{foldercontents:singleline,indent= :: }**

To limit the maximum number of items listed in each category (dirs and files), use the **maxitems** keyword. Note that there is a hard maximum of 20 items. When there are more items than the maximum, the list will be truncated with "...". For example: **{foldercontents:files,maxitems=5}**

To limit the maximum length of each individual item, use the **maxitemlength** keyword. Note that there is a hard maximum of 260 characters. When a name is too long, it will be truncated with "...". For example: **{foldercontents:maxitemlength=20}**

- **{thumbnail}**: This code displays the thumbnail for the file if Opus is able to generate one. You can configure how the thumbnail is displayed by appending a border style value to the code:
  - **{thumbnail:0}** displays the thumbnail with no border (frame)
  - **{thumbnail:1}** displays the thumbnail with a normal border (this is the default if no value is given)
  - **{thumbnail:2}** displays a border if the thumbnail does not have an alpha channel. For 32 bit images with an alpha channel (transparency) no border is shown.
  - **{thumbnail:3}** displays no border for folders, but normal borders for files

Additionally, you can configure the size of the thumbnail. By default thumbnails will appear the same size in the info tip as they do in the file display, but you can append a size value to the code to specify a different size. Note that the border style value must also be provided if you want to provide the size. For example,

- **{thumbnail:1:512}** displays the thumbnail with normal border and 512 pixels in size (the image will be scaled to preserve the correct aspect ratio)
  - **{thumbnail:0:64}** displays the thumbnail with no border, 64 pixels in size
- **{infotip}**: This code causes Opus to display the standard, system info tip for the file (if there is one). This would be the text that is shown in the tooltip in Explorer when you hover over the file. The main use for this is to display information from third-party *tooltip shell extensions* that you may have installed.

Any text you enter into the info tip definition that isn't a **{..}** code is displayed as-is (unless it appears on a line that uses a non-applicable **{..}** code, in which case the whole line is omitted as described above). You can also use some simple HTML-style markup codes to control font styles in the info tip:

- **<#RRGGBB>...</#>**: This sets the text color for text between the tags. The color is given in [hexadecimal](#), e.g. **<#32CD32>Hello!</#>** would display the text **Hello!** in lime green.
- **<b>..</b>**: This displays text between the tags in **bold**.
- **<i>..</i>**: This displays the text between the tags in *italics*.
- **<u>..</u>**: This underlines the text between the tags.

At the bottom of the info tip page are several buttons:

- **Insert Field:** This displays a drop-down list of all the information fields (arranged by category) that you can use in info tips. This list is also automatically displayed in a pop-up menu when you press the { key in the info tip definition field. So don't worry, you don't have to memorise all of the codes!
- **Samples:** This lets you access several sample info tip definitions for reference (the samples are the defaults for the various default file type groups plus generic info tips for other files and folders).
- **Clear:** Clears the info tip definition completely.
- **Never show an InfoTip for this File Type:** If this option is on, the info tip field will be disabled and Opus will be prevented from ever showing an info tip for this type of file (hovering over the file will do nothing).

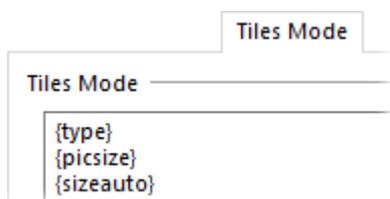
## Tiles Mode

The **Tiles Mode** page in the [file type editor](#) lets you define what is shown for files of that type when the file display is in [tiles mode](#).

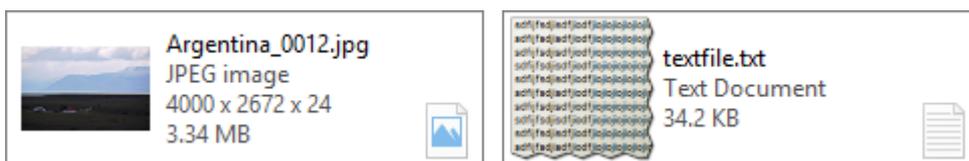
When Opus needs to display a file in tiles mode, it checks file types in the following order:

1. The specific file type for that file (e.g. for a **.jpg** file this might be the **JPEG Image** file type)
2. The file type group that contains that file extension (e.g. **Images**)
3. The **Recognized images** file type if the file is a recognized image file
4. The **All files** file type (or for a folder, **All folders**)
5. The **All files and folders** file type.

The first, most specific, file type with tiles mode text defined is the one used. So if you want to assign tiles mode text to all members of a group you can edit the file type group definition, and then override it on a per file-type basis if needed.



The tiles mode definition (the image above illustrates the default tiles mode text for the **Images** group) uses various **{..}** codes to insert information about the file into the tile's label. Each line in the edit field corresponds to a line in the label. Bear in mind that the amount of text that can be displayed on a tile is limited, but you can still use this for useful effect. You can configure the size of tiles (and therefore how much information can be displayed in the label) on the [Tiles Mode Preferences](#) page.



You can see that the **Images** group tiles mode definition has resulted in the type, image dimensions and file size being displayed in the tile for the **.jpg** file, whereas the **.txt** file only displays the type and file size (this comes from the default tile definition for the [All Files](#) file type).

The keywords used in tiles are the same as used by the [Rename](#) function when [renaming files using metadata](#), and the [Set](#) command when adding and removing columns to the file display. See the [Keywords for Columns](#) page for a full list of supported keywords. Two special keywords that are specific to the tiles mode definition are:

- **{foldersize}**: This code applies to tiles for folders, and its use will cause Opus to calculate the total size of the folder when its tile is displayed.
- **{foldercontent}**: This code also applies to folders; it will result in Opus displaying the names of the first few files and sub-folders contained in the folder.

Any text you enter into the tile definition that isn't a **{..}** code is displayed as-is. You can also use some simple HTML-style markup codes to control font styles in the tile:

- **<#RRGGBB>...</#>**: This sets the text color for text between the tags. The color is given in [hexadecimal](#), e.g. **<#32CD32>Hello!</#>** would display the text **Hello!** in lime green.
- **<b>..</b>**: This displays text between the tags in **bold**.
- **<i>..</i>**: This displays the text between the tags in *italics*.
- **<u>..</u>**: This underlines the text between the tags.

At the bottom of the tiles page are several buttons:

- **Insert Field**: This displays a drop-down list of all the information fields (arranged by category) that you can use in tiles. This list is also automatically displayed in a pop-up menu when you press the **{** key in the tile definition field. So don't worry, you don't have to memorise all of the codes!
- **Samples**: This lets you access several sample tile definitions for reference (the samples are the defaults for the various default file type groups plus generic tile definitions for other files and folders).
- **Clear**: Clears the tile definition completely.

# Scripting

The Directory Opus scripting interface lets you write scripts using any installed ActiveX-scripting language. The advantage of this is that you can leverage skills you may already have in well-known languages, rather than having to learn a potentially arcane and confusing proprietary language as in some other products. For example, VBScript and JScript are built into Windows, and many people are already familiar with them from the web. Other common languages like Perlscript and Python can be obtained from third-party providers.

With the scripting interface you can, for example:

- Query the state of Listers, tabs, paths and toolbars
- Obtain lists of files and folders and discover information about them (basic information like name, size, date modified, etc, as well as metadata like EXIF information, MP3 tags, etc)
- Write buttons and hotkeys functions purely from script code without resorting to "Rename" hacks
- Test the state of certain system settings (similar to `@ifset` in a traditional function)
- Build collections of files and run commands (Opus internal commands and external programs) on them
- Display dialogs and popup menus
- Access the clipboard, environment variables and folder aliases
- Extend the list of Opus internal commands
- Add additional file and folder information columns that can be displayed in file displays and infotips
- Automatically trigger scripts based on certain events
- Save and load configuration (Opus provides an editor that the user can use to edit your script config)

The scripting interface is present as a series of objects that export methods you can invoke and properties you can query (and sometimes set).

There are three ways to use scripts with Opus.

- [Rename Scripts](#) let you write a script within the [Advanced Rename](#) dialog that gives complete control over file renaming.
- [Script Functions](#) are scripts that are [defined directly in a button, menu or hotkey](#).
- [Script Add-ins](#) are script files that are installed in the Opus *Script Addins* folder. Whereas Script Functions are user-driven (e.g. they execute when the user clicks a button to specifically run the script), script add-ins are event driven. They provide one or more defined event handlers that Opus will invoke in certain situations, and are also used to implement [custom commands](#) and [columns](#).

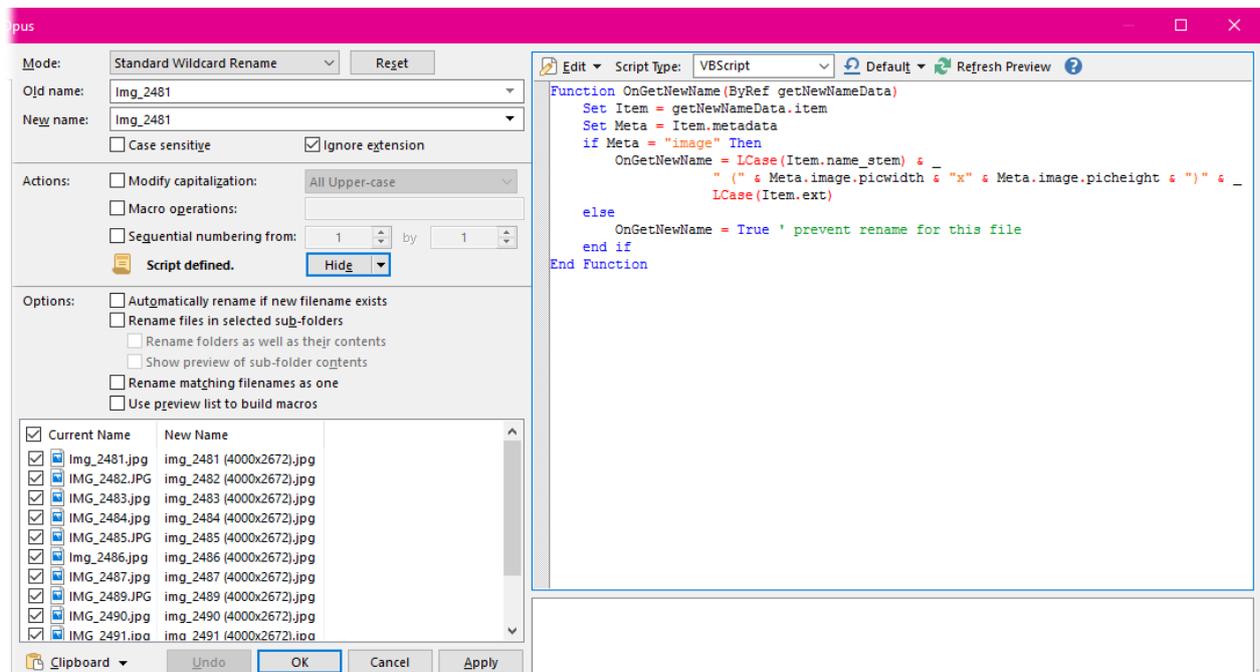
See the [Scripting Reference](#) section for a complete guide to the Opus scripting objects, and the [Example Scripts](#) section for some examples of the various types of scripts.

Any errors/warnings or text output from a script (via the **DOpus.Output** function) will be displayed in the *Other Logs* panel (part of the *Utility Panel*). You can choose the type of information to display in the log using the [Preferences / Miscellaneous / Advanced:](#) **script\_output\_level** option.

You can also use the [CLI](#) tool to design and test ad-hoc scripts.

# Rename Scripts

This is by far the most powerful, but also most complicated, feature of the [Advanced Rename](#) dialog. You can write a rename script that gives you complete control over the outcome of the rename operation.



Here's a reasonably simple example of a scripted rename. You can see that the script editor has been shown. This is initially pre-populated with a do-nothing script that you can edit.

The *Script Type* drop-down at the top is used to specify the scripting language - in the above example, *VBScript* is selected.

1. For each file to be renamed, Opus calls the [OnGetNewName](#) function.
2. The parameter passed to the [OnGetNewName](#) function is a [GetNewNameData](#) object.
3. From this we obtain the **item** property, which returns an [Item](#) object.
4. From the Item object we obtain the **metadata** property, which returns a [Metadata](#) object.
5. We check the default value of this object - if it returns "image" we know the item is an image file.
6. We build a new name consisting of the original file's name stem (the **Item.name\_stem** property, converted to lowercase by the VBScript **LCase** function).
7. We then append a string displaying image's dimensions (extracted from the **Meta.image** object which provides the **picwidth** and **picheight** properties).

8. Finally the original filename extension (**Item.ext**) is appended.
9. The new name is returned from the [OnGetNewName](#) function (the convention in VBScript, shown above, is to return a value by assigning it to the name of the function itself).
10. If the image wasn't an image file we return **True** which tells Opus to skip over that file without renaming it.

Any error messages or other text output from a script can be viewed in the *Other Logs* log window, which is accessed from the [Utility Panel](#) (or the **Logs / Other Logs** command in the **Help** menu). If you find a script isn't behaving as expected, you should check this log to see if any error messages are being generated. You can also output your own text to this log using the **DOpus.Output** script method, which can help with debugging.

The default configuration provides a *Number Files* script as one of the default Rename presets, which is implemented using VBScript, so please feel free to examine this for a more complex example. Finally, please see the [Rename Scripting section on the Opus Resource Centre](#) for example rename scripts you can download and use, and feel free to ask on the forum for help with writing rename scripts.

Opus also supports two alternative functions, for legacy reasons (so that scripts written for earlier versions of Opus will still work). We recommend all new scripts use the **OnGetNewName** event as shown above.

The original function supported was **Rename\_GetNewName**, which takes multiple arguments providing basic information about the file to be renamed:

```
Function Rename_GetNewName ( strFileName, strFilePath,
    fIsFolder, strOldName, ByRef strNewName )
    Rename_strNewName = LCase(strNewName)
End Function
```

The new name is returned in the "by-reference" variable **strNewName**. The second alternative function, **Rename\_GetNewName2**, is provided for scripting languages that don't support "by-reference" variables. The only difference is that **strNewName** is not passed by-reference, and you use the return value of the function to specify the new name. The above example using **Rename\_GetNewName2** would look something like this:

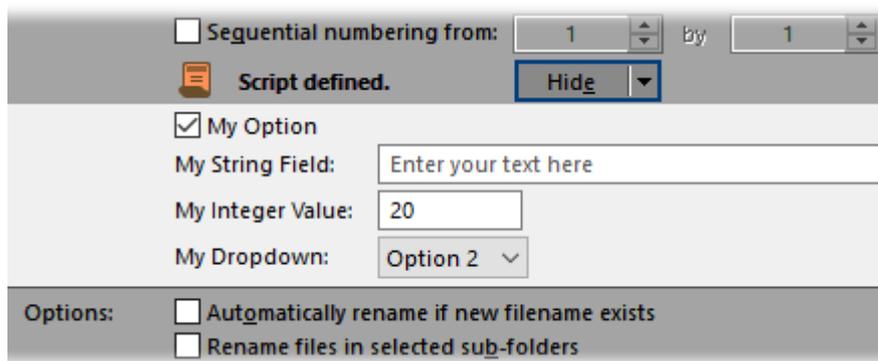
```

Function Rename_GetNewName2 ( strFileName, strFilePath,
    fIsFolder, strOldName, strNewName )
    Rename_GetNewName2 = LCase(strNewName)
End Function

```

## Custom Fields in the Rename Dialog

Rename scripts can add their own fields to the rename dialog itself, by implementing the [OnGetCustomFields](#) event. This lets you provide one or more controls that users can use to pass parameters to your script. Users can also feed parameters to your script using the new **SCRIPTARG** argument for the **Rename** command.



Custom fields can use check boxes, string fields, number fields and drop-downs.

To add custom fields from your rename script, implement the [OnGetCustomFields](#) method. The above fields were added using the following code (in VBScript):

```

Function OnGetCustomFields(ByRef getFieldData)

    ' Add the custom fields
    getFieldData.fields.my_option = True
    getFieldData.fields.my_field = ""
    getFieldData.fields.my_value = 20
    getFieldData.fields.my_combo = DOpus.Create.Vector(1, "Option 1", "Option
2",
        "Option 3")

    ' Assign labels to them
    getFieldData.field_labels("my_field") = "My String Field"
    getFieldData.field_labels("my_option") = "My Option"

```

```

getFieldData.field_labels("my_value") = "My Integer Value"
getFieldData.field_labels("my_combo") = "My Dropdown"

' Set cue text for the text field
getFieldData.field_tips("my_field") = "Enter your text here"

```

End Function

Custom fields are defined in much the same way as [Script add-in](#) defines its configuration using the [ScriptConfig](#) object. The [OnGetCustomFields](#) method is passed a **GetCustomFieldData** object. Fields are added by assigning properties of the **GetCustomFieldData.fields** object to the variable type you want the field to use (e.g. assign **True** or **False** for a Boolean, a string for a text string, etc.). The value you provide will become the default value for the field.

Each field can also have a label, and text fields can have a “cue banner” which is shown when the text field is empty (as seen above). Two [Map](#) objects are provided (**GetCustomFieldData.field\_labels** and **GetCustomFieldData.field\_tips**) which allow you to assign these.

The *Rename* dialog will expand automatically to accommodate your custom fields – obviously, screen space isn’t infinite, so you shouldn’t add too many fields or the dialog will grow too big for the screen!

The values that the user enters into your custom fields are provided to your [OnGetNewName](#) method via the [CustomFieldData](#) object passed as the **GetNewNameData.custom**. Each field you add in **OnGetCustomFields** will appear as a property of this object. For example, this function will print the provided values to the output log.

```

Function OnGetNewName (ByRef getNewNameData)
    DOpus.Output "Option:    " & getNewNameData.custom.my_option
    DOpus.Output "String:    " & getNewNameData.custom.my_field
    DOpus.Output "Number:    " & getNewNameData.custom.my_value
    DOpus.Output "Dropdown:  " & getNewNameData.custom.my_combo
    OnGetNewName = True ' skip rename
End Function

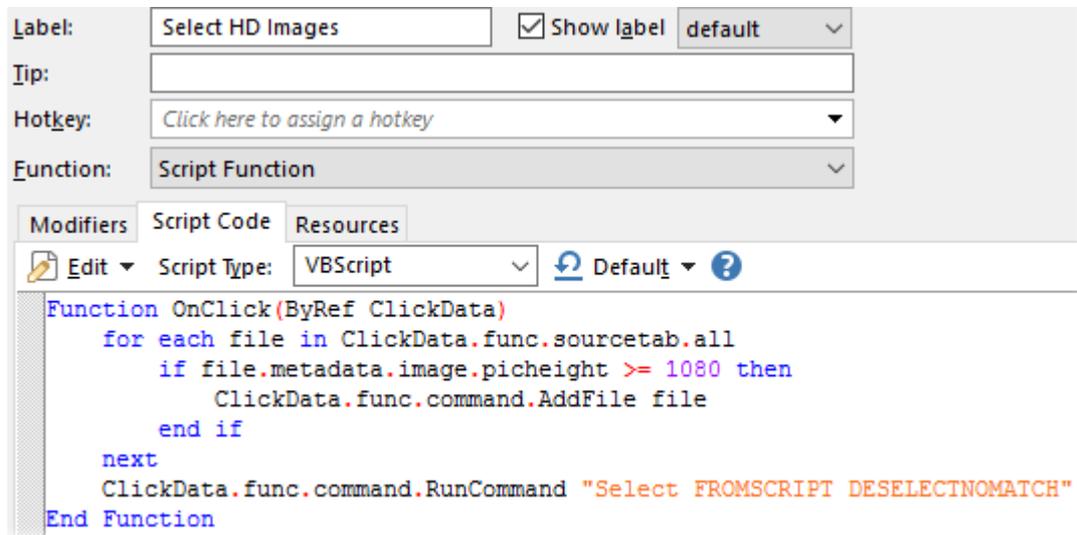
```

When the user automates the [Rename](#) command to run your rename script directly (using the **PRESET** argument), they can use the new **SCRIPTARG** parameter to pass data for your custom fields through. This argument accepts multiple *name:value* pairs. For example, assume the above script was saved as the rename preset “MyRename”. The user might run the following command:

**Rename PRESET MyRename SCRIPTARG my\_option:True my\_field:moocow**

## Script Functions

*Script Functions* are [defined directly in a button or menu](#) - they provide a third type of button function alongside *Standard Function* and *MS-DOS Batch Function*.



The screenshot above is an example of a script function that selects all "high-definition" images in the current source file display (which are defined as images with a vertical resolution greater than or equal to 1080 pixels).

The dropdown at the top of the script editor is used to specify the scripting language - here it is set to *VBScript*. The **Default** button lets you save a script "template" as the default for a particular language, and revert to the default at any time.

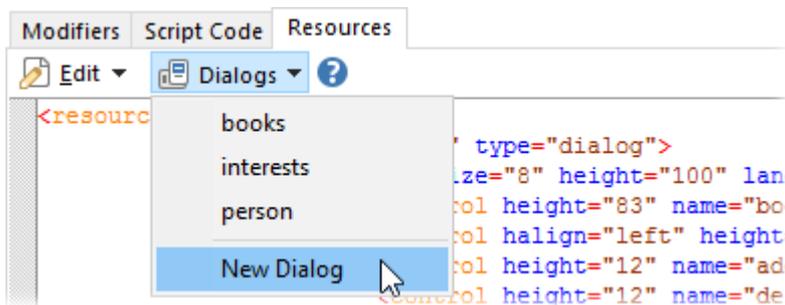
The [OnClick](#) function is a defined script entry point that Opus will call whenever your button is clicked (or hotkey is pressed). The [ClickData](#) object passed to it provides a number of properties and methods that you can use to interact with the Lister that launched the function.

When the function editor has been set to run a *Script function*, it has three separate tabs which split the function into:

- **Modifiers:** Any [command modifiers](#) that apply to the script (e.g. [@filesfromdroponly](#)).
- **Script Code:** The actual code that defines the script.
- **Resources:** Script [resources](#).

At the bottom of the function editor the **Run** button lets you test the current script immediately, without having to exit *Customize* mode. When you use the **Run** button an output panel will appear below the editor which displays any errors or script text output.

The **Resources** tab defines any [resources](#) available for the script to use. Dialogs are the main type of resource, but also supported are string resources which let you define strings in multiple languages.



While you can hand-code dialog resources in XML if you wish, it's much easier to design them using the in-built [dialog editor](#).

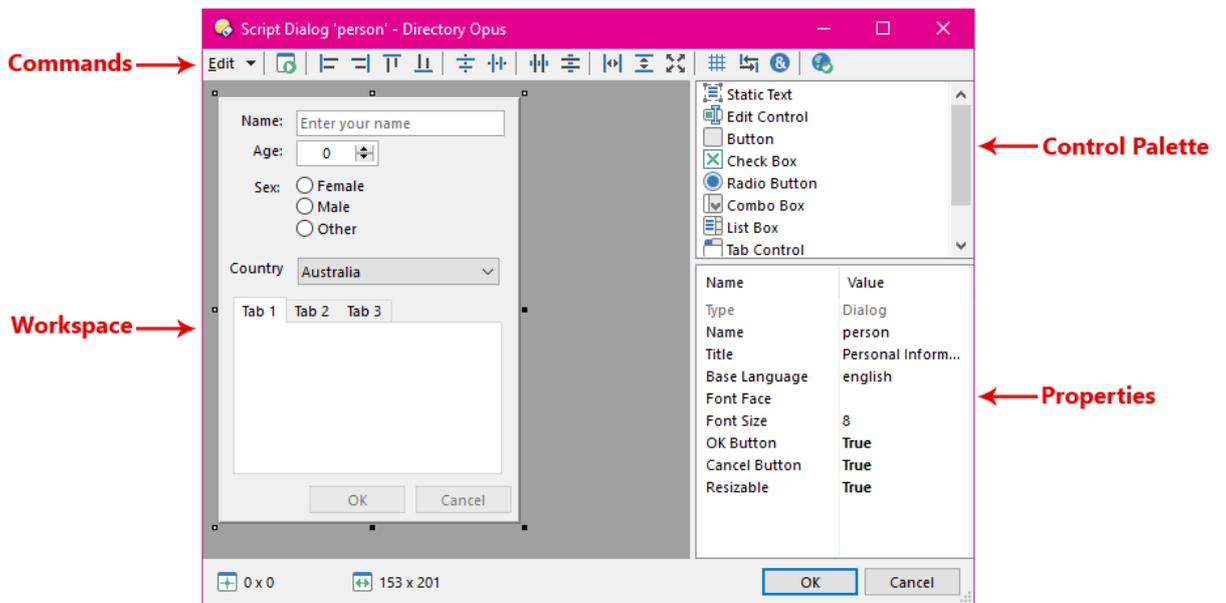
## Script Dialogs

Scripts are able to define free-form dialogs in much the same way that “proper” Windows software can, using many of the standard Windows controls.



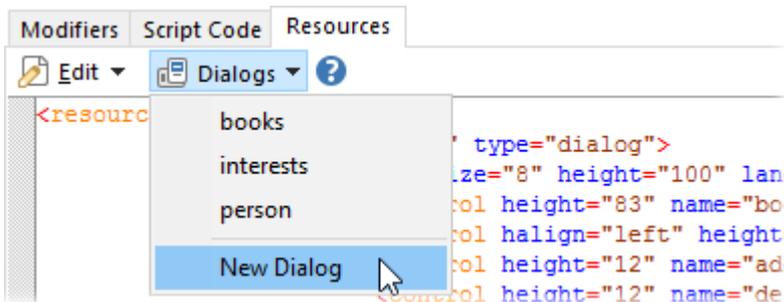
Above is an example of the type of dialog that scripts can create. Dialogs are defined as “resources” – XML formatted data that defines the dialog and control layout. Any script can have resources attached – either a script in a button, or one in the Script Add-Ins folder.

A full GUI-based dialog editor is provided inside the function editor, which makes it very easy to design script dialogs.



## Creating Script Dialogs

When you're in [Customize](#) mode and editing a button that's set as a [Script Function](#), a **Resources** tab is visible:



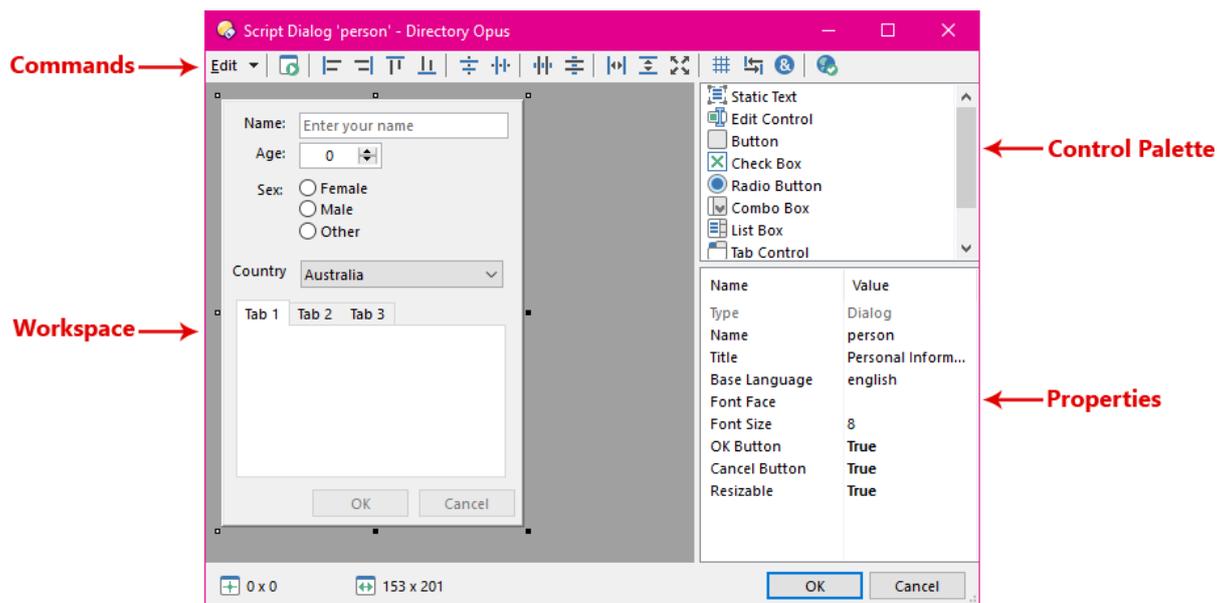
This tab lets you edit the [resources](#) associated with the script function. You can edit them in free-form XML if you like, but for dialogs you can also use the in-built dialog editor which is a much easier way of designing dialogs.

To add a script dialog, select the **Resources** tab and then click the **Dialogs** drop-down, as shown above. The **New Dialog** command will prompt you for a name for your dialog (this is the name your script will use to access it), and then display the [dialog editor](#).

You can see in the above image that three dialogs have already been defined - **books**, **interests** and **person**. You can edit existing dialogs simply by selecting them from the **Dialogs** drop-down.

## Script Dialog Editor

### *An overview of the dialog editor*



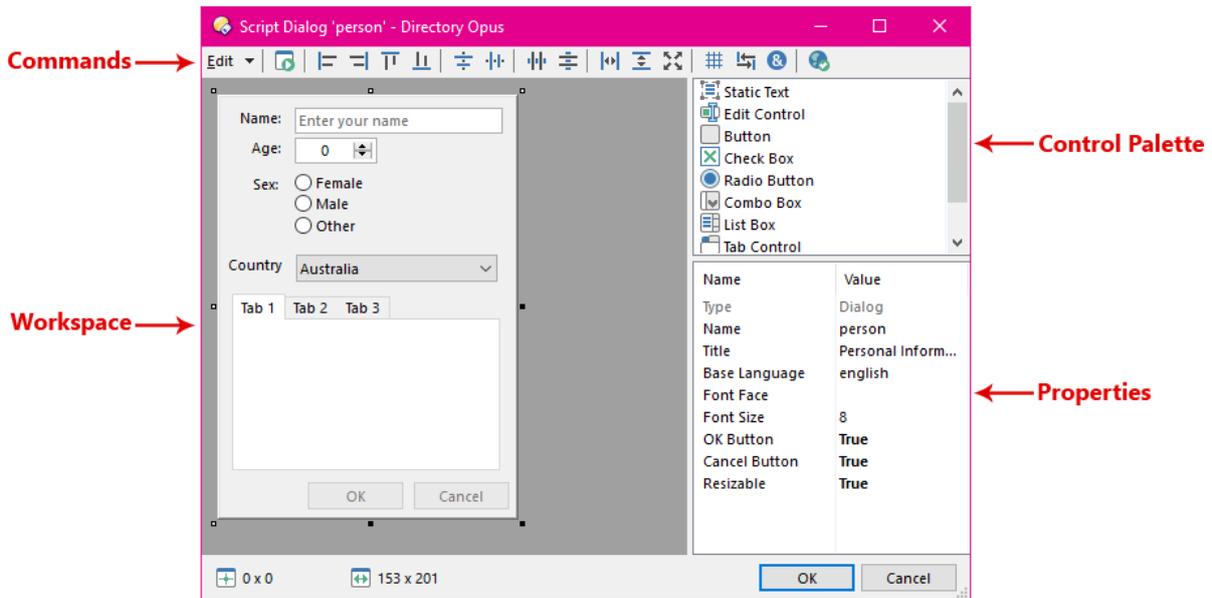
If you're familiar with the dialog editor in Visual Studio then you'll probably recognize the script dialog editor, as it was very heavily influenced by the Visual Studio original.

The four main parts of the dialog editor are:

- **Commands:** Commands for editing the dialog including positioning and alignment tools.
- **Workspace:** The main area where the dialog is designed.
- **Control Palette:** Controls you can add to the dialog.
- **Properties:** Edit the properties of the dialog and controls.

See [Creating Script Dialogs](#) for information on how to get to the dialog editor.

## Dialog Editor Commands



The commands on the dialog editor toolbar are:

- **Edit**: Contains the standard cut/copy/paste/undo edit commands.
-  (**Test**): Lets you test your design as a real dialog from within the editor.
-  (**Align Left, Right, Top, Bottom**): Align multiple controls to the same edge.
-  (**Center Horizontally, Vertically**): Center multiple controls in the dialog.
-  (**Space Evenly Across, Down**): Evenly space multiple controls.
-  (**Make Same Width, Height, Size**): Make multiple controls the same size.
-  (**Grid**): Display a grid. By default, control positions will snap to this grid. The grid size is configurable through the **Edit** menu.
-  (**Tab Order**): Define the tab order of the controls in the dialog.
-  (**Check Mnemonics**): Check for clashing **&** mnemonics in the dialog controls, and auto-assign them if wanted.
-  (**Language Overlays**): Create one or more language overlays for the dialog (lets you provide different versions of the same dialog in other languages).

## Adding Dialog Controls

To add a control to your dialog, simply click it in the control palette, and then click again on the dialog to place it. You can also drag-and-drop directly from the control palette to the workspace.

You can also make copies of existing controls using the clipboard (e.g. **Ctrl-C**, **Ctrl-V**).

## Control types

Script dialogs support a number of the standard Windows control types (other types may be added in the future):

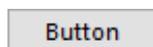
- **Static Text:** Displays a static text string. The control type is used for labels and instructions.

Static Text

- **Edit Control:** A field that you can enter text into. Supports various sub-types including single line, multiple line, password and number.



- **Button:** A push button that you can click to trigger an action.



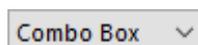
- **Check Box:** A checkbox button; can be on or off (or optionally, a third “indeterminate” state).



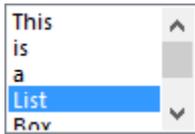
- **Radio Button:** A radio button; can be on or off. Provides mutual exclusion with other radio buttons in the same logical group.



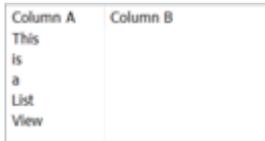
- **Combo Box:** Provides a drop-down list in its default mode. Can also provide an edit field combined with a drop-down list, or an edit field combined with a flat list.



- **List Box:** Provides a flat list that the user can select one or more strings from.



- **List View:** Similar to a List Box, but offers multiple columns and alternate display modes.



- **Tab Control:** A control that can host other dialogs. Each sub-dialog appears as a tab.

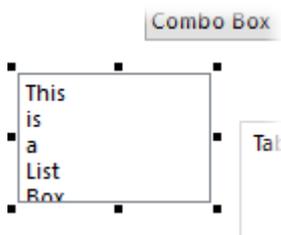


- **Group Box:** Used to draw a box around other controls, or as a header to divide two sections of a dialog.



## ***Sizing and Positioning Dialog Controls***

You can size and position controls manually (using the mouse or keyboard), and you can also size and position them automatically using the dialog layout commands (described below). The first step is to select a control by clicking it with the mouse. You can tell which control is selected because it displays “grips” around the outside, like this:



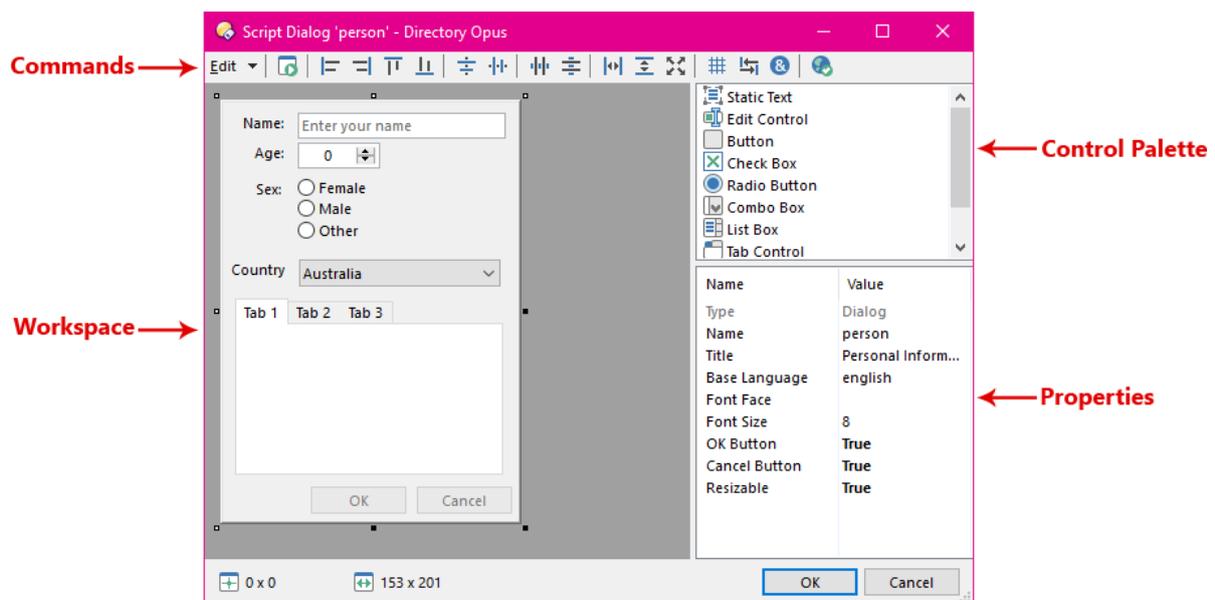
When a control is selected you can resize it by dragging with the mouse from any of the grips. If you click and drag the control itself (i.e. not on one of the grips) you can reposition it.

The dialog itself can also be resized, by clicking it (or the workspace) and resizing it using the grips.

To reposition a control with the keyboard, select it and press the cursor keys. To resize a control, hold **Shift** and press the cursor keys.

You can also move or resize multiple controls at once. To select multiple controls at once, click and drag on the dialog itself to draw a bounding box around the controls you want to select. You can also hold the **Shift** or **Control** keys down when you click to select an additional control without deselecting the ones already selected. When multiple controls are selected and you reposition them using the mouse or keyboard, they maintain their relative positions to each other.

## Using the Layout Commands



The command toolbar at the top of the dialog editor contains a number of layout commands that make it easy to position and size multiple controls.

## Alignment

The alignment commands (     ) let you align one or more controls with another control (known as the “master control”). Each command uses the designated edge of the master control’s rectangle to align the same edge of the other controls.

To use these commands, select all the controls you want to align *except* the master (either by clicking on an empty space in the dialog and dragging a selection rectangle around them, or by

**Shift-** or **Control-**clicking them). Then **Shift-** or **Control-**click on the master control so that it's the last one you select. You can distinguish the master control from the other selected controls by the color of its grips.

Once you have the controls you want aligned and the correct master control selected, click the desired alignment command.

## Sizing

The sizing commands (    ) let you make one or more controls the same size as another control (known as the “master control”). You can copy either the width, height or both width and height from the master to all other selected controls.

To use these commands, select all the controls you want to resize except the master (see the *Alignment* section above for more detail on this process). Once you have the controls selected and the correct master chosen, click the desired resize command.

## Positioning

There are two sets of positioning commands. The first (   ) is used to center one or more controls, relative to the dialog. You can center controls horizontally or vertically (if you want both you need to use both commands!). Note that when you have more than one control selected, they are repositioned as a group – that is, they maintain their relative positions to each other, and the group as a whole is centered.

The second set of positioning commands (   ) is used to space three or more controls evenly. These functions work on the group of selected controls as a whole. For example, using the *Space Evenly Down* command, the selected controls will be spaced evenly between the top of the top control and the bottom of the bottom control.

## Grid

The *Grid* (  ) option displays a regular grid on the dialog, making it much easier to align controls as you place them. When the grid is turned on, controls will snap to the grid when you move or resize them using the mouse or keyboard. To override the snap-to-grid, hold the **Alt** key down when moving or resizing controls.

The grid size can be modified using the **Grid Settings** command in the **Edit** menu.

## *Dialog Control Tab Order*

Tab order refers to the relative order of controls in a dialog when you move from one to the next by pressing the **Tab** key. It's also known as z-order (based on the controls' relative positions on a theoretical z-axis).

Tab order is also important when you have a static text control that acts as a label for another control (e.g. an edit control). A mnemonic assigned to the static text control (via a **&** in its label) will activate the control immediately following it in the tab order.

When you add a new control to a dialog it automatically takes a position after all existing controls in the tab order, but you can modify the order of existing controls using the **Tab order** command (  ).

The **Tab order** command is modal – click the button once to turn it on, and again to turn it off. When this mode is enabled all controls will display their current position in the tab order. To reorder controls, click the control you want to be first (number 1) in the tab order, then the second control, then the third, and so on. If most controls are already correct and you just want to modify one or two, hold the **Shift** key down and click the control immediately **before** the first one that you want to modify – then continue clicking controls to reorder them as normal.

### ***Dialog Control Mnemonics***

A mnemonic, or keyboard accelerator, is a key that you can press (sometimes in conjunction with **Alt**) to activate a control rather than using the mouse. Mnemonics are specified by including an ampersand character (**&**) in a control's title. For example, a button with the title **&Modify** could be activated by pushing the **M** key.

For complicated dialogs it can sometimes be hard to find unique letters in each control to use as a mnemonic. The **Check Mnemonics** command (  ) can help in two ways.

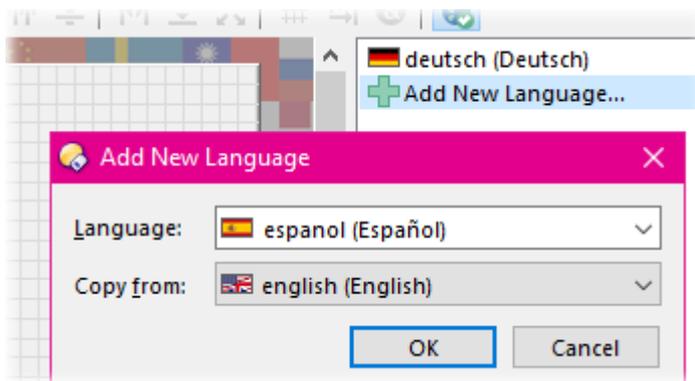
Firstly, it will tell you if the same mnemonic has been assigned to two or more controls. If any mnemonics do clash, you will be given the option to select the controls (so you can identify and fix them manually), or to automatically fix them.

Secondly, if there aren't any clashing mnemonics, but some controls do not have a mnemonic assigned at all, you'll be given the option to automatically assign them. In either case, selecting the auto-assignment option will cause the dialog editor to recalculate mnemonics for the controls in question so that (as much as possible) all controls have a unique mnemonic key.

### ***Language Overlays***

Language overlays allow you to provide translations of your dialog in other languages. A language overlay provides translations of all the text in a dialog and can optionally adjust the size and position of controls or the dialog itself (to accommodate languages which on average have longer or shorter strings than the “base language” of the dialog). When the user invokes your dialog Opus will automatically select the language overlay to match the user's current language (if one has been provided).

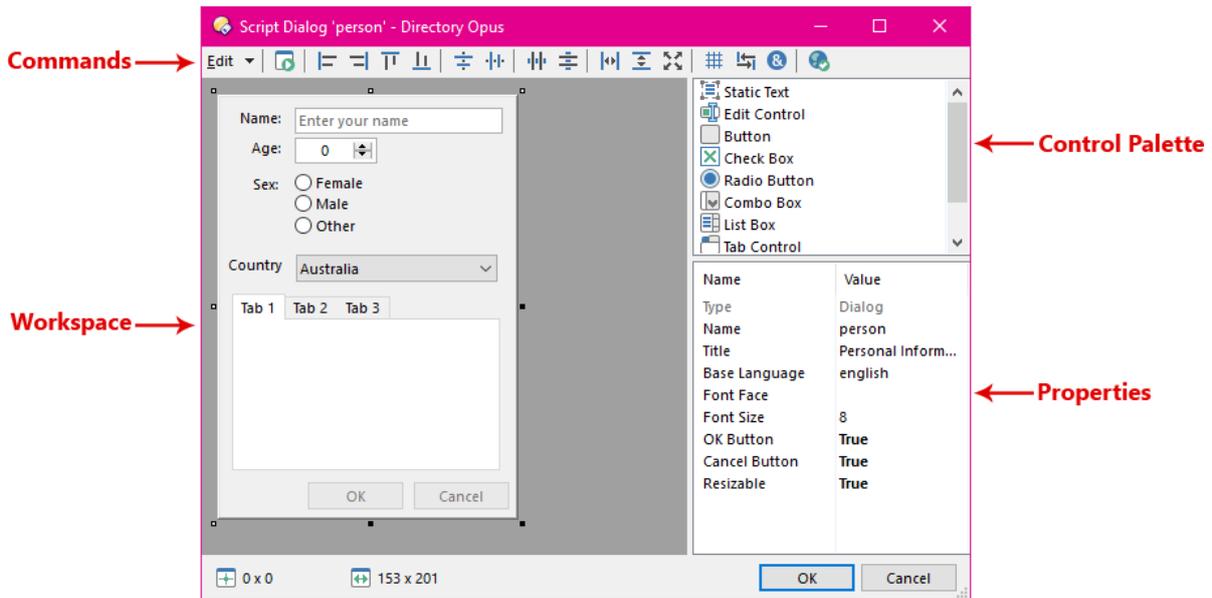
The **Language overlays** command (🌐) is modal. Turn it on to edit language overlays, and off again to go back to the regular design mode. In language overlay mode, the *Control palette* will change to display a list of configured language overlays.



To add a new overlay, click the **Add New Language** item. A dialog will prompt you to select the language you wish to create an overlay for, and also which language you want to copy the overlay from initially. To edit an existing overlay, simply select it from the list.

While editing a language overlay the properties for selected controls are reduced – only the title can be edited. You can reposition and resize controls and the dialog, but you can't add or remove any controls.

## Dialog Control Properties



Every control (and the dialog itself) has a number of properties that controls how it appears and functions. To edit the properties of a control, simply click it to select it. The properties appropriate to the control will appear in the *Properties* panel at the bottom-right of the dialog.

## Common Dialog Properties

There are several properties that all (or some) controls have in common.

- **Name:** All controls and dialogs have a name. This name will be used by your script to refer to the control, for example to read or change its value. Control names must be unique within the one dialog. When you add a control a default name will be assigned to it but you will probably want to change this to make it easier to identify the control in your script.
- **Title:** Many controls have a title, which is a string displayed in the user interface. The dialog itself has a title string which is displayed in the title bar at the top of the dialog window. For some controls (e.g. the edit control) the title string lets you provide the default value of the control.

Use an ampersand (&) in the title to give a control a mnemonic (keyboard accelerator). For example, a *Check Box* control with the label **A&utomatic** would appear as **Automatic** and allow the user to press **Alt + U** to activate the control. For controls like the *Edit control* (that don't display a title) you can assign a mnemonic using a separate *Static text* control as a label. If you want an actual ampersand to appear in the title you need to double it (e.g. **This && That**).

- **Visible:** Set to **True** if the control should initially be visible in the dialog. If set to **False**, the control won't appear until you make it visible in your script.

- **Enable:** Set to **True** if the control should initially be enabled (able to receive user input). If set to **False** the control will be disabled until you enable it in your script.
- **Resize:** Controls how the control handles resizing (if the parent dialog is set to allow resizing). The options available are:
  - **X:** The control will move relative to the width of the dialog.
  - **Y:** The control will move relative to the height of the dialog.
  - **W:** The control will grow horizontally (in width) relative to the width of the dialog.
  - **H:** The control will grow vertically (in height) relative to the height of the dialog.
- **Case:** Some controls (like the edit control) have a **Case** property which lets you force the value of the control to be all upper- or all lower-case.
- **Align Text:** Some controls (like the edit control) have this property, which lets you change the text alignment from the default (left) to center or right.

## Dialog Properties

To edit the properties of the dialog, click on it outside of any control. The specific properties applicable to the dialog are:

- **Base Language:** Specifies the “base language” of the dialog, that is the native language you’re designing it in. You can use the language overlay feature (described later) to provide additional translations of your dialog.
- **Font Face:** Use this to override the default font that the dialog uses.
- **Font Size:** Lets you control the font size used in the dialog. Dialog controls are positioned and sized using “dialog units” which are scaled with the size of the font, so you can change font size without having to re-lay out your dialog.
- **Opacity:** Sets the initial opacity of the dialog, from **255** (totally opaque) to **0** (totally transparent). The opacity can be modified after the dialog has been created using the **Dialog.opacity** property.
- **OK Button:** Set to **True** to get a default **OK** button in the bottom-right of the dialog. If the user clicks the **OK** button or pushes the **Return** key your dialog will close. This option lets you get this standard Windows UI behaviour without having to add the button and code it yourself.
- **Cancel Button:** Set to **True** to get a default **Cancel** button. If the user clicks the **Cancel** button or pushes the **Escape** key your dialog will close.
- **Resizable:** Set to **True** to allow your dialog to be resizable. If you make your dialog resizable you also need to configure the **Resize** property for any controls that you want to respond to resize events.
- **Accept Drops:** Set to **True** to allow users to drag and drop files to your dialog. Your script will receive a **drop** event along with information about the dropped files.

## Static Text Properties

The specific properties applicable to the *Static Text* control are:

- **Ellipsis:** This option configures the ellipsis (...) that will be displayed to indicate when the title string doesn't fit completely in the control. The options are:
  - **None:** No ellipsis will be displayed, text that doesn't fit will be clipped. Selecting **None** also enables word wrapping (so text will wrap to additional lines before being clipped). You can also insert line breaks manually using the `\n` character sequence.
  - **End:** If the end of a string does not fit in the rectangle, it is truncated and ellipses are added. If a word that is not at the end of the string goes beyond the limits of the rectangle, it is truncated without ellipses. Using this style will force the control's text to be on one line with no word wrap.
  - **Word:** Truncates any word that does not fit in the rectangle and adds ellipses. Using this style will force the control's text to be on one line with no word wrap.
  - **Path:** Replaces characters in the middle of the string with ellipses so that the result fits in the specified rectangle. If the string contains backslash (\) characters this style preserves as much as possible of the text after the last backslash. Using this style will force the control's text to be on one line with no word wrap.
- **Edit Control:** The static control duplicates the text-displaying characteristics of a multiline edit control. Specifically, the average character width is calculated in the same manner as with an edit control, and the function does not display a partially visible last line.
- **No Prefix:** When set to **True**, prevents interpretation of any ampersand (&) characters in the control's text as mnemonic prefix characters. These are displayed with the ampersand removed and the next character in the string underlined. This can be useful when filenames or other strings that may contain an ampersand (&) must be displayed in a static control in a dialog box.
- **Image:** When set to **True** the control will display an image file instead of a text string. The value of the **Title** property will be used as the pathname of the image to display (so in your script, you can load a new image by changing the control's title). The image will be scaled to fit the control automatically.

## Edit Control Properties

Note that the *Edit Control* uses the **Title** property to set its default value. The specific properties applicable to the *Edit Control* are:

- **Edit Type:** Sets the type of the edit control. The available types are:
  - **Single line:** Allows the entry of a single line of text.

- **Multiline:** Lets you enter multiple lines of text. When the cursor is on this type of control, pushing the **Return** key will insert a line break in the text.
- **Number:** Allows the entry of a positive, whole number.
- **Password:** Designed for password entry, this control type obscures the text you type into it.
- **Cue Text:** Specifies a “cue” string that’s displayed by the control when it’s empty (i.e. to provide a hint to the user as to what they should enter). Only available in **Single line** mode.
- **Read Only:** Set to **True** to make the control “read only” – text in it can be viewed, selected and copied, but not edited.
- **Max Length:** Sets the maximum length in characters that can be entered into the control. If empty (or set to **0**) there will be no limit on the amount of text that can be entered.
- **Up/Down Control:** Set to **True** to give a **Number** control a set of up/down arrows that the user can click to
- **Minimum Value:** Sets the minimum acceptable value the Up/Down control will allow the user to enter.
- **Maximum Value:** Sets the maximum acceptable value the Up/Down control will allow the user to enter.

## Button Properties

The specific properties applicable to the *Button* control are:

- **Default Button:** Set to **True** to make this button the “default button” for the control. The default button is the one which is triggered when the user pushes the Return key. If you have the automatic **OK** button enabled in the dialog properties, normally that button would be the default.
- **Close Dialog:** Set to **True** to make this button close the dialog when it’s clicked. If set to **False**, nothing will happen automatically when the user clicks the button, but your script will be notified.
- **Return Value:** If **Close Dialog** is set to **True**, this lets you specify the value that the dialog returns when it closes. For example, you might have a three buttons that all close the dialog – specifying a different return value lets you tell which one the user clicked after the dialog closes.

## Check Box Properties

The specific properties applicable to the *Check Box* control are:

- **Tri-state:** Set to **True** to give this check box a third state, “indeterminate”. This third state appears to the user as a solid rectangle inside the checkbox (  ). If set to **False** the check box will have only two states, *on* and *off*.
- **Check State:** Use this to control the default state of the check box. The state can also be changed programmatically by your script.

## Radio Button Properties

The specific properties applicable to the *Radio Button* control are:

- **Check State:** Sets the default state of the radio button.
- **Begin Group:** Set to **True** if this radio button is the first in a connected group of radio buttons. When one radio button is selected, all other radio buttons in the same group are automatically turned off. This property lets you have more than one group of radio buttons on the one dialog.

## Combo Box Properties

The specific properties applicable to the *Combo Box* control are:

- **Content:** Lets you specify the initial contents of the combo box control. Each value you enter (one per line) will appear in the drop-down list. You can also manipulate the contents of the combo box from your script.
- **Sort:** Set to **True** to have the contents of the combo box automatically sorted alphabetically.
- **Combo Type:** This sets the type of the combo box control. Available types are:
  - **Drop-down:** Provides a drop-down list of options that the user can select from. Does not allow the user to enter their own text.
  - **Drop-down Edit:** Provides a drop-down list of options, and also a text field that the user can type into. This lets the user choose from a predefined list or enter their own text.
  - **Simple List:** Provides a flat list of options (always displayed, rather than as a drop-down) and a text field that the user can type into.
- **Max Length:** If the combo allows the user to type their own text, this lets you specify the maximum length of text that can be entered.

## List Box Properties

The specific properties applicable to the *List Box* control are:

- **Content:** Lets you specify the initial contents of the list box control. Each value you enter (one per line) will appear in the list box. You can also manipulate the contents of the list box from your script.
- **Sort:** Set to **True** to have the contents of the list box control automatically sorted alphabetically.
- **Selection:** Controls the selection type of the list box. Options are:
  - **Single:** The user can select at most one item from the list
  - **Multiple:** The user can select more than one item by holding the **Control** key when clicking on the list.
  - **None:** Does not allow items to be selected; the list will be for display only.

## List View Properties

The specific properties applicable to the *List View* control are:

- **Content:** Lets you specify the initial contents of the list view control. Each value you enter (one per line) will appear in the list view. You can also manipulate the contents of the list view from your script.
- **Sort:** Set to **True** to have the contents of the list view control automatically sorted alphabetically.
- **Selection:** Controls the selection type of the list view. Options are:
  - **Single:** The user can select at most one item from the list
  - **Multiple:** The user can select more than one item by holding the **Control** and/or **Shift** keys when clicking on the list, or by dragging a marquee around the items with the mouse.
  - **None:** Does not allow items to be selected; the list will be for display only.
- **Columns:** Lets you add columns to the list view when it's in *Details* mode. Each value you enter (one per line) will appear as a new column in the list view.
- **View Mode:** Controls the initial display mode of the list view. The view mode can be changed later on in your script. Options are:
  - **Large Icons:** Large (32x32 nominal) icons with labels underneath.
  - **Small Icons:** Small (16x16 nominal) icons with labels to the right.
  - **List:** Similar to *Small Icons* but with a horizontal layout.
  - **Details:** Multiple columns, one item per row.
- **Edit Labels:** Set to **True** to allow the item labels (column 0) to be edited.
- **Full-row Select:** Set to **True** if you want the full width of a row in *Details* mode to be used for selection.
- **No Header:** Set to **True** to hide the column headers in *Details* mode.

- **No Sort Header:** Set to **True** to disable the ability to resort the list by clicking the column headers in *Details* mode.
- **Small Icons:** Set to **True** to enable icons in the "small" modes (*Small Icons*, *List*, *Details*). Icons can not be disabled in *Large Icon* mode.
- **Checkboxes:** Set to **Automatic** or **Manual** to enable checkboxes for each list item. In **Automatic** mode, the checkbox state is changed automatically when user clicks the checkbox. In **Manual** mode, you're notified of the click (via a **checked** event) but must change the state yourself using the **DialogListItem.checked** property.

## Tab Control Properties

The specific properties applicable to the *Tab Control* are:

- **Fixed Width:** Set to **True** to make all the tabs the same width; if set to **False** they will auto-size to fit their labels.
- **Tabs:** Specifies which dialogs will appear as tabs in this control (and the order in which they'll appear). You must create the dialogs (in the same script) before you can add them to a tab.
- **Selection:** Sets the initially selected tab index.

## Group Box Properties

The specific properties applicable to the *Group Box* control are:

- **Group Header:** Set to **True** to make this a group header (designed to separate two sections of a dialog) rather than a group box (designed to enclose a group of related controls).
- **No Prefix:** Disables the usual mnemonic (keyboard accelerator) behaviour if the control's title contains an ampersand (this lets you have a group title containing an ampersand).

## The Dialog Message Loop

There are two ways your script can manage the display of a dialog.

- **Simple:** The "simple" way displays the dialog and doesn't return control to your script until the user closes it. Dialogs displayed using the simple method can only be used as "dumb" data entry windows. Once the dialog returns your script can read the value of any controls in the dialog but it can't interact with the dialog while it's open.

- **Detached:** The other, more complex method is known as “detached”. With this method, the script displays the dialog and receives control back immediately. The script is then responsible for running a “message loop” which processes user actions in the dialog. The script can detect when the user clicks buttons or enters data, and is able to interact with the controls in the dialog to, for example, update lists, enable or disable options or display messages.

The detached method is far more flexible and you will probably want to use this method most of the time but if you only have a simple dialog for the user to enter data with, the first method is a lot easier to use.

The [Dialog](#) object is the primary object for creating and displaying script dialogs. To display a script dialog with a [Dialog](#) object, set its **template** parameter to the name of the dialog before you call the **Show** method.

## Simple Dialogs

With the simple method, the script is not responsible for the dialog’s message loop. Instead, the script prepares a [Dialog](#) object, and invokes the **Show** method to display the dialog. The **Show** method doesn’t return until the user has closed the dialog, at which time you can read the values of any controls on the dialog to find out what data the user provided.

To allow the user to close the dialog you should add one or more *Button* controls with the **Close Dialog** property set to **True**. The return value of the **Show** method will be the value you specified for the button’s **Return Value** property.

Below is an example of a script that displays a dialog using the simple method. The raw XML for the dialog is also included. If you want to experiment with this example, create a new toolbar button, and paste the script code into the **Script Code** tab of the command editor, and the resources XML into the **Resources** tab. All the following examples are written in VBScript.

## Script Code

```
Function OnClick(ByRef clickData)

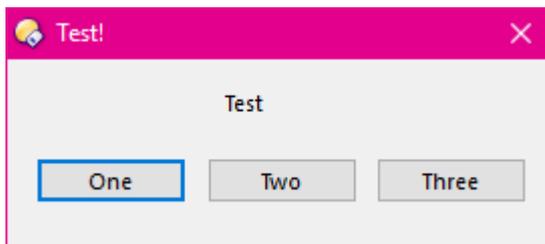
    Set Dlg = DOpus.Dlg
    Dlg.window = clickData.func.sourcetab
    Dlg.template = "testdlg"
    retVal = Dlg.Show
    DOpus.Output "Return code = " & retVal

End Function
```

## Resources

```
<resources>
  <resource name="testdlg" type="dialog">
    <dialog fontsize="8" height="58" lang="" title="Test!" width="180">
      <control halign="left" height="8" name="static1" title="Test
Dialog"
      type="static" width="35" x="72" y="9" />
      <control close="1" default="yes" height="14" name="button1"
title="One"
      type="button" width="50" x="9" y="30" />
      <control close="2" height="14" name="button2" title="Two"
type="button"
      width="50" x="66" y="30" />
      <control close="3" height="14" name="button3" title="Three"
type="button"
      width="50" x="122" y="30" />
    </dialog>
  </resource>
</resources>
```

This script creates and displays the following dialog:



In the script, the [Dialog.Show](#) method is called on the *Dlg* object, and its return value (*retVal*) corresponds to the button you click in the dialog. The value is then printed to the script output log.

When the **Show** method returns your script can read the value of any controls it may have added to the dialog – see the [Reading Dialog Control Values](#) page for an example of this.

### **Detached Dialogs**

Unlike a simple dialog, with a detached dialog the **Show** method returns to the script immediately. It's then the script's responsibility to run a message loop for the dialog (which isn't complicated in itself). It's this interactivity that means the script is able to respond to dialog control input in real time.

To create a detached dialog, set the [Dialog.detach](#) property to **True** before you call the **Show** method. Alternatively, you can call [Dialog.Create](#) instead of [Dialog.Show](#) - this creates the dialog as detached but doesn't show it immediately, allowing you to initialize its controls before calling **Show** to display it.

However you create the detached dialog, after it's been displayed your script has to run a message loop until the dialog closes. Assuming your [Dialog](#) object is called *Dlg*, a basic message loop looks like this:

## Basic Message Loop - VBScript

```
Do
    Set Msg = Dlg.GetMsg
    If Not Msg.result Then Exit Do
    ' Your code to process dialog events will go here.
Loop
```

## Basic Message Loop - JScript

```
while (true) {
    var Msg = Dlg.GetMsg();
    if (!Msg.result) break;
    // Your code to process dialog events will go here.
}
```

Although this message loop doesn't do much, this is the bare minimum needed. The [Dialog.GetMsg](#) method returns a [Msg](#) object, and that object's **result** property is **False** when the user closes the dialog, which will cause the loop to exit.

Until the dialog is closed, actions taken by the user in the dialog will generate various events which can be accessed using the various properties of the [Msg](#) object returned by the [GetMsg](#) method.

Just like in a simple dialog, *Button* controls with their **Close Dialog** property set to **True** cause the dialog to close (and the **Msg.result** property to be **False**). However, with a detached dialog, the **Show** method will have already returned before the message loop started, and long before the dialog was closed, so the result of the **Show** method can't be used to access the return value like it can with simple dialogs. Instead, the [Dialog.result](#) property can be used to find out which button the user pushed to close the dialog.

The following script code, along with the same resources from the previous example, demonstrate a simple detached dialog. If you want to experiment with this example, create a new toolbar button, and paste the script code into the **Script Code** tab of the command editor, and the resources XML into the **Resources** tab.

## Script Code - VBScript

```
Function OnClick(ByRef clickData)
    Set Dlg = DOpus.Dlg
    Dlg.window = clickData.func.sourcetab
    Dlg.template = "testdlg"
    Dlg.detach = True
    Dlg.Show
    Do
        Set Msg = Dlg.GetMsg
        If Not Msg.result Then Exit Do
        DOpus.Output "Msg Event = " & Msg.event
    Loop
    DOpus.Output "Return code = " & Dlg.result
End Function
```

## Script Code - JScript

```
function OnClick(clickData) {
    var Dlg = DOpus.Dlg;
    Dlg.window = clickData.func.sourcetab;
    Dlg.template = "testdlg";
    Dlg.detach = true;
    Dlg.Show();
    while (true) {
        var Msg = Dlg.GetMsg();
        if (!Msg.result) break;
        DOpus.Output("Msg Event = " + Msg.event);
    }
    DOpus.Output("Return code = " + Dlg.result);
}
```

## Resources

```
<resources>
  <resource name="testdlg" type="dialog">
    <dialog fontsize="8" height="58" lang="" title="Test!" width="180">
      <control halighn="left" height="8" name="static1" title="Test
Dialog"
      type="static" width="35" x="72" y="9" />
      <control close="1" default="yes" height="14" name="button1"
title="One"
      type="button" width="50" x="9" y="30" />
```

```

        <control close="2" height="14" name="button2" title="Two"
type="button"
        width="50" x="66" y="30" />
        <control close="3" height="14" name="button3" title="Three"
type="button"
        width="50" x="122" y="30" />
    </dialog>
</resource>
</resources>

```

## Reading Dialog Control Values

The primary use of a script dialog is to obtain information from the user. The [Control](#) object is used to read data from dialog controls. In a [detached dialog](#) you can do this while the dialog is open; in both a simple and a detached dialog you can also do this after the dialog has closed.

The [Control.Control](#) method is used to obtain a [Control](#) object corresponding to a dialog control. The [Control](#) method takes between one and three parameters:

**Dim dlgCtrl**

**Set dlgCtrl = Dialog.Control ( <control>, [<dialog>, [<tab>]] )**

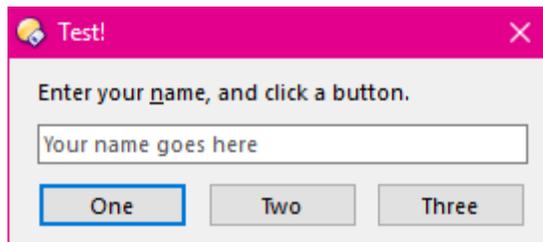
The **<control>** parameter is the name of the control, as specified in the control's properties. This parameter is required.

The other two parameters are only needed if you have any *Tab controls* on your dialog, as these can host other dialogs. The **<dialog>** parameter specifies the name of the dialog (as specified in its properties), and optionally the **<tab>** parameter specifies the name of the tab. You would only need to provide all three if you had multiple tab controls with the same child dialog used in both (an unlikely scenario).

The [Control](#) object returned can be used to read data from the corresponding dialog control. In a detached dialog this will return the current control value as long as the dialog is open, and also lets you interact with the control (e.g. your script can change the control's value after the dialog

has been created). After the user has closed the dialog the [Control](#) object can be used to obtain the final value of each control.

Below is an example of a simple (non-detached) dialog that asks you to enter your name, and then click a button to close it. The name you entered, along with the button you chose, will be displayed in the script output log.



## Script Code

```
Function OnClick(ByRef clickData)

    Set Dlg = DOpus.Dlg
    Dlg.window = clickData.func.sourcetab
    Dlg.template = "testdlg"
    Dlg.Show
    DOpus.Output "Hi, " & Dlg.Control("name").value & "!"
    DOpus.Output "Return code = " & Dlg.result

End Function
```

## Resources

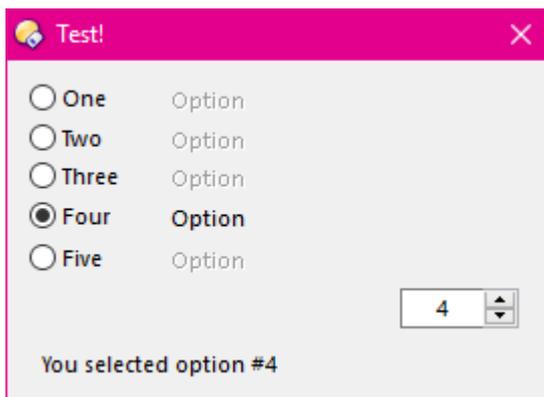
```
<resources>
  <resource name="testdlg" type="dialog">
    <dialog fontsize="8" height="58" lang="english" title="Test!"
width="180">
      <control halign="left" height="8" name="static1"
        title="Enter your &name, and click a button."
        type="static" width="159" x="9" y="6" />
      <control halign="left" height="12" name="name" tip="Your name goes
here"
        type="edit" width="163" x="9" y="20" />
      <control close="1" default="yes" height="14" name="button1"
title="One"
        type="button" width="50" x="9" y="38" />
      <control close="2" height="14" name="button2" title="Two"
type="button"
        width="50" x="66" y="38" />
      <control close="3" height="14" name="button3" title="Three"
type="button"
```

```
        width="50" x="122" y="38" />
    </dialog>
</resource>
</resources>
```

## Interacting with Dialog Controls

When a dialog is [detached](#) (by setting the [Dialog.detach](#) property to **True**) it becomes possible for your script to interact with the controls on the dialog to a much greater extent. You can read their values at any time, not just after the dialog has closed, and you can also modify their values (both after the dialog has been created and in response to control input). The various methods and properties of the [Control](#) object let you read and modify control values while a detached dialog is open.

The following (admittedly artificial!) example demonstrates the script interacting with the controls on a detached dialog.



Clicking one of the five radio buttons performs the following actions:

- The matching static text (*Option one*, etc.) will be enabled.
- The static texts corresponding to the other radio buttons will be disabled.
- The number field in the bottom-right corner will update to reflect the selected radio button.
- The static text at the bottom of the dialog will also update to reflect the selection.

Additionally, you can edit the number field in the bottom-right which will cause the appropriate radio button to be selected.

## Script Code

```
Function OnClick(ByRef clickData)

    Set Dlg = DOpus.Dlg
    Dlg.window = clickData.func.sourcetab
    Dlg.template = "testdlg"
    Dlg.detach = True
    Dlg.Show

    ' message loop
    Do
        Set Msg = Dlg.GetMsg()

        ' click message - see if it's from one of the radio buttons
        if Msg.event = "click" And Left(Msg.control, 5) = "radio" Then
            Call RadioSelected(Dlg, Msg)

        ' edit message from the "val" number input
        elseif Msg.event = "editchange" and Msg.control = "val" Then
            Call EditChanged(Dlg, Msg)

        End If
    Loop While Msg

End Function

Sub RadioSelected(ByRef Dlg, ByRef Msg)

    if Msg.data Then ' checked?

        ' loop through the radio controls
        ' they're called "radio1" through "radio5"
        ' (and the statics are named similarly)
        ' so we can do this programmatically by building the names in the loop

        for i = 1 to 5

            ' was this the radio button clicked?
            if Msg.control = "radio" & i Then

                ' enable its matching static
                Dlg.Control("static" & i).enabled = True

                ' update the text in the static at the bottom of the dialog
                Dlg.Control("seltext").label = "You selected option #" & i

                ' if (and ONLY if) we had focus, update the number field
                ' the focus test is important because it means this event came
                ' from an actual click rather than from us calling SetCheck
                below
                if Msg.focus Then Dlg.Control("val").value = i

            ' this wasn't the button clicked, so disable its matching static
            Else

                Dlg.Control("static" & i).enabled = False

            end if

        next i

    end if

End Sub
```

```

        next

    end if

End Sub

Sub EditChanged(ByRef Dlg, ByRef Msg)

    ' we only want to process this when the control has focus - otherwise
    ' the change is one we made ourselves rather than the user

    if Msg.focus and Msg.value >= 1 and Msg.Value <= 5 Then

        ' check the appropriate radio button
        Dlg.Control("radio" & Msg.Value).value = True

    end if

End Sub

```

## **Resources**

```

<resources>
    <resource name="testdlg" type="dialog">
        <dialog fontsize="8" height="105" lang="english"
title="Test!" width="180">
            <control height="10" name="radio1" title="One"
type="radio" width="25"
                x="7" y="7" />
            <control height="10" name="radio2" title="Two"
type="radio" width="25"
                x="7" y="19" />
            <control height="10" name="radio3" title="Three"
type="radio" width="30"
                x="7" y="31" />
            <control height="10" name="radio4" title="Four"
type="radio" width="27"
                x="7" y="43" />
            <control height="10" name="radio5" title="Five"
type="radio" width="26"
                x="7" y="56" />
            <control halign="left" height="8" name="static1"
title="Option one"
                type="static" width="35" x="54" y="8" enable="no"
            />
            <control halign="left" height="8" name="static2"
title="Option two"
                type="static" width="35" x="54" y="20" enable="no"
            />
            <control halign="left" height="8" name="static3"
title="Option three"
                type="static" width="40" x="54" y="32" enable="no"

```

```

/>
    <control halign="left" height="8" name="static4"
title="Option four"
    type="static" width="36" x="54" y="44" enable="no"
/>
    <control halign="left" height="8" name="static5"
title="Option five"
    type="static" width="34" x="54" y="57" enable="no"
/>
    <control halign="left" height="8" name="seltext"
type="static" width="155"
    x="11" y="89" />
    <control halign="center" height="12" name="val"
number="yes" type="edit"
    updown="yes" val_max="5" val_min="1" width="39"
x="130" y="70" />
    </dialog>
</resource>
</resources>

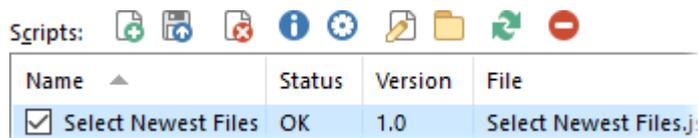
```

## Script Add-ins

*Script Add-ins* are script files that are installed in the Opus *Script Addins* folder (you can find this folder by typing `/dopusdata/Script AddIns` into the location field). Script add-ins are plain text files with an extension that specifies the script language in use (e.g. a file ending with `.js` would be a *JScript* script). One or more script add-ins can be bundled, optionally with a set of icons, in an archive called a [Script Package](#) (a zip file with the `.osp` suffix).

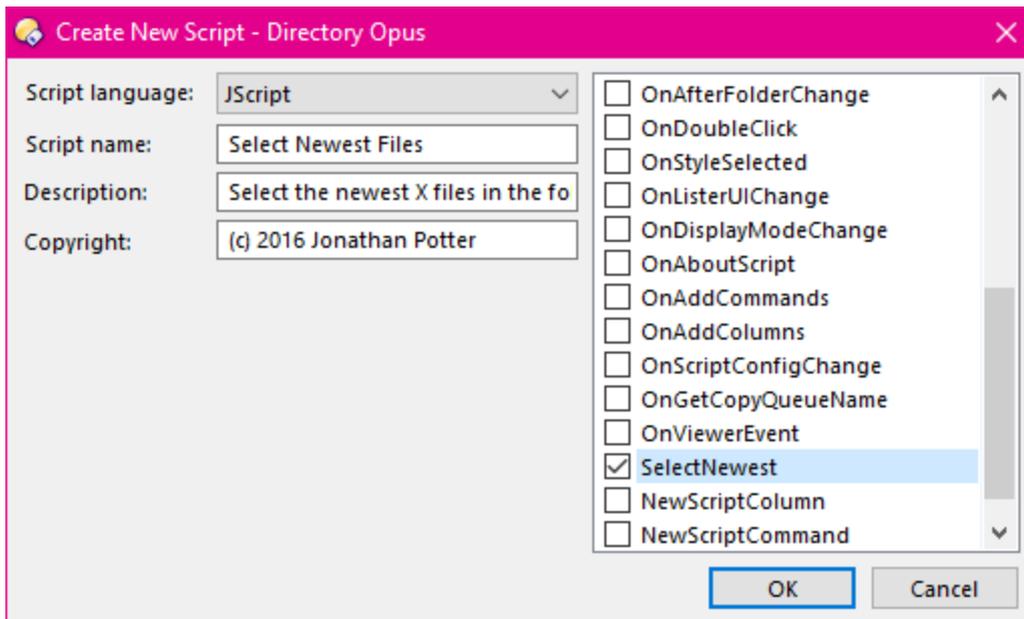
Whereas [script functions](#) are only called whenever the button they reside in is clicked, script add-ins can be invoked automatically in response to one or more [events](#). Each script can provide handling for one or more defined event entry points, and Opus will automatically call on each script that provides handling for the event in question. Some of the events that scripts are notified for include when tabs are activate or deactivated, before and after folder changes, when the view mode is changed and when Listers are opened or closed. Script add-ins can also implement [custom commands](#), which let you extend the Opus internal command set in a similar way to *User commands*, and [custom columns](#), which let you add additional information columns for files and folders.

The [Toolbars / Scripts](#) page in Preferences displays a list of any add-in scripts that have been installed. You can use the checkbox to enable or disable each script.



If you select a script in the list it reveals more information about it (a description, copyright information, and a list of the commands, columns and events that the scripts implements). The **About** and **Configure** buttons will be active if the script supports those actions. Clicking the **Edit** button will open the script in your default text editor (standalone scripts only).

You can easily install new scripts or script packages by dragging and dropping them onto the *Scripts* list. You can also create a new script using the **New Script** button in the toolbar above the list of scripts. This displays the *Create New Script* dialog:



This dialog lets you create a template for a *JScript* or *VBScript* script. Select the desired language, and enter a name, optional description and copyright string, and then use the checkboxes in the list to select the events that you want the script to create functions for.

The last two events in the list, *NewScriptCommand* and *NewScriptColumn*, let you create a script that adds internal commands or columns. When you turn this option on it will activate and let you enter a new name for the command or column. Press return to accept the new name, and another *NewScriptXXX* entry will be added to the list. In this way you can easily create a template for a script that adds multiple internal commands or columns.

## Script Package

To make it easier to distribute your scripts to others, you can bundle one or more script add-ins in an archive called a *Script Package* (a zip file with the **.osp** suffix).

You can also include a set of icons that can provide the default icon images for any internal commands your scripts add.

To make a script package:

- Create a Zip archive (leave its suffix as **.zip** for the moment).
- Copy your script files (**.js**, **.vbs**, etc) into the archive.
- When finished, rename the archive to give it a **.osp** suffix.

If you like you can add **.osp** to the list of recognized Zip file extensions on the [Zip & Other Archives / Zip Files](#) page in Preferences, to make script packages easier to deal with in Opus.

If you want to add icons that your scripts can refer to:

- Create an icon set, following the specifications in the [Icon Sets](#) reference section. *Do not zip this to a .dis file.*
- Create a sub-folder in the script package archive called **icons**.
- Copy all the files of your icon set into the **icons** sub-folder.

In your scripts, you can use the icons in your icon set as the default toolbar button images for any [internal commands](#) you add, using the [ScriptCommand.icon](#) property.

Script packages can also include external [script resource](#) files. These must be XML files with the file extension **.odxml**. You can use the [Script.LoadResources](#) method in your script to load the script resource file from the package.

## Script Resources

You may have noticed in the section on [Script dialogs](#) that scripts can have “resources” associated with them. This is XML-formatted data that provides resources to the script but doesn’t actually form part of the script code.

There are two types of script resource currently in use, dialogs and strings. When you use the command editor to design your script, the resources are split out onto a separate tab to make it easier to work with. For a *Script Add-in*, or when using the *CLI* in script mode, there is no Resources tab. Instead, script resources can be defined in two ways:

- Script resources can be included at the end of the script code itself. A separator line marks the boundary between script code and resources, like this:

```
If BlahBlah Then
    BlahBlah Blah
End If

==SCRIPT RESOURCES
<resources>
  <resource name="blah1" type="dialog">
    <dialog blah blah>
    </dialog>
  </resource>
  <resource name="blah2" type="dialog">
    <dialog blah blah>
    </dialog>
  </resource>
  <resource type="strings">
    <strings lang="blah">
      <string id="blah" text="Blah!" />
    </strings>
  </resource>
</resources>
```

Everything before the line `==SCRIPT RESOURCES` is considered part of the script code, and everything after it is the XML-formatted resources.

- Script resources can be loaded from an external file, or a raw XML string, using the [Script.LoadResources](#) method. Note that if the script is included in a package the resource file must have `.odxml` as a file extension.

## String Resources

As well as *dialog resources*, scripts can also have *string resources* which provides an easy way for a script to support languages for ad-hoc strings (that is, for strings not part of a static dialog but used programmatically by the script). Strings defined in dialogs (e.g. *Button* control labels) can be translated using the [Language overlays](#) feature.

As a very simple example, consider the following VBScript fragment.

```
If DOpus.language = "français" Then
    DOpus.Output "Bonjour!"
ElseIf DOpus.language = "deutsch" Then
    DOpus.Output "Guten Tag!"
Else
    DOpus.Output "Hello!"
End If
```

This tests the current language Opus is running in and prints an appropriate “hello” string in that language, or in English if the language isn’t known. That’s fine for a single string, but having to do that for every string in the script could be a lot of typing. Using string resources, the above script fragment reduces to:

```
DOpus.Output DOpus.Strings.Get("hello")
```

The actual strings themselves are provided as an XML resource, much the same way as dialog definitions are. Here’s the string resource that provides the above strings:

```
<resources>
  <resource type="strings">
    <strings lang="français">
      <string id="hello" text="Bonjour!" />
    </strings>
    <strings lang="deutsch">
      <string id="hello" text="Guten Tag!" />
    </strings>
    <strings lang="english">
      <string id="hello" text="Hello!" />
    </strings>
  </resource>
</resources>
```

As you can see, there are three **strings** tags, each one with a different **lang** attribute (this specifies the language). Inside each **strings** tag are one or more **string** tags which provide the actual strings. The **id** attribute is a name of your choosing – it's how your script refers to the string. The **text** attribute provides the translation of the string in the specified language.

The values for the **lang** attribute correspond to the names of the Directory Opus language libraries (which can be found in the **/home/Language** folder) – **english, deutsch, francais, cat, czech, espanol**, etc.

Note that there isn't currently a GUI for editing string resources like there is for dialogs, so you need to code the XML resources yourself if you want to use them.



# Example Scripts

The following are several example scripts:

- [Example Rename Script](#): An example of a [rename script](#) that adds the resolution of images to the filename.
- [Simple Script Function](#): A simple [script function](#) that displays the name of the largest file in the current folder.
- [Adding a new Internal Command](#): A [script add-in](#) that adds an enhanced GetSizes command to Opus.
- [Adding a new Column](#): A [script add-in](#) that adds an **Is Modified?** column to Opus.
- [Dialogs and Popup Menus](#): Example showing how to display simple dialogs and popup menus from a script.
- [Responding to Events](#): An example script add-in that responds to several events.

## Example Rename Script

This is an example of a [rename script](#) that adds an image file's resolution to its filename.

You would use this script through the [Advanced Rename](#) dialog (turn on the **Script mode** option to display the script editor). You could also [embed the script in a button](#).

```
' Set the script type to VBScript to use this script
Option Explicit

' main Rename entry point. The method is passed a GetNewNameData
object for each file

Function OnGetNewName ( ByRef GetNewNameData )

    Dim item, meta

    ' Get the provided Item object from the GetNewNameData
object's item property
    Set item = GetNewNameData.item

    ' Request its metadata. This will return a Metadata object.
    Set meta = item.metadata

    ' If the primary type of the meta data is "image" then we know
it's an image file
    If meta = "image" Then

        ' Build and return the new name
        ' This is made up of
        '     - the "name stem" (the original filename minus the
file extension)
        '     - the image width and height, obtained from the
```

```

ImageMeta object (which comes from the Metadata.image property)
' - the original filename extension
    OnGetNewName = item.name_stem & " (" & meta.image.picwidth
& "x" & meta.image.picheight & ")" & item.ext

Else

    ' The item wasn't an image, so return True to skip it
    OnGetNewName = True

End If

End Function

```

## Simple Script Function

This is an example of a [script function](#) that displays the name of the largest file in the current folder, and gives you the option to select it.

To use this function you would:

- [Create a new toolbar button](#) or hotkey and [edit](#) it
- Click the **Advanced** button in the [command editor](#) to switch to the [advanced command editor](#)
- Set the **Function** drop-down to **Script Function**

```

// Set the script type to JScript to use this script
// main script entry point. clickData is a ClickData object
function OnClick(clickData) {

    // initialise a variable to keep track of the largest file
    largestFile = null;

    // create an enumerator object that will let us easily
    enumerate the files in the current tab.
    // clickData.func is a Func object
    // func.sourcetab is a Tab object
    // sourcetab.files is a collection of Item objects
    enumFiles = new Enumerator(clickData.func.sourcetab.files);
    enumFiles.moveFirst();

    // enumerate the files in the tab
    while (enumFiles.atEnd() == false) {

        // if this is the largest file encountered so far, keep
        track of it
        // the file size is obtained from the Item object via its
        size property
        // we use the FileSize object's cy property as this makes
        it easier to compare
        // two sizes numerically
        if (largestFile == null || largestFile.size.cy <

```

```

enumFiles.item().size.cy)
    largestFile = enumFiles.item();
    enumFiles.moveNext();
}

// create a Dialog object to display the results
dlg = clickData.func.Dlg;
if (largestFile == null) {

    // there were no files found, so show a warning message
    dlg.message = "There are no files in this folder!";
    dlg.buttons = "OK";
    dlg.icon = "warn";
}
else {

    // build the info message to report on the largest file
    // the dialog will have two buttons, giving the user the
option to select the file
    dlg.message = "The largest file is '" + largestFile.name +
"' - " + largestFile.size.fmt;
    dlg.buttons = "Select It|Cancel";
    dlg.icon = "info";
}

dlg.title = "Largest File Finder";

// display the dialog. if the user clicks the "Select It"
button it will return 1
if (dlg.Show() == 1) {

    // initialise and run the Select command to select the
largest file
    // this makes use of the Command object provided to the
button via clickData.func
    clickData.func.command.ClearFiles();
    clickData.func.command.AddFile(largestFile);
    clickData.func.command.RunCommand("Select FROMSCRIPT
DESELECTNOMATCH MAKEVISIBLE");
}
else {

    // not selecting a file, so make sure anything already
selected is unaffected
    clickData.func.command.deselect = false;
}
}
}

```

## Adding a new Internal Command

This is an example of a [script add-in](#) that adds a new internal command to Opus.

This command, **SelectNewest**, selects one or more of the newest files in the current folder. You would use this by creating a new **.js** file in the *Script Addins* folder (type **/dopusdata/Script Addins** into the location field to find this).

You could then create a button or hotkey to run the **SelectNewest** command just like any other internal Opus command.

The basic procedure to add an internal command from a [script add-in](#) is:

- Either:
  - In the [OnInit](#) event, create a [ScriptCommand](#) object with the [ScriptInitData.AddCommand](#) method.
  - Or, in the [OnAddCommands](#) event, create a [ScriptCommand](#) object with the [AddCmdData.AddCommand](#) method.
- Initialize the properties of the [ScriptCommand](#) object. At a minimum you must populate the **name** and **method** properties.
- Create a separate function in your script-add in, with the **method** name you specified for the command. This will be your command's entry point.

```
// Set the script type to JScript to use this script
// The OnInit function is called by Directory Opus to initialize
the script add-in
function OnInit(initData) {

    // Provide basic information about the script by initializing
the properties of the ScriptInitData object
    initData.name = "Select Newest Files";
    initData.desc = "Select the newest X files in the folder";
    initData.copyright = "(c) 2016 Jonathan Potter";
    initData.default_enable = true;

    // Create a new ScriptCommand object and initialize it to add
the command to Opus
    var cmd = initData.AddCommand();
    cmd.name = "SelectNewest";
    cmd.method = "OnSelectNewest";
    cmd.desc = initData.desc;
    cmd.label = "Select Newest Files";
    cmd.template = "NUMBER/N, FROM/K/N, NODESELECT/S";
}

// Implement the SelectNewest command (this entry point is an
OnScriptCommand event).
// The name of this function must correspond to the value
specified for the method property when the command was added in
OnInit

function OnSelectNewest(scriptCmdData) {

    // scriptCmdData is a ScriptCommandData object, and it
provides the raw command-line and a Func object.
    // The Func object contains an Args object, which in turn
contains our parsed arguments (if any), usually preferable to
```

```

using the raw command-line.
    // we use the got_arg object to test if an argument was
    provided, before reading its value

    // First get the starting index to select from, from the FROM
    argument. Default to 0 (newest) if not supplied.
    iStartPos = 0;
    if ( scriptCmdData.func.args.got_arg.from ) iStartPos =
    scriptCmdData.func.args.from;

    // Next get the number of files to select, from the NUMBER
    argument. Default to 1 if not supplied
    iNumber = 1;
    if ( scriptCmdData.func.args.got_arg.number ) iNumber =
    scriptCmdData.func.args.number;

    // The func.sourcetab property returns a Tab object
    representing the function's source tab
    // We create an enumerator for files in the source tab
    (sourcetab.files).
    // To change this to operate on files and folders rather than
    just files, use sourcetab.all instead

    var objEnum = new Enumerator(
    scriptCmdData.func.sourcetab.files );

// build an Array of the files so we can sort it
var objFiles = new Array();
i = 0;
while (!objEnum.atEnd()) {
    objFiles[i++] = objEnum.item();
    objEnum.moveNext();
}

if (objFiles.length > 0) {

// Sort the array by the "modify" property. This will sort the
file array
    // in order from newest to oldest
    objFiles.sort(function(a, b) {
        If (a.modify < b.modify)
            return 1;
        If (a.modify > b.modify)
            return -1;
        return 0;
    });

// By default we get given a Command object pre-filled for the
current tab
    // we can use that instead of creating our own, but we need
    to clear
    // out its list of files if it has one already
    scriptCmdData.func.command.ClearFiles();

    // Starting from the specified start position, add the
    requested number
    // of files to the command object. Because we have sorted
    the array from newest
    // to oldest, this will automatically add the newest files
    to the command

```

```

        for ( i = iStartPos; i < objFiles.length && i < iStartPos
+ iNumber; i++ ) {
            scriptCmdData.func.command.AddFile( objFiles[i] );
        }

        // Build the Select command line to run that will actually
select the files.
        // The FROMSCRIPT argument is needed to select files from
the command object.
        // The DESELECTNOMATCH argument is used to deselect all
other files, unless the
        // script's NODESELECT argument was used.

        strCommand = "Select FROMSCRIPT";
        if (!scriptCmdData.func.args.got_arg.nodeselect)
            strCommand += " DESELECTNOMATCH";

// run the Select command via the Command object's RunCommand
method
        scriptCmdData.func.command.RunCommand( strCommand );
    }
}

```

## Adding a new Column

This is an example of a [script add-in](#) that adds a new information column to Opus.

This column, **Is Modified?**, indicates whether a file has been modified since it was created. This is performed by simply comparing whether the item's creation and modification dates are the same.

- For files that have been modified, the column displays *Yes*, and for files that haven't been modified it displays *No*. Nothing is displayed for folders.
- When sorting by this column, modified files are displayed above non-modified files.
- When grouping by this column, the two groups are labeled *Modified* and *Never Modified*.

The basic procedure to add a custom column from a [script add-in](#) is:

- Either:
  - In the [OnInit](#) event, create a [ScriptColumn](#) object with the [ScriptInitData.AddColumn](#) method.
  - Or, in the [OnAddColumns](#) event, create a [ScriptColumn](#) object with the [AddColData.AddColumn](#) method.
- Initialize the properties of the [ScriptColumn](#) object. At a minimum you must populate the **name** and **method** properties.
- Create a separate function in your script-add in, with the **method** name you specified for the column. This method will be called when Opus wants to retrieve data for your column.

```

// Set the script type to JScript to use this script
// The OnInit function is called by Directory Opus to initialize
the script add-in
function OnInit(initData) {

    // Provide basic information about the script by initializing
the properties of the ScriptInitData object
    initData.name = "Is Modified Column";
    initData.desc = "Adds a column that indicates if a file has
ever been modified.";
    initData.copyright = "(c) 2014 Jonathan Potter";
    initData.default_enable = true;

    // Create a new ScriptColumn object and initialize it to add
the column to Opus
    var cmd = initData.AddColumn();
    cmd.name = "IsModified";
    cmd.method = "OnIsModified";
    cmd.label = "Is Modified?";
    cmd.autogroup = false; // we provide our own grouping
information
    cmd.autorefresh = true; // refresh column when files
change
    cmd.justify = "center";
    cmd.match.push_back("Yes"); // when searching, these are the
only two options
    cmd.match.push_back("No");

}

// Implement the IsModified column (this entry point is an
OnScriptColumn event).
// The name of this function must correspond to the value
specified for the method property when the column was added in
OnInit
function OnIsModified(scriptColData) {

    // scriptColData is a ScriptColumnData object. first check that
this is the right column (it should be since we only added one)
    if (scriptColData.col != "IsModified")
        return;

    // we don't provide a value for folders since their timestamps
don't mean much
    // we set the sort key to 3 so they come last (in case the
user is mixing files and folders)
    if (scriptColData.item.is_dir) {

scriptColData.value = "";
        scriptColData.sort = 3;
    }

    // for files, are the two dates the same?
    else if (new Date(scriptColData.item.create).valueOf() == new
Date(scriptColData.item.modify).valueOf()) {

```

```

scriptColData.value = "No";
    scriptColData.group = "Never Modified";
    scriptColData.sort = 2;
}

// the dates are different, therefore modified
else {

scriptColData.value = "Yes";
    scriptColData.group = "Modified";
    scriptColData.sort = 1;
}
}

```

## Adding a new Column from Shell Properties

Directory Opus has script support for Windows shell properties, making it easy for a script to enumerate properties in the system and retrieve the properties for a file.

- Use **FSUtil.GetShellPropertyList** to retrieve a list of properties (optionally matching a wildcard pattern).
- Use **FSUtil.GetShellProperty** to get the value of one or more properties for a file.
- Use **Item.ShellProp** to get the value of a single property for the item.

Below is an example script add-in that adds columns to Opus that show the value of shell properties for DWG (AutoCAD) files added by a third party tool.

```

Function OnInit(initData)
    initData.name = "DWG Columns"
    initData.desc = "Adds DWG Columns from the JTB World
extension"
    initData.copyright = "(c) 2016 jpotter"
    initData.version = "1.0"
    initData.default_enable = true
    initData.min_version = "12.0.8"

    Dim props, prop, col
    Set props = DOpus.FSUtil.GetShellPropertyList("dwg.*", "r")

    for each prop in props

        Set col = initData.AddColumn
        col.name = prop.raw_name
        col.method = "OnDWGColumn"
        col.label = prop.display_name
        col.justify = "left"
        col.autogroup = true
        col.userdata = prop.pkey

    next

End Function

```

```

Function OnDWGColumn(scriptColData)

    scriptColData.value =
scriptColData.item.shellprop(scriptColData.userdata)

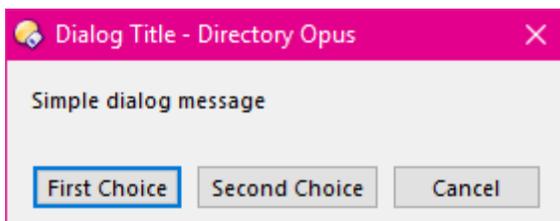
End Function

```

## Simple Dialogs and Popup Menus

The [Dialog](#) object makes it easy for scripts to display various types of dialogs and popup menus. Here are a few examples of its use. You can try all of these in the [CLI](#) set to **script mode** (with the type set to *VBScript*).

### *A simple message dialog with three buttons:*



```

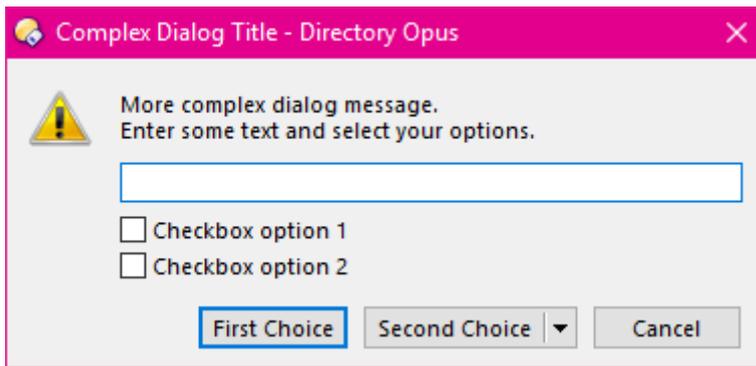
' Create a Dialog object. You can also obtain dialog objects from
the Lister, Tab, Func and Command objects.
Set dlg = DOpus.Dlg

' Initialise the object to display a simple message with three
buttons.
dlg.window = DOpus.Listers(0)
dlg.message = "Simple dialog message"
dlg.title = "Dialog Title"
dlg.buttons = "First Choice|Second Choice|Cancel"

' Show the dialog and print the result to the script log
ret = dlg.Show
DOpus.Output "Dialog.Show returned " & ret

```

### *A more complex dialog with an icon, a drop-down button, a text field and two checkbox options:*



```

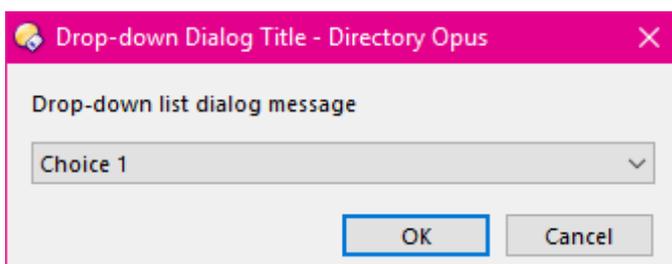
' Create a Dialog object.
Set dlg = DOpus.Dlg

' Initialise the object to display a message with three buttons,
one of which is a drop-down with multiple choices
' This dialog also has a warning icon, a text field allowing the
user to enter a line of text,
' and two checkboxes which the user can turn on or off.
dlg.window = DOpus.Listers(0)
dlg.message = "More complex dialog message." & vbCrLf & "Enter
some text and select your options."
dlg.title = "Complex Dialog Title"
dlg.buttons = "First Choice|Second Choice+Third Choice+Fourth
Choice|Cancel"
dlg.icon = "warn"
dlg.max = 128 ' enable the text field
dlg.options(0).label = "Checkbox option 1"
dlg.options(1).label = "Checkbox option 2"

' Show the dialog and print the results to the script log
ret = dlg.Show
DOpus.Output "Dialog.Show returned " & ret
DOpus.Output "The string you entered was " & dlg.input
DOpus.Output "The two checkboxes were set to " &
dlg.options(0).state & " and " & dlg.options(1).state

```

***A dialog with a drop-down list the user can select from:***



```

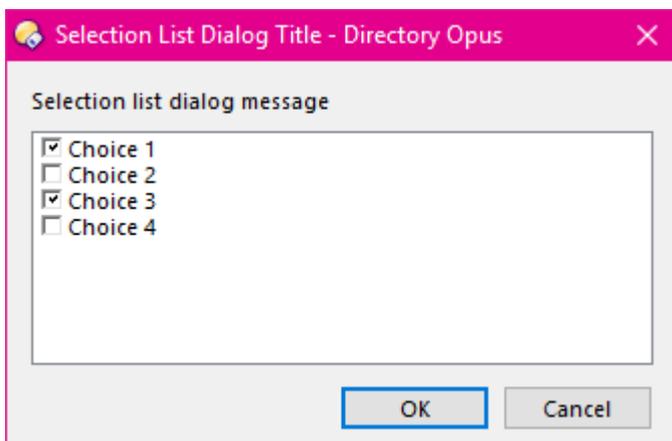
' Create a Dialog object.
Set dlg = DOpus.Dlg

' Initialise the object to display a drop-down control that lets
the user pick one option from a list of many
dlg.window = DOpus.Listers(0)
dlg.message = "Drop-down list dialog message"
dlg.title = "Drop-down Dialog Title"
dlg.buttons = "OK|Cancel"
dlg.choices = DOpus.Create.Vector("Choice 1", "Choice 2", "Choice
3", "Choice 4")
dlg.selection = 0

' Show the dialog and print the results to the script log
ret = dlg.Show
DOpus.Output "Dialog.Show returned " & ret
DOpus.Output "The selected choice was " &
dlg.choices(dlg.selection)

```

***A dialog with a list of items the user can turn on or off:***



```

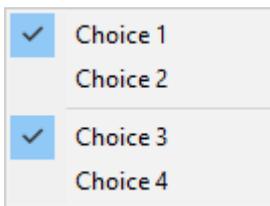
' Create a Dialog object.
Set dlg = DOpus.Dlg

' Initialise the object to display a list of items the user can
turn on or off
dlg.window = DOpus.Listers(0)
dlg.message = "Selection list dialog message"
dlg.title = "Selection List Dialog Title"
dlg.buttons = "OK|Cancel"
dlg.choices = DOpus.Create.Vector("Choice 1", "Choice 2", "Choice
3", "Choice 4")
dlg.list = DOpus.Create.Vector(True, False, True, False)

' Show the dialog and print the results to the script log
ret = dlg.Show
DOpus.Output "Dialog.Show returned " & ret
For i = 0 To dlg.choices.size - 1
    DOpus.Output dlg.choices(i) & " state was " & dlg.list(i)
Next

```

***A popup menu the user can select from:***



```

' Create a Dialog object.
Set dlg = DOpus.Dlg

' Initialise the object to display a popup menu the user can
select from
dlg.window = DOpus.Listers(0)
dlg.choices = DOpus.Create.Vector("Choice 1", "Choice 2", "-",
"Choice 3", "Choice 4")
dlg.menu = DOpus.Create.Vector(2, 0, 0, 2, 0)

' Show the menu and print the results to the script log
ret = dlg.Show
DOpus.Output "Dialog.Show returned " & ret

```

## Script Dialog Example

This is an example of a [script dialog](#); it implements the dialog shown at the start of the [Script Dialogs](#) section. It makes use of a *Tab control* to host two child dialogs, and also demonstrates how to add and remove strings from a *List Box* control.



The *Tab control* has been configured to host two child dialogs; *Interests* and *Favorite Books*. Clicking the tabs lets you switch back and forth between the two pages.

Most of the dialog's controls don't actually do anything – only the *Favorite Books* tab has been implemented. If you switch to that tab you can enter a string in the *Enter title* field and click the **Add** button to add it to the list. Click an item in the list and click the **Delete** button to remove it.

This dialog also provides an example of a resizable dialog. You can make any dialog resizable, but it's particularly useful when working with tabs, as it means the *Tab control* and the parent dialog can automatically size to fit the content displayed inside the tabs.

If you want to experiment with this example, create a new toolbar button (with the type to *Script Function*) and paste the script code into the **Script Code** tab of the command editor, and the resources XML into the **Resources** tab. All the following examples are written in VBScript.

## Script Code

```
Function OnClick(ByRef clickData)

    Set dlg = DOpus.Dlg
    dlg.window = clickData.func.sourcetab
    dlg.template = "person"
    dlg.detach = true
    dlg.Show

    Do
        Set msg = dlg.GetMsg()

        Select Case msg.control

            Case "add"

                ' get the string the user has entered
                str = dlg.Control("title", "books").value

                if Len(str) > 0 Then

                    ' add to the list box and clear the edit field
                    dlg.Control("books", "books").AddItem(str)
                    dlg.Control("title", "books").value = ""

                End If

            Case "del"

                ' get the selection from the list box
                sel = dlg.Control("books", "books").value
                if sel > -1 Then

                    ' remove it from the list
                    dlg.Control("books", "books").RemoveItem(sel)

                End If

            End Select

        Loop While msg

    End Function
```

## Resources

```
<resources>
  <resource name="books" type="dialog">
    <dialog fontsize="8" height="100" lang="english" resize="yes"
      title="Favorite Books" width="180">
      <control height="83" name="books" resize="wh" type="listbox"
width="176"
        x="2" y="2" />
      <control halign="left" height="12" name="title" resize="yw"
        tip="Enter title and press Add" type="edit" width="101" x="2"
y="86" />
      <control height="12" name="add" resize="xy" title="Add"
type="button"
        width="34" x="106" y="86" />
      <control height="12" name="del" resize="xy" title="Delete"
type="button"
        width="35" x="143" y="86" />
    </dialog>
  </resource>
  <resource name="interests" type="dialog">
    <dialog fontsize="8" height="100" lang="english" resize="yes"
      title="Interests" width="180">
      <control halign="left" height="96" multiline="yes" name="interests"
        readonly="yes" resize="wh" title="Nothing here yet :)\n"
type="edit"
        width="176" x="2" y="2" />
    </dialog>
  </resource>
  <resource name="person" type="dialog">
    <dialog fontsize="8" height="201" lang="english" resize="yes"
      standard_buttons="ok, cancel" title="Personal Information"
width="153">
      <control halign="left" height="8" name="static1" title="Name:"
type="static"
        width="21" x="11" y="7" />
      <control halign="left" height="12" name="name" tip="Enter your
name"
        type="edit" width="106" x="39" y="6" />
      <control halign="left" height="8" name="static2" title="Age:"
type="static"
        width="15" x="17" y="21" />
      <control halign="center" height="12" name="age" number="yes"
        title="Edit Control" type="edit" updown="yes" val_max="50"
val_min="10"
        width="42" x="39" y="20" />
      <control halign="left" height="8" name="static3" title="Sex:"
type="static"
        width="14" x="18" y="38" />
      <control height="10" name="f" title="Female" type="radio"
width="34"
        x="39" y="37" />
      <control height="10" name="m" title="Male" type="radio" width="26"
        x="39" y="47" />
      <control height="10" name="o" title="Other" type="radio" width="29"
        x="39" y="57" />
      <control halign="left" height="8" name="static4" title="Country:"
        type="static" width="27" x="5" y="76" />
      <control height="40" name="country" type="combo" width="104" x="39"
y="75">
      <contents>
```

```

        <item text="Australia" />
        <item text="UK" />
        <item text="USA" />
        <item text="Spain" />
        <item text="France" />
        <item text="Germany" />
        <item text="China" />
        <item text="Japan" />
    </contents>
</control>
<control height="84" name="tab1" resize="wh" type="tab" width="138"
    x="7" y="94">
    <tabs>
        <tab dialog="interests" />
        <tab dialog="books" />
    </tabs>
</control>
</dialog>
</resource>
</resources>

```

## Responding to Events

[Script add-ins](#) are invoked in response to one or more [events](#).

This is an example of a script add-in that responds to two different events. You would use this by creating a new **.vbs** file in the *Script Addins* folder (type **/dopusdata/Script Addins** into the location field to find this).

This example responds to both folder change ([OnAfterFolderChange](#)) and view mode change ([OnDisplayModeChange](#)) events.

In both cases, it checks if the current folder is the *Pictures* library (**lib://Pictures** or a child folder). If so, it automatically changes the sort mode based on the current view mode:

- If the display is set to *Thumbnails* mode, the list will be sorted by date
- Otherwise, the list will be sorted by name

```

' Set the script type to VBScript to use this script
' The OnInit event is called by Directory Opus to initialize the
script
Function OnInit(ByRef initData)

' Provide basic information about the script
    initData.name = "Example Event Script"
    initData.desc = "Script that shows an example of responding to
events"
    initData.copyright = "(c) 2016 Jonathan Potter"
    initData.default_enable = True

```

```
End Function
```

```
' The OnAfterFolderChange event is called after a folder has been read
```

```
Function OnAfterFolderChange (ByRef afterFolderChangeData)
```

```
' The result property of the AfterFolderChangeData object tells us if the read was successful
```

```
    If afterFolderChangeData.result Then
```

```
' Check if the path that was read begins with lib://Pictures
```

```
    If Left(afterFolderChangeData.tab.path, 14) = "lib://Pictures" Then
```

```
        ' The path matched, so call our CheckPictureSorting subroutine to update the sort mode if needed
```

```
        ' We pass the Tab that the folder was read in as a parameter to the subroutine
```

```
        Call CheckPictureSorting(afterFolderChangeData.tab)
```

```
    End If
```

```
End If
```

```
End Function
```

```
' The OnDisplayModeChange event is called whenever the view mode is changed
```

```
Function OnDisplayModeChange (ByRef displayModeChangeData)
```

```
' The DisplayModeChangeData object gives us the Tab that the mode was changed in
```

```
' See if the folder currently displayed in the tab is lib://Pictures or a sub-folder
```

```
    If Left(displayModeChangeData.tab.path, 14) = "lib://Pictures" Then
```

```
        Call CheckPictureSorting(displayModeChangeData.tab)
```

```
    End If
```

```
End Function
```

```
' This is a subroutine called by both the above events, and is passed a Tab object as a parameter.
```

```
' It creates a Command object and uses the IsSet method to see if the current view mode in the tab is Thumbnails
```

```
'     - If so it uses the RunCommand method to sets the sort order to "modified date"
```

```
'     - If not it sets the sort order to "name"
```

```
Sub CheckPictureSorting (ByRef objTab)
```

```
    Set objCmd = DOpus.CreateCommand
```

```
    objCmd.SetSourceTab(objTab)
```

```
    If objCmd.IsSet("VIEW=Thumbnails") Then
```

```
        objCmd.RunCommand("Set SORTBY=modified")
```

```
    Else
```

```
        objCmd.RunCommand("Set SORTBY=name")
    End If
    Set objCmd = Nothing

End Sub
```



# Reference

This section contains information about the more technical side of Opus - the internal commands and arguments, wildcard pattern matching syntax, codes for configuring the status bar, scripting, etc.

Please treat this section as a reference guide only - it is not intended to be read as a tutorial for raw beginners. Most pages in this section will link back to the main help file where concepts are introduced and explained in much greater detail.

# Wildcard Reference

There are two types of wildcard (pattern matching) systems used in Opus.

- [Standard pattern matching](#) syntax is a simple wildcard system - if you're familiar with typing \*.doc to mean all document files, you should start with this.
- [Regular expressions](#) are a more powerful but also more complicated system that let you match much more complex patterns. They also let you perform search and replace operations, which lets you perform complex batch renames via the [Advanced Rename](#) function.

## Pattern Matching Syntax

Wildcard patterns can be used in many functions in Opus, including:

- When searching for a file using the [Find Files](#) tool (matching both filename or file contents or metadata)
- When [filtering](#) files in and out of the display
- When [selecting](#) files in the file display
- When [filtering file operations](#) like copy or move

As well as [regular expressions](#), Opus supports a more simple wildcard system, called the *standard pattern matching* system. When using wildcards, you specify a pattern consisting of a mix of literal text and special *wildcard tokens*. Each token is used to match one or more characters in the target string. As a simple example, the pattern \*.doc matches anything that ends with the literal characters .doc (or in other words, document files).

Token	Description
#	<p>The character or expression following the # is repeated 0 or more times.</p> <p>For example:</p> <p><b>a#xb</b> matches <i>ab</i>, <i>axb</i>, <i>axxb</i>, <i>axxxb</i>, etc. <b>a#(xyz)b</b> matches <i>ab</i>, <i>axyzb</i>, <i>axyzxyzb</i>, etc.</p>
?	<p>Matches any single character.</p> <p>For example:</p> <p><b>a?b</b> matches <i>aab</i>, <i>abb</i>, <i>acb</i>, <i>adb</i>, etc. It does <b>not</b> match <i>ab</i>.</p>

	<p>Used to separate multiple expressions, any of which may match (effectively an <b>or</b> operator).</p> <p>For example:</p> <p><b>a(x y z)b</b> matches <i>axb</i>, <i>ayb</i> and <i>azb</i>.  <b>*.(gif bmp jpg)</b> matches anything ending with <i>.gif</i>, <i>.bmp</i> or <i>.jpg</i>.</p>
~	<p>This is the <b>not</b> operator, it negates the following expression.</p> <p>For example:</p> <p><b>~(*.doc)</b> matches anything that doesn't end in <i>.doc</i>  <b>~(*.(gif bmp))</b> matches anything that doesn't end in <i>.gif</i> or <i>.bmp</i></p>
()	<p>Parentheses are used to combine multiple characters into an expression. If we take the above example of <b>a#(xyz)b</b>, the parentheses are used to form <i>xyz</i> into a single expression. Without the brackets (i.e. <b>a#xyzb</b>) only the <i>x</i> would be seen to follow the <b>#</b> character - <i>yzb</i> would be treated literally.</p> <p>Parentheses can be nested (as in the above example for <b>~</b>) to combine multiple expressions into a larger expression.</p> <p>For example:</p> <p><b>NEW-((*.doc) (*_backup_*))</b> matches anything starting with <i>NEW-</i> that either ends in <i>.doc</i>, or is followed by the string <i>_backup_</i>.</p>
[]	<p>Matches any single character in the set of specified characters.</p> <p>You can specify the character set as individual characters (e.g. <b>[abdfg]</b>) or as a range of characters (e.g. <b>[a-j]</b>) or as multiple ranges.</p> <p>For example:</p> <p><b>[abc]</b> matches either <i>a</i>, <i>b</i> or <i>c</i>  <b>[af-j]</b> matches either <i>a</i>, <i>f</i>, <i>g</i>, <i>h</i>, <i>i</i> or <i>j</i>  <b>[a-dh-kq-]</b> matches <i>a</i>, <i>b</i>, <i>c</i>, <i>d</i>, <i>h</i>, <i>i</i>, <i>j</i>, <i>k</i>, or any character from <i>q</i> onwards  <b>IMGP#[0-9].jpg</b> matches <i>IMGP0158.jpg</i> (or any other number).</p>
[~]	<p>Matches any character <b>not</b> in the set of specified characters. See <b>[]</b> for information on how the set is defined.</p>

	<p>For example:</p> <p><b>[~pqr]</b> matches any character except <i>p</i>, <i>q</i> or <i>r</i></p>
*	<p>Matches any number of characters, including zero. This is a synonym for <b>#?</b>.</p> <p><b>*.doc</b> is equivalent to <b>#?.doc</b>.</p> <p>For example:</p> <p><b>abc*xyz.*q</b> matches any filename that begins with <i>abc</i>, has <i>xyz</i> at the end and then has any file extension ending with <i>q</i>.</p>
'	<p>The apostrophe is called the <i>escape character</i>.</p> <p>Any character in the pattern string that isn't a wildcard token is taken as a literal character - literal characters have to match the characters in the target string exactly. However, if actually want to use a wildcard token as a literal character itself, you must escape the token by preceding it with an apostrophe.</p> <p>For example:</p> <p><b>a#xb</b> matches <i>ab</i>, <i>axb</i>, <i>axxb</i>, <i>axxxb</i>, etc, as seen above  <b>a'xb</b> will only match <i>a#xb</i></p>
<b>grp:</b>	<p>When using wildcards to match filenames this is a shorthand way to refer to all the file extensions defined by a <a href="#">file type group</a>.</p> <p>For example, the <b>Images</b> group may contain the file extensions <i>.jpg</i>, <i>.bmp</i> and <i>.gif</i>. If you wanted a command to automatically select files of these types, you could use the command:</p> <p><b>Select *.<a href="#">(jpg bmp gif)</a></b></p> <p>A better way would be to use the <b>grp:</b> token to automatically include all file extensions from that group:</p> <p><b>Select grp:Images</b></p> <p>This is much easier to read (and type!), and if you ever add or remove file types from the group in question, any patterns which refer to the group will automatically reflect the new changes.</p> <p>For example:</p>

(grp:Images|grp:Documents) matches any files in the **Images** or **Documents** file type groups

## Regular Expression Syntax

Regular expressions are a more powerful (and therefore complicated) form of wildcard pattern matching. Like [standard pattern matching](#), they can be used throughout Opus. Generally, you have to specifically enable the use of regular expressions in a given situation - by default, Opus will assume standard pattern matching. For example, the [Advanced Rename](#) dialog has a regular expression mode that you must select before regular expressions can be used.

One advantage regular expressions have over standard pattern matching is they can enable a form of search and replace in certain functions. As an example, this is used in the **Rename** command. The "search" string is specified as a pattern to match against the original names of files. That pattern can indicate *capture groups* - expressions in the source string that are captured, and can be carried over to the new string (which acts as the "replace" string). As an example, imagine the **Rename** dialog is set to regular expression mode, with the following patterns supplied:

**Old Name:** The (\*.\*) Backup\.(.\*)  
**New Name:** \1.\2

The two (\*.\*) tokens in the *old name* string are capture groups - they "capture" whatever is matched by the expression within the parentheses. In this case, the expression inside the brackets is .\* which simply means "match anything". So what this pattern will do is match any filename beginning with *The* and ending in *Backup*, and it will capture the middle of the filename for later use. The second (\*.\*) will capture the file extension. The *new name* string can then re-use the captured text, and this is indicated with the \1 and \2 markers. So as an example, the original filename *The Lord Of The Rings Backup.avi* would be renamed to *Lord Of The Rings.avi*. \1 refers to the first capture group, \2 to the second, and so on.

If you need the *new name* string to contain a literal \, use two together. For example, *abc\\xyz* will turn into *abc\xyz*.

When used with the **Rename** command only, the *old name* pattern can be followed with a # character to indicate that the search and replace operation should be repeated multiple times. For example, the following regular expression rename will remove all spaces from the filename:

**Old Name:** (\*.\*)\s(\*.\*)#  
**New Name:** \1\2

The # causes the search and replace to be repeated until the new name no longer changes. You can also specify a maximum repetition count by appending a number, for example #5 at the end would repeat the operation no more than five times.

There are many different variants of regular expression; by default Opus uses what's called *TR1 ECMAScript*. Microsoft has a [page on TR1](#) that goes into far more detail than this help file can.

Token	Description
^	<p><b>Start of a string.</b> The caret is used to "anchor" the search to the start of the string. If the search is not anchored to either end, the pattern can match a sub-string of the target.</p> <p>For example:</p> <p><b>^abc</b> matches <i>abc, abcdefg, abc123</i>, but not <i>123abc</i> <b>abc</b> also matches <i>123abc</i></p>
\$	<p><b>End of a string.</b> The dollar sign is used to "anchor" the search to the end of the string. If the search is not anchored to either end, the pattern can match a sub-string of the target.</p> <p>For example:</p> <p><b>abc\$</b> matches <i>abc, endsinabc, 123abc</i>, but not <i>abc123</i> <b>^abc(.*?)123\$</b> matches <i>abc123, abcxyz123</i>, but not <i>abcxyz123def</i></p>
.	<p><b>Any single character.</b> The period (full stop) is used to match any single character.</p> <p>For example:</p> <p><b>a.c</b> matches <i>abc, aac, acc, adc</i> but not <i>acd</i></p>
*	<p><b>0 or more of previous expression.</b> Matches zero or more occurrences of the previous expression. Combine with . to form the "match anything" token (.*).</p> <p>For example:</p>

	<p><b>ab*c</b> matches <i>ac, abc, abbc, abbbc, ...</i></p> <p><b>a.*c</b> matches <i>ac, abc, a123456c, aanythingc, ...</i></p> <p><b>.*</b> matches anything</p>
<b>+</b>	<p><b>1 or more of previous expression.</b> Matches one or more occurrences of the previous expression.</p> <p>For example:</p> <p><b>ab+c</b> matches <i>abc, abbc, abbbc</i> but not <i>ac</i></p>
<b>?</b>	<p><b>0 or 1 of previous expression.</b> Matches either zero or one occurrence of the previous expression.</p> <p>For example:</p> <p><b>ab?c</b> matches <i>ac, abc</i> but not <i>abbc</i> or <i>abbbc</i></p>
<b> </b>	<p><b>Alternation (logical or).</b> The vertical bar is used to separate two or more characters or expressions, any of which may match.</p> <p>For example:</p> <p><b>a b</b> matches <i>a</i> or <i>b</i></p> <p><b>a(b c)d</b> matches <i>abd</i> or <i>acd</i></p> <p><b>(bill ted)</b> matches <i>bill</i> or <i>ted</i></p>
<b>{ }</b>	<p><b>Quantifier.</b> Braces are used to indicate that the preceding expression must match an exact number of times.</p> <p>For example:</p> <p><b>ab{2}c</b> matches <i>abbc</i>, but not <i>abc</i> or <i>abbbc</i></p> <p><b>a.{4}z</b> matches <i>abcdez, a1234z, afourz, aaaaaz, etc.</i></p>
<b>[ ]</b>	<p><b>Character set.</b> Matches any single character in the set of specified characters.</p> <p>You can specify the character set as individual characters (e.g. <b>[abdfg]</b>) or as a range of characters (e.g. <b>[a-j]</b>) or as multiple ranges.</p> <p>For example:</p>

	<p><b>[abc]</b> matches either <i>a</i>, <i>b</i> or <i>c</i>  <b>[af-j]</b> matches either <i>a</i>, <i>f</i>, <i>g</i>, <i>h</i> or <i>j</i>  <b>[a-dh-kq-]</b> matches <i>a</i>, <i>b</i>, <i>c</i>, <i>d</i>, <i>h</i>, <i>i</i>, <i>j</i>, <i>k</i>, or any character from <i>q</i> onwards  <b>IMG[P0-9]{4}.jpg</b> matches <i>IMG0158.jpg</i> (or any other four-digit number).</p>																
<b>[^]</b>	<p><b>Negative character set.</b>  Matches any character <b>not</b> in the set of specified characters. See <b>[]</b> for information on how the set is defined.</p> <p>For example:</p> <p><b>[^pqr]</b> matches any character except <i>p</i>, <i>q</i> or <i>r</i></p>																
<b>()</b>	<p><b>Expression / capture group.</b>  Parentheses are used to combine multiple characters into an expression. When used in a "search and replace" like Advanced Rename, they also mark capture groups - see above for a discussion of these.</p> <p>For example:</p> <p><b>a bc</b> matches <i>ac</i> or <i>bc</i>, whereas  <b>a(bc)</b> matches <i>a</i> or <i>bc</i></p>																
<b>\</b>	<p><b>Escape character.</b>  The backslash is used to escape token characters in order to match those characters literally.</p> <p>When used before a non-token character, it is used to indicate the following special escape characters:</p> <table border="1"> <tr> <td><b>\t</b></td> <td>tab character (\$09)</td> </tr> <tr> <td><b>\r</b></td> <td>carriage return (\$0d)</td> </tr> <tr> <td><b>\v</b></td> <td>vertical tab (\$0b)</td> </tr> <tr> <td><b>\f</b></td> <td>form feed (\$0c)</td> </tr> <tr> <td><b>\n</b></td> <td>new line (\$0a)</td> </tr> <tr> <td><b>\e</b></td> <td>escape (\$1b)</td> </tr> <tr> <td><b>\x</b></td> <td>matches an ASCII character specified as a two-digit hexadecimal number, e.g. <b>\x20</b> matches a space</td> </tr> <tr> <td><b>\u</b></td> <td>matches a Unicode character specified as a four-digit hexadecimal number, e.g. <b>\u0020</b> matches a space.</td> </tr> </table>	<b>\t</b>	tab character (\$09)	<b>\r</b>	carriage return (\$0d)	<b>\v</b>	vertical tab (\$0b)	<b>\f</b>	form feed (\$0c)	<b>\n</b>	new line (\$0a)	<b>\e</b>	escape (\$1b)	<b>\x</b>	matches an ASCII character specified as a two-digit hexadecimal number, e.g. <b>\x20</b> matches a space	<b>\u</b>	matches a Unicode character specified as a four-digit hexadecimal number, e.g. <b>\u0020</b> matches a space.
<b>\t</b>	tab character (\$09)																
<b>\r</b>	carriage return (\$0d)																
<b>\v</b>	vertical tab (\$0b)																
<b>\f</b>	form feed (\$0c)																
<b>\n</b>	new line (\$0a)																
<b>\e</b>	escape (\$1b)																
<b>\x</b>	matches an ASCII character specified as a two-digit hexadecimal number, e.g. <b>\x20</b> matches a space																
<b>\u</b>	matches a Unicode character specified as a four-digit hexadecimal number, e.g. <b>\u0020</b> matches a space.																

	<p>It is also used to mark several character classes, which are shorthand ways to specify various common [] character sets (see below).</p> <p>For example:</p> <p><b>a b</b> matches <i>a</i> or <i>b</i>, whereas  <b>a\b</b> matches <i>a/b</i>  <b>a\t</b> matches <i>a</i> followed by a tab character, whereas  <b>a\\t</b> matches <i>a\t</i></p>
<b>\w</b>	<p><b>Word character.</b>  Matches any word character. Equivalent to <b>[a-zA-Z_0-9]</b>.</p> <p>For example:</p> <p><b>^\w+[0-9]{4}.jpg</b> matches <i>IMG0158.jpg</i> (or any other four-digit number preceded by at least one other word character).</p>
<b>\W</b>	<p><b>Non-word character.</b>  Matches any non-word character, equivalent to <b>[^a-zA-Z_0-9]</b>.</p>
<b>\s</b>	<p><b>Space character.</b>  Matches any whitespace character. Equivalent to <b>[\f\n\r\t\v]</b>.</p>
<b>\S</b>	<p><b>Non-space character.</b>  Matches any non-whitespace character. Equivalent to <b>[^\f\n\r\t\v]</b>.</p>
<b>\d</b>	<p><b>Digit character.</b>  Matches any decimal digit. Equivalent to <b>[0-9]</b>.</p>
<b>\D</b>	<p><b>Non-digit character.</b>  Matches any non-decimal digit. Equivalent to <b>[^0-9]</b>.</p> <p>For example:</p> <p><b>^\D+\d{4}.jpg</b> matches <i>IMG0158.jpg</i> (or any other four-digit number preceded by at least one non-digit character).</p>

## Status Bar Codes

The information displayed in the [status bar](#) at the bottom of a Lister can be configured with a number of different codes. The information that can be displayed includes:

- The number and total size of selected files and folders
- How many items are hidden from the display
- Amount of disk space used and free
- Total duration of music and video files
- Various graphical elements like bar graphs and icons

The status bar is configured from the [Display / Status Bar](#) page in Preferences. You can optionally define a completely separate status bar for dual-display Listers.

### Codes for file and folder counts

The following status bar codes are used to display information about the number and size of files and folders (selected, hidden and total) in the Lister. Each code has several different forms - in a dual-display Lister this can be used to control which file display the information refers to.

Description	Code	Details
<b>Number of selected folders</b> Displays the number of selected folders.	{sd} {sdD} {sdL} {sdR}	Selected folders in the source file display - destination file display - left/top file display - right/bottom file display
<b>Number of selected files</b> Displays the number of selected files.	{sf} {sfD} {sfL} {sfR}	Selected files in the source file display - destination file display - left/top file display - right/bottom file display
<b>Number of selected items</b> Displays the total number of selected files and folders.	{si} {siD} {siL} {siR}	Selected items in the source file display - destination file display - left/top file display - right/bottom file display
<b>Size of all selected items</b> Displays the total size of all selected files (and folders, if their size has been <a href="#">calculated</a> ).  The size can be displayed in either	{sb} {sbDb} {sbLb} {sbRb}	Size of selected items in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)

bytes, or an automatic mode where the unit (bytes, KB, MB or GB) is automatically chosen depending on the size.	{sba} {sbDa} {sbLa} {sbRa}	Size of selected items in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)
<b>Size of all selected files</b> Displays the total size of all selected files (does not include selected folders).	{sbf} {sbfDb} {sbfLb} {sbfRb}  {sbfa} {sbfDa} {sbfLa} {sbfRa}	Size of selected files in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)  Size of selected files in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)
<b>Size of all selected folders</b> Displays the total size of all selected folders (does not include selected files). A folder's size must have been previously <a href="#">calculated</a> for it to be included in this total.	{sbd} {sbdDb} {sbdLb} {sbdRb}  {sbda} {sbdDa} {sbdLa} {sbdRa}	Size of selected folders in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)  Size of selected folders in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)
<b>Total number of folders</b> Displays the total number of folders.	{td} {tdD} {tdL} {tdR}	Total folders in the source file display - destination file display - left/top file display - right/bottom file display
<b>Total number of files</b> Displays the total number of files.	{tf} {tfD} {tfL} {tfR}	Total files in the source file display - destination file display - left/top file display - right/bottom file display
<b>Total number of items</b> Displays the total number of files and folders.	{ti} {tiD} {tiL} {tiR}	Total items in the source file display - destination file display - left/top file display - right/bottom file display
<b>Size of all items</b> Displays the total size of all files (and folders, if their size has been	{tb} {tbDb}	Size of items in the source file display (in bytes) - destination file display (bytes)

<p><a href="#">calculated</a>).</p> <p>The size can be displayed in either bytes, or an automatic mode where the unit (bytes, KB, MB or GB) is automatically chosen depending on the size.</p>	<p>{tbLb} {tbRb}</p> <p>{tba} {tbDa} {tbLa} {tbRa}</p>	<p>- left/top file display (bytes) - right/bottom file display (bytes)</p> <p>Size of items in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Size of all files</b> Displays the total size of all files (does not include folders).</p>	<p>{tbf} {tbfDb} {tbfLb} {tbfRb}</p> <p>{tbfa} {tbfDa} {tbfLa} {tbfRa}</p>	<p>Size of files in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)</p> <p>Size of files in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Size of all folders</b> Displays the total size of all folders (does not include files). A folder's size must have been previously <a href="#">calculated</a> for it to be included in this total.</p>	<p>{tbd} {tbdDb} {tbdLb} {tbdRb}</p> <p>{tbda} {tbdDa} {tbdLa} {tbdRa}</p>	<p>Size of folders in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display bytes)</p> <p>Size of folders in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Number of hidden folders</b> Displays the number of folders that have been hidden from the file display via the various <a href="#">filters</a>.</p>	<p>{hd} {hdD} {hdL} {hdR}</p>	<p>Hidden folders in the source file display - destination file display - left/top file display - right/bottom file display</p>
<p><b>Number of hidden files</b> Displays the number of hidden files.</p>	<p>{hf} {hfD} {hfL} {hfR}</p>	<p>Hidden files in the source file display - destination file display - left/top file display - right/bottom file display</p>
<p><b>Number of hidden items</b> Displays the total number of hidden files and folders.</p>	<p>{hi} {hiD} {hiL} {hiR}</p>	<p>Hidden items in the source file display - destination file display - left/top file display - right/bottom file display</p>

<p><b>Size of hidden items</b> Displays the total size of all hidden files (and folders, if their size has been <a href="#">calculated</a>).</p> <p>The size can be displayed in either bytes, or an automatic mode where the unit (bytes, KB, MB or GB) is automatically chosen depending on the size.</p>	<p>{hb} {hbDb} {hbLb} {hbRb}</p> <p>{hba} {hbDa} {hbLa} {hbRa}</p>	<p>Size of hidden items in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)</p> <p>Size of hidden items in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Size of hidden files</b> Displays the total size of all hidden files (does not include folders).</p>	<p>{hbf} {hbfdB} {hbflB} {hbfrB}</p> <p>{hbfa} {hbfaDa} {hbfaLa} {hbfaRa}</p>	<p>Size of hidden files in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)</p> <p>Size of hidden files in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Size of hidden folders</b> Displays the total size of all hidden folders (does not include files). A folder's size must have been previously <a href="#">calculated</a> for it to be included in this total.</p>	<p>{hbd} {hbdbD} {hbdlB} {hbdrB}</p> <p>{hbda} {hbdaDa} {hbdaLa} {hbdaRa}</p>	<p>Size of hidden folders in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display bytes)</p> <p>Size of hidden folders in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Show Everything mode</b> Indicates if <a href="#">Show Everything</a> mode is active.</p>	<p>{hse}</p>	<p>"Everything" if <i>Show Everything</i> mode is active. "Filtering" if <i>Show Everything</i> mode is turned off.</p>
<p><b>Number of file groups</b> If the file display is set to group by a column this code returns the number of distinct file groups.</p>	<p>{grp} {grpD} {grpL} {grpR}</p>	<p>Number of file groups in the source file display - destination file display - left/top file display - right/bottom file display</p>

## Codes for disk space

The following status bar codes are used to display information about the amount of space used and free on the current drive. Each code has several different forms - in a dual-display Lister this can be used to control which file display the information refers to.

Description	Code	Details
<p><b>Free disk space</b> Displays the amount of free space on the drive.</p> <p>The size can be displayed in either bytes, or an automatic mode where the unit (bytes, KB, MB or GB) is automatically chosen depending on the size.</p>	<p>{dfb} {dfDb} {dfLb} {dfRb}</p> <p>{dfa} {dfDa} {dfLa} {dfRa}</p>	<p>Free space on the drive in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)</p> <p>Free space on the drive in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Used disk space</b> Displays the amount of space in use on the drive.</p>	<p>{dub} {duDb} {duLb} {duRb}</p> <p>{dua} {duDa} {duLa} {duRa}</p>	<p>Used space on the drive in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)</p> <p>Used space on the drive in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)</p>
<p><b>Total disk space</b> Displays the total size of the drive.</p>	<p>{dtb} {dtDb} {dtLb} {dtRb}</p>	<p>Total space on the drive in the source file display (in bytes) - destination file display (bytes) - left/top file display (bytes) - right/bottom file display (bytes)</p>

	{dta} {dtDa} {dtLa} {dtRa}	Total space on the drive in the source file display (automatic units) - destination file display (automatic units) - left/top file display (automatic units) - right/bottom file display (automatic units)
<b>Percentage of disk space free</b> Displays the amount of free space on the drive as a percentage of the total size.	{pf} {pfD} {pfL} {pfR}	Percentage of free space on the drive in the source file display - destination file display - left/top file display - right/bottom file display
<b>Percentage of disk space used</b> Displays the amount of used space on the drive as a percentage of the total size.	{pu} {puD} {puL} {puR}	Percentage of used space on the drive in the source file display - destination file display - left/top file display - right/bottom file display

## Codes for music and video duration

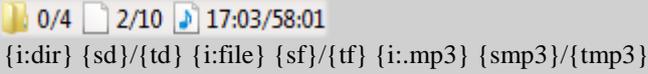
The following status bar codes are used to display information about the duration of music and video files in the file list.

Code	Description
{tmp3}	<b>Total playing time</b> Displays the total playing time (duration) of all recognised music and video files in the file list.
{smp3}	<b>Playing time of selected files</b> Displays the total duration of all selected music and video files.

## Codes for graphical elements

The following codes are used to display various graphical elements in the status bar. Some (like the {i} code for icons) are purely for cosmetic reasons, and others have more functional uses.

Code	Description
------	-------------

<p>{fl} {gfl}</p>	<p><b>Format lock control</b> Displays the format lock icon (🔒). This icon indicates the current state of the <a href="#">format lock</a>, and clicking it will toggle the lock on and off. Hovering the mouse over the icon displays a tooltip indicating <a href="#">how the current folder format was derived</a>.</p> <p>The {gfl} form displays a grayscale rather than color icon.</p>
<p>{i:dir} {i:file} {i:&lt;ext&gt;} {i:&lt;filename&gt;}</p>	<p><b>Icons</b> Lets you display an arbitrary icon in the status bar. For example, instead of status bar text that reads <i>Folders: xxx / yyy</i> you could display a generic folder icon.</p> <p style="text-align: center;">  </p> <p>You may want this to save space on the status bar or for purely cosmetic reasons.</p> <p>You can specify a generic folder icon (<b>{i:dir}</b>), a generic file icon (<b>{i:file}</b>), or an icon for a specific file type (e.g. <b>{i:.mp3}</b> for an MP3 file icon).</p> <p>You can also use your own custom icon or image by providing the path and filename of the icon. For example, <b>{i:/mypictures/myicon.png}</b>. The image file will be automatically scaled down to fit the status bar.</p>
<p>{bg}</p>	<p><b>Bar graphs</b> Bar graphs can be used to visually represent a value as a percentage. The default behaviour of bar graphs is to display the percentage of disk space used on the current drive, but using the various parameters for the {bg} code you can configure a bar graph to represent any other status bar value as a percentage of either another value or an absolute amount.</p> <p>See the page on <a href="#">bar graphs</a> for full details on configuring the graphs.</p>

## Other Codes

The following status bar codes are used to style text, display miscellaneous information about the current drive or Lister, or work with variables and if-conditions.

## Text Styles, Colors and Comments

Code	Description
<code>&lt;b&gt;...&lt;/b&gt;</code>	<b>Bold text</b>
<code>&lt;i&gt;...&lt;/i&gt;</code>	<b>Italic text</b>
<code>&lt;u&gt;...&lt;/u&gt;</code>	<b>Underlined text</b>
	<p>Use HTML-style tags to make text on the status bar bold, italic, or underlined.</p> <pre>&lt;b&gt;This Will Be Bold Text&lt;/b&gt;</pre> <pre>&lt;i&gt;This Will Be Italic Text&lt;/i&gt;</pre> <pre>&lt;u&gt;This Will Be Underlined Text&lt;/u&gt;</pre>
<code>&lt;#RRGGBB&gt;...&lt;/#&gt;</code>	<b>Text color</b>
	<p>Use tags with RGB hex values around text to make it a different color.</p> <pre>&lt;#FF0000&gt;This Will Be Red Text&lt;/#&gt;</pre>
<code>{{</code>	<b>Literal { character</b>
	<p>Add two { characters in a row to insert a single literal { character into the status bar text, without it being interpreted as the start of a status bar code.</p>
<code>//</code>	<b>Comments</b>
	<p>Start lines with // to turn them into comments. Comments can be used to add notes or explanations, or to temporarily remove parts of the status bar.</p> <p>The // must be at the very start of the line, with no spaces or anything else before it, for the line to become a comment.</p>

### ***Current drive, view mode, and Lister state***

<b>Code</b>	<b>Description</b>
{dlet}	<b>Drive letter</b>  Displays the current drive letter. This refers to the folder that is currently shown in the source file display.
{dlab}	<b>Drive label</b>  Displays the label (if any) of the drive that is currently open in the source file display.
{fsys}	<b>File system</b>  Displays the file system type of the current drive.
{vm}	<b>View mode</b> Displays the current <a href="#">view mode</a> in the source file display.
{ls}	<b>Lister state</b>  Displays the current <a href="#">source / destination</a> state of the Lister.

### ***Details about the currently selected file or folder***

<b>Code</b>	<b>Description</b>
{sel:..}	<b>Selected file information</b>  Displays information about the most recently selected file. The <b>sel:</b> must be followed by a keyword to specify the information to display; valid keywords are: <ul style="list-style-type: none"><li>• <b>name:</b> Name of the file or folder.</li><li>• <b>size:</b> File size. Follow this keyword with <b>b</b> or <b>k</b> to specify the units as <i>bytes</i> or <i>KB</i> (otherwise the units are automatically chosen).</li></ul>

- **create**: Creation date stamp. Follow this keyword with **d** or **t** to specify *date* or *time* (otherwise both are shown).
- **write**: Last write (modification) date stamp. Follow this keyword with **d** or **t** to specify *date* or *time* (otherwise both are shown).
- **access**: Last access date stamp.
- **attr**: File or folder attributes.
- **desc**: Description string (the same as is displayed in the *Description* column).
- **path**: Full path of the file or folder.
- **index**: Index in the file display.

For example, `{sel:sizek} {sel:desc}`.

## Variables

Code	Description
<code>{var:...}</code>	<p><b>Variables</b></p> <p>Display the value of a variable. Can also be used for <a href="#">hiding sections on the status bar</a>. Variables can be set from commands using the <a href="#">@set modifier</a> or from scripts using the <a href="#">Vars object</a>.</p> <p>Variable names must to be prefixed with a scope. For example:</p> <ul style="list-style-type: none"> <li>• <code>{var:glob:MyGlobalVariable}</code></li> <li>• <code>{var:lst:MyListerVariable}</code></li> <li>• <code>{var:tab:MyFolderTabVariable}</code></li> <li>• <code>{var:src:MySourceFolderTabVariable}</code></li> <li>• <code>{var:dst:MyDestinationFolderTabVariable}</code></li> <li>• <code>{var:left:MyLeftFolderTabVariable}</code></li> <li>• <code>{var:right:MyRightFolderTabVariable}</code></li> </ul>

The **tab** scope will often be identical to the **src** scope, but not always. If you have Opus configured to use separate status bars for the left and right file displays, you can use the **tab** scope in the status bar definition for both sides and it will look up variables in the appropriate file display for each side.

The same scope prefixes are also used in commands which use variables. In addition, for consistency with other status bar codes, you can also use:

- `{varL:tab:xyz}` as a synonym for `{var:left:xyz}`
- `{varR:tab:xyz}` as a synonym for `{var:right:xyz}`
- `{varD:tab:xyz}` as a synonym for `{var:dst:xyz}`

To [hide part of the status bar](#) if a variable is not set, and show the part if the variable is set, you could use something like this:

- `{h!{var:glob:ShowExtraInfo}} ...extra info is ON... {h!}`

You could then have a menu item, button or hotkey which ran this command to toggle that part of the status bar on and off:

```
@if:$glob:ShowExtraInfo
@set glob:ShowExtraInfo
@if:else
@set glob:ShowExtraInfo=on
@if:common
@toggle:update
```

See the [Command Modifier Reference](#) for details about the `@if`, `@set`, and `@toggle` command directives.

You can negate a variable by adding `!` before its name:

- `{h!{var:!glob:ShowExtraInfo}} ...extra info is OFF... {h!}`

### **Conditional tests**

These all test a condition and return the result. If the condition is true, they will return "1"; if it is false, they will return nothing (an empty string). You would normally use them to show or hide other information. You may find it useful to use them on their own, when testing your status bar codes to check that they return "1" when expected, but once you are done testing you will probably only use them in conjunction with `{h!}` and similar [codes for showing and hiding parts of the status bar](#).

```
{h!{ifpath:C:\}} You are in the root of C:\. I bet you didn't  
already know. {h!}
```

While calculating `{ifpath:...}` is fairly inexpensive, keep in mind that there can be a performance hit from the other conditional tests, especially if they end up touching a network drive or evaluating a complex command. The status bar is updated frequently -- for example, every time a file is selected or deselected -- and any conditional tests you add to it will be re-evaluated each time it is updated. Don't go overboard!

When showing or hiding part of the status bar based on multiple conditions, put the ones which are cheapest to evaluate at the start. For example:

```
{h!{ifpath:/downloads}!{ifexists:.*.dll}} WARNING: DLLs in the  
downloads folder! {h!}
```

The `{ifpath:...}` test is fast, since the current path is already known and just has to be compared to the specified path or wildcard. If that test fails, the `{ifexists:...}` test which comes after it can be skipped. Doing the tests in this order means you only do the more expensive test when in the folder where it matters. `{ifexists:...}` can be expensive because it has to go to the disk/filesystem to check if anything is there, which is much slower than testing something which is already in memory.

If you test the same condition multiple times on the status bar, using the exact same codes, you will only pay a performance penalty for the first test; the others re-use its result.

Code	Description
{ifpath:...}	<p data-bbox="485 427 778 461"><b>Test the current path</b></p> <p data-bbox="485 501 1406 573">Use <b>{ifpath:...}</b> in to test if the current path matches a particular folder, wildcard or regular expression.</p> <p data-bbox="580 685 1289 757"><b>{ifpath:C:\Program Files}</b> -- True when "C:\Program Files" is the current folder.</p> <p data-bbox="485 869 1366 940">Quotes are options, except that if the path contains a { or } character then you <i>must</i> put quotes around it:</p> <p data-bbox="580 1052 1302 1124"><b>{ifpath:"C:\My {Test} Folder"}</b> -- True when "C:\My {Test} Folder" is the current folder.</p> <p data-bbox="485 1236 1008 1270">Put a ! before the path to <b>negate</b> the test.</p> <p data-bbox="580 1382 1302 1453"><b>{ifpath:!C:\Program Files}</b> -- True when "C:\Program Files" is NOT the current folder.</p> <p data-bbox="485 1570 1401 1677">The <a href="#">wildcard pattern matching syntax</a> is used by default. Note that this means that if the path contains ( or ) characters, you need to escape the those characters:</p> <p data-bbox="580 1792 1214 1863"><b>{ifpath:C:\Program Files '(x86)'}</b> -- True when "C:\Program Files (x86)" is the current folder.</p>

[Aliases](#) may be used if you wish:

**{ifpath:/downloads}** -- True when in your Downloads folder.

*Wildcard Tip - Matching folders and sub-folders:*

Ending the path with (**(\\*)**) means it will match both the folder itself and any folders below it. For example:

- Using **C:\MyFolder** as the pattern will only match **C:\MyFolder** itself, and not **C:\MyFolder\Child**.
- If you use **C:\MyFolder\\*** instead, it will match **C:\MyFolder\Child** but will no longer match **C:\MyFolder** itself.
- If you use **C:\MyFolder\***, it will match both folders, but will also match completely unrelated folders such as **C:\MyFolderBackup**.
- Using **C:\MyFolder(\\*)** will match the folder itself and any folders below it (including children-of-children, and so on).

**{ifpath:C:\Program Files(\\*)}** -- True when in or below "C:\Program Files".

You can use [regular expression syntax](#) instead by adding **regex:** before the path (remember to escape backslashes when using regular expressions):

**{ifpath:regex:"C:\\Program Files\\[^\\]+\$"}** -- True in direct children of "C:\Program Files" only.

**{ifpath:!regex:[0-9]}** -- True in folders which have no numbers in their paths.

(Note that you cannot use *ifpathr* as a shorthand for regular expressions mode, unlike the [command modifier](#) equivalent, because it has a different meaning here.)

The **{ifpath:...}** code tests the current, active folder tab's folder. You can also use:

- **{ifpathL:...}** to test the left tab's folder.
- **{ifpathR:...}** to test the right tab's folder.
- **{ifpathD:...}** to test the destination tab's folder.

The **L**, **R** and **D** characters must be uppercase. You would normally only use those side-specific variants when defining a single status bar which is shared by both sides at once (not generally recommended these days). When using a status bar for each side of the lister, you may get unexpected results if you test one side's path from the other side, because each status bar only updates in responses to changes that affect its own side, not the other side.

**{ifexists:...}**

### **Test if a path exists**

Use **{ifexists:...}** to test for the existence of things in the filesystem. The test will return true if a file or folder exists at the specified path; it does not make a difference if a file or a folder is found.

As with the **{ifpath:...}** code, quotes are optional but must be used if the path contains **{** or **}** characters.

You can test if an absolute path exists:

```
{ifexists:C:\Docs\My Special File.txt} -- True if  
"C:\Docs\My Special File.txt" exists.
```

You can also test if a relative path exists (i.e. relative to the folder tab's current location). Relative paths *must* begin with either `.\` (for the current directory) or `..\` (for the parent of the current directory).

`{ifexists:'.\Help'}` -- True if "Help" exists below the current directory.

Basic `*` wildcards can be used in the last path component only:

`{ifexists:'.\*.dll'}` -- True if any `*.dll` files (or folders!) exist below the current directory.

As with `{ifpath:...}`, you can negate the test using a `!` before the path:

`{ifexists:!'C:\Test*.dll'}` -- True if no `*.dll` files exist in "C:\Test".

As with `{ifpath:...}`, and with similar caveats to those discussed in its section above, you can use `{ifexistsL:...}`, `{ifexistsR:...}` and `{ifexistsD:...}` to target the left, right or destination side instead of the active side.

We do not recommend testing paths on network drives, as that can cause major delays if the drive is slow or unreachable. (Windows takes up to 30 seconds to decide a network path cannot be accessed, and can block the program attempting the access until then.) Testing paths on removable drives, or testing the existence of them, may also have unwanted side-effects, but depends on the type of drive you are testing.

`{if:...}`  
`{ifset:...}`

### Test Set and other commands

Similar to the `@if` and `@ifset` [command modifiers](#), you can test the state of [internal commands](#) to determine if status bar parts should be hidden.

The "*state of a command*" generally means whether or not, if the command was placed on a toolbar button, the button would appear "pushed in" or activated. For example, the **Set VIEW=Details** command will appear activated when the file display is in **Details** mode.

As with the **{ifpath:...}** code, quotes are optional but must be used if the command contains { or } characters.

**{if:Set VIEW=Details}** -- True if the file display is in Details mode.

**{if:Toolbar NAME="My Toolbar" TOGGLE}** -- True if "My Toolbar" is turned on.

**{ifset:...}** is simply an synonym for **{if:Set...}**, maintained for compatibility with similar commands in other parts of the program.

**{ifset:VIEW=Details}** -- Identical to **{if:Set VIEW=Details}**

As with **{ifpath:...}**, you can negate the test using a **!** before the command:

**{if:!Set VIEW=Details}** -- True if the file display is NOT in Details mode.

As an example use-case for this, you might want the status bar to display the date and size of the selected file when in modes such as Thumbnails, while removing the extra clutter when in Details mode, since Details mode shows the same information in its columns. You could do that using these codes:

**{h!{if:!Set VIEW=Details}}{sel:size} {sel:write}{h!}**

As with **{ifpath:...}**, and with similar caveats to those discussed in its section above, you can use **{ifL:...}**, **{ifR:...}** and **{ifD:...}** to target the left, right or destination side instead of the active side.

## Bar graphs and Percentages

You can configure the status bar to display various information values as percentages. For example, the percentage of disk space used on the current drive, or the total size of selected files expressed as a percentage of the free space in the destination folder. The percentage values can be displayed in two ways - as a number, or as a bar graph.

- The **{pf}** and **{pu}** codes display the percentage of disk space used and free. These are described in detail on the [Codes for disk space](#) page.
- The **{cp}** code can display a percentage calculated from any other status bar information code.
- The **{bg}** code displays a bar graph representing a calculated percentage.

The **{cp}** code is used to display a percentage that is calculated from either two other status bar information codes, or one code and one absolute value. The template for the **{cp}** code is as follows:

**{cp+V=<num>/<den>}**

In the above template, *<num>* is the value to be divided and *<den>* is the value it will be divided by. The result is then multiplied by 100 to give the final percentage that will be displayed in the status bar. Both *<num>* and *<den>* can refer to any of the status bar codes that produce a number. They can also specify an absolute value, and this value can be given as bytes, kilobytes (with the **kb** suffix), megabytes (with the **mb** suffix) or gigabytes (with the **gb** suffix).

For example:

- To display the percentage of the size of selected items (**{sb}**) relative to the free space in the destination file display (**{dfDb}**), you would specify **{cp+V=sf/dfDb}**.
- To display the percentage of the size of selected files relative to 4.7 GB (the size of an empty DVD), you would specify **{cp+V=sb/4.7gb}**.

If **{cp}** is used by itself (without specifying **+V=**) the default behaviour is to indicate the amount of disk space used - this is equivalent to **V=du/dt**. The exception to this is if a **{bg}** code was used before the **{cp}** - in this case, the percentage reflected by the bar graph will be displayed.

The **{bg}** code is used to display a bar graph in status bars. The default graph shown is the percentage of disk space used on the current drive. You can configure the **{bg}** code with one or more parameters, to control:

- The width of the graph
- The text and fill colors
- The type of gradient fill used (if any)
- The type of frame used (if any)
- Which line it should appear on (lets you position one bar graph above another)
- Which status bar information codes are used in the calculation of the graph

The template for the **{bg}** code is as follows:

**{bg+<param>,<param>,...}**

The available parameters are:

Code	Description
<b>W=&lt;width&gt;</b>	Sets the width of the graph, in pixels. Specify <b>-1</b> to make the graph the same width as status bar section it is in. Positive numbers will scale with system DPI, so 50 will be 50 pixels wide at standard DPI and 100 pixels wide at 200% DPI. Negative values (other than -1) can be used to specify a width that does not scale with DPI.

<b>C=#&lt;rrggb&gt;</b>	Sets the color of the bar. This is used when the bar is solid - use <b>C1</b> and <b>C2</b> when using a gradient bar. The color is given in hexadecimal notation, e.g. <b>C=#ff8000</b> .
<b>C1=#&lt;rrggb&gt;</b>	Sets the left color of the bar when using a gradient fill.
<b>C2=#&lt;rrggb&gt;</b>	Sets the right color of the bar when using a gradient fill.
<b>C3=#&lt;rrggb&gt;</b>	Sets the color of the bar when the calculated percentage is greater than 100%. For example, you could have a graph that indicates the size of selected files compared with the space on a DVD - if the size was greater than 4.7GB the bar could turn red.
<b>B=#&lt;rrggb&gt;</b>	Sets the background color of the graph. This is the color used to fill the space not occupied by the bar (i.e. to the right of the bar).
<b>B=n</b>	No background fill is used for the graph - the background of the status bar will show through to the right of the bar.
<b>T=#&lt;rrggb&gt;</b>	Sets the text color for the percentage value that is drawn over the top of the graph. The color is given in hexadecimal notation, e.g. <b>T=#ffffff</b> .
<b>T=n</b>	The percentage value will not be shown.
<b>G=&lt;gradient&gt;</b>	<p>Sets the gradient type used to fill the bar. The available types are:</p> <ul style="list-style-type: none"> <li><b>0</b> - no gradient (solid fill - color set by <b>C</b>)</li> <li><b>1</b> - normal gradient fill (gradient of colors from <b>C1</b> to <b>C2</b>)</li> <li><b>2</b> - scaled gradient fill</li> <li><b>3</b> - Computer-style gradient type A (unthemed)</li> <li><b>4</b> - Computer-style gradient type B (unthemed)</li> <li><b>5</b> - Computer-style gradient type A (themed)</li> <li><b>6</b> - Computer-style gradient type B (themed)</li> </ul> <p>With a normal gradient fill (<b>1</b>), the color drawn at the very right edge of the bar will be equal to <b>C2</b>. For example, when <b>C1=#FF0000</b> (red) and <b>C2=#FFFF00</b> (yellow), you can see that yellow is drawn at the right of the bar even when the percentage is less than 100%:</p>  <p>With a scaled gradient fill (<b>2</b>), <b>C2</b> will only be used when the bar percentage equals 100%. At values less than 100%, the color drawn at the right edge of the bar will be a color between <b>C1</b> and <b>C2</b> proportional to the percentage being drawn. In the above example, you can see that the right of the bar is drawn in a sort of orange color - full yellow would only be used at the 100% mark.</p>  <p>When using <b>0</b>, <b>1</b> or <b>2</b>, if the <b>C3</b> color is set and the bar's value is over 100% then the entire bar will turn the <b>C3</b> color. For example, if you have a graph which shows how much of an empty DVD is filled by the selection, you may want the graph to change color when the selection is too large to fit.</p>

The Computer-style gradients (**3**, **4**, **5** and **6**) are based on the free-space bar graphs displayed in the Computer folder. The "unthemed" versions are drawn by Opus itself while the "themed" versions are drawn by your Windows visual style. (With Windows Aero, the themed and unthemed bars look very similar, but this may vary with other visual styles.)



When using the Computer-style gradients, the colors set by **C**, **C1** and **C2** are ignored. The "unthemed" versions always use blue and red. With Windows Aero, the "themed" versions also use blue and red, but other visual styles may look different.

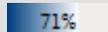
Computer-style Type A gradients (**3** and **5**) are normally used for displaying used space. Type B gradients (**4** and **6**) are normally used for displaying free space graphs. By default, Type A displays blue bars for percentages below 90% and red bars otherwise. Type B is the opposite, displaying red bars for percentages below 10% and blue bars otherwise.

When using the Computer-style gradients, if the **C3** color is set to anything at all then the cut-off point for the blue/red color changes from 90% to 100%. It does not matter what the **C3** color is actually set to when using Computer-style gradients; only that it is set at all.

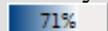
**F**=<frame>

Sets the frame type used to outline the graph. The available types are:

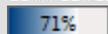
**0** - no frame



**1** - 3D-style frame (raised)



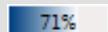
**2** - flat frame (dark)



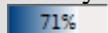
**3** - fill status bar section (see below)

**4** - flat frame (always dark - see below)

**5** - flat frame (light)



**6** - 3D-style frame (sunken)

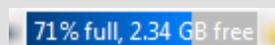


Frame types **2** and **4** are only different when the status bar is using glass on Vista and above. In that case, type **2** will display a light colored frame to improve the appearance by default - type **4** can be used to force a dark colored frame. When not using glass, types **2** and **4** are the same.

Frame type **3** is a special case. It causes the bar graph to be expanded to fill the status bar section it is contained in - the graph actually becomes the background of the section. This lets you display a bar graph without it taking up any additional space - any other text in that status bar section is drawn on top of the graph. For example, the following section will display a

bar graph of used disk space behind text showing both the used and free space.

```
{bg+F=3,T=n,W=-1,B=#BBBBBB}<#ffffff> {pu}% full,
{df} free
```



When the gradient is set to one of the Computer-style types (**G=3** through to **G=6**), the frame type is ignored.

**L=<line>**

Lets you stack one bar graph on top of the other. If **L=** is not specified, the bar graph will be the full height of the status bar, but if you specify either **L=0** or **L=1**, the graph will be reduced to half-height, and the graph with **L=0** will be displayed above the graph with **L=1**.

For example, **{bg+L=0}{bg+L=1,V={df}/{dt}}** displays a graph of disk space used above the inverse graph, one of disk space free.



You can also use this technique to display multiple smaller graphs above or below a larger one. For example,

```
{bg+L=0,W=150}{bg+L=1,V={sb}/{df},W=75}{bg+L=1,V={sb}/{dt},W=74}
```

This displays one larger graph showing the percentage of space used on the disk, and two smaller graphs below it, showing the size of selected files compared with the free space on the disk and the total size on the disk.



As you can see, when the graphs are reduced to half-height, the percentage value can no longer be drawn over the top.

**V=<num>/<den>**  
>

Specifies the two values that are used to calculate the bar graph percentage. This is equivalent to the format used in the **{cp}** code (described above) - **<num>** is the value to be divided and **<den>** is the value it will be divided by. The result is then multiplied by 100 to give the final percentage and this value is displayed by the graph.

Both **<num>** and **<den>** can refer to any of the status bar codes that produce a number. They can also specify an absolute value, and this value can be given as bytes, kilobytes (with the **kb** suffix), megabytes (with the **mb** suffix) or gigabytes (with the **gb** suffix).

For example, to produce a bar graph that indicates the size of selected items (**{sb}**) relative to the free space in the destination file display (**{dfDb}**), you would specify **V=sf/dfDb**.

As another example, a bar graph that indicates the size of selected files relative to 4.7 GB (the size of an empty DVD), you would specify **V=sb/4.7gb**.

The default behaviour of a bar graph is to indicate the amount of disk space used - this is equivalent to **V=du/dt**.

## Hiding sections on the status bar

It's possible to configure a section of the status bar definition to only be displayed if a particular value is non-zero (or is zero). The default status bar definition uses this when it displays the number of hidden items in the current folder. Rather than have a count of zero showing when no items are hidden, the section is hidden completely - it is only displayed when one or more items are actually hidden.

The section that is to be conditionally hidden is surrounded with a pair of special codes. Opus looks at the values of all status bar codes within the section, and applies the desired test.

- **{h!} ... {h!}** - display only if all codes within the section are non-zero
- **{h#} ... {h#}** - display if any code within the section is non-zero
- **{h?} ... {h?}** - display only if all codes within the section are zero

For example:

**{h#}{sd}/{td} folders, {sf}/{tf} files{h#}**

This displays the selected and total number of folders and files. If there are **no** files or folders at all, the section will be hidden.

Status bar codes can also be included within the condition codes themselves. This lets you test status bar values without actually displaying those values on the status bar. For example:

```
{h?{si}}{ti} object(s){h?}{h!{hi}?{si}}{h!}{si} object(s) selected{h!}
```

This contains two conditionally hidden sections. Each section tests the value of the `{si}` code (number of selected items) in a different way, meaning only one of the two sections will be displayed at once.

- The first section uses an embedded code in the hide code to test `{si}`. The `{ti} object(s)` string will be displayed if the number of selected items is zero, but note that the `{si}` code itself is not displayed.
- The second section uses the `{h!}` code to display the `{si} object(s) selected` string if the number of selected items is non-zero.

You can even embed multiple condition codes to perform more complex tests. For example, `{h!{sf}?{sd}} ... {h!}` will display the conditional text only if `{sf}` (selected files) is non-zero and `{sd}` (selected folders) is zero.

While a single hidden section can depend on multiple conditions, and you can have multiple separate hidden sections on the same status bar, you **must not nest or overlap** multiple hidden sections in the same place as it will not work.

**Variables** can be used to show or hide sections of the status bar, and that in turn allows you to create toolbar buttons or hotkeys which toggle extra information, or to write scripts which automatically change the information displayed as you change folders or change display modes, for example. See the [{var:...} code documentation](#) for a detailed discussion.

For example:

```
{h!{var:glob:ShowExtraInfo}} {smp3} / {tmp3} ...other extra info... {h!}
```

**Conditional tests** can also be used to show or hide sections based on the current folder, whether a particular path exists, or the state returned by various [internal commands](#). See the [{ifpath:...}](#), [{ifexists:...}](#) and [{if:...} code documentation](#) for details.

For example:

```
{h!{ifpath:C:\}} You are in the root of C:\! {h!}
```

```
{h!{ifpath:/downloads}!{ifexists:.*.dll}} WARNING: DLLs in the downloads folder! {h!}
```

```
{h!{if:!Set VIEW=Details}}{sel:size} {sel:write}{h!}
```

Keep in mind that some conditional tests may impact performance. See the section about them for a detailed discussion of that.

## Padding sections on the status bar

The following codes let you pad the width of sections to align with various Lister elements. For example, you can pad a section to align with the edge of the folder tree, so that no matter how wide the tree is the section will always be the same width. This is particularly useful in dual-display Listers, where you can define the status bar to show information for both file displays at once - in this instance, you would want to pad the section for the right file display to always align with it.

Code	Description
{padX}	<b>Pad to percentage of total space</b> Pads the width of a status bar section to a percentage of the total width of the status bar.  X is a value from 1 to 100. For example, {pad50} pads the section to 50% of the width of the Lister.
{rpadX}	<b>Pad to percentage of remaining space</b> Pads the width of a status bar section to a percentage of the remaining width on the status bar (that is, the space from the beginning of this section to the right-hand edge of the Lister).  X is a value from 1 to 100. For example, {pad25} pads the section to 25% of the remaining space.
{wtree}	<b>Pad to width of folder tree</b> The section will be padded to the same width as the folder tree. In a dual-display Lister, with dual trees, this will be the width of the left folder tree. This code has no effect if the folder tree is not visible.
{wtree2}	<b>Pad to width of second folder tree</b> The section will be padded to the same width as the second folder tree in

	a dual-display Lister. This code has no effect if the second folder tree is not visible.
{rtree}	<b>Pad to right edge of folder tree</b> The section will be right-aligned with the right edge of the folder tree. In a dual-display Lister, this will be the left folder tree.
{rtree-}	<b>Pad to right edge of folder tree, right-justify contents</b> The section will be right-aligned with the right edge of the folder tree. As well, the contents of this section will be right-justified.
{rtree2}	<b>Pad to right edge of second folder tree</b> The section will be right-aligned with the right edge of the second folder tree.
{rtree2-}	<b>Pad to right edge of second folder tree, right-justify contents</b> The section will be right-aligned with the right edge of the second folder tree. As well, the contents of this section will be right-justified.
{wleft}	<b>Pad to width of left file display</b> The section will be padded to the same width as the left file display (or the only file display in a single-display Lister).
{rleft}	<b>Pad to right edge of left file display</b> The section will be right-aligned with the right edge of the left file display (or the only file display in a single-display Lister).
{rleft-}	<b>Pad to right edge of left file display, right-justify contents</b> The section will be right-aligned with the right edge of the left file display. As well, the contents of this section will be right-justified.
{widthX}	<b>Absolute width, scale with DPI</b> Sets the section to an absolute width in pixels (adjusted for your system DPI).  <b>X</b> is the desired width. For example, <b>{width240}</b> pads the section to 240 pixels wide at standard DPI and 480 pixels wide at 200% DPI.
{width-X}	<b>Absolute width</b> Sets the section to an absolute width in pixels.  <b>X</b> is the desired width. For example, <b>{width-240}</b> pads the section to 240 pixels wide regardless of system DPI.

# Command Reference

[Toolbar](#) buttons, [menu](#) items and [hotkeys](#) are constructed from one or more commands. There are two main types of commands, or instructions: internal and external.

- [Internal commands](#) are the set of functions built-in to Opus - its "native language" if you like. There are around 30 commands that make up the internal command set, and each command takes one or more [arguments](#) that defines or modifies its behavior.
- External commands are other programs outside of Opus. Using one or more special [control codes](#), a toolbar button or hotkey can launch an external program and pass information like the names of selected files to it.

The third type of instruction that can be used in a button or hotkey is the [command modifier](#). These are not commands themselves - instead, they modify the behaviour of the overall button.

## Argument Types

The following qualifiers are used in the internal command templates to indicate the type of each argument. Remember that you **never** type the qualifiers when using arguments - they are merely a clue as to the argument type.

Qualifier	Type	Description
/S	Switch	Indicates a switch argument (a Boolean option that can either be on or off).
/K	Keyword	Indicates a value argument (a value must be provided following the argument keyword).
/O	Optional	Indicates an optional argument (can be used either by itself, as a switch, or with a following value).
/N	Numeric	The value of the argument must be a number.
/M	Multiple	The argument can accept multiple values (e.g. a list of files).
/R	Raw	The argument accepts a "raw" value. For these arguments, the rest of the command line following the argument name is taken as the value.  Arguments of this type are the only ones that do not require quotes around values which contain spaces.

See the [Internal Command Arguments](#) page for a full description of the various argument types.

## Internal Commands

The internal command set consists of the following commands. Most commands perform multiple functions that can be selected or modified by the various arguments, however the primary purpose for each command is as follows. See the documentation for each command for a full list of its arguments and the functions that it can perform.

- [CLI](#): Displays the CLI window letting you enter commands via the keyboard.
- [Clipboard](#): Copy and paste files between folders.
- [Close](#): Close Lister and the whole program.
- [ContextMenu](#): Display and access context menu commands.
- [Copy](#): Copy and move files and folders.
- [CreateFolder](#): Create new folders.
- [Delete](#): Delete files and folders.
- [Favorites](#): Display and manipulate your favorite folder list.
- [FileType](#): Trigger filetype-specific commands and events.
- [Find](#): Search for files and folders.
- [GetSizes](#): Calculate the size of folders.
- [Go](#): Navigate to another location in the Lister.
- [Help](#): Display the help file.
- [Image](#): Perform simple image manipulation.
- [Join](#): Join files together.
- [Marker](#): Display commands provided by third-party namespace extensions.
- [Play](#): Play sound files.
- [Prefs](#): Access the Preferences system.
- [Print](#): Print files and folder listings.
- [Properties](#): Display the system Properties for files and folders.
- [Recent](#): Access the recent folder list.
- [Rename](#): Rename files and folders.
- [Select](#): Perform wildcard selection of files.
- [Set](#): Modify various Lister settings.
- [SetAttr](#): Change file attributes and timestamps.
- [Show](#): Display images.
- [Split](#): Split files into parts.
- [Toolbar](#): Open and close different toolbars.
- [Undo](#): Undo actions like delete to recycle bin.

## CLI

The **CLI** internal command can be used to:

- Open the [Command Line Interpreter](#) window
- Open an MS-DOS prompt (with the current directory set to the current folder in the Lister)
- Cause the [find-as-you-type](#) field to appear in its various modes.

### Command Arguments:

Argument	Type	Possible values	Description
<i>no argument</i>	-	-	Opens an instance of the <a href="#">Command Line Interpreter</a> window.
DOSPROMPT	/O	<i>(no value)</i>	Opens a DOS prompt with the current directory set to the folder displayed in the source file display. You can override the current directory by using the <b>cd</b> directive before the <b>CLI</b> command (requires using the advanced command editor).  <i>Example: CLI DOSPROMPT</i>
		<b>selfolder</b>	Uses the first selected sub-folder in the source display as the CD for the DOS prompt.  <i>Example: CLI DOSPROMPT=selfolder</i>
		<b>admin</b>	On Vista and above, opens the DOS prompt elevated (after a UAC prompt).  <i>Example: CLI DOSPROMPT=admin</i>
		<b>noadmin</b>	Prevents the DOS prompt from being elevated.  <i>Example: CLI DOSPROMPT=noadmin</i>
		<b>powershell</b>	Opens a PowerShell prompt rather than a DOS prompt.  <i>Example: CLI DOSPROMPT=powershell,admin</i>
		<b>powershellise</b>	Opens a PowerShell ISE rather than a DOS prompt.

			<p><i>Example: CLI DOSPROMPT=powershellise</i></p> <p>When launching a PowerShell ISE, any color parameters are ignored. This limitation affects the ISE only; DOS prompts and normal PowerShell windows can both have colors specified.</p>																				
		<b>color=&lt;color&gt;</b>	<p>Sets the text and background colors of the DOS window. You can use this by itself, or in conjunction with the admin argument to override the default color when the prompt is elevated.</p> <p>The &lt;color&gt; value is specified with two hexadecimal digits - the first corresponds to the background color, and the second to the foreground. The possible colors depend on your system's settings but are as below by default:</p> <table border="1" data-bbox="874 952 1437 1189"> <thead> <tr> <th colspan="4">Supported color values</th> </tr> </thead> <tbody> <tr> <td>0 = Black</td> <td>1 = Blue</td> <td>2 = Green</td> <td>3 = Aqua</td> </tr> <tr> <td>4 = Red</td> <td>5 = Purple</td> <td>6 = Yellow</td> <td>7 = White</td> </tr> <tr> <td>8 = Gray</td> <td>9 = Light Blue</td> <td>A = Light Green</td> <td>B = Light Aqua</td> </tr> <tr> <td>C = Light Red</td> <td>D = Light Purple</td> <td>E = Light Yellow</td> <td>F = Bright White</td> </tr> </tbody> </table> <p>Note that you must enclose the entire value of the <b>DOSPROMPT</b> argument in quotes when using the <b>color</b> parameter (otherwise the embedded = sign will confuse the command parser).</p> <p><i>Example: CLI DOSPROMPT="admin,color=97"</i></p>	Supported color values				0 = Black	1 = Blue	2 = Green	3 = Aqua	4 = Red	5 = Purple	6 = Yellow	7 = White	8 = Gray	9 = Light Blue	A = Light Green	B = Light Aqua	C = Light Red	D = Light Purple	E = Light Yellow	F = Bright White
Supported color values																							
0 = Black	1 = Blue	2 = Green	3 = Aqua																				
4 = Red	5 = Purple	6 = Yellow	7 = White																				
8 = Gray	9 = Light Blue	A = Light Green	B = Light Aqua																				
C = Light Red	D = Light Purple	E = Light Yellow	F = Bright White																				
		<b>nocolor</b>	<p>Prevents the color of the DOS prompt from being set when elevated.</p> <p><i>Example: CLI DOSPROMPT=admin,nocolor</i></p>																				
		<b>wsl</b>	<p>Opens a WSL (Windows Subsystem for Linux) shell window. Note that WSL must be installed from the Windows Store.</p> <p><i>Example: CLI DOSPROMPT=wsl</i></p>																				

QUICKCMD	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in Command mode, which lets you enter an ad-hoc Opus command to execute in the current file display. This lets you bind a <a href="#">hotkey</a> to bring the FAYT field up in the specific mode.</p> <p><i>Example: CLI QUICKCMD</i></p>
		<command>	<p>Displays the <a href="#">find-as-you-type</a> field in Command mode, and initialises it with the specified command.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the command instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKCMD Help</i>  <i>Example: CLI QUICKCMD noselect:Help</i></p>
QUICKDOSCMD	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in DOS Command mode, which lets you enter a command to execute in a DOS prompt.</p> <p><i>Example: CLI QUICKDOSCMD</i></p>
		<command>	<p>Displays the <a href="#">find-as-you-type</a> field in DOS Command mode, and initialises it with the specified command.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the command instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKDOSCMD dir</i>  <i>Example: CLI QUICKDOSCMD noselect:dir</i></p>
QUICKFILTER	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in Filter mode, which lets you filter the current file list.</p> <p><i>Example: CLI QUICKFILTER</i></p>
		<pattern>	<p>Displays the <a href="#">find-as-you-type</a> field in Filter mode, and initialises it with the specified pattern.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the pattern instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKFILTER *.jpg png</i>  <i>Example: CLI QUICKFILTER noselect:*.jpg</i></p>
QUICKFIND	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in Find mode, which lets you scroll to the first file matching the entered string.</p>

			<p><i>Example: CLI QUICKFIND</i></p>
		<search string>	<p>Displays the <a href="#">find-as-you-type</a> field in Find mode and initialises it with the specified string.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the string instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKFIND di</i>  <i>Example: CLI QUICKFIND noselect:di</i></p>
QUICKFTPCMD	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in a special mode that lets you enter a command to send directly to a remote FTP server. This command only works when you are currently connected to an FTP site. You can view the results of your command in the <a href="#">FTP log</a>.</p> <p><i>Example: CLI QUICKFTPCMD</i></p>
		<command>	<p>Displays the <a href="#">find-as-you-type</a> field in FTP command mode, and initialises it with the specified command.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the command instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKFTPCMD chmod * 755</i>  <i>Example: CLI QUICKFTPCMD noselect:chmod *</i></p>
QUICKGO	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in a special mode ("go" mode) that lets you navigate to another folder in the current file display.</p> <p><i>Example: CLI QUICKGO</i></p>
		<path>	<p>Displays the <a href="#">find-as-you-type</a> field in "go" mode, and initialises it with the specified path.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the path instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKGO C:\Program Files</i>  <i>Example: CLI QUICKGO noselect:C:\</i></p>
QUICKRANGE	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in Range mode, which lets you select files by index (or by a range of indices). This only works when the Index column has been added to the file display.</p>

			<p><i>Example: CLI QUICKRANGE</i></p>
		<range>	<p>Displays the <a href="#">find-as-you-type</a> field in Range mode, and initialises it with the specified range string.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the range string instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKRANGE 1-10,20-30</i>  <i>Example: CLI QUICKRANGE noselect:1-10</i></p>
QUICKSEARCH	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in Search mode, which lets you initiate a <a href="#">Windows Search</a> of the current folder.</p> <p><i>Example: CLI QUICKSEARCH</i></p>
		<query term>	<p>Displays the <a href="#">find-as-you-type</a> field in Search mode, and initialises it with the specified query term.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the query string instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKSEARCH author:davidson</i>  <i>Example: CLI QUICKSEARCH noselect:author:davidson</i></p>
QUICKSELECT	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in Select mode, which lets you select files in the current folder by <a href="#">wildcard pattern</a>.</p> <p><i>Example: CLI QUICKSELECT</i></p>
		<pattern>	<p>Displays the <a href="#">find-as-you-type</a> field in Select mode and initialises it with the specified pattern.</p> <p>Prefix with <b>noselect:</b> to place the cursor at the end of the pattern instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKSELECT *.doc</i>  <i>Example: CLI QUICKSELECT noselect:*.doc</i></p>
QUICKTABS	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in Tabs mode, which lets you search and switch folder tabs.</p> <p><i>Example: CLI QUICKTABS</i></p>
		<pattern>	<p>Displays the <a href="#">find-as-you-type</a> field in Tabs mode and initialises it with the specified text.</p>

			<p>Prefix with <b>noselect</b>: to place the cursor at the end of the text instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKTABS docu</i>  <i>Example: CLI QUICKTABS noselect:docu</i></p>
QUICKWSLCMD	/O/R	(no value)	<p>Displays the <a href="#">find-as-you-type</a> field in WSL script mode, which lets you enter a command to execute in a WSL (Windows System For Linux) window. Note that WSL needs to be installed from the Windows Store.</p> <p><i>Example: CLI QUICKWSLCMD</i></p>
		<command>	<p>Displays the <a href="#">find-as-you-type</a> field in WSL script mode, and initialises it with the specified command.</p> <p>Prefix with <b>noselect</b>: to place the cursor at the end of the command instead of initially selecting it for typing over.</p> <p><i>Example: CLI QUICKWSLCMD ls</i>  <i>Example: CLI QUICKWSLCMD noselect:ls</i></p>
SCRIPTMODE	/O	(no value)	<p>Displays the CLI in Script Mode, which provides a simple way to test <a href="#">scripts</a> before adding them to buttons.</p> <p><i>Example: CLI SCRIPTMODE</i></p>
		<language>	<p>Displays the CLI in Script Mode with the language type set to the specified language. If you don't specify a language the CLI will remember the previous language used.</p> <p><i>Example: CLI SCRIPTMODE=jscript</i></p>

## Clipboard

The **Clipboard** internal command can be used to:

- Copy and paste files and folders from one folder to another
- Copy the **names** of selected files and folders to the clipboard
- Paste image or text data from the clipboard as files on disk
- Paste files, image or text data from the clipboard into archives (either new or existing)

## Command Arguments:

Argument	Type	Possible values	Description
ADD	/S	(no value)	<p>Adds additional files to those already on the clipboard instead of replacing them. It is used in conjunction with <b>COPY</b> or <b>CUT</b>. This lets you place multiple items on the clipboard from different source folders, and then paste them into the desired destination folder in one go.</p> <p><i>Example: Clipboard COPY ADD</i></p>
AS	/K	<filename>	<p>Overrides the default filename used when pasting images or text to disk. Normally when you paste clipboard image data, Opus creates a file called <i>Clipboard Image.png</i> (or another suffix, depending on the file format), and when you paste text data, Opus creates a file called <i>Clipboard Text.txt</i>. Use this argument to change the filename. The default file extension will be applied automatically.</p> <p>This argument can also be used when pasting the clipboard contents as an archive (e.g. with <b>Clipboard PASTE=zip</b>). In this case the filename for the archive is normally generated automatically from the clipboard contents - this argument lets you override it.</p> <p><i>Example: Clipboard PASTE AS PastedData</i></p>
		ask	<p>When pasting image or text data to a file, this argument causes Opus to prompt you for a name for the created file, and in the case of image data, the image format to use.</p> <div data-bbox="938 1570 1406 1787" data-label="Image"> </div> <p>This argument can also be used when pasting the clipboard contents as an archive (e.g. with <b>Clipboard PASTE=7z</b>). In this case you will be prompted for the name of the new</p>

			<p>archive and any archive format-specific parameters.</p> <p>The <b>Shrink image to compensate for system DPI</b> option will scale the pasted image down if your system DPI is higher than 100%.</p> <p><i>Example: Clipboard PASTE AS=ask</i></p>
COPY	/S	(no value)	<p>Copies all selected files and folders to the clipboard.</p> <p><i>Example: Clipboard COPY</i></p>
COPYNAMES	/O	(no value)	<p>Copies the names of all selected files and folders to the clipboard (the file names themselves are copied, in text format, rather than the actual files).</p> <p>The default format (with no value specified) copies the full path and filename of all selected items to the clipboard, with one file on each line. For example,</p> <pre>C:\Windows\notepad.exe C:\Windows\regedit.exe</pre> <p>The various values for the <b>COPYNAMES</b> argument let you modify the format filenames are copied in. Some values can be combined - see the examples given below for ideas on what you can do. Also see the <b>REGEXP</b> argument which lets you control the format yourself using regular expressions.</p> <p><i>Example: Clipboard COPYNAMES</i></p>
		<b>nopaths</b>	<p>Copies just the names of selected items to the clipboard - the paths are not copied. For example,</p> <pre>notepad.exe regedit.exe</pre> <p><i>Example: Clipboard COPYNAMES=nopaths</i></p>

	<b>url</b>	<p>Copies the names of selected items to the clipboard in URL format. For files on an FTP site this will result in an <b>ftp://</b> style path that should be accepted by other FTP programs. For local files, filenames will be copied as <b>file://</b> style links (e.g. <i>file:///C:/Windows</i>). For example,</p> <pre>file:///C:/Windows/notepad.exe file:///C:/Windows/regedit.exe</pre> <p><i>Example: ClipboardCOPYNAMES=url</i></p>
	<b>hash</b>	<p>Copies the filename of each selected file along with its <b>MD5</b> checksum. For example,</p> <pre>notepad.exe : f2c7bb8acc97f92e987a2d4087d021b 1 regedit.exe : 2e2c937846a0b8789e5e91739284d1 7a</pre> <p><i>Example: Clipboard COPYNAMES=hash</i></p>
	<b>hash2</b>	<p>Copies the <b>MD5</b> checksum of selected files in an alternative format, one compatible with the venerable <i>MD5Sum</i> utility. For example,</p> <pre>f2c7bb8acc97f92e987a2d4087d021b 1 *notepad.exe 2e2c937846a0b8789e5e91739284d1 7a *regedit.exe</pre> <p><i>Example: Clipboard COPYNAMES=hash2</i></p>
	<b>hash3</b>	<p>Copies the <b>MD5</b> checksum of selected files <b>without</b> the filenames. You would probably only want to use this format on a single file at a time. For example,</p> <pre>f2c7bb8acc97f92e987a2d4087d021b 1 2e2c937846a0b8789e5e91739284d1 7a</pre> <p><i>Example: Clipboard COPYNAMES=hash3</i></p>
	<b>hash4</b>	<p>Copies the filename of each selected file along with its <b>SHA-1</b> checksum. For example,</p>

			<pre>notepad.exe : 7eb0139d2175739b3ccb0d11100678 20be6abd29 regedit.exe : f48138dc476e040b8a9925c7d2650b 706178e863</pre> <p><i>Example: Clipboard COPYNAMES=hash4</i></p>
		<b>hash5</b>	<p>Copies the <b>SHA-1</b> checksum of selected files in an alternative format. For example,</p> <pre>7eb0139d2175739b3ccb0d11100678 20be6abd29 *notepad.exe f48138dc476e040b8a9925c7d2650b 706178e863 *regedit.exe</pre> <p><i>Example: Clipboard COPYNAMES=hash5</i></p>
		<b>hash6</b>	<p>Copies the <b>SHA-1</b> checksum of selected files <b>without</b> the filenames. You would probably only want to use this format on a single file at a time. For example,</p> <pre>7eb0139d2175739b3ccb0d11100678 20be6abd29 f48138dc476e040b8a9925c7d2650b 706178e863</pre> <p><i>Example: Clipboard COPYNAMES=hash6</i></p>
		<b>hashcache</b>	<p>Add the <b>hashcache</b> keyword to one of the <b>hash</b> options to make Opus use the checksum cache; if the file has previously had its checksum calculated and does not appear to have changed, the cached value will be used.</p> <p><i>Example: Clipboard COPYNAMES=hash2,hashcache</i></p>
		<b>unc</b>	<p>When the <b>unc</b> value is specified, and the files whose names are being copied reside on a mapped network drive, the function will resolve their mapped paths to a UNC path and copy that to the clipboard instead. For example, if <b>Z:</b> were a mapped network drive, <b>Clipboard COPYNAMES</b> would return:</p>

		<p>Z:\Windows\notepad.exe Z:\Windows\regedit.exe</p> <p>Whereas <b>Clipboard COPYNAMES=unc</b> might return:</p> <p>\\win7vm\c\Windows\notepad.exe \\win7vm\c\Windows\regedit.exe</p> <p>If the current folder is not on a mapped network drive, the unc argument has no effect.</p> <p><i>Example: Clipboard COPYNAMES=unc</i></p>
	<b>short</b>	<p>Copies the short (8.3) filenames of selected files, rather than their long filenames. For example,</p> <p>C:\PROGRA~1\GPSOFT~1\DIRECT~1\dopus.exe C:\PROGRA~1\GPSOFT~1\DIRECT~1\dopuslib.dll</p> <p><i>Example: Clipboard COPYNAMES=short,nopaths</i></p>
	<b>single</b>	<p>When multiple files are selected, this will copy all the names on a single line (separated by spaces) rather than one item per line. If any item names contain a space, they will be surrounded by quotation marks. For example,</p> <p>dopus.exe dopuslib.dll "Opus 10 What's New.pdf"</p> <p><i>Example: Clipboard COPYNAMES=nopaths,single</i></p>
	<b>path</b>	<p>Copies just the path of selected items, without the filenames (the opposite of <b>nopaths</b>). For example,</p> <p>C:\Program Files\GPSSoftware\Directory Opus</p> <p><i>Example: Clipboard COPYNAMES=path</i></p>
	<b>capsman tics</b>	<p>Makes the function behave like the Windows <i>Copy as path</i> context menu command (one file path per line, and lines are always quoted).</p>

			<p><i>Example: Clipboard</i>  <b>COPYNAMES=capsemantics</b></p>
		<b>quote</b>	<p>Forces filenames and paths copied to the clipboard to be enclosed with double-quotes even if they don't contain spaces (and therefore ordinarily wouldn't need quotes).</p> <p><i>Example: Clipboard</i> <b>COPYNAMES=quote</b></p>
		<b>wsl</b>	<p>Copies file paths in WSL (Windows Subsystem for Linux) format. For example, <b>C:\Temp</b> would be converted to <b>/mnt/c/Temp</b>.</p> <p><i>Example: Clipboard</i> <b>COPYNAMES=wsl</b></p>
COPYQUEUE	/O	(no value)	<p>Use in conjunction with the <b>PASTE</b> argument to paste files with <a href="#">copy queuing</a> enabled. With no value specified, copies will be queued automatically if required. This can override the <b>Automatically manage file copy queues</b> option on the <a href="#">File Operations / Copy Options</a> Preferences page.</p> <p><i>Example: Clipboard</i> <b>PASTE COPYQUEUE</b></p>
		<queue name>	<p>When you specify a queue name as the value for this argument, it enables manual copy queuing when pasting files. That is, with a name specified, file pastes will always be queued to the specified queue - if no name is specified for the argument, pastes will only be queued if needed.</p> <p><i>Example: Clipboard</i> <b>PASTE COPYQUEUE=MyQueue</b></p>
		<b>none</b>	<p>Used to disable copy queuing - whether enabled in Preferences, or otherwise enabled by the <b>shift</b> keyword.</p> <p><i>Example: Clipboard</i> <b>PASTE COPYQUEUE=none</b></p>
		<b>shift</b>	<p>Lets you specify two alternate parameters for the <b>COPYQUEUE</b> argument. The value specified before the <b>shift</b> keyword is used if the <b>Shift</b> key is not held down - the value after it is used if it is. For example, you could configure a paste button to queue files to a</p>

			<p>specific queue if the <b>Shift</b> key were held down, and to disable queuing otherwise.</p> <p><i>Example: Clipboard PASTE COPYQUEUE=none,shift,MyQueue</i></p>
		<b>quiet</b>	<p>Specify the <b>quiet</b> keyword to suppress the prompt that normally indicates a copy operation has been queued.</p> <p><i>Example: Clipboard PASTE COPYQUEUE=MyQueue,quiet</i></p>
CUT	/S	(no value)	<p>Cuts all selected files and folders to the clipboard (nothing happens to the files immediately, but when you paste them somewhere else they are moved rather than copied).</p> <p><i>Example: Clipboard CUT</i></p>
CUTNOCOPYQUEUEWHENSAME	/S	(no value)	<p>Used with <b>Clipboard PASTE</b>, this disables the use of a copy queue when the files on the clipboard are being cut (that is, moved), and the source and destination are on the same drive partition.</p> <p><i>Example: Clipboard PASTE CUTNOCOPYQUEUEWHENSAME</i></p>
EXPANDNEWLINES	/S	(no value)	<p>Used with <b>Clipboard SET</b>, allows you to use <b>\n</b> to insert a new line into the string, so that you can set a clipboard string consisting of multiple lines. You can also use <b>\\</b> to insert a literal backslash.</p> <p><i>Example: Clipboard EXPANDNEWLINES SET Hello\nWorld</i></p> <p>When inserting paths into the clipboard string while using <b>EXPANDNEWLINES</b>, you should use <a href="#">escbackslash</a> to prevent the backslashes in the paths from being misinterpreted.</p> <p><i>Example (all on one line): Clipboard EXPANDNEWLINES SET {sourcepath escbackslash}\n{destpath escbackslash}</i></p>
NEWLINEIFADDING	/S	(no value)	<p>Used with <b>Clipboard ADD SET</b>, adds a new line between the existing clipboard data and the string to be added. If the clipboard is currently blank or has non-text data then the</p>

			<p>new line is not added.</p> <p><i>Example: Clipboard ADD NEWLINEIFADDING SET This is a new line.</i></p>
NOFROMFOCUS	/S	(no value)	<p>The default behaviour for the <b>Clipboard</b> command is to operate on either the source file display, or the Folder Tree, depending on which one has the input focus. This lets you use the same command to copy folders in the tree as well as files in the file display. Specify this argument to force the command to always operate on the source file display and ignore the folder tree.</p> <p><i>Example: Clipboard COPY NOFROMFOCUS</i></p>
PASTE	/O	(no value)	<p>Pastes any files and folders previously copied to the clipboard into the current Lister. If the files were placed on the clipboard by a "cut" operation they will be moved, otherwise they will be copied to the new location.</p> <p><i>Example: Clipboard PASTE</i></p>
		<b>jpg</b>	<p>Forces image data on the clipboard to be pasted in JPEG format (overriding the Preferences default setting). You can also specify the JPEG image quality.</p> <p><i>Example: Clipboard PASTE=jpg:85</i></p>
		<b>png</b>	<p>Pastes image data in PNG format.</p> <p><i>Example: Clipboard PASTE=png</i></p>
		<b>gif</b>	<p>Pastes image data in GIF format.</p> <p><i>Example: Clipboard PASTE=gif</i></p>
		<b>bmp</b>	<p>Pastes image data in BMP format.</p> <p><i>Example: Clipboard PASTE=bmp</i></p>
		<b>zip</b>	<p>Pastes clipboard contents (files, image or text data) as a new ZIP archive. The filename of the archive will be generated automatically from the clipboard contents - you can override this with the <b>AS</b> argument.</p>
		<b>7z</b>	<p>Pastes clipboard contents as a new 7Zip archive.</p>

		<i>&lt;archive suffix&gt;</i>	Pastes clipboard contents as a new archive of the specified type (you can specify any archive suffix that Opus supports creation of).
PASTELINK	/O	<i>(no value)</i>	Pastes shortcuts to any files and folders previously copied to the clipboard into the current Lister. For example, if you use the <b>Clipboard COPY</b> command on the <i>C:\Windows</i> folder, navigate to another folder, and run the <b>Clipboard PASTELINK</b> command, it would paste a shortcut called <i>Windows - Shortcut.lnk</i> .  <i>Example: Clipboard PASTELINK</i>
		<b>junction</b>	Creates a junction to any folders that are on the clipboard. Junctions are only supported on NTFS volumes, and you can not create junctions to files - only folders.  <i>Example: Clipboard PASTELINK=junction</i>
		<b>hardlink</b>	Creates a hardlink to any files that are on the clipboard. Hardlinks are only supported on NTFS volumes, and you can not create hardlinks to folders - only to files.  <i>Example: Clipboard PASTELINK=hardlink</i>
		<b>softlink</b>	Creates a softlink to any files or folders that are on the clipboard. Note that the link target is stored as an absolute path. Soft links support both files and folders, but are only supported on Vista and above (and again, only on NTFS volumes). Creating a softlink requires administrator access so Opus will display a UAC prompt if necessary when you run this command.  <i>Example: Clipboard PASTELINK=softlink</i>
		<b>relsoftlink</b>	Creates a softlink to any files or folders that are on the clipboard, storing a relative target path if possible. A regular absolute link will be created if the target can not be expressed relative to the link.  <i>Example: Clipboard PASTELINK=relsoftlink</i>
		<b>auto</b>	Automatically determines the most suitable type of link to create. On Vista and above, this will be a softlink - on Windows XP,

			<p>either a junction or a hardlink depending on the type of item. If a non-filesystem object is on the clipboard (e.g. you are trying to make a link to a virtual folder like the <i>Control Panel</i>) or the target drive is not formatted with NTFS then it will create a shortcut.</p> <p><i>Example: Clipboard PASTELINK=auto</i></p>
		<b>autosoft</b>	<p>Does the same as <b>auto</b> except that it will not try to create softlinks. In other words: It will create either a junction or a hardlink depending on the type of item. If a non-filesystem object is on the clipboard (e.g. you are trying to make a link to a virtual folder like the <i>Control Panel</i>) or the target drive is not formatted with NTFS then it will create a shortcut.</p> <p><i>Example: Clipboard PASTELINK=autosoft</i></p>
PREFERIMAGE	/S	(no value)	<p>When pasting data from the clipboard into a new file, image data is normally given priority over text data. Opus makes an exception to this rule in certain situations.</p> <p>For example, if you use Microsoft Excel and copy some cells to the clipboard, Excel puts both the text from the cells and a screenshot of them into the clipboard. When pasting data from Excel, people usually want the text, not the screenshot. Accordingly, when Opus detects that pasted data is from Excel it gives text priority over images, contrary to its behaviour with data from other programs.</p> <p>If the <b>PREFERIMAGE</b> argument is specified, image data will <i>always</i> be given priority over text data, regardless of where the data came from. This allows you to paste the image data from Excel, if that's what you want. (You'll still get a text file if there is only text data on the clipboard.)</p> <p><i>Example: Clipboard PASTE PREFERIMAGE</i></p>
PREFERTEXT	/S	(no value)	<p>When pasting data from the clipboard into a new file, image data is normally given priority over text data. (There are exceptions</p>

			<p>to this rule; see the <b>PREFERIMAGE</b> argument, above.)</p> <p>Normally, if another program puts both text and image data into the clipboard and you then paste in Opus to save the data into a new file, the text data will be ignored and you'll get an image file (BMP, JPG, GIF or PNG).</p> <p>For example, a paint program may place both a photo and a description of the photo into the clipboard, and pasting into Opus would normally create an image file containing the photo, not a text file containing the description.</p> <p>The <b>PREFERTEXT</b> argument gives text priority over images, so you'll get a text file in that situation instead. (You'll still get an image file if there is only image data on the clipboard.)</p> <p><i>Example: Clipboard PASTE PREFERTEXT</i></p>
REGEXP	/K/ M	<search> <replace> > ...	<p>In conjunction with <b>COPYNAMES</b> lets you perform <a href="#">regular expression</a> manipulation on the filenames as they are put into the clipboard - effectively letting you determine your own clipboard format.</p> <p>The values specified for this argument are one or more pairs of strings - the first of each pair is the pattern to search for, and the second of each pair is the replace string. For example, to strip off the suffixes of all filenames when they are copied to the clipboard, the search string would be <code>(.*)\.(.*)</code> and the replacement string would be <code>\1</code>.</p> <p><i>Example: Clipboard COPYNAMES=nopaths REGEXP "(.*)\.(*)" "\1"</i></p>
SCREENSHOT	/O	(no value)	<p>Takes a screenshot of the active Lister and copies it to the clipboard. Equivalent to pushing <b>Alt+PrtScr</b>.</p> <p><i>Example: Clipboard SCREENSHOT</i></p>

	<b>all</b>	<p>Takes a screenshot of the desktop and copies it to the clipboard. Equivalent to pushing <b>PrtScr</b>.</p> <p><i>Example: Clipboard SCREENSHOT=all</i></p>
	<b>format</b>	<p>When the <b>save</b> argument is specified this lets you specify the file format to save the screenshot in. Supported formats are <b>jpg</b>, <b>png</b>, <b>gif</b> and <b>bmp</b>.</p> <p><i>Example: Clipboard SCREENSHOT=save,format:jpg</i></p>
	<b>name</b>	<p>When the <b>save</b> argument is specified this lets you configure the name to save the screenshot to. If not specified a default name is used. You can insert the current date in the name using the <b>%date%</b> code. Remember to enclose value of the <b>SCREENSHOT</b> argument in quotes if your desired name contains spaces.</p> <p><i>Example: Clipboard SCREENSHOT=save,name:My_Opus_%date</i>  <i>Example: Clipboard SCREENSHOT "save,name:Opus Screenshot"</i></p>
	<b>quality</b>	<p>When the <b>save</b> argument is specified and the screenshot is saved as a jpeg image, this lets you specify the quality of the compressed image.</p> <p><i>Example: Clipboard SCREENSHOT=save,format:jpg,quality:85</i></p>
	<b>quiet</b>	<p>Does not display a confirmation message after taking the screenshot.</p> <p><i>Example: Clipboard SCREENSHOT=all,quiet</i></p>
	<b>save</b>	<p>The screenshot will be saved to the desktop automatically (as well as copied to the clipboard). Use the <b>format</b> and <b>quality</b> arguments to control the file type. You can</p>

			configure the name with the <b>name</b> argument.  <i>Example: Clipboard SCREENSHOT=save</i>
		<b>secure</b>	Filenames will be blurred in the Lister when the screenshot is taken, to obscure potentially sensitive information.  <i>Example: Clipboard SCREENSHOT=all,secure</i>
		<b>time</b>	Displays a countdown timer before taking the screenshot.  <i>Example: Clipboard SCREENSHOT=secure,time:10</i>
SET	/K/ R	<i>user defined</i>	Copies the supplied text to the clipboard.  <i>Example: Clipboard SET {sourcepath}</i>
USESEL	/S	<i>(no value)</i>	Modifies the behaviour of the <b>PASTE</b> and <b>PASTESHORTCUT</b> functions. Normally files are pasted into the current source folder. If you specify the <b>USESEL</b> argument and a sub-folder is currently selected in the source file display, the files will be pasted into that sub-folder. This is most useful when used as a context menu command (so that you can right-click on a folder and paste the clipboard contents into it).  This also works with archives that Opus can write to - for example, if you add the following command to the context menu for the Archives <a href="#">File type Group</a> then you can right-click on an existing archive file and paste the clipboard contents directly into it.  <i>Example: Clipboard PASTE USESEL</i>

## Close

The **Close** command can be used to:

- Close Listers (either the current Lister or all)

- Collapse all open Listers to tabs in a single Lister
- Exit Directory Opus altogether
- Shutdown or restart the system immediately
- Schedule a system shutdown for some time in the future
- Automatically shutdown the system when all file operations are complete

### Command Arguments:

Argument	Type	Possible values	Description
<i>no argument</i>	-	-	Closes the active Lister. If the active Lister was the only one and the <b>Shutdown Directory Opus when the last Lister closes</b> option on the <a href="#">Launching Opus / Startup</a> Preferences page is turned on, this will also exit the program.  <i>Example: Close</i>
ALLLISTERS	/O	(no value)	Closes all currently open Listers. The program will also exit if the <b>Shutdown Directory Opus when the last Lister closes</b> option is turned on. Add the <b>CURRENTDESKTOP</b> argument to only close the ones on the current virtual desktop.  <i>Example: Close ALLLISTERS</i>
		<b>collapse</b>	Collapses all open Listers to tabs in a single Lister. All Listers except the active one will be closed, and tabs will be opened in the remaining Lister for every Lister that closed.  <i>Example: Close ALLLISTERS=collapse</i>
		<b>unique</b>	Combine <b>unique</b> with <b>collapse</b> to prevent duplicate tabs for paths which are already open in the lister. Has no effect unless <b>collapse</b> is also specified. At most one tab for each path will be opened in the left or right of the lister, with all other listers closing at the end as usual. Note that if the lister you are collapsing things into already has multiple tabs for the same folder, they will be left as-is, but no additional tabs will open for it.  <i>Example: Close ALLLISTERS=collapse,unique</i>
ALLOTHERLISTERS	/S	(no value)	Closes all open Listers <b>except</b> the active Lister. Add the <b>CURRENTDESKTOP</b> argument to only close the ones on the current

			virtual desktop.  <i>Example: Close ALLOTHERLISTERS</i>
ALLVIEWERS	/S	(no value)	Closes any open <a href="#">standalone viewer</a> windows. Add the <b>CURRENTDESKTOP</b> argument to only close the ones on the current virtual desktop.  <i>Example: Close ALLVIEWERS CURRENTDESKTOP</i>
AT	/K	<HH:MM:SS>	In conjunction with the <b>SYSTEM</b> argument this schedules an automatic shutdown of your system at a given time. The time to shutdown is given as a 24 hour time in HH:MM:SS format. If the time you specify is earlier than the current time of day, it will be taken to refer to the next day. When the time you specify is reached, a ten second countdown timer is displayed before the system it shutdown.  <i>Example: Close SYSTEM=shutdown AT 04:00:00</i>
AUTOLISTER	/O	(no value)	In conjunction with the <a href="#">Prefs RESTORE</a> command, this argument sets a global flag that specifies that when Opus restarts, it should use "auto-Lister" semantics (which basically means that when Opus restarts, it will open a Lister by default).  <i>Example: Close AUTOLISTER</i>
		no	Disables "auto-Lister" semantics when Opus restarts.  <i>Example: Close AUTOLISTER=no</i>
CANCEL	/S	(no value)	If you have previously scheduled a system shutdown with the <b>Close SYSTEM</b> and <b>AT</b> or <b>IN</b> arguments, this argument will cancel it.  <i>Example: Close CANCEL</i>
CURRENTDESKTOP	/S	(no value)	When combined with <b>ALLLISTERS</b> , <b>ALLOTHERLISTERS</b> , or <b>ALLVIEWERS</b> , the <b>CURRENTDESKTOP</b> switch will make the command only close those windows which are on the currently active virtual desktop. (Virtual desktops are a feature of Windows 10 and above, so this argument has no effect on earlier versions.)

			<i>Example: Close ALLLISTERS=collapse CURRENTDESKTOP</i>
IN	/K	<HH:MM:SS>	Schedules an automatic shutdown in a certain amount of time (contrast with <b>AT</b> , which lets you specify an absolute time of day). The time is specified in HH:MM:SS format (or MM:SS or just SS). When the time you specify is reached, a ten second countdown timer is displayed before the system it shutdown.  <i>Example: Close SYSTEM=force,poweroff IN 60:00</i>
NOSCRIPT	/S	(no value)	Closes the Lister without triggering the <a href="#">OnCloseLister</a> script events that any <a href="#">script add-ins</a> may have provided. You would probably want to use this if running the <b>Close</b> command from within an <b>OnCloseLister</b> event (to, e.g. close other Listers automatically in response to the user closing one).  <i>Example: Close NOSCRIPT</i>
PROGRAM	/O	(no value)	Exits Directory Opus immediately.  <i>Example: Close PROGRAM</i>
		<b>confirm</b>	Exits Directory Opus after displaying a confirmation dialog.  <i>Example: Close PROGRAM=confirm</i>
		<b>onlast</b>	Close the program only if this is the last remaining Lister, otherwise just close the Lister. This command lets you create a unified close/quit command.  <i>Example: Close PROGRAM=onlast</i>
QUIET	/S	(no value)	Suppresses the confirmation dialog normally displayed when shutting the system down with the <b>Close SYSTEM</b> command. The system will immediately shutdown - there will be no chance to cancel. You can also use this in conjunction with the scheduled shutdown arguments <b>AT</b> and <b>IN</b> .  <i>Example: Close SYSTEM=shutdown QUIET</i>
SYSTEM	/O	(no value)	Ends your Windows session and logs you off. A ten second countdown dialog will be displayed giving you a chance to cancel (unless you also use the <b>QUIET</b> argument).

			<i>Example: Close SYSTEM</i>
		<b>restart</b>	Restarts (reboots) the system  <i>Example: Close SYSTEM=restart QUIET</i>
		<b>shutdown</b>	This shuts the system down (depending on your hardware, this may also power off the computer; or it may display the "Your computer is now safe to be shutdown" screen).  <i>Example: Close SYSTEM=shutdown</i>
		<b>poweroff</b>	This shuts the system down and powers off the computer.  <i>Example: Close SYSTEM=poweroff</i>
		<b>force</b>	Use this in addition to the other options to force the system to shutdown even if some programs are not responding. Normally the system will wait for all running programs to exit before shutting down. If you use this option the system will shutdown immediately and you may potentially lose unsaved data if some programs are currently busy.  <i>Example: Close SYSTEM=shutdown,force AT 03:30:00 QUIET</i>
		<b>forceifhung</b>	This is the same as the <b>force</b> option, but only if there actually are any non-responding programs. Otherwise the shutdown proceeds as normal.  <i>Example: Close SYSTEM=restart,forceifhung</i>
		<b>switch</b>	If fast user-switching is enabled, this command will let you switch users. (This cannot be scheduled via the <b>AT</b> argument.)  <i>Example: Close SYSTEM=switch</i>
		<b>unattended</b>	System restart or shutdown will be done in a way which avoids additional confirmation dialogs when running via Remote Desktop or Terminal Services. The dialogs this avoids come from Windows itself; Opus will still show its own confirmation dialogs unless you also use the <b>QUIET</b> argument. If any applications, for any logged-in users, have unsaved work then system dialogs may still be generated unless you also use <b>force</b> or <b>forceifhung</b> (there is no distinction between

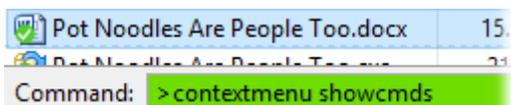
			the two when combined with <b>unattended</b> ).
			<i>Example: Close</i> <b>SYSTEM=shutdown,force,unattended</b> <b>QUIET</b>
TOGGLE	/S	(no value)	Use this argument in conjunction with the scheduled-shutdown options to toggle the scheduled shutdown on or off. If used in a toolbar or menu it causes the button to appear checked whenever a shutdown is scheduled, and unchecked if not (providing a visual indication of the current state of the shutdown scheduler).  <i>Example: Close</i> <b>SYSTEM=poweroff,force</b> <b>WHENFINISHED QUIET TOGGLE</b>
WHENFINISHED	/S	(no value)	Schedules an automatic shutdown when all outstanding file operations have completed. For example, when you are downloading a large amount of data via FTP, you could use this to have the computer automatically shutdown when the download is complete.  Note that if there are currently no executing functions, the shutdown will be triggered immediately!  <i>Example: Close</i> <b>SYSTEM=shutdown</b> <b>WHENFINISHED</b>

## ContextMenu

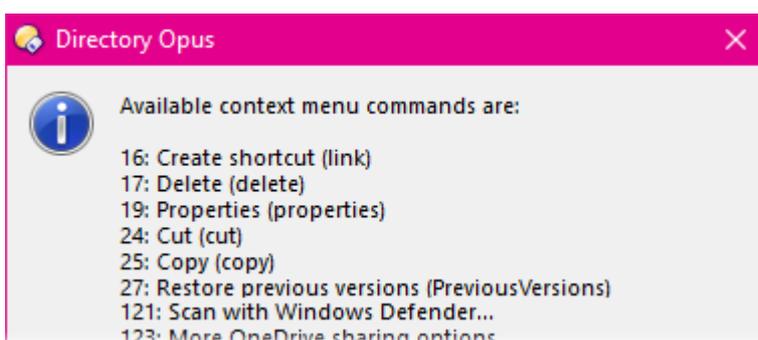
The **ContextMenu** command can be used to:

- Display a list of the context menu commands that are available for selected files, and
- Trigger a context menu command for selected files without needing to first display the context menu (so you can, for example, configure a button or hotkey to execute a command that is normally only available from the context menu)

The first step in using the **ContextMenu** command is to select the file (or a file of the type) in question and use the **SHOWCMDS** argument to view the available context menu commands. The easiest way to do this (unless you want to set up a button dedicated to this) is to use the [find-as-you-type](#) field in **Command** mode.



Opus will display a dialog containing a list of the available context menu commands. Context menu commands have an ID number and some may also have a "verb" - a plain text string that you can use to invoke the command. If a command has a verb displayed you should use this in preference to the ID, as it's possible for the ID number to change if additional context menu extensions are installed.



You can see from this example that the **Restore previous versions** command has a verb (**PreviousVersions**) whereas the **Scan with Windows Defender** command only has an ID.

Once you know what the verb or ID is for the command you want to automate, you can configure your button or hotkey with the appropriate **ContextMenu** command.

### Command Arguments:

Argument	Type	Possible values	Description
CTRL	/S	(no value)	When invoking a context menu command, tells the command handler to act as if the Ctrl key is held down. What difference this makes (if any) is up to the handler.  <i>Example: ContextMenu CTRL VERB delete</i>

EXTENDEDVERBS	/S	(no value)	<p>Enables support for <i>extended verbs</i>. These are context menu commands that are normally only displayed when the <b>Shift</b> key is held down.</p> <p><i>Example: ContextMenu SHOWCMDS EXTENDEDVERBS</i></p> <p>If you use this while invoking a command, you will probably also want to specify the <b>SHIFT</b> argument.</p> <p><i>Example: ContextMenu EXTENDEDVERBS SHIFT VERB delete</i></p>
FILE	/K/M	<filename>, ...	<p>Specify the file or files to operate on. If not specified the command will operate on all currently selected files.</p> <p>You can use this to execute commands on non-filesystem items, for example the Recycle Bin. To do this requires knowing the item's GUID, which unfortunately is beyond the scope of this help file - but as an example, the GUID for the Recycle Bin is <b>{645FF040-5081-101B-9F08-00AA002F954E}</b>. The following command will execute the "empty" verb on the Recycle Bin, causing it to empty.</p> <p><i>Example: ContextMenu FILE=::{645FF040-5081-101B-9F08-00AA002F954E} VERB=empty</i></p>
ID	/K/N	<command ID>	<p>Specify the ID of the context menu command to execute.</p> <p>You should use an item's verb or label (in that order, if it has either) in preference to its ID, because the ID may change (with the old ID potentially representing a completely different action) when you add or remove software, or even each time the menu is used.</p> <p><i>Example: ContextMenu ITEMMODE ID=79</i></p>
ITEMMODE	/S	(no value)	<p>(Windows Vista and above. Has no effect on Windows XP.)</p> <p>Specify <b>ITEMMODE</b> when you wish to list or invoke context menu commands in the same way as when a file or folder is right-clicked.</p> <p>Do not specify <b>ITEMMODE</b> if you wish to list</p>

			<p>or run commands which only appear when you right-click the background of a folder window.</p> <p>The exact difference (if any) that <b>ITEMMODE</b> makes is up to the individual command handlers. As with the <b>EXTENDEDVERBS</b> argument, some handlers will filter their menu items in or out depending on the mode.</p> <p>Typically, <b>ITEMMODE</b> will either be required or will make no difference; it is only off by default for backward-compatibility reasons (to avoid breaking existing commands which invoke items from the folder background menu).</p> <p><i>Example: ContextMenu ITEMMODE VERB PreviousVersions</i></p>
LABEL	/K	<label>	<p>Specify the label of the context menu command to execute.</p> <p>It is better to specify the command's verb, using the <b>VERB</b> argument, instead of its label, but not all commands have verbs. (Not all commands have labels, either. Context menu handlers which draw their own items may not provide the labels in a form which Opus can read.) Labels, unlike verbs, are typically language-dependent and may change between versions of software or when different files are selected. Using the label is still preferable to using the ID, however.</p> <p>You can use the <b>REGEXP</b> or <b>WILD</b> arguments if you need to match the label using a pattern, which is most often needed when the label includes part of the selected filename.</p> <p>When searching for the label in a menu with sub-menus, Opus will check all of the top-level items first, then the items directly below the top-level sub-menus, then any items nested three levels deep, and so on.</p> <p><i>Example: ContextMenu ITEMMODE LABEL "Quarantine file"</i></p>
LOOKUP	/S	(no value)	<p>Use in conjunction with the <b>VERB</b> argument to make Opus lookup the verb's ID on-the-fly instead of simply passing the verb name to the command handler. You should not normally have to do this but some context menu handlers have bugs that prevent them from doing the</p>

			lookup themselves.  <i>Example:</i> <b>ContextMenu WANTSYNC ITEMMODE LOOKUP VERB PickLinkSource FILE "C:\Template Folder"</b>
REGEXP	/S	(no value)	Used in conjunction with the <b>LABEL</b> argument to specify that the label string is a regular expression.  <i>Example:</i> <b>ContextMenu ITEMMODE REGEXP LABEL "Add to *.*.zip.*"</b>
SHIFT	/S	(no value)	When invoking a context menu command, tells the command handler to act as if the Shift key is held down. What difference this makes (if any) is up to the handler. If you use this then you'll probably also want to use <b>EXTENDEDVERBS</b> as holding down the shift key also normally causes the extended verbs to be shown, and the menu handler may expect both to be used together.  <i>Example:</i> <b>ContextMenu SHIFT EXTENDEDVERBS VERB delete</b>
SHOWCMDS	/S	(no value)	Displays a list of context menu commands for the selected files. Depending on the context menu handlers installed on your system, the list of commands you get back may differ based on the <b>EXTENDEDVERBS</b> and <b>ITEMMODE</b> arguments; if you use either of them when finding a command then you should do the same when running that command.  <i>Example:</i> <b>ContextMenu SHOWCMDS ITEMMODE</b>
VERB	default	<verb>	Specify the verb of the context menu command to execute. This is the default argument for the <b>ContextMenu</b> command, so the keyword <b>VERB</b> does not need to be specified.  <i>Example:</i> <b>ContextMenu PreviousVersions</b>  <i>Example:</i> <b>ContextMenu ITEMMODE VERB SevenZipCompressEmail</b>
WANTSYNC	/S	(no value)	Requests that the context menu command be run synchronously. In other words, any other commands should wait until the context menu command has completed, instead of running in parallel. This is just a request and the context menu handler may ignore it. Similarly, context

			<p>menu handlers may choose to run things synchronously even when this argument is not specified.</p> <p><i>Example:</i> <b>ContextMenu WANTSYNC ITEMMODE LOOKUP VERB DropHardLinkClone FILE {sourcepath\$}</b></p>
WILD	/S	(no value)	<p>Used in conjunction with the <b>LABEL</b> argument to specify that the label string is a wildcard expression.</p> <p><i>Example:</i> <b>ContextMenu ITEMMODE WILD LABEL "Add to *.zip*"</b></p>

## Copy

The **Copy** internal command can be used to:

- Copy and move files and folders
- Create and add files to archives
- Extract files from archives
- Perform simple one-way file synchronisation
- Create links, shortcuts and junctions
- Install fonts
- Send files via email
- Add items to [file collections](#), and include folders in [libraries](#)

## Command Arguments:

Argument	Type	Possible values	Description
ADDTOARCHIVE	/O	(no value)	<p>Displays the <a href="#">Add to Archive</a> dialog, to create an archive from the selected files. The dialog will default to creating a Zip file.</p> <p><i>Example:</i> <b>Copy ADDTOARCHIVE</b></p>

			<p>The <b>CREATEFOLDER</b> argument can be used to change the default archive name.</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE CREATEFOLDER="Backup"</b></p>
		<i>&lt;archive format&gt;</i>	<p>Displays the <a href="#">Add to Archive</a> dialog with the archive type set to the specified format (given as the file extension of the desired format).</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE=.rar</b></p> <p>The archive format can be optionally followed by parameters that are specific to the selected format. Currently the only format that defines optional parameters is Zip, those are listed below.</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE=.zip,fullpaths</b></p> <p>You can also use this command to create a self-extracting archive from the selected files, with the following arguments:</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE=.zip+sfx</b></p>
		<b>fullpaths</b>	<p>When creating Zip archives, this turns on the <i>Save full file paths</i> option by default.</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE=.zip,fullpaths</b></p>
		<b>nofullpaths</b>	<p>When creating Zip archives, this disables the <i>Save full file paths</i> option.</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE=.zip,nofullpaths</b></p>
		<b>split:</b> <i>&lt;size &gt;</i>	<p>When creating Zip archives, this sets the default value for the <i>Split archive</i> option.</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE=.zip,fullpaths,split:2.5 MB</b></p>
		<b>nosplit</b>	<p>When creating Zip archives, this disables the <i>Split archive</i> option.</p> <p><i>Example: Copy</i> <b>ADDTOARCHIVE=.zip,nosplit,nofullpaths</b></p>
ARCHIVE	/O	<i>(no value)</i>	<p>Adds all selected files and folders to a Zip file named after the first selected item. Note that if</p>

			<p>only a single folder is selected, the items <i>within</i> that folder are added rather than the folder itself - this prevents you from ending up with a Zip file containing an unnecessary sub-folder.</p> <p><i>Example: Copy ARCHIVE</i></p> <p>The <b>CREATEFOLDER</b> argument can be used to change the default archive name.</p> <p><i>Example: Copy ARCHIVE CREATEFOLDER="Backup"</i></p>
		<archive format>	<p>Adds all selected files and folders to an archive of the specified format.</p> <p><i>Example: Copy ARCHIVE=.rar</i></p>
		<b>single</b>	<p>Each selected item will be added to its own archive, rather than all items being added to the one archive.</p> <p><i>Example: Copy ARCHIVE=single</i></p>
		<b>keepfolder</b>	<p>Modifies the behaviour when only a single folder is selected - instead of the items within the folder being added, the folder itself will be added to the archive.</p> <p><i>Example: Copy ARCHIVE=keepfolder</i></p>
AS	/O	(no value)	<p>When copying or moving files, you will be prompted to enter a new name for each file.</p> <p><i>Example: Copy MOVE AS</i></p>
		<new name>	<p>Specifies the new name or <a href="#">wildcard pattern</a> for the copied or moved files. This only changes the names; if you want to specify the destination path, use the <b>TO</b> argument.</p> <p><i>Example: Copy AS *.bak HERE</i></p>
AUTOSELECT	/K	<b>yes</b>	<p>Overrides the state of the <i>Automatically select newly copied files</i> option on the <a href="#">File Operations / Copy Options</a> Preferences page. Newly copied files will always be selected.</p> <p><i>Example: Copy DUPLICATE AUTOSELECT=yes</i></p>

		<b>no</b>	Newly copied files will not be selected, no matter what the <i>Automatically select newly copied files</i> option is set to.  <i>Example: Copy <b>AUTOSELECT=no</b></i>
BUFSIZE	/K/ N	<size in bytes>	Overrides the <b>copy_buffer_size</b> setting on the <a href="#">Miscellaneous / Advanced</a> page in Preferences. The buffer size is specified in bytes, if no units are specified, and you can also use KB, MB and GB to specify larger sizes.  This buffer is in addition to any buffering provided by the filesystem, hardware, and so on; it is not connected to the non-buffered IO mode controlled by the <b>NONBUFIO</b> argument and <b>copy_nonbufferio_threshold</b> Preferences option.  <i>Example: Copy <b>BUFSIZE 128KB</b></i>
BURNCD	/S	(no value)	Invokes the system CD Burning Wizard, which initiates burning of any files you have previously copied to the CD staging area.  <i>Example: Copy <b>BURNCD</b></i>
CLEARREADONLY	/K	<b>yes</b>	Clear the read-only attribute. When copying files from a CD this overrides the <b>Clear read-only flag when copying from CDs</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page.  <i>Example: Copy <b>CLEARREADONLY=yes</b></i>
		<b>no</b>	Do not clear the read-only flag when copying files.  <i>Example: Copy <b>CLEARREADONLY=no</b></i>
COLLIST	/S	(no value)	Displays a dynamically generated list of <a href="#">file collections</a> - you can add selected files and folders to a file collection simply by selecting it from this list. Acts as a <a href="#">dynamic button</a> . Most useful when used in a drop-down menu or on a <a href="#">context menu</a> .  <i>Example: Copy <b>COLLIST</b></i>
COPYATTR	/K	<b>yes</b>	Preserves file attributes when copying files or folders (overrides the <b>Preserve the attributes of copied files</b> option on the <a href="#">File Operations /</a>

			<p><a href="#">Copy Attributes</a> Preferences page).</p> <p><i>Example: Copy COPYATTR=yes</i></p>
		<b>no</b>	<p>Do not preserve file attributes (the newly copied file will have the default file attributes for a newly created file).</p> <p><i>Example: Copy COPYATTR=no</i></p>
COPYCREATIONTIME	/K	<b>yes</b>	<p>Preserve the creation time of copied files and folders.</p> <p>Overrides the <b>no_copy_creation_time</b> option on the <a href="#">Miscellaneous / Advanced</a> Preferences page.</p> <p><b>COPYCREATIONTIME=yes</b> will cause file and folder creation times to be preserved even if <b>COPYFILETIMES=no</b> or <b>COPYDIRTIMES=no</b> are used.</p> <p>Modified and accessed times are not affected by this argument.</p> <p><i>Example: Copy COPYCREATIONTIME=yes</i></p>
		<b>no</b>	<p>Do not preserve the creation time of copied files and folders. (The current time will be used instead.)</p> <p><b>COPYCREATIONTIME=no</b> will prevent preservation of file and folder creation times even if <b>COPYFILETIMES=yes</b> or <b>COPYDIRTIMES=yes</b> are used.</p> <p>Modified and accessed times are not affected by this argument.</p> <p><i>Example: Copy COPYCREATIONTIME=no</i></p>
COPYDESC	/K	<b>yes</b>	<p>Preserves file descriptions when copying files or folders (overrides the <b>Preserve the descriptions of copied files option</b> on the <a href="#">File Operations / Copy Attributes</a> Preferences page).</p> <p><i>Example: Copy COPYDESC=yes</i></p>
		<b>no</b>	<p>Do not preserve file descriptions.</p> <p><i>Example: Copy COPYDESC=no</i></p>

COPYDIRTIMES	/K	yes	<p>Preserves the created, modified and accessed times of copied folders.</p> <p>Overrides the <b>Preserve the timestamps of copied files</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page and the <b>no_copy_dir_dates</b> option on the <a href="#">Miscellaneous / Advanced</a> Preferences page.</p> <p>You can override the <b>COPYDIRTIMES</b> argument itself, for just the created timestamps, using the <b>COPYCREATIONTIME</b> argument.</p> <p><i>Example: Copy COPYDIRTIMES=yes</i></p>
		no	<p>Prevents preservation of the created, modified and accessed times of copied folders.</p> <p><i>Example: Copy COPYDIRTIMES=no</i></p>
COPYFILETIMES	/K	yes	<p>Preserves the created, modified and accessed times of copied files.</p> <p>Overrides the <b>Preserve the timestamps of copied files</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page.</p> <p>You can override the <b>COPYFILETIMES</b> argument itself, for just the created timestamps, using the <b>COPYCREATIONTIME</b> argument.</p> <p><i>Example: Copy COPYFILETIMES=yes COPYCREATIONTIME=no</i></p>
		no	<p>Prevents preservation of the created, modified and accessed times of copied files.</p> <p><i>Example: Copy COPYFILETIMES=no</i></p>
COPYOWNER	/K	local	<p>Copy file owner information when the copy takes place between local drives only (overrides the <b>Copy owner</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page). Note that under Vista and above, setting the file owner requires elevation and so may produce a UAC prompt.</p> <p><i>Example: Copy COPYOWNER=local</i></p>
		all	<p>Copy file owner information for all drives, not just local ones.</p>

			<i>Example: Copy COPYOWNER=all</i>
		<b>no</b>	Do not copy file owner information.  <i>Example: Copy COPYOWNER=no</i>
COPYPROPERTIES	/K	<b>yes</b>	Copy NTFS properties/metadata when copying files and folders (overrides the <b>Copy metadata</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page).  <i>Example: Copy COPYPROPERTIES=yes</i>
		<b>no</b>	Do not copy metadata when copying files.  <i>Example: Copy COPYPROPERTIES=no</i>
		<b>all</b>	Copy all NTFS data streams, not just the metadata ones.  <i>Example: Copy COPYPROPERTIES=all</i>
COPYSECURITY	/K	<b>yes</b>	Copy security permissions when copying files between NTFS drives (overrides the <b>Copy security permissions</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page).  <i>Example: Copy COPYSECURITY=yes</i>
		<b>no</b>	Do not copy security permissions. Newly copied files will inherit the default security permissions of the destination folder.  <i>Example: Copy COPYSECURITY=no</i>
COPYSPARSE	/K	<b>yes</b>	Recreate sparse regions in the copied file (when the source file is sparse). Overrides the <b>Copy sparse files as sparse</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page).  <i>Example: Copy COPYSPARSE=yes</i>
		<b>no</b>	Do not recreate sparse regions in the copied file. Newly copied files will occupy their full size on disk.  <i>Example: Copy COPYSPARSE=no</i>
COPYTOCOLL	/K	<b>member</b>	When copying (adding) folders to a <a href="#">file collection</a> , add them as members of the collection (overrides the <b>When copying folders to a Collection</b> option on the <a href="#">File Operations / Copy Options</a> Preferences page).

			<i>Example: Copy COPYTOCOLL=member</i>
		<b>sub</b>	Copy folders to collections as sub-collections (the contents of the folder will be added as member items to the newly created sub-collection).  <i>Example: Copy COPYTOCOLL=sub</i>
		<b>ask</b>	Ask how to copy folders to collections.  <i>Example: Copy COPYTOCOLL=ask</i>
CREATEFOLDER	/O	(no value)	Prompts for the name of a new folder and copies the selected files and folders into that folder.  <i>Example: Copy CREATEFOLDER</i>
		<folder name>	Creates a new folder with the specified name and copies selected items into that folder. You can specify an absolute path, or just a name - if only a name is provided the folder is created in the destination (or source if the <b>HERE</b> argument is also specified). You can also use the <a href="#">external control codes</a> to (for example) automatically create a folder based on the current date.  <i>Example: Copy CREATEFOLDER "Backup-{"date yyyyMMdd}"</i>  When archiving files, this specifies the name of the archive to create or update. (If you don't specify an archive name it will default to the name of the first selected file, without its extension, or the name of the current folder if no file selection is used.). The example below adds the selected files to an archive called "Cat Photos.zip" below the current folder:  <i>Example: Copy HERE ARCHIVE=.zip CREATEFOLDER="Cat Photos"</i>
DUPLICATE	/S	(no value)	Create duplicates of the selected items in the same folder. You will be prompted to enter new names (or a <a href="#">wildcard pattern</a> ) for the duplicated files.  <i>Example: Copy DUPLICATE</i>
EXTRACT	/O	(no value)	Extracts the contents of selected archives to the destination folder. You can also use this with folders to copy their contents without

			<p>copying the folder itself.</p> <p><i>Example: Copy EXTRACT</i></p>
		<b>sub</b>	<p>Creates a sub-folder in the destination named after the archive, and extracts the archive contents to that folder.</p> <p><i>Example: Copy EXTRACT=sub</i></p>
		<b>checkout</b>	<p>Extracts the contents of an archive to a temporary folder and automatically opens that folder in a new Lister. You can use this to "check out" the files in an archive before you decide if and where you want to extract them to. Note that this command only works when you are inside the archive itself (e.g. double-click a .zip file to enter it, and then run the "checkout" command to extract it).</p> <p><i>Example: Copy EXTRACT=checkout</i></p>
FILE	/M	<filename> > ...	<p>Specifies the name of the file or files to copy. If you don't provide this argument the command operates on all selected items in the source Lister. This is the default argument for the <b>Copy</b> command - you don't need to specify the <b>FILE</b> keyword.</p> <p>If you only specify the filename instead of the full path of the file or files, Opus will look in the current source folder. You can also specify a <a href="#">wildcard pattern</a>. Remember that if the filename contains spaces you need to enclose it in quotes.</p> <p><i>Example: Copy "C:\Data Directory\*.xls" TO /desktop</i></p>
FILTER	/O	(no value)	<p>Copies with filtering enabled (without having to activate the <a href="#">copy filter</a> in the Lister first). Opus will prompt you to define the filter.</p> <p><i>Example: Copy FILTER</i></p>
		<filter>	<p>Copies using the specified filter. This must have previously been created from the <a href="#">File Operations / Filters</a> page in Preferences. You can also directly specify a <a href="#">simple wildcard pattern</a></p> <p><i>Example: Copy FILTER *.jpg png</i></p>
		<b>shift</b>	<p>Copies with filtering enabled if the <b>Shift</b> key is held down. Opus will prompt you to define</p>

			the filter.  <i>Example: Copy FILTER=shift</i>
		<b>alt</b>	Copies with filtering enabled if the <b>Alt</b> key is held down.  <i>Example: Copy FILTER=alt</i>
		<b>ctrl</b>	Copies with filtering enabled if the <b>Ctrl</b> key is held down.  <i>Example: Copy FILTER=ctrl</i>
FLATVIEWCOPY	/K	<b>single</b>	When copying items in different directories from a <a href="#">flat view</a> file display, the files will all be copied to the same target directory. This overrides the <b>When copying in Flat View mode</b> option on the <a href="#">File Operations / Copy Options</a> Preferences page.  <i>Example: Copy FLATVIEWCOPY=single</i>
		<b>recreate</b>	Recreates the source folder structure when copying items in different directories out of a <a href="#">flat view</a> file display.  <i>Example: Copy FLATVIEWCOPY=recreate</i>
		<b>ask</b>	Opus will ask you what to do when copying items out of a flat view file display.  <i>Example: Copy FLATVIEWCOPY=ask</i>
FORCE	/S	(no value)	Automatically replace existing files without prompting. In-use files that cannot be replaced will be automatically scheduled for replacement next reboot.  <i>Example: Copy FORCE</i>
HERE	/S	(no value)	Use the source folder as the destination folder (for example, archives can be extracted to the same folder instead of the destination).  <i>Example: Copy EXTRACT HERE</i>
IGNOREEXT	/S	(no value)	Makes the copy function ignore file extensions when copying with a wildcard rename (for example, so a button can work on both files and folders using the same wildcard pattern).

			<i>Example: Copy * AS *.bak IGNOREEXT</i>
INCLUDEINLIBRARY	/O	(no value)	<p>Displays a dynamically generated list of <a href="#">libraries</a> - you can include a selected folder in a library simply by selecting it from this list. Acts as a <a href="#">dynamic button</a>. Most useful when used in a drop-down menu or on a <a href="#">context menu</a>.</p> <p>Opus will automatically navigate to show the contents of the library with the newly included folder.</p> <p><i>Example: Copy INCLUDEINLIBRARY</i></p>
		<b>noread</b>	<p>Prevents Opus from automatically navigating to the library when you include a folder in it via the generated list.</p> <p><i>Example: Copy INCLUDEINLIBRARY noread</i></p>
		<b>\$new</b>	<p>Include selected folders in a new library. The new library will be given the name of the first selected folder. Opus will automatically navigate to the new library unless you include the <b>noread:</b> prefix.</p> <p><i>Example: Copy INCLUDEINLIBRARY noread:\$new</i></p>
		<library name>	<p>Include selected folders in the named library. Opus will automatically navigate to the new library unless you include the <b>noread:</b> prefix.</p> <p><i>Example: Copy INCLUDEINLIBRARY "noread:Movie Files"</i></p>
INSTALLFONT	/S	(no value)	<p>Installs new fonts in your system fonts folder. You do not need to specify the destination folder when using this command - the fonts will be copied to your fonts folder and registered automatically. This command has no effect if non-font files are selected.</p> <p><i>Example: Copy INSTALLFONT</i></p>
MAKELINK	/O	(no value)	<p>Creates shortcuts to all selected files and folders. Shortcuts do not require NTFS. Shortcuts may point to things on different drives to themselves.</p> <p><i>Example: Copy MAKELINK TO /desktop</i></p>
		<b>junction</b>	<p>Creates junctions to all selected folders. Junctions are only supported on NTFS drives.</p>

			<p>Junctions only work with folders (not with files). Junctions may point to folders on different drives to themselves.</p> <p><i>Example: Copy MAKELINK=junction</i></p>
		<b>hardlink</b>	<p>Creates hardlinks to all selected files. Hardlinks are only supported on NTFS drives. Hardlinks only work with files (not folders). Hardlinks <i>cannot</i> point to files on different drives to themselves.</p> <p><i>Example: Copy MAKELINK=hardlink</i></p>
		<b>softlink</b>	<p>Creates softlinks to all selected files or folders, using absolute paths. Softlinks require Windows Vista or above and are only supported on NTFS drives. Softlinks work with both files and folders. Softlinks may point to things on different drives to themselves. Creating softlinks requires administrator access and will trigger a UAC prompt if necessary.</p> <p><i>Example: Copy MAKELINK=softlink</i></p>
		<b>relsoftlink</b>	<p>Creates softlinks to any selected files or folders, using relative paths where possible. A regular absolute link will be created if the target can not be expressed relative to the link. (See <b>softlink</b> for more information on softlinks.)</p> <p><i>Example: Copy MAKELINK=relsoftlink</i></p>
		<b>auto</b>	<p>Automatically determines the most suitable type of link to create. On Vista and above, it will usually create softlinks (for both files and folders). On Windows XP, it will usually create junctions (for folders) and hardlinks (for files). Shortcuts will be created instead in cases where the desired link type is not applicable. For example, a shortcut will be created if the drives are not NTFS or if a hardlink is desired but the source and destination are on different drives.</p> <p><i>Example: Copy MAKELINK=auto</i></p>
		<b>autosoft</b>	<p>Does the same as <b>auto</b> except that it will not try to create softlinks. It will usually create junctions (for folders) and hardlinks (for files). Shortcuts will be created instead in cases where the desired link type is not</p>

			<p>applicable. For example, a shortcut will be created if the drives are not NTFS or for files where the source and destination are on different drives.</p> <p><i>Example: Copy MAKELINK=autonosoft</i></p>
MAKESFX	/O	(no value)	<p>Creates a <a href="#">self-extracting Zip file</a> from selected files and folders. If you select a .zip file then it will be converted directly to self-extracting format; otherwise, the selected items will be zipped first.</p> <p><i>Example: Copy MAKESFX</i></p>
MARKDESTARCHIVE	/K	yes	<p>Set the <b>A</b> attribute on newly copied files. You can use this if have a backup solution that uses the <b>A</b> attribute as an indication that a file has been backed up (overrides the <b>Mark copied files as archived</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page).</p> <p><i>Example: Copy MARKDESTARCHIVE=yes</i></p>
		no	<p>Do not set the <b>A</b> attribute on newly copied files.</p> <p><i>Example: Copy MARKDESTARCHIVE=no</i></p>
MARKSOURCEARCHIVE	/K	yes	<p>Set the <b>A</b> attribute on the original files after they have been copied (overrides the <b>Mark original files as archived after being copied</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page).</p> <p><i>Example: Copy MARKSOURCEARCHIVE=yes</i></p>
		no	<p>Do not set the <b>A</b> attribute on the original files after they are copied.</p> <p><i>Example: Copy MARKSOURCEARCHIVE=no</i></p>
MOVE	/S	(no value)	<p>Move selected files and folders to the destination (a.k.a. cut-and-paste). If the destination folder is on the same drive as the source then generally items can be moved through a simple rename operation, which is very quick. When moving files between devices Opus first copies them to the destination, and then deletes them from the</p>

			source.  <i>Example: Copy MOVE</i>
MOVEWHENSAME	/S	(no value)	If the destination folder is on the same drive as the source then selected items will be moved, otherwise they will be copied. This command is used in the default <a href="#">drag-and-drop file type event</a> (which mimics the standard Explorer drag-and-drop behaviour where files are moved if you drag them to a different folder on the same drive, and copied otherwise).  <i>Example: Copy MOVEWHENSAME</i>
MOVEWITHSHIFT	/S	(no value)	Selected items will be moved if the <b>Shift</b> key is held down when the command is executed, otherwise they will be copied.  <i>Example: Copy MOVEWITHSHIFT</i>
NONBUFIO	/K	<b>yes</b>	Changes when the copy operation uses non-buffered mode, where the filesystem buffers provided by Windows are bypassed.  For very large files, copying in non-buffered mode can increase the memory efficiency, copy speed and UI responsiveness. On the other hand, non-buffered mode may slow things down for smaller files or certain devices. In rare cases, non-buffered mode may not work at all (e.g. if you have a device which mis-reports its sector size).  Use of this argument overrides the default file size threshold for non-buffered copies set via the <b>copy_nonbufferio_threshold</b> <a href="#">Advanced</a> Preferences setting.  You can specify "yes" or "no" to force non-buffered mode on or off for all files, or specify the file size above which non-buffered mode should be used.  <i>Example: Copy NONBUFIO=yes</i>
		<b>no</b>	Forces the copy operation to be buffered, even if the file being copied exceeds the threshold size set via the <b>copy_nonbuffer_threshold</b> <a href="#">Advanced</a> Preferences setting.  <i>Example: Copy NONBUFIO=no</i>

		<i>&lt;threshold size&gt;</i>	<p>The copy operation will be non-buffered if the file size exceeds the specified size, and buffered otherwise. When specifying a size, units can be KB, MB or GB. If no units are specified, MB is used by default.</p> <p><i>Example: Copy NONBUFIO=64MB</i></p>
NOQUEUEWHENSAME	/S	<i>(no value)</i>	<p>Disables the use of the copy queue when the source and destination paths are on the same drive partition. You would normally only use this argument when moving files, because moves on the same drive can be done without actually copying any data.</p> <p><i>Example: Copy MOVE NOQUEUEWHENSAME</i></p>
PATTERN	/K	<i>&lt;old name pattern&gt;</i>	<p>Specifies the "old name" or "from" <a href="#">wildcard pattern</a> for the copied or moved files. Use this in conjunction with <b>AS</b> to control wildcard renaming of copied or moved files.</p> <p><i>Example: Copy DUPLICATE PATTERN *.* AS *_{date yyyy-MM-dd}.*</i></p>
QUEUE	/O	<i>(no value)</i>	<p>Enables automatic <a href="#">copy queuing</a>. File copies will be queued automatically if required (based on the source and destination drives). This can override the <b>Automatically manage file copy queues</b> option on the <a href="#">File Operations / Copy Options</a> Preferences page.</p> <p><i>Example: Copy QUEUE</i></p>
		<i>&lt;queue name&gt;</i>	<p>When you specify a queue name as the value for this argument, it enables manual copy queuing when copying files. That is, with a name specified, file copies will always be queued to the specified queue - if no name is specified for the argument, copies will only be queued if needed. The specified name will be shown in the progress dialog's title bar.</p> <p><i>Example: Copy QUEUE=MyQueue</i></p>
		<b>none</b>	<p>Used to disable copy queuing - whether enabled in Preferences, or otherwise enabled by the <b>shift</b> keyword.</p> <p><i>Example: Copy QUEUE=none</i></p>
		<b>shift</b>	<p>Lets you specify two alternate parameters for the <b>QUEUE</b> argument. The value specified before the <b>shift</b> keyword is used if the <b>Shift</b></p>

			<p>key is not held down - the value after it is used if it is. For example, you could configure a copy button to queue files to a specific queue if the <b>Shift</b> key were held down, and to disable queuing otherwise.</p> <p><i>Example: Copy</i>  <b>QUEUE=none,shift,MyQueue TO</b>  <b>\\NAS1\Storage</b></p>
		<b>quiet</b>	<p>Specify the <b>quiet</b> keyword to suppress the prompt that normally indicates a copy operation has been queued.</p> <p><i>Example: Copy QUEUE=MyQueue,quiet</i></p>
RENAMEWHENSAME	/S	(no value)	<p>If the source and destination are the same folder, the newly copied file will be automatically renamed to avoid a clash.</p> <p><i>Example: Copy RENAMEWHENSAME</i></p>
SENDMAIL	/O	(no value)	<p>Send selected files as email attachments. Email settings must be configured on the <a href="#">Internet / Email</a> Preferences page.</p> <p><i>Example: Copy SENDMAIL</i></p>
		<email address>	<p>Send selected files to the specified email recipient. This only works if email sending is set to use the <b>MAPI client</b> on the <a href="#">Internet / Email</a> Preferences page.</p> <p><i>Example: Copy</i>  <b>SENDMAIL=f.bloggs@company.com</b></p>
SENDTO	/K	<send-to target>	<p>Send selected files to the specified "send to" target. This can be any item that appears in the system <i>Send To</i> context menu, and lets you perform the same action without actually displaying the context menu. The value given for the target must be the name of the actual file in the <i>SendTo</i> folder (to find the <i>SendTo</i> folder and see what's in there, use the <a href="#">/sendto folder alias</a>).</p> <p><i>Example: Copy SENDTO "Web Publishing Wizard"</i></p>
TO	/K	<target path>	<p>Specify the target path for the command. By default Copy functions that require a destination folder will use the current destination file display or Lister - this argument allows you to override that. Also see the <b>HERE</b> argument for a way to override the</p>

			<p>destination path. You can use <a href="#">folder aliases</a> and <a href="#">@ftp shortcuts</a>, URL-style paths for <a href="#">virtual filesystems</a> (<a href="#">collections</a>, <a href="#">libraries</a>, etc.) as well as standard file system paths. Remember that if the specified path contains a space you must enclose the whole path in quotes. This only sets the destination folder; if you want to change the names of the copied files as well, use the <b>AS</b> argument.</p> <p><i>Example: Copy TO "lib://Backups/Daily Backup Folder"</i></p>
		<b>ask</b>	<p>Normally if no destination path is specified, and there is no current destination file display or Lister, Opus will prompt for a destination path via a popup dialog. You can use the <b>ask</b> value to force Opus to always prompt for a destination path, even if there already is one.</p> <p>You can also specify a default destination path, by appending a colon and the path. Remember to use quotes if the path contains spaces.</p> <p><i>Example: Copy TO=ask</i>  <i>Example: Copy "TO=ask:c:\My Documents"</i></p>
		<b>ask\$</b>	<p>Force Opus to ask for a destination path. If you have a function that combines multiple <b>Copy</b> commands, you can use <b>ask\$</b> to make Opus only prompt for a destination path once for the whole function, rather than prompting separately for each <b>Copy</b> command.</p> <p>You can also specify a default destination path, by appending a colon and the path.</p> <p><i>Example: Copy TO=ask\$</i>  <i>Example: Copy TO=ask\$:D:\</i></p>
UNATTENDED	/K	<b>yes</b>	<p>Enables <a href="#">unattended copy</a> mode. In this mode, Opus will not display any confirmation prompts or error dialogs - the copy will proceed until the end, and any errors will be</p>

			<p>summarised upon completion. Use the other arguments like <b>WHENEXISTS</b> to control what happens in certain situations.</p> <p><i>Example: Copy UNATTENDED=yes TO=@myftpsite WHENEXISTS=replace</i></p>
		<b>no</b>	<p>Disables unattended copy mode.</p> <p><i>Example: Copy UNATTENDED=no</i></p>
UPDATEALL	/S	(no value)	<p>Update files in the destination folder (<a href="#">a simple form of one-way synchronization</a>). Only files that either do not exist in the destination, or do exist but are different from the source files, will be copied - other files will be skipped. A file is defined as different if either its timestamp or size has changed - the contents of the file are not compared.</p> <p><i>Example: Copy UPDATEALL FORCE</i></p>
UPDATEEXISTING	/O	(no value)	<p>Update existing files in the destination folder (<a href="#">a simple form of one-way synchronization</a>). Only files that already exist in the destination, but are different from the source files, will be copied. Files that do not already exist, as well as files that have not changed, will be skipped. A file is defined as different if either its timestamp or size has changed.</p> <p><i>Example: Copy UPDATEEXISTING</i></p>
		<b>size</b>	<p>Update existing files whose size is different (ignore timestamp).</p> <p><i>Example: Copy UPDATEEXISTING=size</i></p>
		<b>date</b>	<p>Update existing files whose timestamp is different (ignore file size).</p> <p><i>Example: Copy UPDATEEXISTING=date</i></p>
UPDATESECURITY	/K	<b>yes</b>	<p>Update security permissions and encryption settings for moved files, to match the destination folder (overrides the <b>Update permissions/encryption to match the destination when moving files</b> option on the <a href="#">File Operations / Copy Attributes</a> Preferences page). For example, if a folder has the <b>E</b> attribute set, files moved into that folder will be automatically encrypted.</p> <p>This only applies if a file can be moved via a rename operation - for files that are moved to</p>

			<p>different devices, via a copy and delete operation, the newly copied files will inherit the permissions of the destination folder (this behaviour is controlled by the <b>COPYSECURITY</b> argument).</p> <p><i>Example: Copy <b>UPDATESECURITY=yes</b></i></p>
		<b>no</b>	<p>Do not update security permissions or encryption settings for moved files (when files are moved via a rename, their existing permissions will remain).</p> <p><i>Example: Copy <b>UPDATESECURITY=no</b></i></p>
UPDATETOLERANCE	/K/ N	<tolerance seconds>	<p>Sets the maximum number of seconds an existing file's timestamp can vary before it will be considered "different" by the <b>UPDATEALL</b> or <b>UPDATEEXISTING</b> functions. The default is two seconds (that is, a file will be considered different if its timestamp is two or more seconds different to the source file). You can use this to make allowances for daylight-savings time, or for file systems that don't preserve file dates to a high enough resolution.</p> <p><i>Example: Copy <b>UPDATEEXISTING UPDATETOLERANCE=3600</b></i></p>
WHENEXISTS	/K	<b>ask</b>	<p>If a file that is being copied already exists in the destination, ask what to do. This overrides the option <b>Ask for confirmation before overwriting existing files</b> on the <a href="#">File Operations / Copy Options</a> page in Preferences.</p> <p><i>Example: Copy <b>WHENEXISTS=ask</b></i></p>
		<b>skip</b>	<p>If a file that is being copied already exists in the destination, skip over it (leaving existing file intact).</p> <p><i>Example: Copy <b>WHENEXISTS=skip</b></i></p>
		<b>replace</b>	<p>Replace any existing files without prompting.</p> <p><i>Example: Copy <b>WHENEXISTS=replace</b></i></p>
		<b>rename</b>	<p>If a file with the same name already exists in the destination, the newly copied file will be renamed automatically to avoid a clash.</p> <p><i>Example: Copy <b>WHENEXISTS=rename</b></i></p>

	<b>renameold</b>	<p>The old file (the one that already exists in the destination directory) will be renamed before the new file is copied.</p> <p><i>Example: Copy WHENEXISTS=renameold</i></p>
	<b>resume</b>	<p>When copying to FTP sites, a previous incomplete file transfer will be automatically resumed (only applies if the server supports FTP, and the existing file is smaller than the one being copied).</p> <p><i>Example: Copy WHENEXISTS=resume</i></p>
	<b>merge</b>	<p>Automatically merge the contents of an existing folder with that of the folder being copied. This overrides the <b>Ask for confirmation before merging existing folders</b> option on the <a href="#">File Operations / Copy Options</a> page in Preferences. You can use this value in addition to another value to specify the 'when exists' behaviour for files as well as folders.</p> <p><i>Example: Copy WHENEXISTS=replace,merge</i></p>
	<b>keepnewer</b>	<p>Replace any existing files if the files being copied are newer. If the timestamps are the same or older, the already-existing files will be skipped.</p> <p><i>Example: Copy WHENEXISTS=keepnewer</i></p> <p>(This was called <b>replacnewer</b> in the past. It has been renamed for clarity. The old name still works to maintain compatibility.)</p>

## CreateFolder

The **CreateFolder** internal command is used to:

- [Create new folders](#) (either via a dialog, an inline rename field in the file display, or with a pre-determined name)
- [Create new archive files](#)
- Create new [libraries](#), [collections](#) and [stored query collections](#)

## Command Arguments:

Argument	Type	Possible values	Description
ARCHIVE	/O	(no value)	Create a new archive file. The archive format will default to .zip but can be changed via the dialog.  <i>Example: CreateFolder ARCHIVE</i>
		<archive format>	Create a new archive file of the specified format.  <i>Example: CreateFolder ARCHIVE=.7z</i>
ARCHIVEARGS	/K	<archive arguments>	Used to pass format-specific arguments when creating an archive. The format these arguments take is defined by the plugin and archive type in question - currently, none of the standard archive formats Opus supports make use of this.  <i>Example: CreateFolder ARCHIVE=.moo ARCHIVEARGS="/compression=1"</i>
ASK	/S	(no value)	If you specify a name with the <b>NAME</b> argument the <i>Create Folder</i> dialog does not normally display; instead the folder is created immediately. You can add the <b>ASK</b> argument to force the dialog to display, pre-populated with the specified name.  <i>Example: CreateFolder "My New Folder" ASK</i>
COLLQUERY	/S	(no value)	Creates a stored query collection rather than a regular file collection. This only has effect when actually creating a collection (because, for example, you are in the File Collections root folder at the time).  <i>Example: CreateFolder COLLQUERY</i>
FROMCLIPBOARD	/O	(no value)	Uses the text currently on the clipboard for the name of the new folder. Equivalent to <b>CreateFolder "{clip}"</b> .  <i>Example: CreateFolder FROMCLIPBOARD</i>
		<b>multi</b>	Creates multiple folders from the clipboard contents, assuming the clipboard contains multiple lines of CR/LF separated text. Without this keyword only the first line would be used.  <i>Example: CreateFolder FROMCLIPBOARD=multi</i>

INLINE	/S	(no value)	<p>Creates a new folder and lets you edit its name inline (in the file display) - equivalent to selecting <i>New -&gt; Folder</i> from the context or File menu.</p> <p><i>Example: CreateFolder</i> <b>INLINE</b></p>
NAME	/M	<folder name> ...	<p>Specify the name of the folder or archive to create. If this argument is supplied Opus will not prompt for a name (unless <b>ASK</b> is also specified). You can specify a full path, or just the folder name to create the folder in the current source directory. You can also specify multiple names to create more than one folder at once.</p> <p>This is the default argument for the <b>CreateFolder</b> command, so the keyword <b>NAME</b> does not need to be specified. However, if you need to create a folder whose name is one of the other argument names, you must use the more explicit <i>NAME="xyz"</i> form to avoid the <i>xyz</i> part being interpreted as an argument name instead of a folder name. If you need this while creating multiple folders, you can specify the <b>NAME</b> argument multiple times.</p> <p><i>Example: CreateFolder</i> <b>"Folder 1" "Folder 2" "Folder 3"</b></p> <p><i>Example: CreateFolder</i> <b>"C:\Program Files\My New Program"</b></p> <p><i>Example: CreateFolder</i> <b>NAME="Apple" NAME="Ask" NAME="Inline" ASK</b></p>
NOSEL	/S	(no value)	<p>Normally when a folder is created in the currently displayed directory it is selected, and the display scrolled if necessary to make it visible. The <b>NOSEL</b> argument prevents this.</p> <p><i>Example: CreateFolder</i> <b>"New Folder" NOSEL</b></p>
NOUPDATESSETTINGS	/S	(no value)	<p>Prevents settings made by this command from becoming the new defaults. For example, if you specify the <b>READAUTO</b> argument, the value you provide will become the new default setting for the <b>CreateFolder</b> command unless you also specify <b>NOUPDATESSETTINGS</b>.</p>

			<i>Example: CreateFolder "{date yyyyMMdd}"</i> <b>READAUTO NOUPDATESSETTINGS</b>
MULTI	/O	(no value)	Ensures the <i>Create Folder</i> is in multiple-folder mode if it opens.  If <b>MULTI</b> is not specified at all, the dialog remembers its previous mode (subject to <b>NOUPDATESSETTINGS</b> ) when no folder names are specified on the command line, uses single-folder mode when one name is specified, and uses multi-folder mode when multiple names are specified.  <i>Example: CreateFolder "Backup on {date yyyy_MM_dd}" ASK MULTI</i>
		<b>no</b>	Ensures the <i>Create Folder</i> dialog is in single-folder mode if it opens.  <i>Example: CreateFolder MULTI=no</i>
READAUTO	/O	(no value)	Automatically read the newly created folder into the current source file display.  <i>Example: CreateFolder "New Folder" READAUTO</i>
		<b>no</b>	Do not automatically read the newly created folder.  <i>Example: CreateFolder "New Folder" READAUTO=no</i>
		<b>dual</b>	Read the newly created folder into the other file display. If the Lister is not currently in <a href="#">dual-display</a> mode it will be placed in this mode first. You can combine this value with <b>tab</b> to open a new tab in the other file display.  <i>Example: CreateFolder "New Folder" READAUTO=dual</i>
		<b>tab</b>	Create a new tab for the newly created folder. The tab will be opened in the source file display unless <b>dual</b> is also specified.  <i>Example: CreateFolder "New Folder" READAUTO=dual,tab</i>
		<b>nofocus</b>	When opening a new tab, prevents that tab from being made the active one.  <i>Example: CreateFolder "New Folder" READAUTO=tab,nofocus</i>

ZIP	/S	(no value)	Create a new Zip file (equivalent to <b>CreateFolder ARCHIVE=.zip</b> ).  <i>Example: CreateFolder ZIP</i>
-----	----	------------	------------------------------------------------------------------------------------------------------------------

## Delete

The **Delete** internal command can be used to:

- Recycle files and folders (delete them to the recycle bin)
- Delete files and folders permanently (bypassing the recycle bin)
- [Securely erase](#) files so they can't ever be recovered
- Empty the recycle bin
- Remove items from a [file collection](#)

## Command Arguments:

Argument	Type	Possible values	Description
ALL	/S	(no value)	Delete all files and folders without prompting. This can be used to override the <b>Ask for confirmation for each file before deleting</b> and <b>Ask for confirmation for each folder before deleting</b> options on the <a href="#">File Operations / Delete Files</a> page in Preferences.  <i>Example: Delete ALL</i>
EMPTYRECYCLE	/O	(no value)	Empty the system recycle bin. You can combine with the <b>FORCE</b> argument to suppress the confirmation prompt, and the <b>QUIET</b> argument to suppress the progress dialog.  <i>Example: Delete EMPTYRECYCLE FORCE QUIET</i>
		<drive letter>	Empty the recycle bin for a specific drive.  <i>Example: Delete EMPTYRECYCLE=C:</i>
FAILNOTEMPTY	/S	(no value)	Fail when attempting to delete a non-empty folder (must be combined with the <b>NORECYCLE</b> argument).  <i>Example: Delete FAILNOTEMPTY NORECYCLE</i>

FILE	/M	<filename> ...	<p>Specifies the name of the file or files to delete. If you don't provide this argument the command will delete all selected items in the source Lister. This is the default argument for the <b>Delete</b> command - you don't need to specify the <b>FILE</b> keyword.</p> <p>If you only specify the filename instead of the full path of the file or files, Opus will look in the current source folder. You can also specify a <a href="#">wildcard pattern</a>. Remember that if the filename contains spaces you need to enclose it in quotes.</p> <p><i>Example: Delete *.tmp</i></p>
FILTER	/O	(no value)	<p>Delete with filtering enabled (without having to activate the <a href="#">delete filter</a> in the Lister first). Opus will prompt you to define the filter.</p> <p><i>Example: Delete FILTER</i></p>
		<filter>	<p>Delete using the specified filter. This must have previously been created from the <a href="#">File Operations / Filters</a> page in Preferences. You can also directly specify a <a href="#">simple wildcard pattern</a></p> <p><i>Example: Delete FILTER "temp files"</i></p>
		shift	<p>Delete with filtering enabled if the <b>Shift</b> key is held down. Opus will prompt you to define the filter.</p> <p><i>Example: Delete FILTER=shift</i></p>
		alt	<p>Delete with filtering enabled if the <b>Alt</b> key is held down.</p> <p><i>Example: Delete FILTER=alt</i></p>
		ctrl	<p>Delete with filtering enabled if the <b>Ctrl</b> key is held down.</p> <p><i>Example: Delete FILTER=ctrl</i></p>
FORCE	/S	(no value)	<p>Force the deletion of any files that are marked as read-only (have the <b>R</b> attribute set). This overrides the <b>Delete read-only files automatically (without asking)</b> option on the <a href="#">File Operations / Delete Files</a> page in Preferences.</p> <p><i>Example: Delete FORCE</i></p>

NOFROMFOCUS	/S	(no value)	<p>The default behaviour for the <b>Delete</b> command is to operate on either the source file display, or the Folder Tree, depending on which one has the input focus. This lets you use the same command to delete folders in the tree as well as items in the file display. Specify this argument to force the command to always operate on the source file display and ignore the folder tree.</p> <p><i>Example: Delete NOFROMFOCUS</i></p>
NORECYCLE	/S	(no value)	<p>Prevent the use of the recycle bin - files will be permanently deleted. This overrides the <b>Delete to Recycle Bin where possible</b> option on the <a href="#">File Operations / Delete Files</a> page in Preferences.</p> <p><i>Example: Delete NORECYCLE</i></p>
QUIET	/S	(no value)	<p>Prevent the display of any confirmation dialogs or error messages.</p> <p><i>Example: Delete FORCE QUIET</i></p>
RECYCLE	/S	(no value)	<p>Delete files by moving them to the recycle bin (if possible - not all drives support the recycle bin). Deletes to the recycle bin are not permanent - the files can be recovered until the bin is emptied. This overrides the <b>Delete to Recycle Bin where possible</b> option on the <a href="#">File Operations / Delete Files</a> page in Preferences.</p> <p><i>Example: Delete RECYCLE</i></p>
REMOVECOLLECTION	/O	(no value)	<p>Remove selected files and folders from the file collection. If used outside of a file collection this command will do nothing.</p> <p><i>Example: Delete REMOVECOLLECTION</i></p>
		<b>auto</b>	<p>When used on files and folders in a file collection, the items will be removed from the collection. When used outside of a file collection, the items will be deleted as normal.</p> <p><i>Example: Delete REMOVECOLLECTION=auto</i></p>
SECURE	/O	(no value)	<p>Perform a secure delete using the number of passes specified on the <a href="#">File Operations / Delete Files</a> page in Preferences.</p> <p><i>Example: Delete SECURE</i></p>

		<passes>	Perform a secure delete using the specified number of passes (from 1 to 32 depending on your level of paranoia :)  <i>Example: Delete SECURE=5</i>
SHIFT	/S	(no value)	Modifies the behaviour of the <b>Delete</b> command if the <b>Shift</b> key is held down. <ul style="list-style-type: none"> <li>• If the <b>Delete to Recycle Bin</b> option is selected on the <a href="#">File Operations / Delete Files</a> page in Preferences, holding the <b>Shift</b> key down will disable the use of the recycle bin.</li> <li>• If the <b>Delete to Recycle Bin</b> option is not selected, the command acts as if the <b>ALL</b> and <b>FORCE</b> arguments are specified.</li> </ul> <p>In either case, if the <b>Shift</b> key is not held down, the <b>SHIFT</b> argument has no effect on the normal operation of the command. This argument is used to emulate the behaviour of Explorer.</p> <i>Example: Delete SHIFT</i>
SKIPNOTEMPTY	/S	(no value)	Skip over without error when attempting to delete a non-empty folder (must be combined with <b>NORECYCLE</b> ).  <i>Example: Delete NORECYCLE SKIPNOTEMPTY</i>

## Favorites

The **Favorites** internal command can be used to:

- Display a dynamic list of your [Favorite folders](#) (either all Favorites or a specific sub-branch)
- Display a dynamic list of your [SmartFavorite folders](#)
- Add a folder to your Favorites
- Create a new [folder alias](#)
- Edit your Favorite folders (by opening Preferences to the [appropriate page](#))

## Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Displays a dynamically generated list of your <a href="#">favorite folders</a> - you can navigate to

			<p>a folder simply by selecting it from this list. Acts as a <a href="#">dynamic button</a>.</p> <p>Some of the arguments of this command can modify the appearance and behaviour of the dynamic list. Note the <b>PATH</b> argument lets you optionally specify a sub-branch of the Favorites list.</p> <p><i>Example: Favorites</i></p>
ADD	/O	(no value)	<p>Add the current source folder to your Favorites list. The entire path of the folder will be displayed in the Favorites list by default.</p> <p><i>Example: Favorites ADD</i></p>
		<b>nameonly</b>	<p>Add the current source folder to your Favorites list, with the name of the favorite entry set to the name of the folder (so that only the name shows in the Favorites list, not the entire path).</p> <p><i>Example: Favorites ADD=nameonly</i></p>
		<b>alias</b>	<p>Add or modify a folder alias instead of a favorite. You may use the <b>NAME</b> argument to specify the name of the alias, or omit it to automatically use the folder's name. You may use the <b>PATH</b> argument to specify the path the alias should point to, or omit it to automatically use the current path.</p> <p>There are two alternative syntaxes for adding an alias. Using <b>ADD=alias</b> is the same as using <b>ALIAS=set</b>, other than the behavior of the <b>PATH</b> and <b>NAME</b> arguments as noted.</p> <p><i>Example: Favorites ADD=alias PATH "C:\Users\Jon\Documents"</i></p>
ADDIALOG	/S	(no value)	<p>Display a dialog that lets you edit the name and path when adding a new favorite folder.</p> <p><i>Example: Favorites ADDIALOG</i></p>
ALIAS	/K	<b>delete</b>	<p>Deletes a folder alias. The alias name must be given by the <b>NAME</b> argument,</p>

			and is not optional.  <i>Example: Favorites ALIAS=delete NAME=MyAlias</i>
		<b>set</b>	Creates or modifies a folder alias. The alias name must be given by the <b>NAME</b> argument, and the path must be given by the <b>PATH</b> argument, with neither being optional.  <i>Example: Favorites ALIAS=set NAME=MyAlias PATH "C:\Users\Jon\Documents"</i>
BRANCH	/K	<branch path>	Specifies the branch of the favorites tree to add a new favorite to, when used with the <b>ADD</b> or <b>ADDIALOG</b> arguments. To add to the root of the favorites tree, omit the <b>BRANCH</b> argument entirely. Specify nested branches with a \ between each component, similar to a folder path. If you specify a branch where some or all parts do not exist, the missing parts will be created. When used with <b>ADDIALOG</b> , the specified path will be selected by default but can then be changed when interacting with the dialog.  <i>Example: Favorites ADD BRANCH="Test Data\Photos"</i>
COPYTO	/S	(no value)	Modifies the generated list of favorites, turning each item into a "copy" button that will copy selected files to the favorite folder.  <i>Example: Favorites COPYTO</i>
EDIT	/S	(no value)	Display the <a href="#">Favorites and Recent / Favorites List</a> page in Preferences.  <i>Example: Favorites EDIT</i>
KEYARGS	/K/M	<qualifier:arguments> ...	When displaying a list of favorites, this argument lets you assign different behaviour to the items in the list if a qualifier key is held down. This is a multiple value argument - for each qualifier key combination listed, you can define a separate set of arguments that

			<p>will be used when the item in the list is selected.</p> <p>For example, you could configure your favorites menu to open favorite folders in a new tab by default, but in a new Lister if the <b>Control</b> key were held down.</p> <p>The qualifier part of the value consists of one or more keywords that represent the qualifier keys - <b>ctrl</b>, <b>shift</b> and <b>alt</b>. These can be combined, for example <b>ctrlshift</b> means that both the <b>Control</b> and <b>Shift</b> keys must be held down. You can also use the keyword <b>none</b> to indicate arguments that are applied when no qualifiers are held.</p> <p><i>Example: Favorites KEYARGS</i>  <b>"ctrl:NEW"</b>  <b>"none:NEWTAB=findexisting"</b></p>
MOVETO	/S	(no value)	<p>Modifies the generated list of favorites, turning each item into a "move" button that will move selected files to the favorite folder.</p> <p><i>Example: Favorites MOVETO</i></p>
NAME	/K	<name>	<p>Specifies the name of the newly created favorite or alias. If a name is not provided, the name of the folder will be used by default.</p> <p><i>Example: Favorites ADD=alias</i>  <b>NAME=docs PATH</b>  <b>"C:\Users\Jon\Documents"</b></p>
NEW	/S	(no value)	<p>Favorites selected from the list generated by this command will open in a new Lister instead of the current one.</p> <p><i>Example: Favorites NEW</i></p>
NEWTAB	/O	(no value)	<p>Favorites selected from the list generated by this command will open in a <a href="#">new tab</a>.</p> <p><i>Example: Favorites NEWTAB</i></p>
		<b>deflister</b>	<p>If no lister exists, the Default Lister will open with an additional tab for the folder. If a lister exists, the folder will open normally in a new tab within the existing</p>

			<p>lister.</p> <p><i>Example: Favorites NEWTAB=deflister</i></p>
		<b>findexisting</b>	<p>Look for the folder in an existing tab before opening a new one.</p> <p><i>Example: Favorites NEWTAB=findexisting</i></p>
		<b>nofocus</b>	<p>New tabs opened by favorites selected from the list will not be brought to the front.</p> <p><i>Example: Favorites NEWTAB=nofocus OPENINDUAL</i></p>
		<b>tofront</b>	<p>If the folder was found in an existing tab, bring that tab to the front (only used with <b>findexisting</b>).</p> <p><i>Example: Favorites NEWTAB=findexisting,tofront</i></p>
NOLABEL	/S	(no value)	<p>The favorites list displayed by this command will not show any labels for the favorite folders.</p> <p><i>Example: Favorites NOLABEL</i></p>
NOOPENINTABS	/S	(no value)	<p>Do not add the <b>Open in tabs</b> command that is normally displayed at the bottom of the generated favorites list.</p> <p><i>Example: Favorites NOOPENINTABS</i></p>
OPENINDEST	/S	(no value)	<p>Favorites selected from the list will open in the destination file display or Lister.</p> <p><i>Example: Favorites OPENINDEST</i></p>
OPENINDUAL	/S	(no value)	<p>Favorites selected from the list will open in the other file display of a dual-display Lister. The Lister will be set to dual-display mode if it isn't in that mode already.</p> <p><i>Example: Favorites OPENINDUAL</i></p>
OPENINLEFT	/S	(no value)	<p>Favorites will open in the left-hand (or top) display of a dual-display Lister.</p> <p><i>Example: Favorites KEYARGS none:OPENINLEFT ctrl:OPENINRIGHT</i></p>

OPENINRIGHT	/S	(no value)	<p>Favorites will open in the right-hand (or bottom) display of a dual-display Lister.</p> <p><i>Example: Favorites NEWTAB OPENINRIGHT</i></p>
PATH		<folder path>	<p>When used with the <b>ADD</b> argument, specifies the path to add as a favorite or alias. Without this, the current source folder will be used.</p> <p>When used without the <b>ADD</b> argument, this modifies the dynamically generated list of favorite folders to only show the contents of a specified sub-branch of the favorites list. The branch path must be preceded with an asterisk.</p> <p>This is the default argument for the <b>Favorites</b> command; you do not need to specify the <b>PATH</b> keyword itself.</p> <p><i>Example: Favorites /desktop ADD</i>  <i>Example: Favorites *Projects</i></p>
SHOWICONS	/S	(no value)	<p>The favorites list displayed by this command will display icons for the items within it. Note that the button that contains the <b>Favorites</b> command must also have its <b>Show image</b> option turned on.</p> <p><i>Example: Favorites SHOWICONS</i></p>
SMART	/O	(no value)	<p>Generates a dynamic list of <a href="#">SmartFavorites</a> instead of the regular Favorite list. The number of folders displayed in the list is specified on the <a href="#">Favorites and Recent / SmartFavorites</a> Preferences page.</p> <p><i>Example: Favorites SMART</i></p>
		<number of items>	<p>Display the specified number of SmartFavorites (overrides the number set in Preferences).</p> <p><i>Example: Favorites SMART 20 SHOWICONS NEWTAB</i></p>
USEQUALKEYS	/S	(no value)	<p>Activates pre-configured behaviour for the main qualifier keys - <b>Control</b> will open the favorite folder in the dual-display, <b>Shift</b> in a new Lister and <b>Alt</b> in a</p>

			<p>new tab.</p> <p>This is equivalent to <b>KEYARGS ctrl:OPENINDUAL shift:NEW alt:NEWTAB</b>.</p> <p><i>Example: Favorites USEQUALKEYS</i></p>
--	--	--	------------------------------------------------------------------------------------------------------------------------------------------------

## FileType

The FileType internal command can be used to:

- Execute a file-type-specific command or action on selected files
- Display the context menu for selected items in a drop-down menu
- Display the context menu items generated by a specific context menu extension
- Edit the file type definition for the selected file (displays the [File Type editor](#))
- Display the system *New* menu (that lets you create new files and folders)
- Create a new file of a specified type
- Display the system *Open With* menu (that lets you choose the program to open selected files with)
- Display the system *Send To* menu (that lets you send selected files and folders to various locations or programs)

## Command Arguments:

Argument	Type	Possible values	Description
ACTION		<event>	<p>Trigger a file-type-defined event for the selected files. For example, you could use this to trigger the drag-and-drop event for a file from a button or hotkey. This is the default argument for the <b>FileType</b> command - you do not need to specify the <b>ACTION</b> keyword.</p> <p>The value must be one of the following event keywords: <b>open, explorer, find, print, drop, dropshift, dropctrl, dropalt, dblclk, dblclkshift, dblclkctrl, dblclkalt, mdblclk, mdblclkalt, mdblclkshift</b> and <b>mdblclkctrl</b>.</p> <p><i>Example: FileType drop</i></p>
CONTEXTFORCE	/S	(no value)	When used with the <b>CONTEXTMENU</b> argument, this overrides the <b>Hide</b>

			<p><b>Windows items on file context menus</b> option on the <a href="#">Miscellaneous / Windows Integration</a> page in Preferences.</p> <p><i>Example:</i> <b>FileType CONTEXTMENU CONTEXTFORCE</b></p>
CONTEXTMENU	/O	(no value)	<p>Displays the standard system context menu for the selected files and folders (acts as a <a href="#">dynamic button</a>). This lets you embed the context menu in a drop-down menu (used in the default File menu, for example). You could also use this in conjunction with the <b>CONTEXTFORCE</b> option to shunt the full Windows context menu into a sub-menu of the normal context menu.</p> <p><i>Example:</i> <b>FileType CONTEXTMENU</b></p>
		<GUID>	<p>Displays any context menu items added by the specified context menu extension. This gives you the ability (in conjunction with the <b>Hide Windows items on file context menus</b> option) to control exactly which context menu items are shown on your context menu. You need to know the GUID for the context menu extension in question, and finding that out is beyond the scope of this help file - instead, see <a href="#">this article</a> on the Opus Resource Centre for an example of how this might be used.</p> <p><i>Example:</i> <b>FileType CONTEXTMENU {FB314ED9-A251-47B7-93E1-CDD82E34AF8B} CONTEXTFORCE</b></p>
		<file class>	<p>Displays any context menus for the specified file class. This gives you the ability (in conjunction with the <b>Hide Windows items on file context menus</b> option) to control exactly which context menu items are shown on your context menu.</p> <p><i>Example:</i> <b>FileType CONTEXTMENU jpegfile</b></p>
CONTEXTOPTIONS	/K	<b>INCLUDE</b> =<type>	<p>When used in conjunction with <b>CONTEXTMENU Directory\Background</b>, this lets you include specific file types (and exclude</p>

			<p>all others) from the generated shell <i>New</i> menu that is added. Note that you must enclose the value in quotes, as it contains an equals sign that would otherwise confuse the command parser.</p> <p><i>Example:</i> <b>FileType CONTEXTMENU Directory\Background CONTEXTOPTIONS "INCLUDE=.bmp,.jpg,.gif"</b></p>
		<b>EXCLUDE=&lt;type&gt;</b>	<p>Exclude the specified file types from the generated shell <i>New</i> menu when used with <b>CONTEXTMENU Directory\Background</b>. You can also use * to totally disable the New menu in this context.</p> <p><i>Example:</i> <b>FileType CONTEXTMENU Directory\Background CONTEXTOPTIONS "EXCLUDE=*"</b></p>
		<b>windowonly</b>	<p>You can specify <b>CONTEXTOPTIONS=windowonly</b> to add all "Windows" context menu items in one go. (This means all context menu items which are not specific to Opus, and can include menu items added by 3rd party programs using the Windows context menu API.)</p> <p>You can use this in conjunction with the <i>Hide Windows items on file context menus</i> Preferences option to move all "Windows" context menu items to a submenu.</p> <p>Unlike <b>INCLUDE</b> and <b>EXCLUDE</b> (described above), <b>windowonly</b> is <i>not</i> restricted to the <b>Directory\Background</b> menu.</p> <p><i>Example:</i> <b>Filetype CONTEXTMENU CONTEXTOPTIONS=windowonly</b></p>
EDIT	/S	(no value)	<p>Displays the <a href="#">File Type editor</a> for the file type corresponding to the selected file. You could add this command to the context menu for <i>All Files and Folders</i> to enable you to quickly edit the file type definition for any file.</p>

			<i>Example: FileType EDIT</i>
FILE	/K/M	<filename> ...	Specify the filename or names to perform the file type action on. If not specified, all selected files in the current source folder will be used.  <i>Example: FileType open FILE C:\Data\sales.xls</i>
FROMCLIPBOARD	/O	(no value)	Use in conjunction with the <b>NEW</b> argument to create one or more new files with names drawn from the clipboard contents. This is designed to be step two of a two-part operation; the first part being copying the names of one or more files to the clipboard with the <a href="#">Clipboard COPYNAMES</a> command.  Opus will make a new file of the specified type for each filename on the clipboard. If the clipboard contains fully qualified pathnames the new files will be created in the same location as the source files (unless overridden by the <b>PATH</b> argument). If the clipboard contains only filenames without paths, the new files will be created in the current source file display.  <i>Example: FileType NEW=.txt FROMCLIPBOARD</i>
		<b>keepext</b>	Specify this argument to preserve the file extensions of the files on the clipboard.  <i>Example: FileType NEW=.txt FROMCLIPBOARD=keepext</i>
NEW	/K	<file type>	Create a new file of the specified type. This is equivalent to selecting the appropriate item from the <i>New</i> context menu. Only files that have a registered "new" handler can be created in this manner - so if a file type doesn't appear in the <i>New</i> context menu you won't be able to create one with this command. Use the <b>NEWNAME</b> argument to modify the default name of the new file.  <i>Example: FileType NEW .txt</i>

			The <b>FileType NEW</b> command automatically sets a <a href="#">variable</a> called <b>newfile</b> to the name of the newly created file. To insert the variable, use <b>{\$newfile}</b> .
NEWCOUNT	/K/N	<count>	Allows the creation of more than one new file at once. This is the equivalent of running the <b>FileType NEW</b> command multiple times.  <i>Example: <b>FileType NEW .txt NEWCOUNT 10</b></i>
NEWMENU	/S	(no value)	Displays the shell <i>New</i> menu, that lets you create new files of various types by selecting the appropriate item from the menu (acts as a <a href="#">dynamic button</a> ). This lets you embed the <i>New</i> menu in a drop-down menu.  <i>Example: <b>FileType NEWMENU</b></i>
NEWNAME	/K	<new filename>	Specifies the filename when creating a new file via the <b>NEW</b> argument. If you don't specify the name, a default filename is used. By default, when a new file is created Opus will initiate inline rename on the new item allowing you to rename it. If you want to specify your own filename and prevent the inline rename behaviour, prefix your filename with the <b>norename:</b> string.  <i>Example: <b>FileType NEW .txt NEWNAME "norename:Text File.txt"</b></i>
OPENWITHMENU	/S	(no value)	Displays the shell <i>Open With</i> menu, that lets you choose the program to open selected files with (acts as a <a href="#">dynamic button</a> ). This lets you embed the <i>Open With</i> menu in a drop-down menu.  <i>Example: <b>FileType OPENWITHMENU</b></i>
PATH	/K	<path>	Use this in conjunction with the <b>NEW</b> argument to create a new file in a location other than the current source file display. Note that when you use this argument the newly created file will not be put into inline rename mode

			<p>automatically.</p> <p><i>Example: FileType NEW .txt PATH "{destpath}"</i></p>
SENDTOMENU	/S	(no value)	<p>Displays the shell Send To menu, that lets you send selected files and folders to various programs and destinations (acts as a <a href="#">dynamic button</a>).</p> <p><i>Example: FileType SENDTOMENU</i></p>
		<b>nosub</b>	<p>By default the Send To menu is displayed as a sub-menu; by adding the <b>nosub</b> value the items within the Send To list will be added directly to the toolbar or drop-down menu containing the <b>FileType SENDTOMENU</b> command.</p> <p><i>Example: FileType SENDTOMENU=nosub</i></p>
		<b>shift</b>	<p>Forces the Send To menu to be built as if the <b>Shift</b> key were held down. On some versions of Windows an extended menu is shown when <b>Shift</b> is held.</p> <p><i>Example: FileType SENDTOMENU=nosub,shift</i></p>

## Find

The **Find** internal command can be used to:

- Display the [Find Panel](#) where you can perform file searches
- Automate the Find Files function to perform simple searches without displaying a user interface
- Automate the [Duplicate File Finder](#) function

### Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	<p>Display the Find Panel. It will open in whichever mode you last used it.</p> <p><i>Example: Find</i></p>

ADVANCED	/S	(no value)	Display the Find Panel in <a href="#">Advanced</a> mode.  <i>Example: Find ADVANCED</i>
ANYWORD	/S	(no value)	Use when automating Find Files to enable the <i>Any Word</i> option for name searching.  For example, if this was turned on and you used "cat dog" as the <b>NAME</b> argument, Opus would match filenames containing "cat" or "dog" (or both, in any order). This saves you having to construct complicated OR wildcard patterns.  <i>Example: Example: Find NAME="cat dog" ANYWORD RECURSE IN "c:\"</i>
ARCHIVES	/S	(no value)	Search inside archive files (when the Find Files function is being automated).  <i>Example: Find *.doc IN C:\Data ARCHIVES</i>
CLEAR	/O	(no value)	Clear previous results (when the Find Files function is being automated). The output file collection will be cleared before the new Find begins.  <i>Example: Find *.doc IN C:\Data CLEAR</i>
		<b>no</b>	Do not clear previous results.  <i>Example: Find *.doc in C:\Data CLEAR=no</i>
COLLNAME	/K	<collection name>	Specify the name of the file collection that results are added to. If not supplied the default collections ( <i>Find Results</i> and <i>Duplicate Files</i> ) are used.  <i>Example: Find *.doc in C:\Data COLLNAME Output</i>
COMPUTERS	/S	(no value)	Launch the Windows <i>Search for Computers</i> function (depending on your version of Windows, this may have no effect).  <i>Example: Find COMPUTERS</i>
CONTAINING	/K	<search text>	Specify text to search for (when the Find Files function is being automated). The <b>CONTCASE</b> and <b>CONTWILD</b> arguments can modify how the search text is used.  <i>Example: Find *.txt CONTAINING printf IN C:\SourceCode</i>
CONTCASE	/S	(no value)	Use when automating Find Files to specify that the <b>CONTAINING</b> argument should be treated as case-sensitive, similar to the related checkbox

			in the UI.  <i>Example: Find *.txt CONTAINING Apple CONTCASE IN C:\Inventory</i>
CONTWILD	/S	(no value)	Use when automating Find Files to specify that the <b>CONTAINING</b> argument should be treated as a wildcard, similar to the related checkbox in the UI.  <i>Example: Find *.txt CONTAINING (foo bar) CONTWILD IN C:\Things</i>
DELMODE	/S	(no value)	When automating the Duplicate File Finder, activates "delete mode" - once duplicates have been identified, the second and subsequent of each duplicate group will be automatically selected, ready for deletion.  <i>Example: Find IN C:\Data DUPES DELMODE</i>
DUPES	/S	(no value)	Automates the Duplicate File Finder.  <i>Example: Find IN C:\Data DUPES</i>
FILTER	/S	(no value)	Indicates that the value of the <b>NAME</b> argument is the name of a <a href="#">pre-defined filter</a> . This lets you automate more complex searches than just those based on filename and containing text.  <i>Example: Find NAME image_files IN C:\Pictures FILTER</i>
IN	/K/M	<search location> ...	Specify the folder or folders to search in when automating a search. Remember that if the paths contain a space you must enclose the value in quotes.  <i>Example: Find *.txt IN "C:\Folder 1" "C:\Folder 2"</i>
LOADPREV	/K	<b>no</b>	Disable the loading of previous find settings when the Find panel opens. By default, the Find panel will remember its previous settings when it opens unless it has been invoked via an automated search (i.e with both <b>IN</b> and <b>NAME</b> arguments specified). Use this argument to prevent the command ever remembering its previous settings.  <i>Example: Find LOADPREV=no</i>
		<b>yes</b>	Force the Find panel to load its previous settings even when opened by an automated search. This does not restore multiple search paths, however -

			<p><b>LOADPREV=all</b> must be specified for that.</p> <p><i>Example: Find *.txt IN "C:\Folder 1"</i> <b>LOADPREV=yes</b></p>
		<b>all</b>	<p>Forces the Find panel to remember its previous settings, and also restores all paths that were previously listed in the search list.</p> <p><i>Example: Find LOADPREV=all</i></p>
MD5	/O	(no value)	<p>Search for duplicates using the MD5 checksum (used with the <b>DUPES</b> argument). The default behaviour is to search based on filename and size.</p> <p><i>Example: Find IN C:\Data DUPES MD5</i></p>
		<percentage>	<p>Only calculate checksums based on a percentage of the file's contents. This can dramatically speed up duplicate checking for large files, at the expense of some accuracy.</p> <p><i>Example: Find IN C:\Data DUPES MD5=75</i></p>
		<b>cache</b>	<p>Use the checksum cache for large files. If a file's checksum has previously been cached and the file does not appear to have changed, the cached value will be used instead of recalculating its checksum.</p> <p><i>Example: Find IN C:\Downloads DUPES MD5=cache,50</i></p>
NAME		<filename>	<p>This is the default argument for the <b>Find</b> command, which means you do not need to use the <b>NAME</b> keyword before it (unless you are searching for a word which is recognized as another argument keyword).</p> <p>Specify the filename or wildcard pattern to search for. To automate the Find Files function you must specify both the <b>NAME</b> argument, and the location to search using the <b>IN</b> argument.</p> <p>The <b>NOWILD</b>, <b>ANYWORD</b> and <b>NOPARTIAL</b> arguments can be used to control whether or not wildcards, "any word" or partial matching are used when searching names, similar to the checkboxes in the UI.</p>

			<p>You can start the wildcard pattern with <b>regex:</b> to use <a href="#">regular expressions</a> instead of wildcards. (Do not specify the <b>NOWILD</b> argument in this case.)</p> <p>You can also search using a pre-defined filter by providing the filter name for the <b>NAME</b> argument and also specifying the <b>FILTER</b> switch.</p> <p><i>Example: Find *.doc txt bmp IN C:\ RECURSE ARCHIVES</i></p> <p><i>Example: Find regex:foo.+bar\.txt IN C:\ RECURSE</i></p> <p><i>Example: Find NAME="Text File (1).txt" NOWILD IN C:\ RECURSE</i></p> <p>When used with the <b>DUPES</b> argument, this lets you provide a filename filter for the duplicates search. When used like this, you can specify a regular expression pattern by prefixing the wildcard with the <b>regex:</b> prefix.</p> <p><i>Example: Find *.pdf IN C:\Documents RECURSE CLEAR MD5 DUPES</i></p>
NAMEONLY	/O	(no value)	<p>Search for duplicates based on filename only (used with the <b>DUPES</b> argument). The default behaviour is to search based on filename and size.</p> <p><i>Example: Find IN C:\Data DUPES NAMEONLY</i></p>
		<b>noext</b>	<p>Search for duplicates based on filename only, ignoring their file extensions.</p> <p><i>Example: Find IN C:\Data DUPES NAMEONLY=noext</i></p>
NOAUTORUN	/S	(no value)	<p>Normally when both the <b>NAME</b> and <b>IN</b> arguments are provided, the Find Files operation begins automatically. To prevent this, specify the <b>NOAUTORUN</b> argument as well.</p> <p><i>Example: Find *.doc IN /mydocuments RECURSE NOAUTORUN</i></p>
NOPARTIAL	/S	(no value)	<p>Use when automating Find Files to prevent partial name matching (names must then match</p>

			<p>the pattern you specify exactly).</p> <p><i>Example: Find *.bak IN "c:\\" RECURSE NOPARTIAL</i></p>
NOWILD	/S	(no value)	<p>Use when automating Find Files to prevent wildcard name matching. Like the similar checkbox in the UI.</p> <p><i>Example: Find "File (1).txt" IN "c:\\" RECURSE NOWILD</i></p>
NUMBERGROUPS	/S	(no value)	<p>The <b>Duplicate Finder</b> creates groups for each set of duplicate files found, and normally these groups are named after the criteria used for the duplicate search. If you turn this option on then each group of duplicates will be numbered (from 1 to X, in the order they are found).</p> <p><i>Example: Find IN C:\Data DUPES MD5 NUMBERGROUPS</i></p>
OF	/K/M	<file to match> ...	<p>Specifies one or more files to search for duplicates of. If you don't use this argument, all duplicate files in the specified locations will be found. Remember that if the file path you are providing contains spaces, you must enclose the value in quotes.</p> <p><i>Example: Find DUPES OF C:\Report.txt IN C:\Backups MD5</i></p>
QUERY	/K/R	<query string>	<p>Performs a search in the specified location using the <a href="#">Windows Search</a> system.</p> <p>If used on its own, without other arguments, it is like typing a query string into the search field at the top-right of the lister to search below the current folder. You can also run a Windows Search query on a specified folder or into a custom results collection.</p> <p>Everything which follows the <b>QUERY</b> argument will be passed to Windows Search as the query string, so it <i>must</i> be the last argument to the command.</p> <p><i>Example: Find QUERY filename:*.doc author:"Leo Davidson"</i>  <i>Example: Find IN "C:\Documents"</i></p>

			<b>COLLNAME "Leo's Docs" QUERY filename:*.doc author:"Leo Davidson"</b>
RECURSE	/O	(no value)	When automating the Find Files and Duplicate File Finder functions, the search will extend into sub-folders below the specified locations.  <i>Example: Find IN C:\Data DUPES RECURSE</i>
		<b>no</b>	Prevent the search from extending into sub-folders.  <i>Example: Find IN C:\Data DUPES RECURSE=no</i>
SAVEQUERY	/S	(no value)	When you are currently viewing the results of a <a href="#">Windows Search</a> query, this command will save the search as a <a href="#">stored query collection</a> .  <i>Example: Find SAVEQUERY</i>
SHOWRESULTS	/K	<b>source</b>	Display the results of the automated search in the current source file display or Lister.  <i>Example: Find *.txt IN C:\ RECURSE SHOWRESULTS=source</i>
		<b>dest</b>	Display the results in the destination file display.  <i>Example: Find *.jpg IN C:\Data SHOWRESULTS=dest</i>
		<b>tab</b>	Display the results in a new tab. You can combine this with the <b>source</b> or <b>dest</b> arguments.  <i>Example: Find IN C:\Data DUPES RECURSE SHOWRESULTS dest,tab</i>
SIMPLE	/S	(no value)	Display the Find Panel in <a href="#">simple mode</a> .  <i>Example: Find SIMPLE</i>
SIZEONLY	/S	(no value)	Search for duplicates based on file size only (used with the <b>DUPES</b> argument). The default behaviour is to search based on filename and size.  <i>Example: Find IN C:\Data DUPES SIZEONLY</i>

## GetSizes

The **GetSizes** internal command can be used to:

- Calculate and display the total sizes of selected folders

- Calculate the total sizes of all folders in the current file display
- Calculate the MD5 checksum for all selected files

### Command Arguments:

Argument	Type	Possible values	Description
<i>(no argument)</i>	-	-	Calculate the total size of all selected folders. If there are no folders currently selected, all folders in the current file display will have their sizes calculated.  <i>Example: <b>GetSizes</b></i>
IGNOREJUNCTIONS	/O	<i>(no value)</i>	Ignores junctions and softlinks within folders when calculating their size. This overrides the <b>Preferences / Folders / Folder Behaviour / Ignore junctions and softlinks when calculating folder sizes</b> Preferences option.  <i>Example: <b>GetSizes IGNOREJUNCTIONS</b></i>
		<b>no</b>	Does not ignore junctions and softlinks when calculating the sizes of folders.  <i>Example: <b>GetSizes IGNOREJUNCTIONS=no</b></i>
MD5	/S	<i>(no value)</i>	Calculate the MD5 checksum for all selected files. If the MD5 column is not currently displayed in the file display it will be added automatically. Using this command overrides the <b>max_md5_file_size</b> setting on the <a href="#">Miscellaneous / Advanced</a> page in Preferences.  <i>Example: <b>GetSizes MD5</b></i>
NODESELECT	/S	<i>(no value)</i>	Prevent the <b>GetSizes</b> command from deselected files and folders once their sizes/checksums have been calculated. This option only works if the <b>Postpone file deselection until end of function</b> option on the <a href="#">File Operations / Options</a> page in Preferences is turned on.  <i>Example: <b>GetSizes NODESELECT</b></i>
SHA	/S	<i>(no value)</i>	Calculate the SHA-1 checksum for all selected files. If the SHA-1 column is not currently displayed in the file display it will be added automatically. Using this command overrides the <b>max_md5_file_size</b> setting on the <a href="#">Miscellaneous / Advanced</a> page in Preferences.

			<i>Example:</i> <b>GetSizes SHA</b>
USEHASHCACHE	/S	(no value)	In conjunction with the <b>MD5</b> and <b>SHA</b> options, this enables the use of the checksum cache. If a file has previously had its checksum cached, and the file doesn't appear to have changed, the cached value will be used.  <i>Example:</i> <b>GetSIZES MD5 USEHASHCACHE</b>

## Go

The **Go** internal command can be used to:

- [Navigate](#) to a different folder in the Lister
- Move back and forward in the [history list](#), and display the contents of the history list
- Connect and disconnect network shares
- Display a list of drives on your toolbar
- Display the contents of a folder in a drop-down menu
- Switch back and forth between a folder and its [compatibility store](#)
- Display a list of sites in the [FTP address book](#)
- Open new Listers
- Open and manipulate [folder tabs](#) and [tab groups](#).

## Command Arguments:

Argument	Type	Possible values	Description
BACK	/S	(no value)	Navigate to the previous folder in the <a href="#">history list</a> . The history list preserves the file selection and scroll offset state of the folder. The <b>BACK</b> argument can be combined with the <b>UP</b> argument; in that case, if the previous folder in the history list is the current folder's parent, Opus will go back (preserving

			<p>selections, etc) rather than <b>UP</b>.</p> <p><i>Example: Go BACK</i></p>
BACKLIST	/O	(no value)	<p>Display a list of all previous folders in the history list (acts as a <a href="#">dynamic button</a>). This is used on the drop-down menu attached to the Back button on the default toolbar.</p> <p><i>Example: Go BACKLIST</i></p>
		<b>noicons</b>	<p>Does not display icons on the generated history list.</p> <p><i>Example: Go BACKLIST=noicons</i></p>
		<b>keys</b>	<p>Assigns the accelerator keys <b>0</b> through <b>9</b> to the first ten items displayed on the generated history list.</p> <p><i>Example: Go BACKLIST=noicons,keys</i></p>
		<b>sort</b>	<p>Sorts the history list alphabetically instead of in chronological order.</p> <p><i>Example: Go BACKLIST=sort</i></p>
COMPATIBILITYFILES	/S	(no value)	<p>Switch between a folder and its <a href="#">compatibility store</a> (Vista and above only). Not all folders have compatibility stores - this command will do nothing in that case. For example, from <b>C:\Program Files</b> you would be taken to <b>C:\Users\...\VirtualStore\Program Files</b>, and vice versa.</p> <p><i>Example: Go COMPATIBILITYFILES</i></p>
CONNECT	/O	(no value)	<p>Display the system dialog that allows you to map a network share to a drive letter.</p> <p><i>Example: Go CONNECT</i></p>
		<network path>	<p>Map the specified network path to a drive letter.</p> <p><i>Example: Go CONNECT \\server\share</i></p>
CURDIR	/O	(no value)	<p>Activates "current directory" mode for the <b>Go</b> command. When used with a drive letter for the <b>PATH</b> argument, this mode makes Opus navigate to the most recently accessed folder on the specified drive. Opus will remember the "current directory" for each drive in your system, even from one session to the next.</p> <p>When used in conjunction with the</p>

			<p><b>DRIVEBUTTONS</b> argument, the generated drive letter buttons will have the same behaviour, and will highlight to indicate the "current drive". In this way you can click around from one drive to another remembering the previously used folder on each drive.</p> <p><i>Example: <b>Go D: CURDIR</b></i>  <i>Example: <b>Go DRIVEBUTTONS=fixed CURDIR</b></i></p>
		<b>rootmode</b>	<p>Modifies "current directory" mode so that clicking the button for the drive you are already on takes you to the root of the drive (ordinarily it would do nothing).</p> <p><i>Example: <b>Go DRIVEBUTTONS CURDIR=rootmode</b></i></p>
CURRENT	/S	(no value)	<p>Indicates the current source folder. This argument is used to open the current folder in another tab, Lister or file display.</p> <p><i>Example: <b>Go CURRENT OPENINDUAL</b></i></p>
DESTPATH	/S	(no value)	<p>Indicates the current destination folder. This argument is used to open the destination folder in the source file display (or another Lister, or tab).</p> <p><i>Example: <b>Go DESTPATH</b></i></p>
DISCONNECT	/S	(no value)	<p>Display the system dialog that allows you to disconnect (unmap) a network share.</p> <p><i>Example: <b>Go DISCONNECT</b></i></p>
DRIVEBUTTONS	/O	(no value)	<p>Display a list of all the drives currently on your system (acts as a <a href="#">dynamic button</a>). Clicking a button navigates the source folder to that drive's root. The drive buttons can also be right-clicked to display the context menu for each drive.</p> <p>The various values for this argument can be used to restrict the drives that are shown.</p> <p><i>Example: <b>Go DRIVEBUTTONS</b></i></p>
		<b>fixed</b>	<p>Only display the fixed drives (hard drives).</p> <p><i>Example: <b>Go DRIVEBUTTONS=fixed</b></i></p>

	<b>network</b>	Only display network (mapped) drives.  <i>Example: Go <b>DRIVEBUTTONS=network</b></i>
	<b>cdrom</b>	Only display CD-ROM (and DVD) drives.  <i>Example: Go <b>DRIVEBUTTONS=cdrom</b></i>
	<b>removable</b>	Only display removable drives (floppies, and some USB drives).  <i>Example: Go <b>DRIVEBUTTONS=removable,cdrom</b></i>
	<b>ramdisk</b>	Only display RAM drives.  <i>Example: Go <b>DRIVEBUTTONS=ramdisk</b></i>
	<b>mtp</b>	Only display MTP (portable) devices.  <i>Example: Go <b>DRIVEBUTTONS=mtp</b></i>
	<b>iconlettersoff</b>	Disable the display of small drive letters as part of each drive's icon.  <i>Example: Go <b>DRIVEBUTTONS=fixed,iconlettersoff</b></i>
	<b>iconletterson</b>	Enable the display of small drive letters as part of each drive's icon.  <i>Example: Go <b>DRIVEBUTTONS=iconletterson</b></i>
	<b>labels</b>	Displays the label of each drive. By default only each drive's letter is shown.  <i>Example: Go <b>DRIVEBUTTONS=fixed,labels</b></i>
	<b>noletters</b>	Prevents the display of each drive's letter, if labels are being shown using the <b>labels</b> keyword. Note that if you want to completely disable any text being displayed in the button you need to turn off the button's <b>Show Label</b> checkbox in the <a href="#">button editor</a> .  <i>Example: Go <b>DRIVEBUTTONS=labels,noletters</b></i>
	<b>multifunc</b>	The generated drive buttons will be <a href="#">multiple function buttons</a> (three-button buttons) - clicking them with the left mouse button will act as if <b>OPENINLEFT</b> were set, the right button will act as if <b>OPENINRIGHT</b> were

		set, and the middle mouse button will act as if <b>NEW</b> were set.  <i>Example: Go</i> <b>DRIVEBUTTONS=fixed,network,multifunc</b>
	<b>multifunctabs</b>	Similar to <b>multifunc</b> , except the left and right mouse button functions will open a new tab on the appropriate side of the Lister. You can control how new tabs are opened with the <b>NEWTAB</b> argument.  <i>Example: Go</i> <b>DRIVEBUTTONS=multifunctabs,labels,fi</b> <b>xed</b>
	<b>lettersbeforelabels</b>	When showing both drive letters and labels, the letters will be displayed first. Without this letters are shown following the labels.  <i>Example: Go</i> <b>DRIVEBUTTONS=fixed,labels,lettersbeforelabels</b>
	<b>offline</b>	When showing network drives, only offline drives will be shown (by default both connected and offline drives are shown).  <i>Example: Go</i> <b>DRIVEBUTTONS=network,offline</b>
	<b>online</b>	When showing network drives, only online (connected) drives will be shown.  <i>Example: Go</i> <b>DRIVEBUTTONS=network,online</b>
	<b>hideempty</b>	Hides the display of empty drives. Removable disks (floppies, card readers, DVDs) that have no media inserted in them will not be displayed.  <i>Example: Go</i> <b>DRIVEBUTTONS=cdrom,removable,hideempty</b>
	<b>+&lt;letters&gt;</b>	Only display the specified drive letters. Any drives not specified will be hidden.  <i>Example: Go</i> <b>DRIVEBUTTONS=removable,+fhjm</b>
	<b>-&lt;letters&gt;</b>	Do not display the specified drive letters.  <i>Example: Go</i> <b>DRIVEBUTTONS=-d</b>

DUALPATH	/K	<path to read>	Specify a path to read into the destination file display of a dual-display Lister (the standard <b>PATH</b> argument reads into the source file display).  <i>Example: Go C:\ DUALPATH D:\</i>
EJECT	/S	(no value)	Trigger an eject of the media in the drive specified by the <b>PATH</b> argument. This command has no effect if the drive does not have an eject mechanism.  <i>Example: Go D: EJECT</i>
EXISTINGLISTER	/O	(no value)	If the specified folder is already open in an existing lister (including inactive folder tabs in other windows) then the command will activate that lister and the appropriate tab within it and do nothing else. If another window is found then the rest of the command's arguments are ignored. If the specified path is not already open in another window then the command continues as if the <b>EXISTINGLISTER</b> argument had not been given.  <i>Example: Go "C:\Program Files" NEW EXISTINGLISTER</i>
		<b>seltabonly</b>	Only selected folder tabs will be considered. If the specified folder is open in an existing lister, but in a folder tab which is not currently selected, then that tab will be ignored.  <i>Example: Go "C:\Program Files" NEW EXISTINGLISTER=seltabonly</i>
EXPANDTREE	/S	(no value)	Automatically expands the folder tree to display the contents of the new folder. This is only of use if the <b>Automatically expand to current folder</b> option on the <a href="#">Folder Tree / Options</a> Preferences page is turned off.  <i>Example: Go "C:\Program Files\GPSoftware\Directory Opus" NEW EXPANDTREE</i>
FDBBUTTONS	/O	(no value)	This command acts as a <a href="#">dynamic button</a> . It lets you add buttons to a toolbar that mimic the standard File Display Border buttons. The command is designed to be used on the File Display toolbar as an alternative to the standard buttons.

			<p>By default the command will add <i>Back</i>, <i>Forward</i>, <i>Up</i>, <i>Copy</i>, <i>Swap</i> and <i>Toggle Layout</i> buttons. You can use the various keywords to control exactly which buttons are added.</p> <p><i>Example: Go FDBBUTTONS</i></p>
		< <i>button keywords</i> >	<p>Use combinations of the keywords <b>back</b>, <b>forward</b>, <b>up</b>, <b>copy</b>, <b>swap</b> and <b>layout</b> to control exactly which buttons are added.</p> <p><i>Example: Go FDBBUTTONS</i>  <b>back,forward</b> - show <i>Back</i> and <i>Forward</i> buttons  <i>Example: Go FDBBUTTONS -layout</i> - show all but the <i>Toggle Layout</i> button</p>
		<b>noicons</b>	<p>Do not display icons for the generated buttons.</p> <p><i>Example: Go FDBBUTTONS -layout,noicons</i></p>
		<b>nolabels</b>	<p>Do not display labels for the generated buttons.</p> <p><i>Example: Go FDBBUTTONS</i>  <b>swap,layout,nolabels</b></p>
		<b>dropdowns</b>	<p>Make the generated buttons drop-down buttons where applicable.</p> <p><i>Example: Go FDBBUTTONS</i>  <b>back,forward,up,dropdowns</b></p>
FINDTITLE	/K	< <i>title string</i> >	<p>Finds all currently open Listers with titles that match the specified string, and brings them to the front. The string to search for can be a specific title or a <a href="#">wildcard pattern</a>.</p> <p><i>Example: Go FINDTITLE *Projects*</i></p>
FOLDERCONTENT	/O	( <i>no value</i> )	<p>Display the contents of the path specified by the <b>PATH</b> argument in drop-down menus (acts as a <a href="#">dynamic button</a>). Sub-folders in the generated list can be selected to navigate to that location, and files in the generated list can be selected to open that file. You can also right-click on items to display their context menus, and drag-and-drop files to folders (to copy or move them) or over other files (to open them). If you hover the mouse over a sub-folder it will expand to display another menu showing the contents of the sub-folder.</p>

		<p>When used with <b>FOLDERCONTENT</b> only, the <b>PATH</b> argument supports multiple paths separated with a vertical bar ( ). For example, this would let you reproduce (to some extent anyway) the Windows start menu in a drop-down menu in a Lister, by showing both the current user's start folder and the common (all users) start folder.</p> <p><i>Example: Go /start /commonstartmenu</i> <b>FOLDERCONTENT</b></p>
	<b>button</b>	<p>Sub-folders in the generated <b>FOLDERCONTENT</b> list will appear as menu buttons. Clicking the button part will read the sub-folder, expanding the drop-down will display its contents.</p> <p><i>Example: Go C:</i> <b>FOLDERCONTENT=button</b></p>
	<b>norecuse</b>	<p>Prevents sub-folders in the generated list from being expanded; the list will be limited to a single folder level (although sub-folders will still be displayed, you will not be able to expand them).</p> <p><i>Example: Go C:</i> <b>FOLDERCONTENT=norecuse</b></p>
	<b>nodirs</b>	<p>Excludes sub-folders from the generated list - only files will be shown.</p> <p><i>Example: Go /temp</i> <b>FOLDERCONTENT=nodirs</b></p>
	<b>nofiles</b>	<p>Excludes files from the generated list - only folders will be shown.</p> <p><i>Example: Go C:</i> <b>FOLDERCONTENT=nofiles</b></p>
	<b>showhidden</b>	<p>Includes hidden files and folders in the list - without this, items with the <b>H</b> attribute set will be excluded.</p> <p><i>Example: Go C:</i> <b>FOLDERCONTENT=nofiles,showhidden</b></p>
	<b>noparselinks</b>	<p>Prevents shortcuts from being resolved. Without this, a shortcut to a folder will be expandable just like a normal sub-folder.</p>

		<p><i>Example: Go /profile</i>  <b>FOLDERCONTENT=noparselinks</b></p>
	<b>showempty</b>	<p>Empty sub-folders will be included in the generated list. Without this, empty sub-folders are excluded.</p> <p><i>Example: Go CURRENT</i>  <b>FOLDERCONTENT=showempty</b></p>
	<b>filefilter=&lt;pattern&gt;</b>	<p>Specifies a <a href="#">wildcard pattern</a> that the names of files must match to be included in the generated list (without this, all files are included). Because this keyword requires an embedded equals sign, you must enclose the entire argument value in quotes to avoid confusing the command parser.</p> <p><i>Example: Go C:</i>  <b>FOLDERCONTENT="filefilter=*.exe"</b></p>
	<b>dirfilter=&lt;pattern&gt;</b>	<p>Specifies a <a href="#">wildcard pattern</a> that folder names must match to be included in the generated list (without this, all folders are included). Because this keyword requires an embedded equals sign, you must enclose the entire argument value in quotes to avoid confusing the command parser.</p> <p><i>Example: Go C:\Work</i>  <b>FOLDERCONTENT="dirfilter=~(.svn)"</b></p>
	<b>maxdepth=&lt;levels&gt;</b>	<p>Specify the maximum number of levels deep that sub-folders can be expanded in the generated list. Because this keyword requires an embedded equals sign, you must enclose the entire argument value in quotes to avoid confusing the command parser.</p> <p><i>Example: Go C:</i>  <b>FOLDERCONTENT="maxdepth=10"</b></p>
	<b>hideext</b>	<p>Do not show the filename extensions for files in the generated list.</p> <p><i>Example: Go /mydocuments</i>  <b>FOLDERCONTENT=hideext</b></p>
	<b>sortext</b>	<p>Sorts files in the generated list by file extension.</p> <p><i>Example: Go /temp</i>  <b>FOLDERCONTENT=nodirs,sortext</b></p>
	<b>sortsize</b>	<p>Sorts files in the generated list by size.</p>

		<i>Example: Go "C:\Program Files"</i> <b>FOLDERCONTENT=sortsize</b>
	<b>sortdate</b>	Sorts files and folders in the generated list by timestamp.  <i>Example: Go /downloads</i> <b>FOLDERCONTENT=sortdate</b>
	<b>sortreverse</b>	Reverse the normal sort order of items in the generated list.  <i>Example: Go /downloads</i> <b>FOLDERCONTENT=sortdate,sortreverse</b>
	<b>copy</b>	The generated list is dedicated to copying selected files to the folders shown in the list. This modifies the standard behaviour when you select a sub-folder from the drop-down menu. For example, if you select some files in the current folder, and then choose a sub-folder from the drop-down <b>Go FOLDERCONTENT</b> menu, the files would be copied to that folder.  <i>Example: Go \\NAS\Music</i> <b>FOLDERCONTENT=nofiles,copy</b>
	<b>move</b>	Selecting a sub-folder from the generated list will move selected files to that folder.  <i>Example: Go D:\Archive\Documents</i> <b>FOLDERCONTENT=nofiles,move</b>
	<b>nomenusel</b>	Hovering over a sub-folder will no longer expand it - instead, you can expand a sub-folder by clicking it. This means it will not be possible to select a sub-folder to navigate to it (unless the button keyword is also given).  <i>Example: Go C:</i> <b>FOLDERCONTENT=nomenusel</b>
	<b>dblclickmenu</b>	Both hovering and single-clicking a sub-folder will expand it, and a double-click will select it to navigate to it.  <i>Example: Go C:</i> <b>FOLDERCONTENT=dblclickmenu</b>
	<b>useshell</b>	Normally folder paths like C:\ are enumerated using the native Windows API functions. If you specify the <b>useshell</b> keyword, they'll instead be enumerated using the shell (i.e. Explorer). This may give you localized names in some cases, as well as

			different ordering and different contents.  <i>Example: Go C: FOLDERCONTENT=useshell</i>
FORWARD	/S	(no value)	Navigate to the next folder in the <a href="#">history list</a> .  <i>Example: Go FORWARD</i>
FORWARDLIST	/O	(no value)	Display a list of all subsequent folders in the history list (acts as a <a href="#">dynamic button</a> ). This is used on the drop-down menu attached to the Forward button on the default toolbar.  <i>Example: Go FORWARDLIST</i>
		<b>noicons</b>	Does not display icons on the generated history list.  <i>Example: Go FORWARDLIST=noicons</i>
		<b>keys</b>	Assigns the accelerator keys <b>0</b> through <b>9</b> to the first ten items displayed on the generated history list.  <i>Example: Go FORWARDLIST=noicons,keys</i>
		<b>sort</b>	Sorts the history list alphabetically instead of in chronological order.  <i>Example: Go FORWARDLIST=sort</i>
FROMSEL	/S	(no value)	Indicates the first selected folder in the source file display. This argument is used to open the selected folder (in the current file display, another tab, a new Lister, etc).  <i>Example: Go FROMSEL NEW</i>
FTP	/S	(no value)	Displays the <a href="#">FTP Connect</a> dialog, allowing you to make an ad-hoc connection to an FTP site.  <i>Example: Go FTP</i>
FTPCMD	/K/ R	<command>	Sends a raw command to the remote FTP server (when currently viewing an FTP directory). If no FTP site is currently connected the command has no effect. The effects of the command, if any, can be viewed in the <a href="#">FTP log</a> .  <i>Example: Go FTPCMD chmod * 755</i>

FTPSITE	/K	<site name>	<p>Connect to an FTP site listed in the <a href="#">FTP address book</a>. The site must be specified by name, and if the site is in a sub-folder of the address book you must include the complete path of the entry. The <b>FTPSITE</b> argument is the equivalent of prefixing the site entry name with @ in the <b>PATH</b> argument.</p> <p><i>Example: Go FTPSITE "Work Servers\Dallas\Production"</i></p>
FTPSITEICONS	/S	(no value)	<p>In conjunction with the <b>FTPSITELIST</b> argument, displays icons for all sites in the list generated by the command.</p> <p><i>Example: Go FTPSITELIST FTPSITEICONS</i></p>
FTPSITELIST	/O	(no value)	<p>Displays a list of the sites in the <a href="#">FTP address book</a>(acts as a <a href="#">dynamic button</a>).</p> <p><i>Example: Go FTPSITELIST</i></p>
		<site prefix>	<p>Displays the FTP site list starting from a specified sub-folder of the address book.</p> <p><i>Example: Go FTPSITELIST "Work Servers"</i></p>
GROUPCOLLAPSE	/K	<group name>	<p>When the file display is <a href="#">grouped</a>, this command can be used to collapse a specified group. The group name must match exactly, but you can also use a <a href="#">wildcard pattern</a> to collapse all groups matching that pattern.</p> <p><i>Example: Go GROUPCOLLAPSE *</i></p>
GROUPEXPAND	/K	<group name>	<p>Expand a specified file group. The group name must match exactly, but you can also use a <a href="#">wildcard pattern</a> to expand all groups matching that pattern.</p> <p><i>Example: Go GROUPEXPAND *</i></p>
HISTORYLIST	/O	(no value)	<p>Display the contents of the history list (acts as a <a href="#">dynamic button</a>). This is a combination of the previous (<b>BACKLIST</b>) and subsequent (<b>FORWARDLIST</b>) folders.</p> <p><i>Example: Go HISTORYLIST</i></p>
		<b>noicons</b>	<p>Does not display icons on the generated history list.</p> <p><i>Example: Go HISTORYLIST=noicons</i></p>

		<b>keys</b>	<p>Assigns the accelerator keys <b>0</b> through <b>9</b> to the first ten items displayed on the generated history list.</p> <p><i>Example: Go</i> <b>HISTORYLIST=noicons,keys</b></p>
		<b>sort</b>	<p>Sorts the history list alphabetically instead of in chronological order.</p> <p><i>Example: Go</i> <b>HISTORYLIST=sort</b></p>
IGNOREQUAL	/S	(no value)	<p>Override the default behaviour of the <b>Go</b> command when various qualifiers are held down. By default, holding the <b>Shift</b> key opens a new Lister, the <b>Control</b> key reads the folder into the opposite file display, and the <b>Alt</b> key opens a new tab. If you specify this argument this functionality is disabled. You would use this if you wanted to define your own qualifier-specific behaviour using the <b>@keydown</b> <a href="#">command modifier</a>.</p> <p><i>Example: Go</i> <b>IGNOREQUAL</b></p>
INITIALDIR	/S	(no value)	<p>Returns the file display to the very first folder that it read.</p> <p><i>Example: Go</i> <b>INITIALDIR</b></p>
KEYARGS	/K/ M	<qualifier:arguments> ...	<p>Provides an alternate way to modify the behaviour of the <b>Go</b> command depending on which qualifier keys are held down (instead of using the <b>@keydown</b> <a href="#">command modifier</a>). This is a multiple value argument - for each qualifier key combination listed, you can define a separate set of arguments that will be used when command is run and that key combination is held.</p> <p>For example, you could configure a <b>Go</b> button to open a folder in a new tab by default, but in a new Lister if the <b>Control</b> key were held down.</p> <p>The qualifier part of the value consists of one or more keywords that represent the qualifier keys - <b>ctrl</b>, <b>shift</b> and <b>alt</b>. These can be combined, for example <b>ctrlshift</b> means that both the <b>Control</b> and <b>Shift</b> keys must be held down. You can also use the keyword <b>none</b> to indicate arguments that are applied when no qualifiers are held.</p>

			<p><i>Example: Go FROMSEL KEYARGS "ctrl:NEW" "none:NEWTAB=findexisting"</i></p>
LASTACTIVELISTER	/S	(no value)	<p>Brings the most recently active Lister to the front. If there is no valid Lister currently open, any other arguments provided to the command are used instead. So for example, you could have a global hotkey that brings the previous Lister to the front, or opens a new one if no Lister is open.</p> <p><i>Example: Go LASTACTIVELISTER NEW</i></p>
LASTCRUMB	/S	(no value)	<p>If the file display has a <a href="#">breadcrumbs location field</a> associated with it, and a ghost path is currently shown, this command will go to the last "crumb" in the ghost path.</p> <p><i>Example: Go LASTCRUMB</i></p>
LAYOUT	/K	<layout name>	<p>Opens the folder in a new Lister loaded from the specified <a href="#">layout</a>. If the layout contains more than one Lister, only the first Lister is used. The other arguments of the <b>Go</b> command can be used to override the settings in the layout.</p> <p><i>Example: Go FROMSEL LAYOUT=Pictures VIEW=details</i></p>
NEW	/O	(no value)	<p>Opens a new Lister. The <a href="#">Default Lister</a> settings are used for the newly opened Lister, although the various other arguments for the <b>Go</b> command can be used to override the settings of the Default Lister.</p> <p><i>Example: Go C:\ NEW</i></p>
		<x>, <y>, <w>, <h>	<p>Specify the position and size of the new Lister window. &lt;x&gt; and &lt;y&gt; represent the left and top edge coordinates of the window, and &lt;w&gt; and &lt;h&gt; the width and height.</p> <p><i>Example: Go NEW 240,300,640,480</i></p>
		<b>max</b>	<p>Maximize the new Lister window. You can use the &lt;x&gt; and &lt;y&gt; parameters to control which monitor the window appears maximized on.</p> <p><i>Example: Go NEW -1024,0,max</i></p>
		<b>min</b>	<p>Minimize the new Lister window. If a size and position is specified as well, it will</p>

		represent the restored position once you un-minimize (restore) the Lister.  <i>Example: Go NEW 1800,50,1024,768,min</i>
	<b>norm</b>	The new Lister window is to be neither minimized nor maximized. Use this to override if the Default Lister is maximized, for example.  <i>Example: Go NEW norm</i>
	<b>source</b>	Set the new Lister to be the current source.  <i>Example: Go FROMSEL NEW=source</i>
	<b>dest</b>	Set the new Lister to be the destination.  <i>Example: Go C:\Backup NEW=dest</i>
	<b>lockoff</b>	The new Lister will be set as "off" - neither <a href="#">source</a> nor <a href="#">destination</a> .  <i>Example: Go NEW=off</i>
	<b>tree</b>	Opens the folder tree in the new Lister.  <i>Example: Go C:\Windows NEW=tree</i>
	<b>notree</b>	Does not open the folder tree in the new Lister.  <i>Example: Go NEW=0,0,640,480,notree</i>
	<b>dual</b>	Open the new Lister in <a href="#">dual-display</a> mode, laid out vertically.  <i>Example: Go NEW=dual C:\ DUALPATH D:\</i>
	<b>dualhoriz</b>	Opens the new Lister in dual-display mode, laid out horizontally.  <i>Example: Go NEW=dualhoriz FROMSEL</i>
	<b>nodual</b>	Opens the new Lister in single-display mode.  <i>Example: Go NEW=nodual</i>
	<b>viewpane</b>	Displays the <a href="#">viewer pane</a> in the new Lister.  <i>Example: Go NEW=viewpane</i>
	<b>noviewpane</b>	Does not display the viewer pane in the new Lister.  <i>Example: Go NEW=noviewpane,notree</i>
	<b>findpanel</b>	Displays the <a href="#">utility panel</a> in Find Files mode in the new Lister.

			<i>Example: Go CURRENT NEW=findpanel</i>
		<b>syncpanel</b>	Displays the utility panel in Synchronize mode in the new Lister.  <i>Example: Go {sourcepath} DUALPATH {destpath} NEW=syncpanel</i>
		<b>dupepanel</b>	Displays the utility panel in Duplicate File Finder mode in the new Lister.  <i>Example: Go {sourcepath} NEW=dupepanel</i>
		<b>noutiltypanel</b>	Do not display the utility panel in the new Lister.  <i>Example: Go NEW=noutiltypanel</i>
		<b>metapane</b>	Displays the <a href="#">metadata pane</a> in the new Lister.  <i>Example: Go NEW=metapane</i>
		<b>nometapane</b>	Does not display the metadata pane.  <i>Example: Go NEW=nometapane,noutiltypanel,noviewpane,notree,nodual</i>
NEWTAB	/O	(no value)	Opens a new <a href="#">folder tab</a> . If no path is specified (e.g. via the <b>PATH</b> argument) then an empty tab is opened, otherwise the specified path will be loaded into the new tab. You can use the <b>TABPOS</b> argument to control where the newly-opened tab is positioned.  <i>Example: Go C:\ NEWTAB</i>
		<b>deflister</b>	If the command is run without a Lister then the Default Lister will open along with a new tab for the specified folder. If a Lister already exists then a new tab for the specified folder will open normally in the existing Lister.  <i>Example: Go C:\ NEWTAB=deflister,findexisting</i>
		<b>findexisting</b>	Look for the specified path in existing tabs. If found, the existing tab will be brought to the front - otherwise a new tab will be opened.  <i>Example: Go CURRENT NEWTAB=findexisting OPENINDUAL</i>

		<b>nofocus</b>	The new tab will not be made active.  <i>Example: Go CURRENT NEWTAB=nofocus</i>
		<b>tofront</b>	Brings the Lister to the front. This is useful when opening a tab in the destination Lister.  <i>Example: Go CURRENT NEWTAB=tofront OPENINDEST</i>
NEXTCRUMB	/S	(no value)	If the file display has a <a href="#">breadcrumbs location field</a> associated with it, and a ghost path is currently shown, this command will go to the next "crumb" in the ghost path.  <i>Example: Go NEXTCRUMB</i>
NOSCRIPT	/S	(no value)	Allows a <a href="#">script</a> to run <b>Go</b> commands without triggering other scripts (or itself). Adding the <b>NOSCRIPT</b> argument disables the <a href="#">OnBeforeFolderChange</a> , <a href="#">OnAfterFolderChange</a> , <a href="#">OnOpenTab</a> and <a href="#">OnOpenLister</a> events that would otherwise be triggered by the command.
OPENCONTAINER	/O	(no value)	Opens the container (parent folder) of the selected item. In normal folders this is not that useful (since the parent of the selected item is the folder you're already in), but in <a href="#">file collections</a> , <a href="#">libraries</a> and <a href="#">flat view</a> it lets you quickly go to the actual folder where a file is located.  This command also works when multiple items are selected and may cause multiple windows or tabs to open if the selected items come from multiple containers.  This command is used on the default collection item context menu.  <i>Example: Go OPENCONTAINER</i>
		<b>target</b>	Dereferences a shortcut or junction. This is similar to the <i>Find Target</i> button in the system Properties dialog for shortcuts. When you run this command with a shortcut selected, the folder containing the target of the shortcut will be loaded and the target itself will be automatically selected (unless <b>noselect</b> is also given; see below). Similarly, if a junction is selected, you'll be taken the parent of its target and its target will be

			<p>selected. If the selected item is neither a shortcut nor a junction then the command functions the same as if <b>target</b> had not been specified.</p> <p><i>Example: Go OPENCONTAINER=target NEW</i></p>
		<b>noselect</b>	<p>Normally, the item or items in question are selected and made visible in the containing folder. Specifying this argument prevents the selection and just opens the containing folder.</p> <p><i>Example: Go OPENCONTAINER=target,noselect NEWTAB</i></p>
OPENINDEST	/S	(no value)	<p>The specified folder will be read into the destination file display. If the current Lister is not in dual-display mode, then could mean the folder is read into another Lister altogether. You can combine this with <b>NEWTAB</b> to open tabs in the destination.</p> <p><i>Example: Go CURRENT OPENINDEST</i></p>
OPENINDUAL	/O	(no value)	<p>The specified folder will be read into the destination file display in a dual-display Lister. The difference between this argument and <b>OPENINDEST</b> is that this will force a single-display Lister into dual-display mode if it is not in that mode already. The default layout (horizontal or vertical) will be used in this case.</p> <p><i>Example: Go CURRENT OPENINDUAL</i></p>
		<b>horiz</b>	<p>Force the layout of the dual-display Lister to horizontal (one display above the other).</p> <p><i>Example: Go C:\Windows OPENINDUAL=horiz</i></p>
		<b>vert</b>	<p>Force the layout to vertical (side-by-side displays).</p> <p><i>Example: Go CURRENT NEWTAB OPENINDUAL=vert</i></p>
OPENINLEFT	/S	(no value)	<p>Reads the specified folder into the left-hand (or top) file display in a dual-display Lister. In a single-display Lister, this argument has no effect (the folder will be read into the</p>

			single display as normal).  <i>Example: Go {rightpath} OPENINLEFT</i>
OPENINRIGHT	/O	(no value)	Reads the specified folder into the right-hand (or bottom) file display in a dual-display Lister. If the current Lister is not already in dual-display mode it will be set to that mode automatically. The default layout (horizontal or vertical) will be used in this case.  <i>Example: Go {leftpath} OPENINRIGHT</i>
		<b>horiz</b>	Force the layout of the dual-display Lister to horizontal (one display above the other).  <i>Example: Go C:\Windows OPENINRIGHT=horiz</i>
		<b>vert</b>	Force the layout to vertical (side-by-side displays).  <i>Example: Go {leftpath} NEWTAB OPENINRIGHT=vert</i>
PATH		(no value)	Specify the path to read (or in conjunction with the <b>FOLDERCONTENT</b> argument, the path to display the contents of).  Opus supports paths in many formats, for example: <ul style="list-style-type: none"> <li>• An absolute path like <i>C:\Windows</i></li> <li>• A relative path like <i>..\.</i> (goes up two levels from the current folder)</li> <li>• A virtual filesystem <a href="#">URL-style path</a> (e.g. <i>ftp://ftp.microsoft.com</i> or <i>lib://Documents</i>)</li> <li>• An FTP <a href="#">address-book shortcut</a> (e.g. <i>@MyFtpSite</i>)</li> <li>• A <a href="#">folder alias</a> like <i>/mydocuments</i></li> <li>• An environment variable like <i>%USERPROFILE%</i></li> <li>• An <a href="#">external control code</a> like <i>{sourcepath}</i></li> <li>• A file URI like <i>file:///C:/Temp</i>.</li> </ul> <p>This is the default argument for the <b>Go</b> command and so you do not need to specify the <b>PATH</b> keyword. Remember that if the path contains spaces it needs to be enclosed in quotes.</p>

			<i>Example: Go "C:\Program Files\GPSSoftware\Directory Opus"</i>
PATHENTRY	/O	(no value)	Displays the <a href="#">find-as-you-type</a> field in a special mode ("go" mode) that lets you navigate to another folder in the current file display. This command is equivalent to <a href="#">CLI QUICKGO</a> .  <i>Example: Go PATHENTRY</i>
		<b>dest</b>	The "Go" field will act on the destination file display rather than the source. Note that it will still appear at the bottom of the source file display, but once you press <b>Enter</b> the folder will be read into the destination.  <i>Example: Go PATHENTRY=dest</i>
		<b>left</b>	The "Go" field will act on the left-hand file display, whether it is the source or destination.  <i>Example: Go PATHENTRY=left</i>
		<b>right</b>	The "Go" field will act on the right-hand file display.  <i>Example: Go PATHENTRY=right</i>
REBUILDTREE	/O	(no value)	Rebuilds the contents of the folder tree attached to the source file display. This is the equivalent of turning the tree off and then back on again.  <i>Example: Go REBUILDTREE</i>
		<b>dest</b>	Rebuilds the folder tree attached to the destination file display (when dual trees are enabled).  <i>Example: Go REBUILDTREE=dest</i>
		<b>left</b>	Rebuilds the folder tree attached to the left/top file display.  <i>Example: Go REBUILDTREE=left</i>
		<b>right</b>	Rebuilds the folder tree attached to the right/bottom file display (when dual trees are enabled).  <i>Example: Go REBUILDTREE=right</i>

		<b>both</b>	Rebuilds both folder trees (or the single tree, whichever is applicable).  <i>Example: Go REBUILDTREE=both</i>
REFRESH	/O	(no value)	Refresh the display of the current folder in the source file display.  <i>Example: Go REFRESH</i>
		<b>tree</b>	Refresh the folder tree.  <i>Example: Go REFRESH=tree</i>
		<b>both</b>	Refresh both file displays in a dual-display Lister.  <i>Example: Go REFRESH=both</i>
		<b>all</b>	Refresh all file displays and folder trees.  <i>Example: Go REFRESH=all</i>
		<b>source</b>	Refresh the source file display and its tree.  <i>Example: Go REFRESH=source</i>
		<b>dest</b>	Refresh the destination file display and its tree.  <i>Example: Go REFRESH=dest</i>
		<b>viewpane</b>	Refresh the <a href="#">viewer pane</a> . The image or file currently displayed in the pane will be reloaded.  <i>Example: Go REFRESH=viewpane</i>
		<b>left</b>	Refresh the left-hand file display.  <i>Example: Go REFRESH=left</i>
		<b>right</b>	Refresh the right-hand file display.  <i>Example: Go REFRESH=right</i>
REFRESHTHUMBS	/O	(no value)	Refreshes thumbnails displayed in the current folder. If thumbnail caching is enabled the cache for the current folder is cleared, forcing thumbnails to be regenerated.  <i>Example: Go REFRESHTHUMBS</i>
		<b>shift</b>	Refreshes thumbnails only if the <b>Shift</b> key is held down. This lets you combine a normal folder refresh with a thumbnail refresh on the one command - it could perform an ordinary folder refresh by default, and also

			force the regeneration of thumbnails with the <b>Shift</b> key held down.  <i>Example: Go REFRESH REFRESHTHUMBS=shift</i>
		<b>alt</b>	Refresh thumbnails only if the <b>Alt</b> key is held down.  <i>Example: Go REFRESHTHUMBS=alt</i>
		<b>ctrl</b>	Refresh thumbnails only if the <b>Control</b> key is held down.  <i>Example: Go REFRESHTHUMBS=ctrl REFRESH=all</i>
ROOT	/O	(no value)	Navigate to the root of the current folder. For example, the root of <i>C:\Program Files\GPSsoftware\Directory Opus</i> is <i>C:</i> .  <i>Example: Go ROOT</i>
		<b>collapse</b>	Collapses the current drive's branch in the folder tree at the same time as navigating to the root folder.  <i>Example: Go ROOT=collapse</i>
ROOTTREE	/S	(no value)	Roots the folder tree to the specified path. Opus will read the path provided, and the tree will be rebuilt to start from that path.  <i>Example: Go C:\ ROOTTREE</i>
RUNEMBEDDEDIFNOTFOUND	/S	(no value)	This argument is used when <a href="#">embedding a function</a> in the <b>Go FINDTITLE</b> command. Normally the embedded function will not be run if no Lister exists that match the supplied string. If the <b>RUNEMBEDDEDIFNOTFOUND</b> argument is specified, the embedded command will be run in the current Lister if a matching Lister is not found.  <i>Example: Go FINDTITLE PhotoWork RUNEMBEDDEDIFNOTFOUND [Set VIEW=Thumbnails]</i>
SWAP	/S	(no value)	Swaps the folders displayed in the source and destination file displays.  <i>Example: Go SWAP</i>
SWITCHPATH	/K/ M	<alternate path> ...	Switches between (or cycles through) two or more paths. The <b>PATH</b> argument is used to provide the first path in the sequence, and

			<p>then the <b>SWITCHPATH</b> argument provides the second and subsequent paths. When you run this command, Opus looks at the current path shown in the source file display. If it matches one of the provided paths, the next path in the sequence is read (and then the next, and then the next, and so on). If the current path does not match one of the provided ones the first path in the sequence is read.</p> <p><i>Example: Go C:\ SWITCHPATH D:\ E:\ F:\ G:\</i></p>
TABCLOSE	/O	(no value)	<p>Close the current <a href="#">folder tab</a> in the source file display.</p> <p>You can combine this with the <b>PATH</b> argument to specify the path of a tab that should be closed. When used in this context the <b>PATH</b> argument can accept wildcards - all tabs matching the supplied pattern will be closed. The <b>PATH</b> argument can also be used to specify the index (starting from 0) of a specific tab to close.</p> <p><i>Examples:</i>  Close the current tab: <b>Go TABCLOSE</b>  Close all tabs showing a drive on C: <b>Go TABCLOSE PATH=C:*</b>  Close the first tab: <b>Go TABCLOSE PATH=0</b></p> <p>When used from a <a href="#">script</a>, you can pass the default value of a <a href="#">Tab</a> object to specify the tab you wish to close.</p> <p><i>Example: Go TABCLOSE=&lt;tab&gt;</i></p>
		<b>left</b>	<p>Close a folder tab in the left (or top) file display, irrespective of whether it is the source or not.</p> <p><i>Example: Go TABCLOSE=left</i></p>
		<b>right</b>	<p>Close a folder tab in the right (or bottom) file display.</p> <p><i>Example: Go TABCLOSE=right PATH=1</i></p>
		<b>dest</b>	<p>Close a folder tab in the destination file display.</p>

			<i>Example: Go <b>TABCLOSE=dest</b> <b>PATH=C:\*</b></i>
TABCLOSEALL	/O	(no value)	<p>Close all folder tabs except the current one.</p> <p>When combined with the <b>TABGROUPLOAD</b> command, this overrides the tab group's <i>Close existing Folder Tabs</i> setting, and forces existing tabs to be closed.</p> <p><i>Example: Go <b>TABCLOSEALL</b></i></p>
		<b>left</b>	<p>Close all folder tabs to the left of the current tab.</p> <p><i>Example: Go <b>TABCLOSEALL=left</b></i></p>
		<b>right</b>	<p>Close all folder tabs to the right of the current tab.</p> <p><i>Example: Go <b>TABCLOSEALL=right</b></i></p>
		<b>dest</b>	<p>Lets you close tabs in the destination file display in a dual-display Lister.</p> <p><i>Example: Go <b>TABCLOSEALL=dest,right</b></i></p>
		<b>force</b>	<p>Forces locked tabs to be closed. Normally locked tabs are not closed by this command.</p> <p><i>Example: Go <b>TABCLOSEALL=right,force</b></i></p>
		<b>expand</b>	<p>Expands tabs to Listers. The tabs will be closed in the current Lister, and each folder opened as a new Lister.</p> <p><i>Example: Go <b>TABCLOSEALL=right,expand</b></i></p>
		<b>no</b>	<p>When combined with the <b>TABGROUPLOAD</b> command, this overrides the tab group's <i>Close existing Folder Tabs</i> setting, and forces existing tabs to be retained.</p> <p>This value has no meaning when not used in conjunction with <b>TABGROUPLOAD</b>.</p> <p><i>Example: Go <b>TABGROUPLOAD=MyTabs</b> <b>TABCLOSEALL=no</b></i></p>
TABCOLOR	/K	<color>	<p>Assigns a custom color to the current tab. You can specify the color in the decimal form R,G,B (e.g. 127,192,55) or the hex form</p>

			#RRGGBB (e.g. #ff0033).  <i>Example: Go TABCOLOR #ff8000</i>
		<b>reset</b>	Resets the current tab's color.  <i>Example: Go TABCOLOR=reset</i>
TABDUPLICATE	/O	(no value)	Duplicates the current folder tab.  <i>Example: Go TABDUPLICATE</i>
		<b>dual</b>	Creates a duplicate of the current folder tab in the other file display of a dual-display Lister.  <i>Example: Go TABDUPLICATE=dual</i>
TABFINDEXISTING	/S	(no value)	If the specified folder is open in another tab Opus will switch to that tab, otherwise the folder will be read into the current tab. Enables similar behavior to the <b>NEWTAB=findexisting</b> argument, except that a new tab is not opened if the path is not found.  <i>Example: Go /mydocuments TABFINDEXISTING</i>
TABGROUPFORCE	/S	(no value)	Use in conjunction with the <b>TABGROUPLOAD</b> or <b>TABGROUPLIST</b> arguments to force folders in the loaded tabs to load immediately even if they would normally be blocked by the Auto-Loading settings in <a href="#">Preferences</a> .  <i>Example: Go TABGROUPLOAD "My Tab Group" TABGROUPFORCE</i>
TABGROUPLIST	/O	(no value)	Displays a list of your saved <a href="#">Folder Tab Groups</a> (acts as a <a href="#">dynamic button</a> ). Selecting a group from the list will open those tabs in the current file display.  This argument works when used in conjunction with the <b>USEQUALKEYS</b> , <b>TABGROUPFORCE</b> and <b>KEYARGS</b> arguments.  <i>Example: Go TABGROUPLIST</i>

	<b>keys</b>	<p>Assigns the accelerator keys <b>0</b> through <b>9</b> to the first ten items displayed on the generated tab group list.</p> <p><i>Example: Go <b>TABGROUPLIST=keys USEQUALKEYS</b></i></p>
	<b>icons</b>	<p>Displays icons for items on the generated tab group list.</p> <p><i>Example: Go <b>TABGROUPLIST=keys,icons</b></i></p>
	<b>nohighlight</b>	<p>Prevents the active tab group from being highlighted.</p> <p><i>Example: Go <b>TABGROUPLIST=icons,nohighlight</b></i></p>
	<b>savecurrent</b>	<p>Saves the current folder tab group (if any) before loading a new one. Without this, any changes to a tab group will be lost when loading another one, unless you explicitly save them.</p> <p>As a special case, clicking the button for the current tab group will reload that group without first saving over it, giving you a way to reset to the group's last saved state.</p> <p>If no tab group is currently loaded, nothing will be saved before loading the new tab group.</p> <p><i>Example: Go <b>TABGROUPLIST=savecurrent</b></i></p> <p>Note that if you want Opus to save tab groups automatically in this way, you may also want to add something to trigger <b>Go TABGROUPSAVE=!current,!quiet</b> when listers close, or on other events such as opening/closing tabs or changing folders, depending on when you want the tab group to be updated. This probably also only makes sense if you only use one lister, since multiple listers may have different views of the currently open groups and save over each other. But if you want auto-saving tab groups, you're probably already using a single window all the time, as other people would use separate windows for separate tab</p>

			groups and switch between windows instead of switching groups in a single window.
TABGROUPLOAD	/K	<group name>	<p>Loads the named folder tab group. This can be combined with <b>OPENINDUAL</b> etc. to open the tab group in another file display. You can also combine the <b>TABCLOSEALL</b> argument to override the tab group's <i>Close existing Folder Tabs</i> setting.</p> <p><i>Example: Go TABGROUPLOAD "My Tab Group" TABCLOSEALL=no</i></p>
TABGROUPSAVE	/O	(no value)	<p>Save the current set of folder tabs as a new tab group. You will be prompted to provide a name for the group.</p> <p><i>Example: Go TABGROUPSAVE</i></p>
		<group name>	<p>Save the current set of folder tabs using the specified group name.</p> <p><i>Example: Go TABGROUPSAVE "My Tab Group"</i></p>
		<b>!both</b>	<p>In a dual-display Lister, this will save the tabs from both file displays to a single group using the "specific sides" option. You can also specify a group name by following the keyword with a comma. (The group name, if any, must come after the keyword and not before.)</p> <p><i>Example: Go TABGROUPSAVE="!both,My Tab Group"</i></p>
		<b>!closeall</b>	<p>Saves the tab group such that it will close all existing tabs when it is loaded (unless overridden at the time of loading). If neither <b>!closeall</b> nor <b>!nocloseall</b> are specified, saving over an existing group will preserved its mode, new groups created non-interactively will close existing tabs by default, and new groups created interactively will default to the mode used the last time the interactive dialog was displayed. (The group name, if any, must come after the keyword and not before.)</p> <p><i>Example: Go TABGROUPSAVE="!both,!closeall,My Tab Group"</i></p>

	<b>!nocloseall</b>	<p>Saves the tab group such that it will not close any existing tabs when it is loaded (unless overridden at the time of loading). See <b>!closeall</b>, above, for more information. (The group name, if any, must come after the keyword and not before.)</p> <p><i>Example: Go</i>  <b>TABGROUPSAVE="!both,!nocloseall,My Tab Group"</b></p>
	<b>!current</b>	<p>Saves over the folder tab group which was last loaded. The <b>!both</b> keyword is ignored, and instead the mode is inherited from the existing group. Add the <b>!quiet</b> keyword to make the command do nothing if there is no current tab group to save over; otherwise you will be prompted for the name of a new group in that situation.</p> <p><i>Example: Go</i>  <b>TABGROUPSAVE="!current"</b></p>
	<b>!quiet</b>	<p>Combine <b>!quiet</b> with <b>!current</b> to prevent being asked to name a new tab group if there isn't one to save over.</p> <p><i>Example: Go</i>  <b>TABGROUPSAVE="!current,!quiet"</b></p>
	<b>!unless</b>	<p>Combine <b>!unless</b> with <b>!current</b> to tell the command to save the current group unless it has a particular name. You probably won't need to use this directly, but it is how the <b>Go TABGROUPLIST=savecurrent</b> command avoids saving over the current group if you click its button to reload it.</p> <p><i>Example: Go</i>  <b>TABGROUPSAVE="!current,!quiet,!unless,My Tab Group"</b></p>
	<b>!forget</b>	<p>Tells the file display to forget the name of the last tab group it loaded. Combine with <b>!both</b> to make both sides of a dual-display lister forget. You may wish to use this if you load a tab group and then modify it to the point that it no longer has any connection to the original group, and you do not want it to remain selected in any tab group lists.</p> <p><i>Example: Go</i>  <b>TABGROUPSAVE="!forget"</b></p>

TABLINK	/K	<b>on</b>	In a dual-display Lister, links the current active tab in the source file display with the current active tab in the destination file display.  <i>Example: <b>Go TABLINK=on</b></i>
		<b>off</b>	If the currently active tab is linked, this will unlink it.  <i>Example: <b>Go TABLINK=off</b></i>
		<b>toggle</b>	Toggles tab linking on or off for the currently active tabs in both left and right file displays.  <i>Example: <b>Go TABLINK=toggle</b></i>
		<b>navlock</b>	Use in conjunction with <b>on</b> or <b>toggle</b> to specify that the tab is to be linked in <a href="#">navigation lock</a> mode (so that it follows the other tab whenever the folder is changed).  <i>Example: <b>Go TABLINK=toggle,navlock</b></i>
		reset	When two tabs are linked in navigation lock mode, the reset keyword lets you reset the sync position to the current locations (for recovering when they get out of sync).  <i>Example: <b>Go TABLINK=reset</b></i>
		<b>unlinkall</b>	Unlinks all linked tabs in the current Lister.  <i>Example: <b>Go TABLINK=unlinkall</b></i>
		<tab1>,<tab2>	When used from a <a href="#">script</a> , you can pass the default value of two <a href="#">Tab</a> objects to specify the precise tabs you wish to link. The two values must be comma-separated. If only one <a href="#">Tab</a> object is specified, it will be unlinked if it is currently linked.  <i>Example: <b>Go TABLINK=&lt;tab1&gt;,&lt;tab2&gt;,navlock</b></i>
TABLIS	/O	(no value)	Displays a list of the currently open folder tabs (acts as a <a href="#">dynamic button</a> ). Selecting a tab from the list will switch to that tab. You can also drag and drop files to the items in the list to copy them to that tab's folder.  By default the tabs shown are those from the current source file display, but you can combine with the <b>OPENINLEFT</b> , <b>OPENINRIGHT</b> and <b>OPENINDEST</b> arguments to display tabs from other file

			displays.  <i>Example: Go TABLIST</i>
		<b>keys</b>	Assigns the accelerator keys <b>0</b> through <b>9</b> to the first ten items displayed on the generated tab list.  <i>Example: Go TABLIST=keys</i>
		<b>sort</b>	Sorts the tab list alphabetically (without this, the items in the list are listed in the same order as the physical tabs). The sort order is based on the folder name (or tab label).  <i>Example: Go TABLIST=sort,keys</i>
		<b>sortpath</b>	Sorts the tab list alphabetically, based on the full path of each folder.  <i>Example: Go TABLIST=sortpath</i>
		<b>icons</b>	Displays folder icons for each tab in the list.  <i>Example: Go TABLIST=sortpath,icons</i>
		<b>nameonly</b>	Only displays the folder name for each tab rather than the full path. This results in the items displaying the same label as the tabs themselves.  <i>Example: Go TABLIST=sort,nameonly</i>
		<b>maxwidth=&lt;width&gt;</b>	Specifies a maximum width (in characters) for each item in the list. When showing the full path of each tab, you may want to use this to stop the tab list from being too wide. If not specified, a maximum width of 50 is used by default - you can disable this (and have no maximum width at all) by specifying <b>maxwidth=0</b> .  Note that because this parameter requires an equals sign, you must enclose the whole value of the <b>TABLIST</b> argument in quotes.  <i>Example: Go TABLIST="keys,maxwidth=70"</i>
TABLOCK	/K	<b>lock</b>	Lock the current <a href="#">folder tab</a> to prevent any folder changes. Attempts to change the folder will result in a new tab being created.  <i>Example: Go TABLOCK=lock</i>
		<b>lockchanges</b>	Lock the current tab. Folder changes will be allowed, but the tab will revert to its original

			(locked) directory if it is clicked, or if the focus is moved to another tab and then back to this one. You can also use the command <b>Go TABLOCKDIR</b> to revert to the original directory.  <i>Example: Go TABLOCK=lockchanges</i>
		<b>lockreuse</b>	Lock the current tab. Attempts to change folder will result in the first already existing, unlocked tab being used to read the new directory. A new tab will be opened if there are no unlocked tabs that can be used.  <i>Example: Go TABLOCK=lockreuse</i>
		<b>off</b>	Unlock the current folder tab.  <i>Example: Go TABLOCK=off</i>
		<b>toggle</b>	Combine with the various <b>lock</b> keywords to toggle that lock mode on or off.  <i>Example: Go TABLOCK=lockchanges,toggle</i>
		<b>all</b>	Lock or unlock all folder tabs.  <i>Example: Go TABLOCK=lock,all</i>
		<b>left</b>	Lock or unlock all tabs to the left of the current tab.  <i>Example: Go TABLOCK=lockchanges,toggle,left</i>
		<b>right</b>	Lock or unlock all tabs to the right of the current tab.  <i>Example: Go TABLOCK=off,right</i>
TABLOCKDIR	/S	(no value)	Indicates the original (locked) directory of the current folder tab. This is the directory that was shown in the tab when it was set to 'Lock Allow Changes' mode. You can use this to return to the locked folder, or open it in another tab or Lister. If the current tab is not locked this argument behaves the same as the <b>CURRENT</b> argument.  <i>Example: Go TABLOCKDIR</i>
TABMOVE	/O	(no value)	Move the current folder tab to the other file display. (See <b>TABPOS</b> for repositioning tabs without changing sides.) The Lister will be set to dual-display mode if it's not in that mode already.

			<i>Example: Go TABMOVE</i>
		<b>split</b>	Splits the folder tabs currently open in a single-display Lister. The Lister will be set to dual-display mode, and all tabs from the current tab onwards will be moved to the other display. If the Lister is already in dual-display mode this command has no effect.  <i>Example: Go TABMOVE=split</i>
		<b>splitlister</b>	Splits the folder tabs in the current file display to a new Lister. All tabs from the current tab onwards will be closed in the current display, and reopened in a new Lister.  <i>Example: Go TABMOVE=splitlister</i>
		<new position>	Specify the new position for the moved tab. You can specify an absolute position, <b>-1</b> to indicate the same relative position, and <b>-2</b> to position the moved tab to the right of the currently active tab. If not specified, the moved tab will be positioned at the end of all existing tabs.  <i>Example: Go TABMOVE=-2</i>
		<b>active</b>	If specified, the moved tab will be made active. Otherwise the previously active tab in the other file display remains active.  <i>Example: Go TABMOVE=active,-2</i>
TABNAME	/O	(no value)	Clears the name of the current tab if one has been assigned. This will reset the tab label back to the default which is to show the name of the current folder.  <i>Example: Go TABNAME</i>
		<tab name>	Assigns the specified name to the current tab. If combined with the <b>NEWTAB</b> argument the name will be assigned to the newly created tab.  <i>Example: Go C:\Work NEWTAB TABNAME "My Work Dir"</i>
TABPOS	/K	<b>first</b>	When used with the <b>NEWTAB</b> argument, causes the newly created tab to appear as the first on the tab bar. If used without the <b>NEWTAB</b> argument, repositions the active folder tab so it is first on the tab bar.

		<p><i>Example: Go C:\ NEWTAB TABPOS=first</i></p> <p>When used from a <a href="#">script</a>, you can pass the default value of a <a href="#">Tab</a> object to specify the tab you wish to reposition as the second parameter.</p> <p><i>Example: Go TABPOS=first,&lt;tab&gt;</i></p>
	<b>last</b>	<p>When used with the <b>NEWTAB</b> argument, causes the newly created tab to appear as the last on the tab bar. If used without the <b>NEWTAB</b> argument, repositions the active folder tab so it is last on the tab bar.</p> <p><i>Example: Go TABPOS=last</i></p> <p>When used from a <a href="#">script</a>, you can pass the default value of a <a href="#">Tab</a> object to specify the tab you wish to reposition as the second parameter.</p> <p><i>Example: Go TABPOS=last,&lt;tab&gt;</i></p>
	<b>+1</b>	<p>When used with the <b>NEWTAB</b> argument, causes the newly created tab to appear as to the right of the currently active tab. If used without the <b>NEWTAB</b> argument, repositions the active folder tab to the right of its current position on the tab bar. You can also use +2 for two positions to the right, and so on.</p> <p><i>Example: Go TABPOS=+1</i></p> <p>When used from a <a href="#">script</a>, you can pass the default value of a <a href="#">Tab</a> object to specify the tab you wish to reposition as the second parameter.</p> <p><i>Example: Go TABPOS=+3,&lt;tab&gt;</i></p>
	<b>-1</b>	<p>When used with the <b>NEWTAB</b> argument, causes the newly created tab to appear to the left of the currently active tab. If used without the <b>NEWTAB</b> argument, repositions the active folder tab to the left of its current position on the tab bar. You can also use -2 for two positions to the left, and so on.</p> <p><i>Example: Go TABPOS=-1</i></p>

			<p>When used from a <a href="#">script</a>, you can pass the default value of a <a href="#">Tab</a> object to specify the tab you wish to reposition as the second parameter.</p> <p><i>Example: Go TABPOS=-2,&lt;tab&gt;</i></p>
		<index>	<p>When used with the <b>NEWTAB</b> argument, specifies the position for the newly created tab. When used without the <b>NEWTAB</b> argument, repositions the active folder tab to a specific place on the tab bar. 0 is the first tab (you can also use "first"), 1 the second, 2 the third, and so on.</p> <p><i>Example: Go D:\ NEWTAB TABPOS=2</i></p> <p>When used from a <a href="#">script</a>, you can pass the default value of a <a href="#">Tab</a> object to specify the tab you wish to reposition as the second parameter.</p> <p><i>Example: Go TABPOS=3,&lt;tab&gt;</i></p>
TABSCROLL	/O	(no value)	<p>If there are more tabs open than will fit in the display, this command displays the tab overflow menu (useful if you want to bind this to a hotkey). You can combine this with the <b>OPENINDEST</b>, <b>OPENINLEFT</b> and <b>OPENINRIGHT</b> arguments to display the overflow menu in other file displays.</p> <p><i>Example: Go TABSCROLL</i></p>
		<delta>	<p>Scroll the folder tabs left or right if there are more tabs open than will fit in the display.</p> <p><i>Example: Go TABSCROLL=-1</i></p>
TABSELECT	/K	<b>first</b>	<p>Select (make active) the first folder tab in the current file display.</p> <p><i>Example: Go TABSELECT=first</i></p>
		<b>last</b>	<p>Select the last folder tab in the current file display.</p> <p><i>Example: Go TABSELECT=last</i></p>
		<b>prev</b>	<p>Reselect the previously selected folder tab.</p> <p><i>Example: Go TABSELECT=prev</i></p>
		<b>+1</b>	<p>Select the next folder tab.</p> <p><i>Example: Go TABSELECT +1</i></p>

		<b>-1</b>	Select the previous folder tab.  <i>Example: <b>Go TABSELECT -1</b></i>
		<index>	Select a specific folder tab. The index is 0 relative (so <b>Go TABSELECT 0</b> is equivalent to <b>Go TABSELECT first</b> ).  <i>Example: <b>Go TABSELECT=5</b></i>
		<b>home</b>	Resets a locked tab to its "home" directory. This is equivalent to <b>Go TABLOCKDIR</b> . If the current tab is not set to 'Locked Allow Changes' mode this command has no effect.  <i>Example: <b>Go TABSELECT home</b></i>
TABSWAP	/O	(no value)	Swaps the active tab in the left file display with the active tab in the right file display.  <i>Example: <b>Go TABSWAP</b></i>
		<index1>,<index2>	Specify indices to swap a specific tab from the left file display with a specific tab in the right file display.  <i>Example: <b>Go TABSWAP=0,0</b></i>
TABUNDOCLOSE	/S	(no value)	Undoes the last action that resulted in one or more tabs being closed.  <i>Example: <b>Go TABUNDOCLOSE</b></i>
TITLE	/K	<custom title>	Specifies a custom title when opening a new Lister using the <b>NEW</b> argument. You can use several special "tokens" in the title string to insert various pieces of information:  <ul style="list-style-type: none"> <li><b>%P</b> - full path of the current (source) folder</li> <li><b>%N</b> - name of the current (source) folder</li> <li><b>%R</b> - drive root of the current (source) folder</li> <li><b>%D</b> - full path of the destination folder</li> <li><b>%M</b> - name of the destination folder</li> <li><b>%L</b> - name of the Layout the Lister came from (if any)</li> <li><b>%T</b> - complete original title (useful for simply adding a prefix or suffix)</li> </ul> <i>Example: <b>Go NEW TITLE "Directory Opus - %N"</b></i>
TOFRONT	/S	(no value)	Makes the Lister the active window and brings it to the front. Typically used when

			<p>sending commands from outside of Opus, to make sure the window that reads the folder is visible.</p> <p><i>Example: Go C:\ TOFRONT</i></p>
UNDOCLOSELISTER	/O	(no value)	<p>Reopens the last Lister that was closed (so if, for example, you accidentally close the Lister you're working with, this command would bring it back).</p> <p><i>Example: Go UNDOCLOSELISTER</i></p>
		<b>closeexisting</b>	<p>Reopens the last Lister that was closed, and closes all other open Listers.</p> <p><i>Example: Go UNDOCLOSELISTER=closeexisting</i></p>
UP	/S	(no value)	<p>Navigate <a href="#">up</a> to the parent of the current folder. You can combine this with <b>OPENINDUAL</b> etc. to open the parent of the current folder in another file display or Lister. This can also be combined with the <b>BACK</b> argument - in that case, if the parent of the current folder is also the previous folder in the history list, Opus will move back rather than up - preserving the file selection and other state of the previous folder.</p> <p><i>Example: Go UP</i></p>
USEQUALKEYS	/S	(no value)	<p>Activates pre-configured behaviour for the main qualifier keys - <b>Control</b> will open the favorite folder in the dual-display, <b>Shift</b> in a new Lister and <b>Alt</b> in a new tab.</p> <p>This is equivalent to <b>KEYARGS ctrl:OPENINDUAL shift:NEW alt:NEWTAB</b>.</p> <p><i>Example: Favorites USEQUALKEYS</i></p>
USER	/K	<user name>	<p>Can be used when certain folder aliases are supplied for the <b>PATH</b> argument. This lets you specify an alternative user name, to access a specific user's instance of a system folder (providing you have the appropriate permissions, of course). If the specified alias doesn't support multiple users this argument has no effect.</p>

			<i>Example: Go /desktopdir USER="Leo Nudelson"</i>
VIEW	/K	<b>largeicons</b>	Changes the <a href="#">view mode</a> of the new folder to <i>Large Icons</i> mode.  <i>Example: Go CURRENT NEWTAB VIEW=largeicons</i>
		<b>smallicons</b>	Changes the view mode of the new folder to <i>Small Icons</i> mode.  <i>Example: Go C:\ VIEW smallicons</i>
		<b>list</b>	Changes the view mode to <i>List</i> mode.  <i>Example: Go {destpath} VIEW list</i>
		<b>details</b>	Changes the view mode to <i>Details</i> mode.  <i>Example: Go CURRENT LAYOUT=PhotoLayout VIEW=details</i>
		<b>power</b>	Changes the view mode to <i>Power</i> mode.  <i>Example: Go NEW VIEW=power</i>
		<b>thumbnails</b>	Changes the view mode to <i>Thumbnails</i> mode.  <i>Example: Go /desktop VIEW=thumbnails</i>
		<b>tiles</b>	Changes the view mode to <i>Tiles</i> mode.  <i>Example: Go @WorkFTP NEWTAB VIEW tiles</i>
WHENDUAL	/K	<b>checkmouse</b>	This works in conjunction with the <b>BACK</b> , <b>FORWARD</b> and <b>UP</b> arguments and is designed to be used with "app command" hotkeys like the Back button on a mouse. It lets you make navigation commands triggered from the mouse act on the file display underneath the mouse pointer in a dual display Lister, rather than, as is the default, the source file display.  <i>Example: Go BACK WHENDUAL=checkmouse</i>
	r	<b>anydevice</b>	Use with the <b>checkmouse</b> argument to cause the <b>WHENDUAL</b> argument to activate when the command is run from a non-mouse device (e.g. when the command is bound to a <b>Back</b> button on a keyboard).

			<i>Example: Go BACK</i> <b>WHENDUAL=checkmouse,anydevice</b>
		<b>deffocus</b>	Add the deffocus argument to cause the command to fall back to the original behavior if the mouse pointer isn't currently over either of the file displays.  <i>Example: Go BACK</i> <b>WHENDUAL=checkmouse,deffocus</b>

## Help

The **Help** internal command can be used to:

- Display the program help file
- Display information about the program (the About dialog)
- Check for [program updates](#)
- Access the [Licence Manager](#)
- Send a quick email

## Command Arguments:

Argument	Type	Possible values	Description
<i>(no argument)</i>	-	-	Displays the program help file.  <i>Example: Help</i>
ABOUT	/S	<i>(no value)</i>	Displays the About dialog, showing information about the current version of the program, copyright and contact details for technical support.  <i>Example: Help ABOUT</i>
CHECKUPDATE	/O	<i>(no value)</i>	Checks online for <a href="#">program updates</a> .  <i>Example: Help CHECKUPDATE</i>
		<b>quiet</b>	Perform a silent check for updates - a dialog will only be displayed if an update is found.  <i>Example: Help CHECKUPDATE=quiet</i>
LANG	/K	<language name>	Displays a help file in the specified language, if one exists.  <i>Example: Help LANG=deutsch</i>

LICENCEMANAGER	/S	(no value)	Displays the <a href="#">Licence Manager</a> , where you can view and manage your current program licence, or apply for a free evaluation licence.  <i>Example: <b>Help LICENCEMANAGER</b></i>
NEWEMAIL	/S	(no value)	Sends a quick email message. You must have configured Opus to send email via SMTP (on the <a href="#">Miscellaneous / Email</a> page in Preferences) for this function to work.  <i>Example: <b>Help NEWEMAIL</b></i>
REF	/K	<b>commands</b> <b>dopusrt</b> <b>metadata</b> <b>statusbar</b> <b>wildcards</b> <b>scripting</b> <b>cmd_&lt;command name&gt;</b>	Provides a shortcut to displays certain reference sections of the help file. For example, you could define a hotkey to take you directly to the <a href="#">Wildcards</a> reference section.  <i>Example: <b>Help REF=wildcards</b></i> <i>Example: <b>Help REF=cmd_CreateFolder</b></i>

## Image

The **Image** internal command can be used to:

- Change the format of images
- Resize and rotate images
- Upload or synchronize images with an online service
- Locate an image using its embedded GPS information

The **Image CONVERT** command displays the [image conversion](#) dialog in interactive mode, letting you select the conversion options to apply to selected images. Using the various arguments of this command it is possible to automate the image conversion function. The image conversion function can accept as input any image format that Opus is able to view (including those supported by plugins), but can only output in JPEG, PNG, GIF or Bitmap formats.

### Command Arguments:

Argument	Type	Possible values	Description
ADDSUFFIX	/O	(no value)	Add a suffix to the output filename when resizing images. The suffix used

			<p>indicates the new image size. If the image is not resized, no suffix is added.</p> <p><i>Example:Image CONVERT=jpg WIDTH=1024 HEIGHT=768 ADDSUFFIX</i></p>
		<suffix>	<p>Adds the specified suffix to the output filename.</p> <p><i>Example:Image CONVERT=jpg WIDTH=128 HEIGHT=128 ADDSUFFIX=thumb</i></p> <p>The specified suffix is used provided a new image is written, but there are cases where no image will be written by default, such as when converting images in-place (not to a separate destination directory) and the source image already matches the specified criteria. If you need to ensure that a second copy of the image is created no matter what, using the specified suffix to modify its name, then you should prefix the suffix with the keyword <b>always</b>:</p> <p><i>Example:Image CONVERT=jpg WIDTH=128 HEIGHT=128 ADDSUFFIX=always:thumb</i></p>
AS	/K	<output filename>	<p>Specify the output filename when converting images. By default the output filename is the same as the input filename, with the possibility of an additional suffix (with the <b>ADDSUFFIX</b> argument) and a different file extension if the image has been converted to a different format.</p> <p><i>Example:Image CONVERT=jpg AS=thumbnail.jpg WIDTH=128 HEIGHT=128</i></p>
BACKGROUND	/K	<r>, <g>, <b> (dec) #rrggbb (hex)	<p>When an image with an alpha channel (transparency) is converted to a format that doesn't support the alpha channel, this argument is used to specify the background color that replaces the transparent area. The color can be specified in either decimal or hex format.</p>

			<i>Example:Image</i> <b>CONVERT=jpg BACKGROUND=#ff8000</b>
CONVERT	/O	(no value)	Displays the <a href="#">image conversion</a> dialog in interactive mode when no other arguments are provided.  <i>Example:Image</i> <b>CONVERT</b>
		<b>jpg</b>	Automates the image conversion function; the converted image will be saved in JPEG format.  <i>Example:Image</i> <b>CONVERT=jpg</b>
		<b>png</b>	The converted image will be saved in PNG format.  <i>Example:Image</i> <b>ROTATE=EXIF CONVERT=png</b>
		<b>gif</b>	The converted image will be saved in GIF format.  <i>Example:Image</i> <b>WIDTH=128 HEIGHT=128 CONVERT=gif</b>
		<b>bmp</b>	The converted image will be saved in BMP format.  <i>Example:Image</i> <b>CONVERT=bmp</b>
FLICKRACCOUNT	/K	<account name>	When used with <b>SYNCPHOTOS=flickr</b> , specifies the Flickr account to <a href="#">synchronize</a> . By default all configured accounts are synchronized.  <i>Example:Image</i> <b>SYNCPHOTOS=flickr FLICKRACCOUNT gpsoftware</b>
FLIP	/K	<b>h</b>	Flip (mirror) the image horizontally.  <i>Example: Image</i> <b>FLIP=h</b>
		<b>v</b>	Flip the image vertically.  <i>Example:Image</i> <b>FLIP=v ROTATE=90 HERE</b>
FROM	/M	<filename> ...	Specify the image file or files to operate on. Without this argument the command will operate on all currently selected files. This is the default argument for the Image command and so you do not need

			<p>to specify the FROM keyword. Remember that if the filename contains a space it needs to be enclosed in quotes.</p> <p><i>Example: <b>Image CONVERT=png FROM C:\MyPhotos\*.jpg HERE</b></i></p>
HEIGHT	/K/N	<height>	<p>Resize the image to the specified height.</p> <p><i>Example: <b>Image HEIGHT=768 PRESERVEASPECTRATIO CONVERT=jpg</b></i></p>
HERE	/S	(no value)	<p>Write converted images to the source folder. Without this argument converted images are written to the current destination folder.</p> <p><i>Example: <b>Image CONVERT=bmp WIDTH=128 HEIGHT=128 ADDSUFFIX HERE</b></i></p>
LOCATE	/K	<keyword>	<p>Locates the real-world position of the selected image file from its embedded GPS information, using a third-party mapping service. With the exception of Google Earth (which must be installed on your machine), all services open a web browser.</p> <p>The list of location services is configurable via <a href="#">Preferences / Miscellaneous / Advanced: image_locate_services</a>. Several services are defined by default, these are listed below.</p> <p>If the selected image does not have GPS information this command will have no effect.</p> <p><i>Example: <b>Image LOCATE=&lt;keyword&gt;</b></i></p>
		<b>bing</b>	<p>Locates the image using Bing maps (will open in a web browser).</p> <p><i>Example: <b>Image LOCATE=bing</b></i></p>
		<b>google</b>	<p>Locates the image using Google maps (will open in a web browser).</p>

			<i>Example:</i> <b>Image LOCATE=google</b>
		<b>googleearth</b>	Locates the image using Google Earth (the software must be installed on your computer for this to work).  <i>Example:</i> <b>Image LOCATE=googleearth</b>
		<b>osm</b>	Locates the image using Open Street Map (will open in a web browser).  <i>Example:</i> <b>Image LOCATE=osm</b>
NOENLARGE	/S	(no value)	Prevents images from being enlarged if the resize operation would otherwise cause this. Selected images that are already smaller than the specified size will remain untouched.  <i>Example:</i> <b>Image CONVERT=jpg WIDTH=1024 HEIGHT=768 NOENLARGE</b>
NOLOSSLESS	/S	(no value)	Disables the ability of Opus to perform lossless JPEG rotation. Normally Opus will rotate JPEG images losslessly if possible, but you may specifically want to recompress the image to a lower quality (to make it smaller) and this keyword allows you to do that. You can also use this without performing a rotation, if all you want to do is recompress a JPEG image to a different quality setting.  <i>Example:</i> <b>Image CONVERT=jpg QUALITY=50 ROTATE=EXIF NOLOSSLESS</b>
NOREDUCE	/S	(no value)	Prevents images from being reduced in size if the resize operation would otherwise cause this. Selected images that are already larger than the specified size will remain untouched.  <i>Example:</i> <b>Image CONVERT=jpg WIDTH=800 HEIGHT=600 NOREDUCE</b>
NOUSEIMAGEDATA	/S	(no value)	When used with the <b>CONVERT</b> argument (in the standalone image viewer), overrides the <b>@useimagedata</b>

			<p>command modifier and makes the image converter load the image from disk rather than obtaining it from the viewer.</p> <p><i>Example:Image CONVERT NOUSEIMAGEDATA</i></p>
PERCENT	/K/N	<resize percent>	<p>Resize the image to the specified percentage of the original size. This can enlarge images as well as reduce them.</p> <p><i>Example:Image CONVERT=jpg PERCENT=125</i></p>
PRESERVEASPECTRATIO	/S	(no value)	<p>Preserve the original aspect ratio when resizing images. The output width or height will be automatically adjusted to ensure the aspect ratio is maintained. Using this switch means you can resize an image by just supplying a new width or a new height - this missing dimension will be calculated automatically.</p> <p><i>Example:Image CONVERT=png WIDTH=1280 PRESERVEASPECTRATIO HERE</i></p>
PRESERVEDATE	/S	(no value)	<p>When converting images, this option preserves the <i>creation</i> and <i>last modified</i> timestamps of the original file. By default, when this option is not specified, the <i>last modified</i> timestamp will be updated to the current time, as will the <i>creation</i> timestamp if the operation creates a new file. (Operations which convert an existing file "in place", overwriting the original with <b>REPLACE HERE</b>, will preserve the <i>creation</i> timestamp regardless of this option.)</p> <p><i>Example:Image ROTATE=EXIF REPLACE HERE PRESERVEDATE</i></p>
QUALITY	/K/N	<quality>	<p>Specify the quality (1 - 100) when an image is saved in JPEG format.</p> <p><i>Example:Image CONVERT=jpg QUALITY=10 NOLOSSLESS</i></p>
REPLACE	/O	(no value)	<p>Automatically replaces existing files in the destination folder. Use this in conjunction with the <b>HERE</b> argument to</p>

			<p>convert an image "in-place". This only applies if the output filename is the same as the input filename - if the output filename has changed (via <b>ADDSUFFIX</b>, etc) and the file already exists, you will still be prompted for confirmation.</p> <p><i>Example:Image ROTATE=EXIF REPLACE</i></p>
		<b>always</b>	<p>Always replace existing files, even if the output filename has changed.</p> <p><i>Example:Image PERCENT=50 ADDSUFFIX HERE REPLACE=always</i></p>
		<b>readonly</b>	<p>Replace existing files, even if they are read-only, without prompting. (Has no effect if read-only prompts are turned off in Preferences.)</p> <p><i>Example:Image PERCENT=50 REPLACE=readonly</i></p> <p>Can be combined with <b>always</b>:</p> <p><i>Example:Image PERCENT=50 REPLACE=always,readonly</i></p>
ROTATE	/K	<angle>	<p>Rotate the image the specified angle (in degrees). Positive values rotate clockwise, negative values counter-clockwise.</p> <p><i>Example:Image ROTATE=90 HERE REPLACE</i></p>
		<b>EXIF</b>	<p>Uses the rotation (orientation) information stored in the images' EXIF tags to rotate the image. The effect of this is to negate the original orientation of the camera, resulting in a "right way up" image. If the selected image does not have an EXIF rotation tag this operation has no effect.</p> <p><i>Example:Image ROTATE=EXIF HERE REPLACE</i></p>
		<b>RESET</b>	<p>Does not actually rotate the image data, but will clear out the rotation (orientation) field from the images' EXIF tags.</p>

			<i>Example:Image ROTATE=RESET HERE REPLACE</i>
SYNCPHOTOS	/K	<b>flickr</b>	Initiate online photo synchronization. Currently the only supported service is <a href="#">Flickr</a> . Use the <b>FLICKRACCOUNT</b> argument to specify an account to synchronize. You can configure the Flickr synchronization function from the <a href="#">Photo Sharing / Flickr</a> page in Preferences.  <i>Example:Image SYNCPHOTOS=flickr</i>
TO	/K	<destination path>	Specifies the destination path for converted images. Remember to enclose the path in quotes if it contains spaces. If not provided, and the <b>HERE</b> argument is not specified, the current destination file display will be used as the target path.  <i>Example:Image CONVERT=jpg WIDTH=80 HEIGHT=80 TO "C:\Photos\Image Thumbnails"</i>
WIDTH	/K/N		Resize the image to the specified width.  <i>Example:Image WIDTH=1024 PRESERVEASPECTRATIO CONVERT=jpg</i>

## Join

The **Join** internal command can be used to join multiple files together into a single larger file. It is mainly used when you have a file that has been split into multiple parts, say for transmission via email.

### Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Displays the <a href="#">Join Files</a> dialog. All currently selected files will be added to the list; you can re-order this list, remove items from it and add additional items to it from within the dialog.

			<i>Example: <b>Join</b></i>
FROM	/M	<filename> ...	Specify the files that will be joined. Remember to enclose each file in quotes if it contains spaces.  <i>Example: <b>Join /temp/part1.bin /temp/part2.bin</b></i>
HERE	/S	(no value)	Writes the joined file to the source folder instead of the destination.  <i>Example: <b>Join HERE</b></i>
TO	/K	<output file>	Specifies the name of the joined file.  <i>Example: <b>Join part1.bin part2.bin part3.bin TO original.jpg HERE</b></i>

## Marker

The **Marker** internal command is used to display toolbar buttons and menu items that are added dynamically by third-party namespace extensions (it acts as a [dynamic button](#)). For example, an FTP namespace extension may add buttons to the toolbar to switch between ASCII and binary transfer modes.

Explorer allows a third-party namespace extension to totally replace the toolbar and menu contents. However, Directory Opus gives full control to the user over the state of the toolbars, and therefore the **Marker** command is needed to mark the place where these commands are to appear.

The system that allows namespace extensions to add toolbar buttons has been deprecated by Microsoft in later versions of Windows, so these days this command is often not needed - however it remains for backwards compatibility reasons.

## Command Arguments:

Argument	Type	Possible values	Description
ID	/K/N	<id>	Send a namespace-specific command direct to the namespace folder currently displayed in the active Lister. You need to know the exact command ID that the namespace uses, which can be hard to determine. Ordinarily you will never use this option directly - Opus uses it when generating the dynamic buttons that are added by the <b>Marker</b> command.  <i>Example: <b>Marker ID 1002</b></i>
MENU	/K	<menu name>	Marks the spot where namespace-specific menu items will be displayed. The <menu name> parameter is a keyword corresponding to one of the standard Explorer

			menus ( <b>file, edit, view, tools, help</b> ), <b>other</b> (commands not fitting into any of those menus) and <b>all</b> (all namespace-specific menu items).  <i>Example: Marker MENU=file</i>
TOOLBAR	/S	(no value)	Marks the spot where namespace-specific toolbar buttons are displayed.  <i>Example: Marker TOOLBAR</i>

## Play

The internal **Play** command is used to [play](#) selected sound files. This is a very simple built-in utility - it's not supposed to replace a dedicated sound player. It may also not support all audio formats - fundamentally, only **.wav** files are supported, although **.mp3** and other formats can also work if suitable codecs are installed for them.

### Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Plays selected sound files in the current folder using the internal <a href="#">sound player</a> .  <i>Example: Play</i>
FILE	/M	<filename> ...	Plays the specified sound file or files. This lets you play a specific sound - for example, you might add this command to the end of a button to signify that a function had completed. You would normally want to use the <b>QUIET</b> argument as well in these cases.  Remember that if the filename contains spaces you need to enclose it in quotes. This is the default argument for the <b>Play</b> command so you do not need to provide the <b>FILE</b> keyword.  <i>Example: Play C:\Data\Sounds\Complete.wav QUIET</i>
QUIET	/S	(no value)	Plays the specified (or selected) sounds without displaying the Play dialog.  <i>Example: Play QUIET</i>

## Prefs

The **Prefs** internal command can be used to:

- Display the [Preferences](#) dialog which lets you change most configuration options
- Display the [Customize](#) dialog to edit toolbars, menus and hotkeys
- Display the [File Types](#) dialog to configure context menus, double-click actions and other file-type specific settings
- Display and edit the [FTP Address Book](#)
- [Backup and restore](#) your configuration
- Load and save [Lister layouts](#)
- Update the [Default Lister](#) settings
- Load and save [Lister styles](#)
- Load and save [Lister themes](#)
- Manage [VFS plugins](#)

### Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Displays the <a href="#">Preferences</a> dialog.  <i>Example: Prefs</i>
ADDBACKGROUND	/S	(no value)	Adds selected images to the list on the <a href="#">Display / Images</a> Preferences page. This makes them available to be used as background images in Listers and toolbars. For example, you could add this command to the context menu for the <b>Images</b> <a href="#">file type group</a> .  <i>Example: Prefs ADDBACKGROUND</i>
ADDFTPSITE	/S	(no value)	Adds the currently connected FTP site as a new entry in the <a href="#">FTP Address Book</a> . If you are not currently viewing an FTP directory this command has no effect.  <i>Example: Prefs ADDFTPSITE</i>
BACKUP	/O	(no value)	Automates the configuration backup process. By default all your configuration settings, toolbars, menus and hotkeys are included in the backup, but extra data like images and sounds are not. The optional values for this argument can be used to control which extra data is included in the backup. Use the <b>TO</b> argument to specify the name of the backup file, and the <b>PASSWORD</b> , <b>DESC</b> and <b>QUIET</b> arguments provide additional control.

			<p><i>Example:</i> <b>Prefs BACKUP TO="/desktop/PrefsBackup"</b></p>
		<b>images</b>	<p>When automating a configuration backup, includes image files in your backup (corresponds to the <b>Backup images</b> option in the <a href="#">Backup and Restore Configuration</a> wizard).</p> <p><i>Example:</i> <b>Prefs BACKUP=images TO="/desktop/PrefsBackup"</b></p>
		<b>sounds</b>	<p>Includes sound files in the backup (corresponds to the <b>Backup sounds</b> option).</p> <p><i>Example:</i> <b>Prefs BACKUP=sounds TO="/desktop/PrefsBackup"</b></p>
		<b>data</b>	<p>Include miscellaneous data in the backup (corresponds to the <b>Backup miscellaneous data</b> option).</p> <p><i>Example:</i> <b>Prefs BACKUP=data TO="/desktop/PrefsBackup"</b></p>
		<b>localstate</b>	<p>Include local state data (corresponds to the <b>Backup local state data</b> option).</p> <p><i>Example:</i> <b>Prefs BACKUP=data,localstate TO="/desktop/PrefsBackup"</b></p>
		<b>all</b>	<p>Backup everything (equivalent to <b>BACKUP=images,sounds,data,localstate</b>).</p> <p><i>Example:</i> <b>Prefs BACKUP=all TO="/desktop/PrefsBackup"</b></p>
BACKUPRESTORE	/S	(no value)	<p>Initiates the <a href="#">Backup and Restore Configuration</a> wizard, which lets you backup and restore your configuration, as well as export Opus to a USB drive. You can use the <b>TO</b> argument to override the default backup name, and the <b>PASSWORD</b> or <b>DESC</b> arguments to pre-supply a default password or descriptions when creating a backup.</p> <p><i>Example:</i> <b>Prefs BACKUPRESTORE</b>  <i>Example:</i> <b>Prefs BACKUPRESTORE TO="/desktop\Backup for %username% on {date yyyy-MM-dd}" PASSWORD="cat" DESC="Quick Backup."</b></p> <p>If you wish to automate the backup or restore process, without showing the interactive wizard, use the separate <b>BACKUP</b> or <b>RESTORE</b> arguments.</p>

CUSTOMIZE	/O	(no value)	Displays the <a href="#">Customize</a> dialog. The dialog will open on the page that was last used.  <i>Example: Prefs CUSTOMIZE</i>
		<page>	Displays the specified tab on the <b>Customize</b> dialog. Allowable values for this argument are <b>commands</b> , <b>toolbars</b> , <b>keys</b> and <b>menus</b> .  <i>Example: Prefs CUSTOMIZE=toolbars</i>
DESC	/K	<description>	Assign a description to a configuration backup (used with the <b>BACKUP</b> or <b>BACKUPRESTORE</b> argument). Remember that if the description string contains spaces you need to enclose it with quotes.  <i>Example: Prefs BACKUP TO="/desktop/PrefsBackup" DESC="My Opus Config"</i>
FILETYPES	/S	(no value)	Displays the <a href="#">File Types</a> dialog.  <i>Example: Prefs FILETYPES</i>
FROM	/K	<backup file>	Specifies the configuration backup file to restore (used with the <b>RESTORE</b> argument to automate the configuration restore process).  <i>Example: Prefs RESTORE FROM="/desktop/PrefsBackup"</i>
FTPSITES	/O	(no value)	Displays the <a href="#">FTP Address Book</a> .  <i>Example: Prefs FTPSITES</i>
		<site name>	Displays the <a href="#">FTP Address Book</a> and automatically selects the named site. If the site is in a sub-folder in the address book you need to provide the full path of the site.  <i>Example: Prefs FTPSITES="WorkServers\Production"</i>
KEYS	/S	(no value)	Displays the <b>Keys</b> page of the <b>Customize</b> dialog (equivalent to <b>Prefs CUSTOMIZE=keys</b> ).  <i>Example: Prefs KEYS</i>
LAYOUT	/K	<layout name>	Loads the named <a href="#">Lister layout</a> .  <i>Example: Prefs LAYOUT="Music Albums"</i>
LAYOUTCLOSELISTERS	/K	<b>yes</b>	When loading a layout with the <b>LAYOUT</b> argument, this overrides the <b>Close all existing Listers when loading this layout</b> flag set for the layout, and forces all existing Listers to close. When saving a layout with the <b>LAYOUTSAVE</b> argument, it sets the state

			of that flag within the layout.  <i>Example: Prefs LAYOUT="FTP Sync" LAYOUTCLOSELISTERS=yes</i>
		<b>no</b>	Does not close all existing Listers when loading a layout.  <i>Example: Prefs LAYOUT="Photo Album" LAYOUTCLOSELISTERS=no</i>
LAYOUTEDIT	/S	(no value)	Opens the Preferences dialog and displays the <a href="#">Layouts and Styles / Layouts</a> page (equivalent to <b>Prefs PAGE=layouts</b> ).  <i>Example: Prefs LAYOUTEDIT</i>
LAYOUTIGNOREFORMATS	/K	<b>yes</b>	When loading a layout with the <b>LAYOUT</b> argument, this overrides the <b>Ignore folder formats saved within this layout</b> flag set for the layout, and forces the layout's <a href="#">folder formats</a> to be ignored. When saving a layout with the <b>LAYOUTSAVE</b> argument, it sets the state of that flag within the layout.  <i>Example: Prefs LAYOUT="Photo Viewing" LAYOUTIGNOREFORMATS=yes</i>
		<b>no</b>	Does not ignore the layout's folder formats.  <i>Example: Prefs LAYOUT="Photos" LAYOUTIGNOREFORMATS=no</i>
LAYOUTIGNORETOOLBARS	/K	<b>yes</b>	When loading a layout with the <b>LAYOUT</b> argument, this overrides the <b>Ignore toolbars saved within this layout</b> flag set for the layout, and forces the layout's toolbars to be ignored, and the default toolbar set to be used instead. When saving a layout with the <b>LAYOUTSAVE</b> argument, it sets the state of that flag within the layout.  <i>Example: Prefs LAYOUT="Photos" LAYOUTIGNORETOOLBARS=yes</i>
		<b>no</b>	Does not ignore the layout's toolbars.  <i>Example: Prefs LAYOUT="Photos" LAYOUTIGNORETOOLBARS=no</i>
LAYOUTLIST	/S	(no value)	Displays a list of your saved <a href="#">Lister layouts</a> (acts as a <a href="#">dynamic button</a> ). Selecting an item from the generated list loads the specified layout. The <b>LAYOUTCLOSELISTERS</b> , <b>LAYOUTIGNOREFORMATS</b> and <b>LAYOUTMOUSERELATIVE</b> flags can be used in conjunction with this argument to control the

			behaviour of the generated buttons.  <i>Example: Prefs LAYOUTLIST LAYOUTCLOSELISTERS=yes</i>
LAYOUTMOUSERELATIVE	/K	yes	When loading a layout with the <b>LAYOUT</b> argument, this overrides the <b>Open layout relative to the monitor the mouse is currently on</b> flag set for the layout, and forces the layout's position to be relative to the mouse. When saving a layout with the <b>LAYOUTSAVE</b> argument, it sets the state of that flag within the layout.  <i>Example: Prefs LAYOUT="Find" LAYOUTMOUSERELATIVE=yes</i>
		no	Does not open the layout relative to the mouse pointer.  <i>Example: Prefs LAYOUT="BackupFiles" LAYOUTMOUSERELATIVE=no</i>
LAYOUTNAME	/K	<layout name>	Specifies a name when saving a <a href="#">layout</a> using the <b>LAYOUTSAVE</b> argument. If not provided Opus will prompt for a name for the new layout.  <i>Example: Prefs LAYOUTNAME="My Layout" LAYOUTSAVE</i>
LAYOUTSAVE	/O	(no value)	Saves the currently open Listers as a <a href="#">layout</a> . You can use the <b>LAYOUTNAME</b> argument to specify the layout name. You can also use the <b>LAYOUTCLOSELISTERS</b> , <b>LAYOUTIGNOREFORMATS</b> and <b>LAYOUTMOUSERELATIVE</b> arguments to control the state of those flags for the saved layout.  <i>Example: Prefs LAYOUTSAVE</i>
		single	Saves only the active Lister as a layout. Any other currently open Listers are not included.  <i>Example: Prefs LAYOUTSAVE=single LAYOUTNAME="CurrentLister"</i>
		noupdatesettings	If you do not use the <b>LAYOUTNAME</b> argument to provide a name for the layout, Opus will display a dialog prompting for the name - this dialog also contains various options for the layout (ignore formats, mouse relative, etc). Changes you make to those options will be saved as the default settings for the <b>LAYOUTSAVE</b> function unless you specify the <b>noupdatesettings</b> value.  <i>Example: Prefs LAYOUTSAVE=noupdatesettings</i>

			<b>LAYOUTMOUSERELATIVE=yes</b> <b>LAYOUTNAME="My Layout"</b>
		<b>updatecurrent</b>	<p>If the current lister is already part of a layout, that layout will be updated and you will not be prompted for a layout name or any further options.</p> <p><i>Example: Prefs LAYOUTSAVE=updatecurrent</i></p>
LAYOUTTHISLISTER	/O	(no value)	<p>In conjunction with the <b>LAYOUT</b> argument, this applies the settings from the specified layout to the current Lister instead of opening a new Lister. If no value is given, all the settings from the layout are used - otherwise, only the specified settings are used.</p> <p><i>Example: Prefs LAYOUT="PhotoViewing" LAYOUTTHISLISTER</i></p>
		<b>size</b>	<p>Use the window size from the specified layout.</p> <p><i>Example: Prefs LAYOUT="SmallLister" LAYOUTTHISLISTER=size</i></p>
		<b>pos</b>	<p>Use the window position from the specified layout.</p> <p><i>Example: Prefs LAYOUT="NiceSizedLister" LAYOUTTHISLISTER=pos,size</i></p>
		<b>paths</b>	<p>Use the paths/tabs (and folder formats) from the specified layout.</p> <p><i>Example: Prefs LAYOUT="CurrentWork" LAYOUTTHISLISTER=paths</i></p> <p>You can also use <b>LAYOUTIGNOREFORMATS</b> in conjunction with <b>LAYOUTTHISLISTER=paths</b> to override whether or not the layout's folder formats are applied (which is normally determined by a flag you can set when saving and editing each layout).</p> <p><i>Example: Prefs LAYOUT="My Layout" LAYOUTTHISLISTER=paths LAYOUTIGNOREFORMATS=yes</i></p>
NOSCRIPT	/S	(no value)	<p>Allows a <a href="#">script</a> to run the <b>Prefs LAYOUT</b> command without triggering other scripts (or itself). Adding the <b>NOSCRIPT</b> argument disables the <a href="#">OnBeforeFolderChange</a>, <a href="#">OnAfterFolderChange</a>, <a href="#">OnOpenTab</a> and <a href="#">OnOpenLister</a> events that would otherwise be triggered by opening a Lister layout.</p> <p><i>Example: Prefs LAYOUT="My Layout" NOSCRIPT</i></p>

PAGE	/K	<prefs page>	<p>Opens the Preferences dialog to the specified page. If this argument is not supplied the Preferences dialog will open to the last-used page. Valid page keywords are:</p> <table border="1"> <tr><td>advanced</td><td>aliases</td><td>archivecont ext</td></tr> <tr><td>assignedlabels</td><td>arcoptions</td><td>autoloadin g</td></tr> <tr><td>colors</td><td>copyattr</td><td>copyopts</td></tr> <tr><td>dblclkfiles</td><td>dblclktask bar</td><td>deflister</td></tr> <tr><td>deleting</td><td>details</td><td>display</td></tr> <tr><td>displaymodetoo lbars</td><td>doubleclik k</td><td>email</td></tr> <tr><td>explorerrep</td><td>faves</td><td>fayt</td></tr> <tr><td>fdb</td><td>fields</td><td>filedisplay</td></tr> <tr><td>fileop</td><td>filterbar</td><td>filters</td></tr> <tr><td>flickr</td><td>folderbeha vior</td><td>folderdispl ay</td></tr> <tr><td>folderformats</td><td>foldertabs</td><td>globalfilter s</td></tr> <tr><td>iconsets</td><td>images</td><td>jumplist</td></tr> <tr><td>labels</td><td>language</td><td>layouts</td></tr> <tr><td>logging</td><td>metadata</td><td>misc</td></tr> <tr><td>mouse</td><td>plugins</td><td>powerdetai ls</td></tr> <tr><td>powermodebut tons</td><td>progress</td><td>proxy</td></tr> <tr><td>recent</td><td>rename</td><td>scripts</td></tr> <tr><td>smartfaves</td><td>sounds</td><td>startup</td></tr> <tr><td>status</td><td>styles</td><td>tabappeara nce</td></tr> <tr><td>tabgroups</td><td>thumbnail s</td><td>tiles</td></tr> <tr><td>toolbarappeara nce</td><td>toolbars</td><td>toolbarsets</td></tr> <tr><td>transitionanim ations</td><td>treeactions</td><td>treeappear ance</td></tr> <tr><td>treebehavior</td><td>treeconten ts</td><td>updates</td></tr> <tr><td>vfsplugins</td><td>viewer1</td><td>viewer2</td></tr> <tr><td>viewer3</td><td>viewerpan e</td><td>virtualfolde rs</td></tr> </table>	advanced	aliases	archivecont ext	assignedlabels	arcoptions	autoloadin g	colors	copyattr	copyopts	dblclkfiles	dblclktask bar	deflister	deleting	details	display	displaymodetoo lbars	doubleclik k	email	explorerrep	faves	fayt	fdb	fields	filedisplay	fileop	filterbar	filters	flickr	folderbeha vior	folderdispl ay	folderformats	foldertabs	globalfilter s	iconsets	images	jumplist	labels	language	layouts	logging	metadata	misc	mouse	plugins	powerdetai ls	powermodebut tons	progress	proxy	recent	rename	scripts	smartfaves	sounds	startup	status	styles	tabappeara nce	tabgroups	thumbnail s	tiles	toolbarappeara nce	toolbars	toolbarsets	transitionanim ations	treeactions	treeappear ance	treebehavior	treeconten ts	updates	vfsplugins	viewer1	viewer2	viewer3	viewerpan e	virtualfolde rs
advanced	aliases	archivecont ext																																																																												
assignedlabels	arcoptions	autoloadin g																																																																												
colors	copyattr	copyopts																																																																												
dblclkfiles	dblclktask bar	deflister																																																																												
deleting	details	display																																																																												
displaymodetoo lbars	doubleclik k	email																																																																												
explorerrep	faves	fayt																																																																												
fdb	fields	filedisplay																																																																												
fileop	filterbar	filters																																																																												
flickr	folderbeha vior	folderdispl ay																																																																												
folderformats	foldertabs	globalfilter s																																																																												
iconsets	images	jumplist																																																																												
labels	language	layouts																																																																												
logging	metadata	misc																																																																												
mouse	plugins	powerdetai ls																																																																												
powermodebut tons	progress	proxy																																																																												
recent	rename	scripts																																																																												
smartfaves	sounds	startup																																																																												
status	styles	tabappeara nce																																																																												
tabgroups	thumbnail s	tiles																																																																												
toolbarappeara nce	toolbars	toolbarsets																																																																												
transitionanim ations	treeactions	treeappear ance																																																																												
treebehavior	treeconten ts	updates																																																																												
vfsplugins	viewer1	viewer2																																																																												
viewer3	viewerpan e	virtualfolde rs																																																																												

			windowsint	winehotkey	zip
			<i>Example: Prefs PAGE=transitionanimations</i>		
PASSWORD	/K	<password>	Encrypt a configuration backup (used with the <b>BACKUP</b> or <b>BACKUPRESTORE</b> argument). You will need to enter the password when you restore the backup.  <i>Example: Prefs BACKUP TO="/desktop/PrefsBackup" PASSWORD="abc123"</i>		
QUIET	/S	(no value)	Prevents the <b>Backup and Restore Configuration</b> wizard from appearing when the backup or restore are automated (when used with the <b>BACKUP</b> and <b>RESTORE</b> arguments). If specified when restoring a configuration, Opus will be restarted automatically - you can use the <b>Close AUTOLISTER</b> command to control whether a new Lister is opened when Opus restarts this way.  <i>Example: Prefs RESTORE FROM="/desktop/PrefsBackup" QUIET</i>		
RESTORE	/O	(no value)	Automates the configuration restore process. By default all the configuration settings, toolbars, menus and hotkeys from the backup will be restored, but extra data like images and sounds are not. The optional values for this argument can be used to control which extra data is included in the restore. Use the <b>FROM</b> argument to specify the name of the backup file. If the backup file was encrypted you can provide the password with the <b>PASSWORD</b> argument. The <b>QUIET</b> argument prevents any user interface from being displayed, and Opus will automatically restart when the restore is complete.  <i>Example: Prefs RESTORE FROM="/desktop/PrefsBackup"</i>		
		<b>images</b>	Restores images from the configuration backup (corresponds to the <b>Restore images</b> option in the <a href="#">Backup and Restore Configuration</a> wizard).  <i>Example: Prefs RESTORE=images FROM="/desktop/PrefsBackup"</i>		
		<b>sounds</b>	Restores sounds from the configuration backup (corresponds to the <b>Restore sounds</b> option in the		

			wizard).  <i>Example: Prefs RESTORE=images,sounds FROM="/desktop/PrefsBackup"</i>
		<b>data</b>	Restores miscellaneous data (corresponds to the <b>Restore miscellaneous data</b> option in the wizard).  <i>Example: Prefs RESTORE=data FROM="/desktop/PrefsBackup" QUIET</i>
		<b>localstate</b>	Restores local state data (corresponds to the <b>Restore local state data (window positions, etc)</b> option in the wizard).  <i>Example: Prefs RESTORE=data,localstate FROM="/desktop/PrefsBackup" QUIET</i>
		<b>replace</b>	Replaces your existing configuration completely (corresponds to the similarly named option in the wizard). If you select this, your existing configuration is deleted before the restore is done. Without this option, the restored configuration files are written over the top of your existing configuration - but the effect of this is to merge things like toolbars and images with different names.  <i>Example: Prefs RESTORE=replace,images,sounds FROM="/desktop/PrefsBackup"</i>
		<b>all</b>	Restores all items in the configuration backup (the equivalent of <b>RESTORE=images,sounds,data,localstate</b> ).  <i>Example: Prefs RESTORE=replace,all FROM="/desktop/PrefsBackup"</i>
SETDEFAULTLISTER	/O	(no value)	Saves the currently active Lister as the <a href="#">Default Lister</a> .  <i>Example: Prefs SETDEFAULTLISTER</i>
		<b>force</b>	Suppress the confirmation and success prompts when setting the Default Lister manually.  <i>Example: Prefs SETDEFAULTLISTER=force</i>
		<b>quiet</b>	Suppress the success prompt when setting the Default Lister.  <i>Example: Prefs SETDEFAULTLISTER=quiet</i>
SHOWICONS	/S	(no value)	Displays icons for dynamic lists generated by the <b>LAYOUTLIST</b> , <b>STYLELIST</b> and <b>VFSPLUGINLIST</b> arguments.

			<i>Example: Prefs LAYOUTLIST SHOWICONS</i>
STYLE	/K	<style name>	Applies the specified <a href="#">style</a> to the current Lister.  <i>Example: Prefs STYLE="Dual Horizontal"</i>
		<b>^prev</b>	This refers to the initial appearance of the Lister when it was opened, and lets you return to that state as if it were a normal style.  <i>Example: Prefs STYLE=^prev</i>
STYLEEDIT	/S	(no value)	Displays the <a href="#">Layouts and Styles / Styles</a> page in Preferences (equivalent to <b>Prefs PAGE=styles</b> ).  <i>Example: Prefs STYLEEDIT</i>
STYLELIST	/O	(no value)	Displays a list of your saved <a href="#">Lister styles</a> (acts as a <a href="#">dynamic button</a> ). Selecting an item from the generated list applies the specified style to the current Lister.  <i>Example: Prefs STYLELIST</i>
		<b>showprevious</b>	Includes the special "Previous" style in the generated list. This refers to the initial appearance of the Lister when it was opened, and lets you return to that state as if it were a normal style.  <i>Example: Prefs STYLELIST=showprevious</i>
STYLESAVE	/O	(no value)	Saves the current appearance of the active Lister as a style.  <i>Example: Prefs STYLESAVE</i>
		<style name>	Specifies the name for the style to create. Opus will prompt for a name if one is not provided.  <i>Example: Prefs STYLESAVE="My Style"</i>
THEMES	/S	(no value)	Displays the <a href="#">Lister Themes</a> dialog.  <i>Example: Prefs THEMES</i>
TO	/K	<backup file>	Specifies the filename for the configuration backup (used with the <b>BACKUP</b> or <b>BACKUPRESTORE</b> argument).  <i>Example: Prefs BACKUP TO="/mydocuments/OpusPrefsBackup {date yyyy-MM-dd}"</i>
TOOLBARS	/S	(no value)	Displays the <b>Toolbars</b> tab in the <b>Customize</b> dialog (equivalent to <b>Prefs CUSTOMIZE=toolbars</b> ).  <i>Example: Prefs TOOLBARS</i>

VFSPLUGINABOUT	/K	<plugin name>	Displays the <b>About</b> dialog for the specified <a href="#">VFS plugin</a> .  <i>Example: Prefs VFSPLUGINABOUT opus7zip.dll</i>
VFSPLUGINCONFIG	/K	<plugin name>	Displays the <b>Configuration</b> dialog for the specified VFS plugin.  <i>Example: Prefs VFSPLUGINCONFIG opus7zip.dll</i>
VFSPLUGINDISABLE	/K	<plugin name>	Enable or disable the specified VFS plugin. If the plugin is currently enabled it will be disabled, and vice versa.  <i>Example: Prefs VFSPLUGINDISABLE opus7zip.dll</i>
		<b>enable</b>	Enable the specified VFS plugin.  <i>Example: Prefs VFSPLUGINDISABLE opus7zip.dll,enable</i>
		<b>disable</b>	Disable the specified VFS plugin.  <i>Example: Prefs VFSPLUGINDISABLE opus7zip.dll,disable</i>
VFSPLUGINLIST	/S	(no value)	Displays a list of the installed VFS plugins (acts as a <a href="#">dynamic button</a> ). Each plugin in the list has a sub-menu containing <i>about</i> , <i>configure</i> , and <i>enable/disable</i> commands.  <i>Example: Prefs VFSPLUGINLIST</i>
VFSPLUGINMANAGER	/S	(no value)	Displays the <a href="#">Zip &amp; Other Archives / Archive and VFS Plugins</a> page in Preferences (equivalent to <b>Prefs PAGE=vfsplugins</b> ).  <i>Example: Prefs VFSPLUGINMANAGER</i>

## Print

The **Print** internal command can be used to:

- Print photos and other images using the Windows Photo Printing wizard
- Print supported image files directly
- Print other files via their registered handlers
- Generate a folder listing and either print it, save it to a file or copy it to the clipboard
- Export a folder listing in **CSV** format for import into Excel, etc.
- Display a list of installed printers and let you modify the default printer

## Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Prints selected files. If only image files are selected (or no files at all are selected), the Windows Photo Printing wizard will be invoked. Non-image files will be printed via their registered print handler.  <i>Example: <b>Print</b></i>
AS	/K	txt	Use plain text format when printing a folder to disk or the clipboard.  <i>Example: <b>Print FOLDER TO=clip AS=txt QUIET</b></i>
		csv	Use CSV (comma-separated value) format when printing a folder to disk or the clipboard.  <i>Example: <b>Print FOLDER TO /desktop/dirprint.csv AS=csv QUIET</b></i>
		tab	Use tab-separated format when printing a folder to disk or the clipboard.  <i>Example: <b>Print FOLDER TO clip AS tab QUIET</b></i>
CALCSIZES	/K	yes	Calculate folder sizes when printing a folder.  <i>Example: <b>Print FOLDER TO clip CALCSIZES=yes QUIET</b></i>
		no	Do not calculate folder sizes.  <i>Example: <b>Print FOLDER TO clip CALCSIZES=no QUIET</b></i>
DEFAULTLIST	/S	(no value)	Displays a generated list of installed printers (acts as a <a href="#">dynamic button</a> ). You can use this list to change the default printer, and right-click the items in the list to display the printer's context menu. You can also print a file by dropping it on the generated button for a printer.  <i>Example: <b>Print DEFAULTLIST</b></i>
ENCODING	/K	ansi	When printing a folder listing to a file, specifies that <b>ANSI</b> encoding should be used. This is the default encoding type initially, but if the <b>Print Folder</b> dialog is used in interactive mode it will remember the last encoding type manually selected. Using this argument lets you override the last used encoding

			<p>type.</p> <p><i>Example:</i> <b>PRINT FOLDER TO /desktop/dirprint.txt ENCODING=ansi QUIET</b></p>
		<b>utf8</b>	<p>Sets the encoding type to <b>UTF8</b> with a BOM (Byte Order Mark).</p> <p><i>Example:</i> <b>PRINT FOLDER TO /desktop/dirprint.txt ENCODING=bom QUIET</b></p>
		<b>utf8nobom</b>	<p>Sets the encoding type to <b>UTF8</b> without a BOM.</p> <p><i>Example:</i> <b>PRINT FOLDER TO /desktop/dirprint.txt ENCODING=utf8nobom</b></p>
<b>FILTER</b>	<b>/K</b>	<i>&lt;filter&gt;</i>	<p>Use a <a href="#">filter</a> when printing the contents of sub-folders (via the <b>FLATVIEW</b> argument). This can be the name of a filter you have previously created via the <a href="#">File Operations \ Filters</a> page in Preferences, or it can be a simple <a href="#">wildcard pattern</a> to filter by filename.</p> <p><i>Example:</i> <b>Print FOLDER FILTER *.jpg FLATVIEW=nofolders QUIET</b></p>
<b>FLATVIEW</b>	<b>/K</b>	<b>no</b>	<p>Do not print the contents of sub-folders when printing a folder listing.</p> <p><i>Example:</i> <b>Print FOLDER FLATVIEW=no QUIET</b></p>
		<b>mixed</b>	<p>Prints the contents of sub-folders in "mixed" mode. This mixes files and folders together in a flat list.</p> <p><i>Example:</i> <b>Print FOLDER FLATVIEW=mixed TO clip QUIET</b></p>
		<b>nofolders</b>	<p>Prints the contents of sub-folders in "mixed - no folders mode". All files from sub-folders are listed, but the folders themselves are not shown.</p> <p><i>Example:</i> <b>Print FOLDER FLATVIEW=nofolders QUIET</b></p>
		<b>grouped</b>	<p>Prints the contents of sub-folders in "grouped" mode. Files and folders are indented to reflect the tree hierarchy.</p> <p><i>Example:</i> <b>Print FOLDER TO /desktop/dirtree.txt FLATVIEW=grouped QUIET</b></p>
<b>FOLDER</b>	<b>/O</b>	<i>(no value)</i>	<p>Displays the <a href="#">Print Folder</a> dialog, which lets you print or export the contents of the current folder</p>

			displayed in the Lister.  <i>Example:Print FOLDER</i>
		<b>selected</b>	Only selected files in the current folder will be printed by the Print Folder function.  <i>Example:Print FOLDER=selected</i>
		<path>	Specify the folder path to print.  <i>Example:Print FOLDER /desktop TO clip QUIET</i>
FONT	/K	<name>,<size>	Specify the font to use when printing to the printer.  <i>Example:Print FOLDER FONT Arial,20 QUIET</i>
FORMAT	/K	<format name>	Use the specified favorite folder format to control the appearance of the printed format. The named format must first have been created from the <a href="#">Folders / Folder Formats</a> page in Preferences.  As well as the name of a favorite format, this argument accepts the following special keywords: <ul style="list-style-type: none"> <li>• <b>!factory</b>: Reset to factory defaults.</li> <li>• <b>!user</b>: Reset to the user default format.</li> <li>• <b>!default</b>: Resets to Folder Type format applicable to current folder.</li> <li>• <b>!folder</b>: Resets to the format for the folder that a brand new Lister would use.</li> <li>• <b>!current</b>: Uses the current format shown in the Lister.</li> </ul> <i>Example:Print FOLDER FORMAT PrintDirFmt QUIET</i>
HEADER	/K	<b>top</b>	Print a header at the top of each page (or when printing to disk or the clipboard, at the top of the listing).  <i>Example:Print FOLDER HEADER=top QUIET</i>
		<b>bottom</b>	Print a footer at the bottom of each page (or the bottom of the listing).  <i>Example:Print FOLDER HEADER=bottom QUIET</i>
		<b>both</b>	Print both a header and a footer.  <i>Example:Print FOLDER HEADER=both QUIET</i>

		<b>none</b>	Do not print a header or a footer.  <i>Example:Print FOLDER HEADER=none QUIET</i>
NOWIZARD	/S	(no value)	Bypass the Windows Photo Printing Wizard for image files; Opus can natively print any image format that it is able to view.  <i>Example:Print NOWIZARD</i>
QUIET	/S	(no value)	Print the folder using the specified options without displaying the <b>Print Folder</b> dialog first.  <i>Example:Print FOLDER QUIET</i>
SETDEFAULT	/K	<printer name>	Set the named printer as the system default printer. The name you provide must be the full name of the printer as shown in the Printers Control Panel. Make sure you enclose the name in quotes if it contains a space.  <i>Example:Print SETDEFAULT "Brother HL-4050CDN"</i>
TO	/K	<printer name>	Print selected files or folders to the specified printer (overriding the default printer). A button with this command can also accept files dropped on it to print them.  <i>Example:Print TO "Brother HL-4050CDN"</i>
		<file name>	Print the folder contents to a specified disk file.  <i>Example:Print TO dirlist.txt FOLDER C:\Data QUIET AS txt</i>
		<b>clip</b>	Print the folder listing to the clipboard.  <i>Example:Print FOLDER TO clip AS csv</i>

## Properties

The **Properties** internal command can be used to:

- Display the system *Properties* dialog for files and folders
- Display the [Folder Options](#) dialog for the current folder

- Display the FTP [Site Properties](#) dialog for the currently connected FTP site
- Display a drop-down menu of your [favorite folder formats](#)
- Assign a [label](#) to selected files and folders
- Set a selected image file as the system wallpaper image

### Command Arguments:

Argument	Type	Possible values	Description
<i>(no argument)</i>	-	-	Display the system <i>Properties</i> dialog for selected files and folders.  <i>Example: Properties</i>
ADDLABEL	/O	<i>(no value)</i>	In conjunction with the <b>SETLABEL</b> argument, this lets you add one or more labels without clearing any existing ones.  <i>Example: Properties SETLABEL green ADDLABEL</i>
		<b>ctrl</b>	If the keyword <b>ctrl</b> is specified, the labels will only be added if the <b>Control</b> key is held down when the function is run - otherwise the ADDLABEL argument will be ignored and they will replace existing labels as normal.  <i>Example: Properties SETLABEL red ADDLABEL=ctrl</i>
FILE		<filename>	Specifies the filename rather than using selected files. This is the default argument for the <b>Properties</b> command so you do not need to specify the <b>FILE</b> keyword. If the filename includes spaces make sure you enclose it in quotes.  <i>Example: Properties "C:\Windows\System32\notepad.exe"</i>
FOLDEROPTIONS	/S	<i>(no value)</i>	Display the <b>Folder Options</b> dialog for the current folder.  <i>Example: Properties FOLDEROPTIONS</i>
FORMATLIST	/S	<i>(no value)</i>	Displays a generated list of your favourite folder formats (acts as a

			<p><a href="#">dynamic button</a>). This displays one item for each format saved in the Favorite Formats category on the <a href="#">Folder Formats</a> page in Preferences. Selecting a format from the generated list applies its settings to the current file display.</p> <p><i>Example: Properties FORMATLIST</i></p>
FTPSITE	/S	(no value)	<p>Display the FTP <a href="#">Site Properties</a> dialog for the currently connected FTP site. If you are not currently viewing an FTP directory this command has no effect.</p> <p><i>Example: Properties FTPSITE</i></p>
LABELCATEGORY	/K	<category>	<p>When used on a command that generates a list of labels (e.g. <b>Properties SETLABEL</b> or <b>Properties SETLABEL !menu</b>) this argument lets you filter the generated list by category. It accepts one or more comma-separated wildcard strings which let you match the name of categories to include.</p> <p>The specified categories will also be used when resetting labels using the <b>Properties SETLABEL !reset</b> command - if <b>LABELCATEGORY</b> is used as well, only labels in the specified categories will be cleared. This is used in the default <b>Properties</b> drop-down menu to provide a command that clears status icons without affecting other labels.</p> <p>You can match uncategorized labels using the pattern ~* (which means "not anything").</p> <p>Special handling exists for the two predefined categories, <i>Status</i> and <i>Colors</i>; these can be referenced using the English names prefixed with <b>raw:</b> and will work in any language.</p>

			<p><i>Example: Properties SETLABEL !menu LABELCATEGORY raw:~(Status)</i></p>
LISTER	/S	(no value)	<p>Displays the system <i>Properties</i> dialog for the folder currently displayed in the source file display.</p> <p><i>Example: Properties LISTER</i></p>
NOFROMFOCUS	/S	(no value)	<p>The default behaviour for the <b>Properties</b> command is to operate on either the source file display, or the Folder Tree, depending on which one has the input focus. This lets you use the same command to access the <i>Properties</i> dialog for folders in the tree as well as files and folders in the file display. Specify this argument to force the command to always operate on the source file display and ignore the folder tree.</p> <p><i>Example: Properties NOFROMFOCUS</i></p>
SETLABEL	/O	(no value)	<p>Displays a generated list of your configured <a href="#">labels</a> (acts as a <a href="#">dynamic button</a>). Selecting a label from this list applies it to all selected files and folders.</p> <p>This command supports <a href="#">embedded functions</a> when it's used to generate dynamic buttons.</p> <p><i>Example: Properties SETLABEL</i></p>
		<label>[,<label>,...]	<p>Applies the specified label or labels to all selected files and folders. Multiple label names must be comma-separated. Commas and back-slashes in label names must be escaped with a back-slash.</p> <p>You can combine this with the <b>ADDLABEL</b> argument to add labels to existing ones rather than replacing them.</p> <p>The keyword <b>stoponmatch</b> can be used to add the "stop on match" flag to a file, which prevents any wildcard or label</p>

			filters from applying to the file.  <i>Example: Properties SETLABEL=Green,Blue</i>
		<b>!menu</b>	Places the generated label list inside a menu.  <i>Example: Properties SETLABEL !menu</i>
		<b>!submenu</b>	Generates submenus for each label category.  <i>Example: Properties SETLABEL !submenu</i>
		<b>!submenu2</b>	Generates submenus for each label category, and places labels without a category in an <i>Uncategorized</i> group.  <i>Example: Properties SETLABEL !menu,!submenu2</i>
		<b>!nogroup</b>	Labels are grouped by category by default; specify <b>!nogroup</b> to ignore categories and generate a flat list of labels, sorted only by name, with labels from different categories intermingling.  <i>Example: Properties SETLABEL !menu,!nogroup</i>
		<b>!noreset</b>	After automatically generated lists of labels, a Reset option is normally added. You can prevent this by specifying <b>!noreset</b> .  <i>Example: Properties SETLABEL !noreset</i>
		<b>!nostoponmatch</b>	After automatically generated lists of labels, a Stop On Match option is normally added, if multiple labels matching a single file is enabled in Preferences. You can prevent it being added by specifying <b>!nostoponmatch</b> .

			<p><i>Example: Properties</i> <b>SETLABEL !nostoponmatch,!noreset LABELCATEGORY raw&gt;Status</b></p> <p>The example above would give you top-level buttons, directly on the toolbar, to toggle each of the Status labels, with no extra buttons cluttering up the toolbar.</p>
		<b>!reset</b>	<p>Removes the label from all selected items. You can combine this with the <b>LABELCATEGORY</b> argument to only remove labels in certain categories.</p> <p><i>Example: Properties</i> <b>SETLABEL=!reset</b></p>
SETLABELINFS	/K	<b>yes</b>	<p>Specify in conjunction with the <b>SETLABEL</b> argument to override the state of the <a href="#">Preferences / Favorites and Recent / File and Folder Labels / Automatically store labels in the file system if possible</a> option. Labels will be stored in the file system even if that option is disabled.</p> <p><i>Example: Properties</i> <b>SETLABEL=Green SETLABELINFS=yes</b></p>
		<b>no</b>	<p>The labels will be stored in your Opus configuration.</p> <p><i>Example: Properties</i> <b>SETLABEL=Blue SETLABELINFS=no</b></p>
SETLABELTOGGLE	/O	<i>(no value)</i>	<p>Specify in conjunction with the <b>SETLABEL</b> argument to toggle labels. If the targeted file or folder already has the specified label, the label will be reset instead.</p> <p>(Only affects labels explicitly applied to individual folders and files. Labels picked up via wildcards and filters can be overridden by more explicit labels but cannot be toggled on individual items.)</p> <p><i>Example: Properties</i> <b>SETLABEL=Bold SETLABELTOGGLE</b></p>

		<b>shift</b>	<p>If specified, the <b>SETLABELTOGGLE</b> argument will only apply if the <b>Shift</b> key is held down when the function is run. If <b>Shift</b> isn't held down, the label will only be turned on - it won't be toggled off again.</p> <p><i>Example: Properties</i>  <b>SETLABEL=Bold</b>  <b>SETLABELTOGGLE=shift</b></p>
SETWALLPAPER	/O	(no value)	<p>Sets the selected image file as the system wallpaper. Opus will make a copy of the selected image file (and convert its format if necessary) - you can change where the copy is stored with the <b>setwallpaper_file</b> option on the <a href="#">Miscellaneous / Advanced</a> page in Preferences.</p> <p><i>Example: Properties</i>  <b>SETWALLPAPER</b></p>
		<b>center</b>	<p>The wallpaper mode will be set to center the image on screen.</p> <p><i>Example: Properties</i>  <b>SETWALLPAPER=center</b></p>
		<b>tile</b>	<p>The image will be tiled across the screen.</p> <p><i>Example: Properties</i>  <b>SETWALLPAPER=tile</b></p>
		<b>stretch</b>	<p>The image will be stretched (or shrunk) to fill the screen completely.</p> <p><i>Example: Properties</i>  <b>SETWALLPAPER stretch</b></p>
		<b>fit</b>	<p>The image will be cropped if necessary to fill the screen. The aspect ratio of the image will be preserved. This is only available on Windows 7 and above.</p> <p><i>Example: Properties</i>  <b>SETWALLPAPER=fit</b></p>
		<b>fill</b>	<p>The image will be stretched or shrunk to fill screen, but the aspect ratio will be preserved - and so black bars may be displayed on the sides or top and bottom of the image. This is only available on</p>

			Windows 7 and above.  <i>Example: Properties</i> <b>SETWALLPAPER fill</b>
		<b>span</b>	The image will be spanned across a multiple monitor desktop. This is only available on Windows 8 and above.  <i>Example: Properties</i> <b>SETWALLPAPER span</b>
		<b>menu</b>	Displays a drop-down menu of the various wallpaper modes (acts as a <a href="#">dynamic button</a> ). Selecting a mode from this menu sets the selected image file as the wallpaper using that mode. This is useful when added to the context menu for the <b>Images</b> <a href="#">file type group</a> .  <i>Example: Properties</i> <b>SETWALLPAPER=menu</b>
SINGLE	/S	(no value)	Modifies the behaviour of the <b>Properties</b> command when more than one file or folder is selected. By default a combined <i>Properties</i> dialog is shown for all selected items, but if <b>SINGLE</b> is specified an individual <i>Properties</i> dialog is shown for each item.  <i>Example: Properties SINGLE</i>

## Recent

The **Recent** internal command can be used to:

- Display a dynamic list of your [Recent folders](#)
- Filter the recent to display only folders beneath a certain path
- Clear the Recent list

## Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Displays a dynamically generated list of your <a href="#">recently visited folders</a> - you can

			<p>navigate to a folder simply by selecting it from this list. Acts as a <a href="#">dynamic button</a>.</p> <p>Some of the arguments of this command can modify the appearance and behaviour of the dynamic list.</p> <p><i>Example:</i> <b>Recent</b></p>
CLEAR	/S	(no value)	<p>Clears the recent folders list.</p> <p><i>Example:</i> <b>Recent CLEAR</b></p>
COPY	/O	<Copy command arguments>	<p>Selecting a folder from the recent list will trigger the <b>Copy</b> command with the specified arguments. For example, you could have one drop-down recent list for <b>Copy</b> and another for <b>Move</b>, using different arguments. Note that if more than one <b>Copy</b> command argument is specified they must be enclosed in quotes.</p> <p><i>Example:</i> <b>Recent COPY</b></p> <p><i>Example:</i> <b>Recent COPY MOVE</b></p>
KEYARGS	/K/M	<qualifier:arguments> ...	<p>When displaying the recent list, this argument lets you assign different behaviour to the items in the list if a qualifier key is held down. This is a multiple value argument - for each qualifier key combination listed, you can define a separate set of arguments that will be used when the item in the list is selected.</p> <p>For example, you could configure your Recent menu to open folders in a new tab by default, but in a new Lister if the <b>Control</b> key were held down.</p> <p>The qualifier part of the value consists of one or more keywords that represent the qualifier keys - <b>ctrl</b>, <b>shift</b> and <b>alt</b>. These can be combined, for example <b>ctrlshift</b> means that both the <b>Control</b> and <b>Shift</b> keys must be held down. You can also use the keyword <b>none</b> to indicate arguments that are applied when no qualifiers are held.</p>

			<p>You can also assign the <b>Copy</b> command to a particular key combination; selecting an item from the recent list with the appropriate key held down would then copy (or move) selected files to the recent location.</p> <p><i>Example:Recent KEYARGS</i>  <b>"ctrl:NEW" "shift:Copy MOVE"</b>  <b>"none:NEWTAB=findexisting"</b></p>
NEW	/S	(no value)	<p>Recent folders selected from the list generated by this command will open in a new Lister instead of the current one.</p> <p><i>Example:Recent NEW</i></p>
NEWTAB	/O	(no value)	<p>Recent folders selected from the list generated by this command will open in a <a href="#">new tab</a>.</p> <p><i>Example:Recent NEWTAB</i></p>
		<b>deflister</b>	<p>If no lister exists, the Default Lister will open with an additional tab for the folder. If a lister exists, the folder will open normally in a new tab within the existing lister.</p> <p><i>Example:Recent NEWTAB=deflister</i></p>
		<b>findexisting</b>	<p>Look for the folder in an existing tab before opening a new one.</p> <p><i>Example:Recent NEWTAB=findexisting</i></p>
		<b>nofocus</b>	<p>New tabs opened by recent folders selected from the list will not be brought to the front.</p> <p><i>Example:Recent NEWTAB=nofocus</i>  <b>OPENINDUAL</b></p>
		<b>tofront</b>	<p>If the folder was found in an existing tab, bring that tab to the front (only used with <b>findexisting</b>).</p> <p><i>Example:Recent</i>  <b>NEWTAB=findexisting,tofront</b></p>
NOLABEL	/S	(no value)	<p>The recent list displayed by this command will not show any labels for the folders.</p>

			<i>Example:</i> <b>Recent NOLABEL</b>
OPENINDEST	/S	(no value)	Recent folders selected from the list will open in the destination file display or Lister.  <i>Example:</i> <b>Recent OPENINDEST</b>
OPENINDUAL	/S	(no value)	Recent folders selected from the list will open in the other file display of a dual-display Lister. The Lister will be set to dual-display mode if it isn't in that mode already.  <i>Example:</i> <b>Recent OPENINDUAL</b>
OPENINLEFT	/S	(no value)	Recent folders will open in the left-hand (or top) display of a dual-display Lister.  <i>Example:</i> <b>Recent KEYARGS none:OPENINLEFT ctrl:OPENINRIGHT</b>
OPENINRIGHT	/S	(no value)	Recent folders will open in the right-hand (or bottom) display of a dual-display Lister.  <i>Example:</i> <b>Recent NEWTAB OPENINRIGHT</b>
PATH		<path>	Filters the displayed recent list to only show folders matching or below the specified path. This is the default argument for the <b>Recent</b> command and so you do not need to specify the <b>PATH</b> keyword. Make sure you enclose the path in quotes if it contains spaces.  <i>Example:</i> <b>Recent C:\</b>
SHOWICONS	/S	(no value)	The recent list displayed by this command will display icons for the items within it. Note that the button that contains the <b>Recent</b> command must also have its <b>Show image</b> option turned on.  <i>Example:</i> <b>Recent SHOWICONS</b>
USEQUALKEYS	/S	(no value)	Activates pre-configured behaviour for the main qualifier keys - <b>Control</b> will open the recent folder in the dual-display, <b>Shift</b> in a new Lister and <b>Alt</b> in a new tab.  This is equivalent to <b>KEYARGS ctrl:OPENINDUAL shift:NEW</b>

**alt:NEWTAB.**

*Example:Recent USEQUALKEYS*

## Rename

The **Rename** internal command can be used to:

- Trigger [inline rename](#) mode, letting you quickly edit the name of a file or folder
- Display the [Rename](#) dialog, letting you perform wildcard and scripted renames on multiple files at once
- Perform automated rename operations
- [Embed a rename script](#) in a button or hotkey

### Command Arguments:

Argument	Type	Possible values	Description
<i>(no argument)</i>	-	-	Displays the <a href="#">Rename</a> dialog. The rename operation will be performed on all selected files and folders. The dialog will open in the last mode it was used in (either <a href="#">simple</a> or <a href="#">advanced</a> ).  <i>Example: <b>Rename</b></i>
ADVANCED	/S	<i>(no value)</i>	Displays the Rename dialog in <a href="#">advanced</a> mode.  <i>Example: <b>Rename ADVANCED</b></i>  When combined with rename patterns or preset names, the <b>ADVANCED</b> argument can be used to ensure the command opens the Rename dialog instead of immediately renaming all selected items.  <i>Example: <b>Rename ADVANCED PATTERN=* TO=*.bak</b></i> <i>Example: <b>Rename ADVANCED PRESET="Number Files"</b></i>
AUTORENAME	/S	<i>(no value)</i>	Automatically adds an incrementing number to the end of new filenames if they clash with existing files.

			<i>Example: Rename PATTERN "The *" TO * AUTORENAME</i>
BY	/K/N	<increment>	Specifies the increment when renaming files with automatic numbering (using the <b>NUMBER</b> option). If not specified the default increment is 1.  <i>Example: Rename NUMBER BY 2</i>
CASE	/K	<b>upper</b>	Converts filenames to all-upper case.  <i>Example: Rename CASE=upper</i>
		<b>lower</b>	Converts filenames to all-lower case.  <i>Example: Rename CASE=lower NUMBER</i>
		<b>firstword</b>	Capitalizes the first letter of the filename.  <i>Example: Rename CASE=firstword</i>
		<b>allwords</b>	Capitalizes the first letter of each word of the filename.  <i>Example: Rename CASE=allwords</i>
		<b>extupper</b>	Converts the filename extension to upper case.  <i>Example: Rename CASE=extupper</i>
		<b>extlower</b>	Converts the filename extension to lower case.  <i>Example: Rename CASE=extlower</i>
FINDREP	/O	(no value)	Enable find-and-replace mode. The <b>PATTERN</b> argument must be used to provide the string to find, and the <b>TO</b> argument provides the string to replace it with. You can optionally combine this with the <b>REGEXP</b> argument.  <i>Example: Rename FINDREP PATTERN="jones" TO="smith"</i>
		<b>ext</b>	Makes find-and-replace operate in the filename extension as well as in the stem of the filename.  <i>Example: Rename FINDREP=ext PATTERN="jpg" TO="jpeg"</i>
FROM		<filename> ...	Renames the files specified on the command line, rather than those selected

			<p>in the source file display. This argument can accept filenames or <a href="#">wildcard patterns</a>. Remember to enclose each filename in quotes if it contains spaces.</p> <p><i>Example: Rename FROM C:\Spool\*.tmp TO *.temp</i></p>
IGNOREEXT	/S	(no value)	<p>Turns on the <b>Ignore extensions</b> option for the rename function. Filename extensions won't be affected and don't need to be accounted for in any wildcard patterns.</p> <p><i>Example: Rename PATTERN * TO *_backup IGNOREEXT</i></p>
INLINE	/O	(no value)	<p>Activates <a href="#">inline renaming</a> on the item in the source file display that currently has focus.</p> <p><i>Example: Rename INLINE</i></p>
		<b>all</b>	<p>Automatically selects the entire filename for editing.</p> <p><i>Example: Rename INLINE=all</i></p>
		<b>name</b>	<p>Automatically selects the filename's stem (but not the extension). Selects the whole name for a folder.</p> <p><i>Example: Rename INLINE=name</i></p>
		<b>endstem</b>	<p>Positions the cursor at the end of the filename's stem (before the extension), but does not automatically select any part of the name.</p> <p><i>Example: Rename INLINE=endstem</i></p>
		<b>ext</b>	<p>Automatically selects the filename's extension. Selects the whole name for a folder.</p> <p><i>Example: Rename INLINE=ext</i></p>
		<b>home</b>	<p>Positions the cursor at the beginning of the filename.</p> <p><i>Example: Rename INLINE=home</i></p>
		<b>end</b>	<p>Positions the cursor at the end of the filename.</p> <p><i>Example: Rename INLINE=end</i></p>

		<b>single</b>	<p>If more than one item is selected, the <b>Rename</b> dialog will be displayed. Without this, inline rename will begin on the focused item only.</p> <p><i>Example: <b>Rename</b> <b>INLINE=name,single</b></i></p>
MACRO	/K		<p>Lets you specify a rename <a href="#">macro operation</a> string. You can use the macro builder in the Rename dialog to generate these strings and then hardcode them in a command.</p> <p><i>Example: <b>Rename MACRO R0-6/L0+Final</b></i></p>
MATCHCASE	/S	(no value)	<p>Makes the rename operation case-sensitive. Patterns and search strings must match the exact case of the filename.</p> <p><i>Example: <b>Rename PATTERN *.jpG TO *.JPG MATCHCASE</b></i></p>
NOFILEINFO	/S	(no value)	<p>Disable file information metadata insertion. For example, the string <b>{mp3title}</b> would normally insert the title of an MP3 file in the new filename. With <b>NOFILEINFO</b> specified, the literal string "{mp3title}" would be inserted in the new filename.</p> <p><i>Example: <b>Rename PATTERN * TO *_{name} NOFILEINFO</b></i></p>
NOMATCHNOFAIL	/S	(no value)	<p>Files which do not match the rename pattern are not 'failed'. This can be useful with multi-line functions which need to do some optional renaming before passing all of the files to additional commands.</p> <p>By default, when <b>NOMATCHNOFAIL</b> is not used, if a file does not match the rename pattern then it will be flagged as a failure and skipped by the rest of the function (unless the very next command is another <b>Rename</b>, in which case the file gets a second chance).</p>

			<p>If <b>NOMATCHNOFAIL</b> is used then files which do not match the rename pattern are still passed to subsequent commands.</p> <p><i>Example: Rename PATTERN "*" Backup.*" TO *.*</i> <b>NOMATCHNOFAIL</b></p>
NUMBER	/O	(no value)	<p>Automatically number files. Selected files will be numbered in the order they are presented in the file display, so you should make sure the list is sorted as desired before using this command. The number 1 will be assigned to the first selected file, and the number for each subsequent file will be incremented by the value given for the <b>BY</b> argument (or by 1 if <b>BY</b> is not provided).</p> <p>By default the number is added to the end of the filename, in front of the extension. You can specify a different location for the number by providing the [#] marker in the new filename.</p> <p><i>Example: Rename PATTERN * TO [#]* NUMBER</i></p>
		<start>	<p>Number files starting with the specified number. Providing a value for NUMBER also lets you zero-pad the assigned number. For example, the value <b>0010</b> means to start numbering at 10, and zero-pad to four digits. The [#] marker can also be used to specify zero-padding - [#5] would zero-pad to five digits.</p> <p><i>Example: Rename NUMBER 00001 BY 2</i></p> <p>If you add a ! before the number, you can use this argument to specify the default value for the corresponding field in the Rename dialog without actually turning on the sequential numbering option. For example, you may want a default of "01" so you get padding to two digits without having to type the extra "0" every time you turn on the option in the UI. If you make use of that,</p>

			<p>you should also include the <b>ADVANCED</b> argument so the Rename dialog appears, instead of it silently performing a rename operation.</p> <p><i>Example: Rename <b>ADVANCED NUMBER=!01</b></i></p>
PATTERN	/K	<pattern>	<p>Specifies a wildcard pattern that represents the old (original) filename. This argument is used when performing a wildcard rename. The Rename command supports a simple wildcard syntax where one or more asterisks supplied for the <b>PATTERN</b> argument can be used to copy parts of the original name to the new name. The <b>PATTERN</b> argument is also used in conjunction with <b>REGEXP</b> to provide the search pattern for a regular expression rename, and in conjunction with <b>FINDREP</b> to provide the search string for a find-and-replace rename.</p> <p>See the section on <a href="#">Renaming Files</a> for a full discussion of the various renaming modes.</p> <p><i>Example: Rename <b>PATTERN IMG(*).jpg TO "Image \1.jpg" REGEXP</b></i></p> <p>Specifying the <b>FROM</b> argument as well will mean the command ignores the current file selection and applies the specified rename on all matching files. Only files matching both the <b>TO</b> and <b>PATTERN</b> arguments would be renamed.</p> <p><i>Example: Rename <b>FROM *.jpg PATTERN IMG(*).jpg TO "Image \1.jpg" REGEXP</b></i></p> <p>If the <b>PATTERN</b> and <b>TO</b> arguments are both given, the command will normally apply the rename immediately, without prompting for any further interaction; you can add the <b>ADVANCED</b> argument to instead display the Rename dialog, with the</p>

			<p>specified pattern, so that you can preview the operation and make adjustments, or cancel it entirely, as needed.</p> <p><i>Example: Rename ADVANCED PATTERN * TO *.bak</i></p>
PRESET	/K	<b>regex</b>	<p>Opens the <b>Rename</b> dialog in <i>Regular Expression</i> mode.</p> <p><i>Example: Rename PRESET=regex</i></p>
		<b>findrep</b>	<p>Opens the <b>Rename</b> dialog in <i>Find and Replace</i> mode.</p> <p><i>Example: Rename PRESET=findrep</i></p>
		<b>last</b> <b>!last</b>	<p>Specifying <b>PRESET=last</b> opens the <b>Rename</b> dialog showing the settings from the last time the dialog was used. Similar to opening the dialog and clicking the <i>Last Rename</i> button.</p> <p><i>Example: Rename PRESET=last</i></p> <p>You can also use <b>!last</b> to apply the last rename automatically, without showing the <b>Rename</b> dialog first.</p> <p><i>Example: Rename PRESET=!last</i></p>
		<b>!list</b>	<p>Displays a generated list of saved rename presets (acts as a <a href="#">dynamic button</a>). Selecting a preset from the list will apply that rename operation to the currently selected files and folders.</p> <p>The generated list can be controlled with the addition of the following keywords:</p> <ul style="list-style-type: none"> <li>• <b>favesonly</b>: Only displays presets marked as favorites.</li> <li>• <b>nofaves</b>: Only displays presets not marked as favorites.</li> </ul>

			<ul style="list-style-type: none"> <li>• <b>nogroup</b>: Don't not group favorite and non-favorite presets separately.</li> </ul> <p><i>Example: Rename</i> <b>PRESET=!list,favesonly</b></p>
		<preset>	<p>Applies the named saved rename preset to all currently selected files and folders. Rename presets are created through the <a href="#">Advanced Rename</a> dialog.</p> <p><i>Example: Rename</i> <b>PRESET="Number Files"</b></p> <p>You can add the <b>ADVANCED</b> argument to make the dialog open and allow you to make changes, or cancel the operation entirely, before the rename is applied.</p> <p><i>Example: Rename</i> <b>ADVANCED PRESET="Number Files"</b></p>
RECURSE	/S	(no value)	<p>The rename operation will operate recursively on all files inside selected sub-folders.</p> <p><i>Example: Rename</i> <b>PATTERN *.jpg TO *.jpeg RECURSE</b></p>
REGEXP	/S	(no value)	<p>Enables regular expression mode. The search pattern must be provided with the <b>PATTERN</b> argument, and the replace pattern with the <b>TO</b> argument. You can optionally combine this with the <b>FINDREP</b> argument.</p> <p><i>Example: Rename</i> <b>PATTERN "(.*) - (.*)\.(*)" TO "\2 - \1.\3" REGEXP</b></p>
RENAMEMATCHING	/S	(no value)	<p>Keeps files with the same stem and different extensions together when renumbering. For example, <b>IMG1032.JPG</b> and <b>IMG1032.WAV</b> when renamed would be given the same number.</p> <p><i>Example: Rename</i> <b>NUMBER 0001 RENAMEMATCHING</b></p>
SCRIPTARG	/K/M	<option:value>, [<option:value>,...]	<p>Used to pass custom field values to a <a href="#">rename script</a>.</p>

			<i>Example: Rename PRESET MyRename SCRIPTARG my_option:True</i>
SIMPLE	/S	(no value)	Displays the Rename dialog in <a href="#">simple</a> mode.  <i>Example: Rename SIMPLE</i>
TO	/O	<new name>	Specifies the new name of the file to rename. This argument is also used to specify the 'to' pattern when renaming using wildcards or regular expression, and the 'replace' string when renaming in find-and-replace mode. This argument can also be provided without a value when doing a find-and-replace rename - in that case, the search string would be replaced with nothing.  <i>Example: Rename PATTERN "(Copy)" TO FINDREP</i>
TYPE	/K	files	Force the rename to only operate on files - any selected folders will be ignored.  <i>Example: Rename PATTERN * TO *.bak TYPE=files</i>
		dirs	Force the rename to only operate on folders.  <i>Example: Rename PATTERN * TO Copy_* TYPE=dirs</i>
WHENEXISTS	/K	ask	Controls what happens if the new filename already exists. The default action is to <b>ask</b> the user for each existing file.  <i>Example: Rename PATTERN * TO *.bak WHENEXISTS=ask</i>
		delete	Performs the rename as requested, deleting the file that already existed.  <i>Example: Rename PATTERN * TO *.bak WHENEXISTS=delete</i>
		keep	Performs the rename as requested, renaming the old file that already existed.

			<i>Example: Rename PATTERN * TO *.bak WHENEXISTS=keep</i>
		<b>rename</b>	Modifies the new name by adding a suffix to make it unique.  <i>Example: Rename PATTERN * TO *.bak WHENEXISTS=rename</i>
		<b>skip</b>	Skips any files that already exist, without performing the rename.  <i>Example: Rename PATTERN * TO *.bak WHENEXISTS=skip</i>

## Select

The **Select** internal command is used to:

- Display the **Select** dialog (in either [simple](#) or [advanced](#) modes)
- Select all, deselect all, and invert the current selection of all files and folders
- Select or deselect files by filename with a wildcard pattern
- Select all files in the source that are selected in the destination, and vice versa
- Select all files with extensions matching those already selected
- Convert [checkmarks](#) into selection and vice versa
- Hide files based on whether they are selected or not
- Select a range of files by index
- Select files by date and size

### Command Arguments:

Argument	Type	Possible values	Description
<i>(no arguments)</i>	-	-	Displays the <b>Select</b> dialog (in either <a href="#">simple</a> or <a href="#">advanced</a> modes, depending on which was last used).

			<i>Example: Select</i>
ADVANCED	/O	(no value)	Displays the <b>Select</b> dialog in <a href="#">advanced</a> mode.  <i>Example: Select ADVANCED</i>
		<filter name>	Displays the <b>Select</b> dialog in advanced mode, with the specified saved filter already loaded.  <i>Example: Select ADVANCED=MyFilter</i>
ALL	/S	(no value)	Select all files and folders in the current source file display.  <i>Example: Select ALL</i>
ALLDIRS	/S	(no value)	Select all folders in the current source file display.  <i>Example: Select ALLDIRS</i>
ALLFILES	/S	(no value)	Select all files in the current source file display.  <i>Example: Select ALLFILES</i>
DATE	/K	<date>	Select files whose last modification timestamps match the specified date. You can specify: <ul style="list-style-type: none"> <li>• Just a date, in the format <i>YYYY-MM-DD</i></li> <li>• Just a time, in the format <i>HH:MM</i> (seconds are ignored)</li> <li>• Both a date and time, in the format <i>YYYY-MM-DD HH:MM</i></li> </ul> <p>Note that specifying both a date and time requires quotes around the value, because of the space character separating the two.</p> <p>You can also use &gt; (greater than) before the date to match all files newer than the specified date, or &lt; (less than) before the date to match all files older than the specified date.</p> <p>You can also specify an <i>age</i> rather than a <i>date</i> to test for. For example, to select all files older than 5 days, you might specify</p>

			<p><b>Select DATE "&gt;5 days"</b>. Valid keywords for age selection are <b>day, week, month, year, hour, minute, second</b>.</p> <p><i>Example: Select *.jpg DATE "&gt;2012-06-15 10:00"</i></p>
		<date1>.. <date2>	<p>Select files whose last modification timestamps falls between the two specified dates. Both dates are supplied in the format described above.</p> <p><i>Example: Select DATE 2012-01-01..2012-12-31 TYPE=files</i></p>
		<b>oldest</b>	<p>Select the oldest item in the current source file display. You can combine this with the <b>PATTERN</b> argument to select the oldest of a specific type of file.</p> <p><i>Example: Select *.doc DATE=oldest</i></p>
		<b>newest</b>	<p>Select the newest item in the current file display.</p> <p><i>Example: Select DATE=newest DESELECTNOMATCH</i></p>
		<b>created</b>	<p>Normally this command considers the last modification timestamp of each file, however by specifying this keyword you can make it look at the creation time instead.</p> <p><i>Example: Select *.zip 7z rar) DATE=created,2010-03-10..2010-03-17</i></p>
		<b>both</b>	<p>Considers both created and last modification timestamps.</p> <p><i>Example: Select DATE=both,newest</i></p>
		<b>next</b>	<p>Modifies the behavior of the <b>newest</b> and <b>oldest</b> arguments. Normally, <b>Select DATE=newest</b> would select the newest file in the list. If it were already selected, nothing would change. If you add the <b>next</b> keyword, Opus will progressively select the next newest file each time the command is run.</p> <p><i>Example: Select DATE=next,newest</i></p>
DESELECT	/S	(no value)	<p>Instead of selecting files, the command will deselect them. This is used in conjunction with the <b>PATTERN, ALLDIRS</b> and <b>ALLFILES</b> arguments.</p>

			<i>Example: Select *.jpg</i> <b>DESELECT</b>
DESELECTNOMATCH	/S	(no value)	Files that don't match the pattern will be deselected (normally files that don't match are left alone). This also works when using the <b>FILTER</b> argument to select files with a predefined filter.  <i>Example: Select *.doc</i> <b>DESELECTNOMATCH</b>
DESELECTOTHERTYPE	/S	(no value)	When used with the <b>TYPE</b> argument to restrict a selection to either files or folders (or with the <b>ALLFILES</b> and <b>ALLDIRS</b> arguments), <b>DESELECTOTHERTYPE</b> causes all items of the other type to be deselected  <i>Example: Select * TYPE=files</i> <b>DESELECTOTHERTYPE</b>
DESTTOSOURCE	/O	(no value)	Selects all files and folders in the source file display that are currently selected in the destination. The comparison is only done on the filename - the files are not actually compared.  <i>Example: Select DESTTOSOURCE</i>
		<b>in</b>	Selects all files and folders in the source file display that exist in the destination.  <i>Example: Select DESTTOSOURCE=in</i>
		<b>noext</b>	Does not consider file extensions when comparing selected files in the source and destination. For example, if <b>IMG1234.JPG</b> was selected in the destination, and <b>IMG1234.WAV</b> existed in the source, it would be selected.  <i>Example: Select DESTTOSOURCE=noext</i>
		<b>notin</b>	Selects all files and folders in the source file display that don't exist in the destination.  <i>Example: Select DESTTOSOURCE=notin</i>
EXACT	/S	(no value)	Indicates that the <b>PATTERN</b> argument is a literal file name and not a wildcard or regular expression. This allows you to specify an exact filename without having to escape wildcard characters like '(' and ')'

			<i>Example: Select "Cat Photo (1).jpg"</i> <b>EXACT</b>
FILTER	/S	(no value)	Performs file selection using a pre-defined filter. The name of the filter must be given as the value of the <b>PATTERN</b> argument. Filters must have previously been configured through the <a href="#">File Operations / Filters</a> page in Preferences.  You can use this with the <b>TYPE</b> argument to restrict the filter to either files or folders only.  <i>Example: Select "Image Files" FILTER</i>
FILTERFLAGS	/K	<b>select</b>	Select files that match the filter (this argument is used in conjunction with the <b>FILTER</b> argument). This is the default behaviour.  <i>Example: Select Documents FILTER FILTERFLAGS=select</i>
		<b>deselect</b>	Deselect files that match the filter.  <i>Example: Select "Music Files" FILTER FILTERFLAGS=deselect</i>
		<b>hide</b>	Hide files that match the filter.  <i>Example: Select "Temp Files" FILTER FILTERFLAGS=hide</i>
		<b>hidenomatch</b>	Hide files that don't match the filter.  <i>Example: Select "Image Files" FILTER FILTERFLAGS=hidenomatch</i>
FIRST	/S	(no value)	Select the first item in the source file display, deselect all other items.  <i>Example: Select FIRST</i>
FROMCHECKS	/S	(no value)	Convert the state of checked items to selections (checked items will be selected, non-checked items will be deselected). This only applies in <a href="#">checkbox mode</a> .  <i>Example: Select FROMCHECKS</i>
FROMSCRIPT	/S	(no value)	This command should be used when running a Select command from a script (e.g. via <b>Command.RunCommand</b> ). It tells the

			<p>command to select the files in the <b>Command</b> object itself.</p> <p><i>Example: <b>Func.Command.RunCommand("Select FROMSCRIPT");</b></i></p>
GROUPNAME	/O	(no value)	<p>When the file display is <a href="#">grouped</a>, this lets you select files based on the group they are in. When <b>GROUPNAME</b> is used without an associated value, the value of the <b>PATTERN</b> argument is used as the name of the group to match.</p> <p>The example below selects all files in groups beginning with <b>X</b>.</p> <p><i>Example: <b>Select X* GROUPNAME</b></i></p>
		<group name>	<p>When a value is provided for the <b>GROUPNAME</b> argument it specifies the name (or wildcard pattern) of the file group. The selection operation will be confined to files and folders in matching groups.</p> <p>You can also use this in conjunction with the <b>SETFOCUS</b> argument to give input focus to a group header.</p> <p><i>Example: <b>Select *.jpg GROUPNAME Today</b></i>  <i>Example: <b>Select NOPATTERN GROUPNAME Yesterday SETFOCUS</b></i></p>
HIDSEL	/O	(no value)	<p>Hide all selected items (both files and folders). This is used either with the <b>PATTERN</b> argument to hide all files that match the pattern, or with the <b>NOPATTERN</b> argument to hide all currently selected files.</p> <p><i>Example: <b>Select *.tmp HIDSEL</b></i></p>
		<b>dirs</b>	<p>Hide all selected directories.</p> <p><i>Example: <b>Select HIDSEL=dirs NOPATTERN</b></i></p>
		<b>files</b>	<p>Hide all selected files.</p> <p><i>Example: <b>Select HIDSEL=files NOPATTERN</b></i></p>

HIDEUNAFFECTED	/S	(no value)	When used with the <a href="#">synchronize</a> tool, this hides any items from the list that are not marked to be synchronized (either copied or deleted).  <i>Example: <b>Select HIDEUNAFFECTED</b></i>
HIDEUNSEL	/O	(no value)	Hide all unselected items (both files and folders). This is used either with the <b>PATTERN</b> argument (files that don't match the pattern will be hidden), or with the <b>NOPATTERN</b> argument (all currently unselected files will be hidden).  <i>Example: <b>Select NOPATTERN HIDEUNSEL</b></i>
		<b>dirs</b>	Hide all unselected directories.  <i>Example: <b>Select HIDEUNSEL=dirs NOPATTERN</b></i>
		<b>files</b>	Hide all unselected files.  <i>Example: <b>Select HIDEUNSEL=files NOPATTERN</b></i>
INVERT	/S	(no value)	Inverts the selection state of all items in the source file display.  <i>Example: <b>Select INVERT</b></i>
LAST	/S	(no value)	Selects the last item in the source file display, deselect all other items.  <i>Example: <b>Select LAST</b></i>
MAKEVISIBLE	/O	(no value)	Ensures that the first selected item is visible in the file display. The list will be scrolled if needed. Similar to the <b>SETFOCUS</b> argument except the viewer pane will not update to show the new selection.  <i>Example: <b>Select *.doc MAKEVISIBLE</b></i>
		<b>immediate</b>	Prevents the short delay that normally occurs before the selected file is scrolled into view.  <i>Example: <b>Select NEXT MAKEVISIBLE=immediate</b></i>
NEXT	/O	(no value)	Selects the next item in the file display. The first item immediately following the first currently selected item will be selected, and all other items deselected.

			<i>Example: <b>Select NEXT</b></i>
		<b>mark</b>	Toggles the selection state of the currently focused item, and moves the input focus to the next item in the list. This is the equivalent of pressing the <b>Insert</b> key in the file display.  <i>Example: <b>Select NEXT=mark</b></i>
		<b>nodeselect</b>	Prevents any currently selected items from being deselected.  <i>Example: <b>Select NEXT=nodeselect</b></i>
		<b>row</b>	In the icon display modes (e.g. Thumbnails mode) this will move the selection down one row (vertically instead of horizontally). Ignored in Details and Power modes.  <i>Example: <b>Select NEXT=row,mark</b></i>
NONE	/S	(no value)	Deselects all items in the source file display.  <i>Example: <b>Select NONE</b></i>
NOPATTERN	/S	(no value)	The <b>Select</b> command normally requires a value for the <b>PATTERN</b> argument to operate, but in some cases you may need it to operate without supplying a pattern. For example, the <b>HIDSEL</b> and <b>HIDEUNSEL</b> arguments can be used to hide all currently selected or unselected items without applying a new wildcard selection first.  <i>Example: <b>Select HIDSEL NOPATTERN</b></i>
PATTERN		<pattern>	Specify a wildcard pattern. All items matching the supplied pattern will be selected (or deselected, hidden, etc. based on the other arguments for this command). The pattern can be specified using <a href="#">standard pattern matching</a> syntax, or <a href="#">regular expressions</a> if the <b>REGEXP</b> argument is supplied. The <b>PATTERN</b> argument is also used to provide the name of a pre-defined filter in conjunction with the <b>FILTER</b> argument.  This is the default argument for the <b>Select</b> command and so the <b>PATTERN</b> keyword does not need to be supplied.

			<i>Example: Select *.<a href="#">(bmp jpg gif)</a> <b>HIDEUNSEL</b></i>
PREV	/O	<i>(no value)</i>	Select the previous item in the file display. The first item immediately preceding the last currently selected item will be selected, and all other items deselected.  <i>Example: Select <b>PREV</b></i>
		<b>mark</b>	Toggles the selection state of the currently focused item, and moves the input focus to the previous item in the list. Similar to pressing the <b>Insert</b> key, except the focus moves to the previous rather than the next item.  <i>Example: Select <b>PREV=mark</b></i>
		<b>nodeselect</b>	Prevents any currently selected items from being deselected.  <i>Example: Select <b>PREV=nodeselect</b></i>
		<b>row</b>	In the icon display modes (e.g. Thumbnails mode) this will move the selection up one row (vertically instead of horizontally). Ignored in Details and Power modes.  <i>Example: Select <b>PREV=row,mark</b></i>
RANGE	/K	<range>	Selects a range of items based on their index (their position in the list). This command is equivalent to the range selection mode of the <a href="#">find-as-you-type</a> field. The <range> value consists of one or more comma-separated ranges; each range can be a single number, or two numbers separated by a hyphen to indicate all numbers within that range.  <i>Example: Select <b>RANGE 3,8-15,22-25,30</b></i>
REGEXP	/S	<i>(no value)</i>	Use <a href="#">regular expression</a> mode instead of standard pattern matching.  <i>Example: Select <b>.*\.</b> <b>REGEXP</b></i>
RESELECT	/S	<i>(no value)</i>	Reselects all files and folders that were used (and deselected) by the previously executed command.  <i>Example: Select <b>RESELECT</b></i>
SETFOCUS	/S	<i>(no value)</i>	Ensures that the first selected item is visible in the file display. The list will be scrolled if

			<p>needed. Additionally, if the viewer pane is open the first selected file will be automatically viewed if possible.</p> <p><i>Example: Select *.jpg SETFOCUS</i></p>
SHOWFOCUS	/S	(no value)	<p>If necessary, scrolls the file display to make the currently focused item visible. The selection will not be modified.</p> <p><i>Example: Select SHOWFOCUS</i></p>
SHOWHIDDEN	/O	(no value)	<p>Reveal any files or folders that have previously been hidden by commands using the <b>HIDSEL</b> or <b>HIDEUNSEL</b> arguments. The other way to reveal files hidden this way is by re-reading the folder (e.g. press <b>F5</b>).</p> <p><i>Example: Select NOPATTERN SHOWHIDDEN</i></p>
		<b>dirs</b>	<p>Reveals all hidden directories.</p> <p><i>Example: Select SHOWHIDDEN=dirs NOPATTERN</i></p>
		<b>files</b>	<p>Reveals all hidden files.</p> <p><i>Example: Select SHOWHIDDEN=files NOPATTERN</i></p>
SHOWUNAFFECTED	/S	(no value)	<p>When used with the <a href="#">synchronize</a> tool, this reveals any items that have previously been hidden because they were not marked to be synchronized (either copied or deleted).</p> <p><i>Example: Select SHOWUNAFFECTED</i></p>
SIMILAR	/S	(no value)	<p>Selects all files with the same file extensions as the currently selected files. For example, if a single <b>.jpg</b> and a single <b>.gif</b> file are currently selected, this command would select <i>all</i> <b>.jpg</b> and <b>.gif</b> files in the source file display.</p> <p><i>Example: Select SIMILAR</i></p>
SIMILARBASE	/S	(no value)	<p>Selects all files with the same base-names as the currently selected files. For example, if <b>cat.jpg</b> and <b>dog.gif</b> are currently selected, this command would select <i>all</i><b>cat.*</b> and <b>dog.*</b> files in the source file display.</p> <p><i>Example: Select SIMILARBASE</i></p>

SIMPLE	/S	(no value)	Displays the <b>Select</b> dialog in <a href="#">simple</a> mode.  <i>Example: <b>Select SIMPLE</b></i>
SIZE	/K	<size>	Select files whose size matches the specified size. By default the size specified is treated as bytes, but you can use the following suffixes to use different units: <ul style="list-style-type: none"> <li>• <b>kb</b> - kilobytes</li> <li>• <b>mb</b> - megabytes</li> <li>• <b>gb</b> - gigabytes</li> </ul> <p>You can also use &gt; (greater than) before the size to match all files larger than the specified size, or &lt; (less than) before the size to match all files smaller than the specified size.</p> <i>Example: <b>Select *.png SIZE &gt;2mb</b></i>
		<size1>..<size2>	Select files whose size falls between the two specified sizes. Both sizes are supplied in the format described above.  <i>Example: <b>Select SIZE 500kb..5mb DESELECTNOMATCH</b></i>
		<b>largest</b>	Select the largest item in the current source file display. You can combine this with the <b>PATTERN</b> argument to select the largest of a specific type of file.  <i>Example: <b>Select *.doc SIZE=largest</b></i>
		<b>smallest</b>	Select the smallest item in the current file display.  <i>Example: <b>Select SIZE=smallest</b></i>
SOURCETODEST	/O	(no value)	Selects all files and folders in the destination file display that are currently selected in the source. The comparison is only done on the filename - the files are not actually compared.  <i>Example: <b>Select SOURCETODEST</b></i>

		<b>in</b>	Selects all files and folders in the destination file display that exist in the source.  <i>Example: Select SOURCETODEST=in</i>
		<b>noext</b>	Does not consider file extensions when comparing selected files in the source and destination. For example, if <b>IMGP1234.JPG</b> was selected in the source, and <b>IMGP1234.WAV</b> existed in the destination, it would be selected.  <i>Example: Select SOURCETODEST=noext</i>
		<b>notin</b>	Selects all files and folders in the destination file display that don't exist in the source.  <i>Example: Select SOURCETODEST=notin</i>
THIS	/S	(no value)	Selects the current focus entry. It is possible for the entry with input focus to not be selected (for example, if you move the focus highlight with <b>Control +Cursor-Down</b> ) and this command will select whichever entry is currently focused.  <i>Example: Select THIS</i>
TOCHECKS	/S	(no value)	Convert item selection states to check states. Selected items will be checked, and unselected items will be unchecked. If the file display is not currently in <a href="#">checkbox mode</a> it will be turned on automatically.  <i>Example: Select TOCHECKS</i>
TYPE	/K	<b>files</b>	Force the selection to only affect files - even if folders match the pattern they will be unaffected.  You can add the <b>DESELECTOTHERTYPE</b> argument to deselect all items of the "other" type.  <i>Example: Select a* TYPE=files</i>
		<b>dirs</b>	Force the selection to only affect folders.  <i>Example: Select "new *" TYPE=dirs</i>

## Set

The **Set** internal command can be used to:

- Turn the various Lister elements ([folder tree](#), [metadata pane](#), [viewer pane](#), [dual display](#), etc) on or off in the current Lister
- Add or remove information columns from the current file display
- Turn [Checkbox Mode](#), [Flatview Mode](#) and [Navigation Lock](#) on or off
- Make changes to [filtering](#), [sorting](#), [grouping](#) and [view mode](#) settings
- Turn [Administrator mode](#) on or off
- Enable or disable the Copy and Delete [recursive filters](#)
- Enable or disable full-row selection and grid lines
- Show or hide [compatibility files](#)
- Adjust the size and position of the current Lister window
- ... and lots of other things :)

You can combine multiple **Set** command arguments on the one command line to make multiple changes to the Lister at once. For example, **Set DUAL=on TREE=off** would turn dual-display mode on, and the folder trees off, in one operation.

### Command Arguments:

Argument	Type	Possible values	Description
ADMIN	/K	<b>on</b>	Turns <a href="#">Administrator mode</a> in in the active Lister. Opus will prompt you for the timeout, after which Administrator mode is automatically deactivated. This function has no effect on Windows XP, or if UAC is disabled.  <i>Example: Set ADMIN=on</i>

		<b>off</b>	Turns Administrator mode off in the active Lister.  <i>Example: Set ADMIN=off</i>
		<b>toggle</b>	Toggles Administrator mode on or off.  <i>Example: Set ADMIN=toggle</i>
		<timeout>	Specify a timeout (in minutes) to suppress the timeout dialog from appearing.  <i>Example: Set ADMIN=toggle,10</i>
AUTOSIZE	/K	<b>on</b>	Turns the <i>Auto-size columns</i> option on in the current file display.  <i>Example: Set AUTOSIZE=on</i>
		<b>off</b>	Turns the <i>Auto-size columns</i> option off.  <i>Example: Set AUTOSIZE=off</i>
		<b>toggle</b>	Toggles the <i>Auto-size columns</i> option on or off in the current file display.  <i>Example: Set AUTOSIZE=toggle</i>
AUTOSIZECOLUMNS	/O	(no value)	Automatically resize all columns in the current source file display (applies to details and power modes only).  <i>Example: Set AUTOSIZECOLUMNS</i>
		<b>dest</b>	Automatically resize all columns in the current destination file display.  <i>Example: Set AUTOSIZECOLUMNS=dest</i>
		<b>left</b>	Resize all columns in the left (or top) file display of a dual display Lister.  <i>Example: Set AUTOSIZECOLUMNS=left</i>
		<b>right</b>	Resize all columns in the right (or bottom) file display.  <i>Example: Set AUTOSIZECOLUMNS=right</i>
		<b>both</b>	Resize all columns in both file displays of a dual display Lister (or the sole display of a single display Lister).

			<p><i>Example: Set</i> <b>AUTOSIZECOLUMNS=both</b></p>
		<b>focus</b>	<p>Resize all columns in the file display that currently has input focus.</p> <p><i>Example: Set</i> <b>AUTOSIZECOLUMNS=focus</b></p>
		<b>widest</b>	<p>Resizes the columns on both sides of a dual display Lister, setting the columns on both sides to the same width (the widest of the two).</p> <p><i>Example: Set</i> <b>AUTOSIZECOLUMNS=widest</b></p>
BLURFILENAMES	/K	<b>on</b>	<p>Turns on blurring of filenames in the source Lister. You might want to do this in order to use an external screenshot tool to take a screenshot while hiding potentially sensitive information.</p> <p><i>Example: Set</i> <b>BLURFILENAMES=on</b></p>
		<b>off</b>	<p>Turns filename blurring off in the source Lister.</p> <p><i>Example: Set</i> <b>BLURFILENAMES=off</b></p>
		<b>toggle</b>	<p>Toggles filename blurring on or off.</p> <p><i>Example: Set</i> <b>BLURFILENAMES=toggle</b></p>
CALCFOLDERSIZES	/K	<b>all</b>	<p>Turns on the Preferences option to calculate folder sizes automatically (for all folders). Note that this simply modifies the Preferences setting - any currently open Listers won't calculate their folder sizes until they are refreshed.</p> <p><i>Example: Set</i> <b>CALCFOLDERSIZES=all</b></p>
		<b>local</b>	<p>Turns on the option to calculate folder sizes automatically for all local drives.</p>

			<p><i>Example: Set</i> <b>CALCFOLDERSIZES=local</b></p>
		<b>fixed</b>	<p>Turns on the option for all fixed local drives.</p> <p><i>Example: Set</i> <b>CALCFOLDERSIZES=fixed</b></p>
		<b>off</b>	<p>Turns automatic folder size calculation off.</p> <p><i>Example: Set</i> <b>CALCFOLDERSIZES=off</b></p>
		<b>toggle</b>	<p>Toggles calculation on or off for the specified folder type.</p> <p><i>Example: Set</i> <b>CALCFOLDERSIZES=local,toggle</b></p>
		<b>skipjunctions</b>	<p>Specify this flag as well to turn on the <i>Skip junctions and softlinks</i> option.</p> <p><i>Example: Set</i> <b>CALCFOLDERSIZES=local,toggle,skipjunctions</b></p>
CHECKBOXMODE	/K	<b>on</b>	<p>Turns <a href="#">Checkbox mode</a> on in the source file display.</p> <p><i>Example: Set</i> <b>CHECKBOXMODE=on</b></p>
		<b>off</b>	<p>Turns Checkbox mode off in the source file display.</p> <p><i>Example: Set</i> <b>CHECKBOXMODE=off</b></p>
		<b>toggle</b>	<p>Toggles Checkbox mode on or off.</p> <p><i>Example: Set</i> <b>CHECKBOXMODE=toggle</b></p>
CLEARFILTERS	/S	<i>(no value)</i>	<p>Clears all file and folder filters in the current source file display. The filters that are cleared are those controlled by the <b>HIDEFILTERFILENAME</b>, <b>HIDEFILTERFOLDERS</b>, <b>SHOWFILTERFILENAME</b> and <b>SHOWFILTERFOLDERS</b> arguments.</p> <p><i>Example: Set</i> <b>CLEARFILTERS</b></p>

COLUMNS	/K	<column>, ...	<p>Changes which columns are displayed in the current source file display. You can specify one or more comma-separated <a href="#">column keywords</a> - the columns will be displayed in the order specified. Note that the <b>Name</b> column must always be present and will be added automatically if you don't specify it.</p> <p><i>Example: Set COLUMNS name,sizeauto,desc,attr</i></p>
COLUMNSADD	/K	<column>, ...	<p>Adds the specified columns to the current source file display. Can also be used to move or resize existing columns. You can specify one or more comma-separated <a href="#">column keywords</a>.</p> <p>Each column name can optionally be followed by the position to insert the column, and the width to make the new column. The format of this is as follows:</p> <ul style="list-style-type: none"> <li>• (&lt;pos&gt;) - Specify the position of the column (0 is the left-most).</li> <li>• (&lt;pos&gt;,&lt;size&gt;) - Specify both the position and size.</li> <li>• (&lt;pos&gt;,&lt;size&gt;,&lt;max&gt;) - Specify position, size and maximum width.</li> </ul> <p>The &lt;pos&gt; argument indicates the position to insert the column. This can be a number where <b>0</b> represents the left-most column, <b>1</b> the second column and so on. The position can also be specified relative to existing columns. For example, to add a column immediately after the <i>Name</i> column you would specify <b>1+Name</b> for the position. To add it immediately before the <i>Name</i> column you would use <b>1-Name</b>. You can also set pos to <b>!</b> to leave it unchanged, or prefix it with <b>!</b> to say it should be unchanged if the column is already present, but use the</p>

position specified after the **!** if it is being added.

To specify a size the `<pos>` argument must be included first - if you want to specify a size without a position, use `*` for `<pos>` to have the column added at the end.

The `<size>` and `<max>` values of existing columns can also be changed by specifying a **!** for `<pos>`.

The `<size>` argument indicates the width of the new column. This can be a number in pixels, **a** for *Auto*, **f** for *Fill*, **e** for *Expand* and **c** for *Collapse*.

The `<max>` argument lets you specify the maximum width for automatically-sized columns. This can be a number in pixels, or **f** for *Fill*.

Within the **Lister Column Header Context Menu**, you can use two special values for the position:

- **%header%** - Turns into the name of the column which was right-clicked. (Typically used to remove the column which was right-clicked.)
- **%headeritem%** Turns into the position nearest where you right-clicked. (Typically used to insert a column nearest where you right-clicked.)

If the position is not specified, the column will be added to the end of the existing columns.

			<p>Specifying the size of columns only works if the auto-size flag is turned off in Folder Options. If an asterisk (*) is specified for the size value, the column will be automatically sized to fit the content.</p> <p><i>Example: Set COLUMNSADD picwidth</i>  <i>Example: Set COLUMNSADD desc(2),author(*,*)</i>  <i>Example: Set COLUMNSADD picwidth(*,a)</i>  <i>Example: Set COLUMNSADD=Status(!1+Name)</i></p>
COLUMNSREMOVE	/O	<column>, ...	<p>Removes the specified columns from the current source display. You can specify one or more comma-separated <a href="#">column keywords</a>.</p> <p><i>Example: Set COLUMNSREMOVE=mp3bitrate, mp3samplerate</i></p> <p>Within the <b>Lister Column Header Context Menu</b>, you can use <b>%header%</b> to refer to the column which was right-clicked.</p> <p><i>Example: Set COLUMNSREMOVE=%header%</i></p>
COLUMNSTOGGLE	/K	<column>, ...	<p>Toggles the specified columns on or off in the current source display. The position and size of added columns can be given as for <b>COLUMNSADD</b>.</p> <p>You can specify one or more comma-separated <a href="#">column keywords</a>. If more than one column name is provided, the named columns will only be turned off if <b>all</b> specified columns are currently present. Otherwise, those columns not currently present will be added.</p> <p>You can also specify the <b>columnlist</b> keyword, which causes Opus to</p>

			<p>automatically generate a list of columns (in sub-menus for categories).</p> <p>Within the <b>Lister Column Header Context Menu</b> only:</p> <ul style="list-style-type: none"> <li>You can use <b>%headeritem%</b> for the position to insert columns nearest where you right-clicked instead of at the end.</li> <li>Specifying <b>columnlist,insert</b> generates a menu of columns which insert themselves nearest where you right-clicked (and you do not need to use <b>%headeritem%</b> for the position as it is done automatically).</li> </ul> <p><i>Example: Set</i>  <b>COLUMNSTOGGLE=desc(2),auth or</b>  <i>Example: Set</i>  <b>COLUMNSTOGGLE=picwidth(%headeritem%)</b>  <i>Example: Set</i>  <b>COLUMNSTOGGLE=columnlist,insert</b>  <i>Example: Set</i>  <b>COLUMNSTOGGLE=Status(1+Name)</b></p>
COMBINESINGLEGROUPS	/K	<b>on</b>	<p>Turns on the <i>When grouped, combine groups with only one member into the "Other" group</i> option for the current file display.</p> <p><i>Example: Set</i>  <b>COMBINESINGLEGROUPS on</b></p>
		<b>off</b>	<p>Turns off the <i>Combine groups</i> option.</p> <p><i>Example: Set</i>  <b>COMBINESINGLEGROUPS=off</b></p>
		<b>toggle</b>	<p>Toggles the <i>Combine groups</i> option on or off.</p> <p><i>Example: Set</i>  <b>COMBINESINGLEGROUPS toggle</b></p>
CONTENTFORMAT	/K	<content group>	<p>Sets the current source file display to use the named <a href="#">content type</a> folder</p>

			format.  <i>Example: Set CONTENTFORMAT Images</i>
COPYFILTER	/K	<b>on</b>	Turns on the recursive <a href="#">copy filter</a> for the active Lister.  <i>Example: Set COPYFILTER=on</i>
		<b>off</b>	Turns off the copy filter in the active Lister.  <i>Example: Set COPYFILTER=off</i>
		<b>toggle</b>	Toggles the copy filter on or off in the active Lister.  <i>Example: Set COPYFILTER=toggle</i>
DELFILTER	/K	<b>on</b>	Turns on the recursive <a href="#">delete filter</a> for the active Lister.  <i>Example: Set DELFILTER=on</i>
		<b>off</b>	Turns off the delete filter in the active Lister.  <i>Example: Set DELFILTER=off</i>
		<b>toggle</b>	Toggles the delete filter on or off in the active Lister.  <i>Example: Set DELFILTER=toggle</i>
DEST	/K	<b>left</b>	Sets the left (or top) file display in a dual-display Lister to be the destination.  <i>Example: Set DEST=left</i>
		<b>right</b>	Sets the right (or bottom) file display to be the destination.  <i>Example: Set DEST=right</i>
		<b>focus</b>	Sets the file display that currently has the input focus to be the destination.  <i>Example: Set DEST=focus</i>
		<b>toggle</b>	Toggles the state (source/destination) of the left and right file displays.  <i>Example: Set DEST=toggle</i>
DISABLEGLOBALHOTKEYS	/K	<b>on</b>	Temporarily disables all global hotkeys. Hotkeys local to a Lister will

			continue to function.  <i>Example: Set <b>DISABLEGLOBALHOTKEYS=on</b></i>
		<b>off</b>	Re-enables all global hotkeys.  <i>Example: Set <b>DISABLEGLOBALHOTKEYS=off</b></i>
		<b>toggle</b>	Toggles all global hotkeys on and off.  <i>Example: Set <b>DISABLEGLOBALHOTKEYS=toggle</b></i>
DUAL	/K	<b>on</b>	Turns on <a href="#">dual-display</a> mode in the active Lister.  <i>Example: Set <b>DUAL=on</b></i>
		<b>off</b>	Turns off dual-display mode in the active Lister.  <i>Example: Set <b>DUAL=off</b></i>
		<b>toggle</b>	Toggles dual-display mode on or off in the active Lister.  <i>Example: Set <b>DUAL=toggle</b></i>
		<b>horiz</b>	Sets dual-display mode to use horizontal layout (one file display above the other). By itself this value will turn dual-display mode on, but you can combine it with <b>toggle</b> to toggle horizontal dual-display on or off. If dual-display mode is already on but the layout is set to vertical, the layout will change to horizontal.  <i>Example: Set <b>DUAL=horiz,toggle</b></i>
		<b>vert</b>	Sets dual-display mode to use vertical layout (one file display next to the other).  <i>Example: Set <b>DUAL=vert</b></i>
		<b>togglelayout</b>	Toggles the layout of dual-display mode between horizontal and vertical. If dual-display mode is not currently active this command has no effect unless the <b>toggle</b> keyword and either the <b>horiz</b> or <b>vert</b> keywords are also given.

			<p>If combined with those other keywords the function will turn on dual-display mode if it's not on already, switch the layout (from horizontal to vertical or vice versa) if dual-display is already on but not in the desired orientation, and close dual-display mode if it is on and already in the desired orientation.</p> <p><i>Example: Set DUAL=togglelayout</i></p>
		<b>source</b>	<p>When dual-display mode is turned on, the newly opened file display will become the source. This value must be combined with one of the other values that actually causes dual-display mode to be switched on.</p> <p><i>Example: Set DUAL=toggle,source</i></p>
		<b>dest</b>	<p>When dual-display mode is turned on, the newly opened file display will become the destination.</p> <p><i>Example: Set DUAL=toggle,dest</i></p>
		<b>right</b>	<p>When dual-display mode is turned off, it will be the right (or bottom) file display that closes.</p> <p><i>Example: Set DUAL=off,right</i></p>
		<b>left</b>	<p>When dual-display mode is turned off, it will be the left (or top) file display that closes.</p> <p><i>Example: Set DUAL=toggle,left</i></p>
		<b>remember</b>	<p>Use this value with the <b>toggle</b> keyword to cause the second file display to remember its path when it is closed and then opened again. If this isn't specified, the newly opened file display's path will be controlled by the <b>Specify initial folder when switching to dual file display</b> option on the <a href="#">File Displays / Options</a> page in Preferences.</p> <p><i>Example: Set DUAL=toggle,horiz,remember</i></p>

DUALSIZE	/K	<size>[,<size>]	<p>Adjusts the splitter between the dual file displays in the active Lister. If the displays are arranged vertically, the command will affect their widths; if they are arranged horizontally it will affect their heights.</p> <p>The size is given as a percentage, specifying how much of the available space the first file display should use, with the second file display getting whatever is left.</p> <p>For example, specify 50 to make both file displays the same size (the same as double-clicking the splitter between them):</p> <p><i>Example: Set DUALSIZE 50</i></p> <p>As another example, specify 75 to make the first file display use 75% of the space to leave 25% of the space for the second:</p> <p><i>Example: Set DUALSIZE 75</i></p> <p>It is also possible to specify two sizes to make the command toggle between them. This lets you create a button or hotkey to quickly toggle between giving most of the space to one display and making them equal again.</p> <p><i>Example: Set DUALSIZE 75,50</i></p> <p>You can also resize the splitter by a relative amount by specifying a positive or negative delta.</p> <p><i>Example: Set DUALSIZE +10</i></p>
ENABLELABELFILTER	/K	<name>	<p>This command allows named <a href="#">wildcard labels and label filters</a> to be turned on or off. The specified name must have been assigned to the filter before you can control it via this command. Both global and Folder Format-based label filters are supported. You can specify <b>local:&lt;name&gt;</b> and <b>global:&lt;name&gt;</b> to restrict the type of filter you want to</p>

			control, or just provide the name and Opus will look for it in both types of filter.  <i>Example: Set</i> <b>ENABLELABELFILTER</b> <b>my_filter,toggle</b>
		<b>on</b>	Turns the specified label filter on.  <i>Example: Set</i> <b>ENABLELABELFILTER</b> <b>my_filter,on</b>
		<b>off</b>	Turns the specified label filter off.  <i>Example: Set</i> <b>ENABLELABELFILTER</b> <b>my_filter,off</b>
		<b>toggle</b>	Toggles the specified label filter on or off.  <i>Example: Set</i> <b>ENABLELABELFILTER</b> <b>my_filter,toggle</b>
FDBTOOLBAR	/O	<name>	This command lets you change which toolbar is used for the <a href="#">File Display</a> . If you don't specify a name the default File Display Toolbar is selected.  <i>Example: Set FDBTOOLBAR "My FDB Toolbar"</i>
		<b>!static</b>	Turns off the FDB toolbar altogether (reverting to a static header).  <i>Example: Set FDBTOOLBAR !static</i>
FILTERS	/K	<b>on</b>	Turns both the copy and delete <a href="#">recursive filters</a> for the active Lister.  <i>Example: Set FILTERS=on</i>
		<b>off</b>	Turns both recursive filters off in the active Lister.  <i>Example: Set FILTERS=off</i>
		<b>toggle</b>	Toggles both recursive filters on or off in the active Lister.  <i>Example: Set FILTERS=toggle</i>

FLATVIEW	/K	<b>on</b>	Turns <a href="#">Flat View</a> mode on in the source file display.  <i>Example: Set <b>FLATVIEW=on</b></i>
		<b>off</b>	Turns Flat View mode off in the source file display.  <i>Example: Set <b>FLATVIEW=off</b></i>
		<b>toggle</b>	Toggles Flat View on or off in the current source file display. If combined with one of the mode keywords ( <b>group</b> , <b>mixed</b> , <b>mixednofolders</b> ), Flat View will only be turned off if it is currently in the specified mode - otherwise, it will be set to that mode (and turned on if needed).  <i>Example: Set <b>FLATVIEW=toggle,grouped</b></i>
		<b>toggleoff</b>	Toggles Flat View on or off. Unlike <b>toggle</b> , Flat View will be turned off if it is currently enabled in <i>any</i> mode, even if the mode does not match the specified keyword.  <i>Example: Set <b>FLATVIEW=mixednofolders,toggleoff</b></i>
		<b>grouped</b>	Sets Flat View to <i>Grouped</i> mode.  <i>Example: Set <b>FLATVIEW=grouped</b></i>
		<b>mixed</b>	Sets Flat View to <i>Mixed</i> mode.  <i>Example: Set <b>FLATVIEW=toggle,mixed</b></i>
		<b>mixednofolders</b>	Sets Flat View to <i>Mixed (No Folders)</i> mode.  <i>Example: Set <b>FLATVIEW=mixednofolders,on</b></i>
FOCUS	/K	<b>left</b>	Sets the input focus to the left-hand file display in a dual-display Lister.  <i>Example: Set <b>FOCUS=left</b></i>
		<b>right</b>	Sets the input focus to the right-hand file display in a dual-display Lister.  <i>Example: Set <b>FOCUS=right</b></i>

	<b>source</b>	Sets the input focus to the current source file display.  <i>Example: Set FOCUS=source</i>
	<b>dest</b>	Sets the input focus to the destination file display.  <i>Example: Set FOCUS=dest</i>
	<b>tree</b>	Sets the input focus to the folder tree. In a dual-display Lister, with dual trees, focus will go to the tree attached to the source file display.  <i>Example: Set FOCUS=tree</i>
	<b>lefttree</b>	Sets the input focus to the left-hand folder tree in a dual-display, dual-tree Lister.  <i>Example: Set FOCUS=lefttree</i>
	<b>righttree</b>	Sets the input focus to the right-hand folder tree in a dual-display Lister.  <i>Example: Set FOCUS=righttree</i>
	<b>toggle</b>	Toggles the input focus between the left and right file displays.  <i>Example: Set FOCUS=toggle</i>
	<b>pathfield</b>	Sets the focus to the source breadcrumbs path field.  <i>Example: Set FOCUS=pathfield</i>
	<b>leftpathfield</b>	Sets the focus to the left/top path field in a dual-display Lister.  <i>Example: Set FOCUS=leftpathfield</i>
	<b>rightpathfield</b>	Sets the focus to the right/bottom path field in a dual-display Lister.  <i>Example: Set FOCUS=rightpathfield</i>
	<b>filedisplay</b>	When testing which control has focus using <a href="#">@if:set</a> you can use this to mean "any file display" rather than having to specify source/destination etc.  <i>Example: @if:set FOCUS=filedisplay</i>

FOLDERTREESIZE	/K	<size>[,<size>][,left right dest]	<p>Adjusts the size of the folder tree pane in the active Lister. The size is given as an absolute width in pixels. It is possible to specify two separate sizes, and the command will toggle between them. You can also make the command operate on a folder tree other than the one attached to the current source file display by appending the <b>left</b>, <b>right</b> or <b>dest</b> keywords.</p> <p><i>Example: Set FOLDERTREESIZE 200,300</i></p>
FONTSCALE	/K	<absolute factor>	<p>Sets the font scaling in the file display to the specified factor. <b>100</b> (meaning 100%) is the baseline level, and represents the actual point size configured on the <a href="#">Display / Colors and Fonts</a> page in Preferences. <b>200</b> would represent twice as large, <b>50</b> would represent half as large, and so on.</p> <p>Note that font scaling only applies to Details and Power <a href="#">view modes</a>.</p> <p><i>Example: Set FONTSCALE=125</i></p>
		<relative factor>	<p>Adjusts the font scaling in the file display by the specified delta. Use a positive value to increase the scaling and a negative value to decrease it.</p> <p><i>Example: Set FONTSCALE=-10</i></p>
		<factor1>,<factor2>	<p>Specify two absolute scale factors to create a command that toggles between the two.</p> <p><i>Example: Set FONTSCALE=100,150</i></p>
		<b>left</b>	<p>Scales the font in the left-hand file display, whether it is the source or not.</p> <p><i>Example: Set FONTSCALE=50,left</i></p>
		<b>right</b>	<p>Scales the font in the right-hand file display.</p> <p><i>Example: Set FONTSCALE=right,-25</i></p>
		<b>dest</b>	<p>Scales the font in the destination file display.</p>

			<p><i>Example: Set</i> <b>FONTSCALE=dest,+50</b></p>
		<b>both</b>	<p>Scales the font in both the left and right file displays.</p> <p><i>Example: Set</i> <b>FONTSCALE=50,125,both</b></p>
FORMAT	/K	< <i>format</i> >	<p>Applies the named favorite folder format to the current source file display. The format must have been previously created through the <a href="#">Folders / Folder Formats</a> Preferences page.</p> <p><i>Example: Set</i> <b>FORMAT "Photo Viewing"</b></p>
		<b>!factory</b>	<p>Applies the Factory default folder format to the current source file display.</p> <p><i>Example: Set</i> <b>FORMAT !factory</b></p>
		<b>!user</b>	<p>Applies the <b>User default</b> folder format to the current source file display.</p> <p><i>Example: Set</i> <b>FORMAT !custom</b></p>
		<b>!default</b>	<p>Applies the default folder format type for the current path in the source file display. For example, if the current path is a network drive, the <b>Network Drives</b> format would be applied.</p> <p><i>Example: Set</i> <b>FORMAT=!default</b></p>
		<b>!folder</b>	<p>Finds and applies a folder format by using the same rules as when the folder was initially loaded. This gives you the same folder format as going out of the current directory and back into it again. See the <a href="#">Folders / Folder Formats</a> Preferences page for a description of the rules applied when Opus chooses a folder format for a path.</p> <p><i>Example: Set</i> <b>FORMAT !folder</b></p>
FORMATLOCK	/K	<b>on</b>	<p>Turns the <a href="#">format lock</a> on in the current Lister.</p> <p><i>Example: Set</i> <b>FORMATLOCK=on</b></p>

		<b>off</b>	Turns the format lock off in the current Lister.  <i>Example: Set <b>FORMATLOCK=off</b></i>
		<b>toggle</b>	Toggles the format lock on or off in the current Lister. This command can replace the padlock icon in the default status bar.  <i>Example: Set <b>FORMATLOCK=toggle</b></i>
		<b>left</b>	Applies the format lock to only the left (or top) file display in a dual-display Lister. This offers more flexibility than the padlock icon in the status bar - the padlock icon applies to the Lister as a whole, whereas using this command lets you set the format lock on or off for individual file displays.  <i>Example: Set <b>FORMATLOCK=toggle,left</b></i>
		<b>right</b>	Applies the format lock to only the right (or bottom) file display.  <i>Example: Set <b>FORMATLOCK=toggle,right</b></i>
		<b>source</b>	Applies the format lock to only the source file display in a dual-display Lister.  <i>Example: Set <b>FORMATLOCK=toggle,source</b></i>
		<b>dest</b>	Applies the format lock to only the destination file display.  <i>Example: Set <b>FORMATLOCK=toggle,dest</b></i>
FTPMODE	/K	<b>ascii</b>	Sets the file transfer mode for the current FTP connection to ASCII. This command has no effect if the source file display is not currently viewing a remote FTP site.  <i>Example: Set <b>FTPMODE=ascii</b></i>
		<b>binary</b>	Sets the transfer mode to binary for the current FTP connection.  <i>Example: Set <b>FTPMODE=binary</b></i>

		<b>auto</b>	Automatically selects the transfer mode based on the file type being transferred.  <i>Example: Set FTPMODE=auto</i>
FULLROWSELECT	/K	<b>on</b>	Turns full-row selection on. There are separate full-row settings in Preferences for both power ( <a href="#">File Display Modes / Power Mode</a> ) and details ( <a href="#">File Display Modes / Details view modes</a> ), and by default this command will affect the setting for the current view mode in the source file display. You can use the other keywords for this argument to control which view mode is affected.  <i>Example: Set FULLROWSELECT=on</i>
		<b>off</b>	Turns full-row selection off.  <i>Example: Set FULLROWSELECT=off</i>
		<b>toggle</b>	Toggles full-row selection on or off.  <i>Example: Set FULLROWSELECT=toggle</i>
		<b>display</b>	Affects the <b>Always highlight full row</b> option in Preferences for the specified view mode. This flag lets the full row be highlighted, but only the filename is active for selection.  <i>Example: Set FULLROWSELECT=display,toggle</i>
		<b>select</b>	Use with the display keyword to toggle between the two modes (full-row selection, and full-row display).  <i>Example: Set FULLROWSELECT=display,select</i>
		<b>power</b>	Only affects the setting for Power mode, irrespective of the current view mode.  <i>Example: Set FULLROWSELECT=toggle,power</i>
		<b>details</b>	Only affects the setting for details mode.

			<p><i>Example: Set</i>  <b>FULLROWSELECT=display,select,details</b></p>
GLOBALHIDEFILENAME	/O	(no value)	<p>Clears the <b>Global hide filter</b> filename filter (on the <a href="#">Folders / Folder Display</a> page in Preferences).</p> <p><i>Example: Set</i>  <b>GLOBALHIDEFILENAME</b></p>
		<pattern>	<p>Sets the <b>Global hide filter</b> filename filter to the specified <a href="#">wildcard pattern</a>.</p> <p>The supplied pattern can be prefixed with <b>regex:</b> to specify the pattern is a regular expression.</p> <p>If the filter is already set to the specified pattern, it will be cleared, making the command automatically act as a toggle.</p> <p><i>Example: Set</i>  <b>GLOBALHIDEFILENAME</b>  <b>"(desktop.ini *.db)"</b></p> <p><i>Example: Set</i>  <b>GLOBALHIDEFILENAME</b>  <b>regex:.db\$</b></p>
GLOBALHIDEFILTER	/K	on	<p>Turns the <b>Enable global wildcard filters</b> option on (on the <a href="#">Folders / Folder Display</a> page in Preferences).</p> <p><i>Example: Set</i>  <b>GLOBALHIDEFILTER on</b></p>
		off	<p>Turns the <b>Enable global wildcard filters</b> option off.</p> <p><i>Example: Set</i>  <b>GLOBALHIDEFILTER off</b></p>
		toggle	<p>Toggles the <b>Enable global wildcard filters</b> option on or off.</p> <p><i>Example: Set</i>  <b>GLOBALHIDEFILTER=toggle</b></p>

GLOBALHIDEFOLDERS	/O	(no value)	<p>Clears the <b>Global hide filter</b> folder filter (on the <a href="#">Folders / Folder Display</a> page in Preferences).</p> <p><i>Example: Set</i> <b>GLOBALHIDEFOLDERS</b></p>
		<pattern>	<p>Sets the <b>Global hide filter</b> folder filter to the specified <a href="#">wildcard pattern</a>.</p> <p>The supplied pattern can be prefixed with <b>regex:</b> to specify the pattern is a regular expression.</p> <p>If the filter is already set to the specified pattern, it will be cleared, making the command automatically act as a toggle.</p> <p><i>Example: Set</i> <b>GLOBALHIDEFOLDERS .svn</b></p>
GLOBALHIDEHIDDEN	/K	on	<p>Turns the global <b>Hide hidden files</b> option on (on the <a href="#">Folders / Folder Display</a> page in Preferences).</p> <p><i>Example: Set</i> <b>GLOBALHIDEHIDDEN on</b></p>
		off	<p>Turns the global <b>Hide hidden files</b> option off.</p> <p><i>Example: Set</i> <b>GLOBALHIDEHIDDEN off</b></p>
		toggle	<p>Toggles the global <b>Hide hidden files</b> option on or off.</p> <p><i>Example: Set</i> <b>GLOBALHIDEHIDDEN=toggle</b></p>
GRIDLINESH	/K	on	<p>Turns horizontal grid lines on in the current file display (only visible in power or details <a href="#">view modes</a>). This command overrides the settings in Preferences (on either the <a href="#">File Display Modes / Power Mode</a> or <a href="#">File Display Modes / Details</a> pages), but changes are only applicable to the current source file display - the global</p>

		<p>Preferences settings are not modified.</p> <p><i>Example: Set <b>GRIDLINESH on</b></i></p>
	<b>off</b>	<p>Turns horizontal grid lines off in the current file display.</p> <p><i>Example: Set <b>GRIDLINESH off</b></i></p>
	<b>toggle</b>	<p>Toggles horizontal grid lines on or off in the current file display. If the <b>reset</b> keyword is also given, the command will toggle between the grid lines specified in the command line, and the current Preferences settings.</p> <p><i>Example: Set <b>GRIDLINESH=toggle</b></i></p>
	<b>reset</b>	<p>Resets the horizontal grid lines settings in the current file display to those defined in Preferences. You can combine this with the <b>toggle</b> keyword to toggle between the Preferences settings and another set of custom settings.</p> <p><i>Example: Set <b>GRIDLINESH=reset,toggle,solid</b></i></p>
	<b>&lt;style&gt;</b>	<p>Sets horizontal grid lines to use the specified style. Use this keyword in conjunction with the <b>on</b>, <b>off</b> or <b>toggle</b> keywords to control which style is displayed by the command.</p> <p>Supported styles are <b>solid</b>, <b>alternate</b>, <b>dot</b>, <b>dash</b>, <b>dashdot</b>, <b>dashdotdot</b> and <b>fill</b>. Note that <b>solid</b> indicates a solid unbroken (single pixel line), whereas <b>fill</b> indicates a solid color fill of alternating colors the height of each row.</p> <p><i>Example: Set <b>GRIDLINESH=toggle,solid</b></i></p>
	<b>color=&lt;color&gt;</b>	<p>Sets the color of the horizontal grid lines. <b>&lt;color&gt;</b> can be specified in either decimal format (<b>rrr,ggg,bbb</b>) or hex format (<b>#rrggbb</b>). Because the <b>color=</b> keyword contains an equals sign, you must enclose the whole value for the <b>GRIDLINES</b> argument in quotes to avoid confusing the</p>

			<p>command parser.</p> <p><i>Example: Set GRIDLINESH</i> <b>"toggle,fill,color=#ff8000"</b></p>
		<b>opacity=&lt;opacity&gt;</b>	<p>Sets the opacity of the horizontal gridlines. &lt;opacity&gt; must be a value from <b>1</b> (nearly transparent) to <b>100</b> (solid). Because the <b>opacity=</b> keyword contains an equals sign, you must enclose the whole value for the <b>GRIDLINES</b> argument in quotes to avoid confusing the command parser.</p> <p><i>Example: Set GRIDLINESH</i> <b>"toggle,solid,color=#808080,opacity=50"</b></p>
GRIDLINESV	/K	<b>on</b>	<p>Turns vertical grid lines on in the current file display (only visible in power or details <a href="#">view modes</a>). This command overrides the settings in Preferences (on either the <a href="#">File Display Modes / Power Mode</a> or <a href="#">File Display Modes / Details</a> pages), but changes are only applicable to the current source file display - the global Preferences settings are not modified.</p> <p><i>Example: Set GRIDLINESV on</i></p>
		<b>off</b>	<p>Turns vertical grid lines off in the current file display.</p> <p><i>Example: Set GRIDLINESV off</i></p>
		<b>toggle</b>	<p>Toggles vertical grid lines on or off in the current file display. If the <b>reset</b> keyword is also given, the command will toggle between the grid lines specified in the command line, and the current Preferences settings.</p> <p><i>Example: Set GRIDLINESV=toggle</i></p>
		<b>reset</b>	<p>Resets the vertical grid lines settings in the current file display to those defined in Preferences. You can combine this with the <b>toggle</b> keyword to toggle between the Preferences settings and another set of custom settings.</p>

			<p><i>Example: Set</i> <b>GRIDLINESV=reset,toggle,solid</b></p>
		<style>	<p>Sets vertical grid lines to use the specified style. Use this keyword in conjunction with the <b>on</b>, <b>off</b> or <b>toggle</b> keywords to control which style is displayed by the command.</p> <p>Supported styles are <b>solid</b>, <b>alternate</b>, <b>dot</b>, <b>dash</b>, <b>dashdot</b>, and <b>dashdotdot</b>.</p> <p><i>Example: Set</i> <b>GRIDLINESV=toggle,solid</b></p>
		<b>color=&lt;color&gt;</b>	<p>Sets the color of the vertical grid lines. &lt;color&gt; can be specified in either decimal format (<b>rrr,ggg,bbb</b>) or hex format (<b>#rrggbb</b>). Because the <b>color=</b> keyword contains an equals sign, you must enclose the whole value for the <b>GRIDLINES</b> argument in quotes to avoid confusing the command parser.</p> <p><i>Example: Set GRIDLINESV</i> <b>"toggle,fill,color=#ff8000"</b></p>
		<b>opacity=&lt;opacity&gt;</b>	<p>Sets the opacity of the vertical gridlines. &lt;opacity&gt; must be a value from <b>1</b> (nearly transparent) to <b>100</b> (solid). Because the <b>opacity=</b> keyword contains an equals sign, you must enclose the whole value for the <b>GRIDLINES</b> argument in quotes to avoid confusing the command parser.</p> <p><i>Example: Set GRIDLINESV</i> <b>"toggle,solid,color=#808080,opacity=50"</b></p>
GROUPBY	/K	<column>	<p><a href="#">Groups</a> the current file display by the specified column. The value must be one of the valid <a href="#">column keywords</a>.</p> <p>As well as the column keywords, <b>GROUPBY</b> recognizes the special keyword synonyms <b>accessed</b>, <b>created</b>, <b>date</b>, <b>disksize</b>, <b>modified</b>, <b>path</b> and <b>size</b>. This lets you group by date, size or path without needing to know the exact column that is displayed (e.g. the column could be <b>size</b>, <b>sizekb</b> or <b>sizerel</b> - but the sorting is the same in</p>

			<p>all cases, and <b>Set GROUPBY=size</b> would work for any column).</p> <p>The <b>GROUPBY</b> argument also recognizes the special keywords <b>dupes</b> and <b>cdstage</b>. These do not correspond with columns, and are only valid in certain folders.</p> <p><b>Set GROUPBY=dupes</b> can only be used in a <a href="#">file collection</a> that has been used as the target of a <a href="#">Duplicate Files</a> search (to group the list by the duplicate file groups found), and <b>Set GROUPBY=cdstage</b> can only be used on a recordable CD/DVD to group the list into files waiting to be burned and files already on the disk.</p> <p><i>Example: Set GROUPBY=picsize</i></p> <p>Within the <b>Lister Column HeaderContext Menu</b>, you can use <b>%header%</b> to refer to the column which was right-clicked.</p> <p><i>Example: Set GROUPBY=%header%,toggle</i></p>
		<b>toggle</b>	<p>Toggles grouping by the specified column on or off. Note that the column name must come first.</p> <p><i>Example: Set GROUPBY=picsize,toggle</i></p>
		<b>off</b>	<p>Turns grouping off in the current file display.</p> <p><i>Example: Set GROUPBY=off</i></p>
		<b>grouplist</b>	<p>When used on a toolbar or menu, the command will turn into a dynamic list of available columns which can be grouped by.</p> <p><i>Example: Set GROUPBY=grouplist</i></p>
GROUPCOLLAPSE	/K	<b>on</b>	<p>Turns on the Collapsed option for grouping in the current file display.</p>

			<p><i>Example: Set</i> <b>GROUPCOLLAPSE=on</b></p>
		<b>off</b>	<p>Turns off the Collapsed option.</p> <p><i>Example: Set</i> <b>GROUPCOLLAPSE off</b></p>
		<b>toggle</b>	<p>Toggles the Collapsed option on and off for the current file display. Note that you need to reread the folder to see the result of the change.</p> <p><i>Example: Set</i> <b>GROUPCOLLAPSE=toggle</b></p>
GROUPFOLDERSATTOP	/K	<b>on</b>	<p>Turns on the <b>Keep folders at top when grouped</b> option in the current file display.</p> <p><i>Example: Set</i> <b>GROUPFOLDERSATTOP=on</b></p>
		<b>off</b>	<p>Turns off the <b>Keep folders at top when grouped</b> option.</p> <p><i>Example: Set</i> <b>GROUPFOLDERSATTOP=off</b></p>
		<b>toggle</b>	<p>Toggles the Keep folders at top when grouped option on or off in the current file display.</p> <p><i>Example: Set</i> <b>GROUPFOLDERSATTOP=toggle</b></p>
GROUPINDIVIDUAL	/K	<b>on</b>	<p>Turns on the <b>Individual Groups</b> option. If this is turned on and the file display is grouped, one group will be created for each distinct value rather than a range of values falling into a single group (e.g. instead of A-H you would have A, B, C, D, E, F, G, H). This is only really useful for text fields like "User description".</p> <p><i>Example: Set</i> <b>GROUPINDIVIDUAL=on</b> <b>GROUPBY=userdesc</b></p>
		<b>off</b>	<p>Turns individual grouping off.</p> <p><i>Example: Set</i> <b>GROUPINDIVIDUAL=off</b></p>

		<b>toggle</b>	Toggles individual grouping on or off in the current file display.  <i>Example: Set <b>GROUPINDIVIDUAL=toggle</b></i>
GROUPREVERSE	/K	<b>on</b>	Reverses the direction of grouping in the current file display. The actual order of the groups is reversed, not the order of files within the groups.  <i>Example: Set <b>GROUPREVERSE=on GROUPBY=picsize</b></i>
		<b>off</b>	Turns reverse grouping off.  <i>Example: Set <b>GROUPREVERSE=off</b></i>
		<b>toggle</b>	Toggles reverse grouping on or off in the current file display.  <i>Example: Set <b>GROUPREVERSE=toggle</b></i>
HIDE	/S	(no value)	Hides any <b>Set</b> command toolbar button that would ordinarily be disabled because the function is not available. This argument does nothing on its own - it is only used in conjunction with other <b>Set</b> command arguments.  For example, the command <b>Set FTPMODE=ascii</b> would normally be disabled on the toolbar when not currently in an FTP folder, but the command <b>Set FTPMODE=ascii HIDE</b> would cause the button to be removed from the toolbar instead of just being disabled.  <i>Example: Set <b>FLATVIEW=toggle,grouped HIDE</b></i>
HIDEEXT	/K	<b>on</b>	Turns the <i>Hide file extension in Filename column</i> option on in the current file display.  <i>Example: Set <b>HIDEEXT=on</b></i>
		<b>off</b>	Turns the <i>Hide file extension</i> option off.  <i>Example: Set <b>HIDEEXT=off</b></i>

		<b>toggle</b>	<p>Toggles the <i>Hide file extension</i> option on or off in the current file display.</p> <p><i>Example: Set HIDEEXT=toggle</i></p>
HIDEFILTERATTR	/O	(no value)	<p>Clears the attributes hide filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder Options</a> dialog is <i>Filters / Hide Filter / Attributes</i>.</p> <p><i>Example: Set HIDEFILTERATTR</i></p>
		<attributes>	<p>Sets the attributes hide filter in the source file display. Files that have all the specified attributes set will be hidden from the display of the current folder.</p> <p>The &lt;attributes&gt; value is one or more of the following letters: <b>R</b> (read-only), <b>A</b> (archive), <b>H</b> (hidden), <b>S</b> (system), <b>E</b> (encrypted), <b>C</b> (compressed), <b>O</b> (offline), <b>I</b> (non-indexed), <b>P</b> (pinned). See <a href="#">Changing Attributes</a> for detailed descriptions.</p> <p><i>Example: Set HIDEFILTERATTR hs</i></p> <p>If the specified attributes are already set as the filter, the filter will be cleared, making the command automatically act as a toggle. You can also specify two sets of attributes, and the command will alternate between them each time it is run.</p> <p><i>Example: Set HIDEFILTERATTR hs,rhs</i></p>
HIDEFILTERFILENAME	/O	(no value)	<p>Clears the filename hide filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder</a></p>

			<p><a href="#">Options</a> dialog is <i>Filters / Hide Filter / Filename</i>.</p> <p><i>Example: Set</i> <b>HIDEFILTERFILENAME</b></p>
		<pattern>	<p>Sets the filename hide filter in the source file display to the specified <a href="#">wildcard pattern</a>. Files that match the pattern will be hidden from the display of the current folder.</p> <p>The supplied pattern can be prefixed with <b>regex</b>: to specify the pattern is a regular expression.</p> <p>If the specified pattern is already set as the filter, the filename filter will be cleared, making the command automatically work as a toggle.</p> <p><i>Example: Set</i> <b>HIDEFILTERFILENAME</b> <b>*.(jpg bmp png gif)</b></p>
HIDEFILTERFOLDERATTR	/O	(no value)	<p>Clears the folder attributes hide filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder Options</a> dialog is <i>Filters / Hide Filter / Folder Attributes</i>.</p> <p><i>Example: Set</i> <b>HIDEFILTERFOLDERATTR</b></p>
		<attributes>	<p>Sets the folder attributes hide filter in the source file display. Folders that have all the specified attributes set will be hidden from the display of the current folder.</p> <p>The &lt;attributes&gt; value is one or more of the following letters: <b>R</b> (read-only), <b>A</b> (archive), <b>H</b> (hidden), <b>S</b> (system), <b>E</b> (encrypted), <b>C</b> (compressed), <b>O</b> (offline), <b>I</b> (non-indexed), <b>P</b> (pinned). See <a href="#">Changing Attributes</a> for detailed descriptions.</p>

			<p><i>Example: Set</i> <b>HIDEFILTERFOLDERATTR h</b></p> <p>If the specified attributes are already set as the filter, the filter will be cleared, making the command automatically act as a toggle. You can also specify two sets of attributes, and the command will alternate between them each time it is run.</p> <p><i>Example: Set</i> <b>HIDEFILTERFOLDERATTRh,ce</b></p> <p><i>Example: Set</i> <b>HIDEFILTERFOLDERATTRh,off</b></p>
		<b>off</b>	<p>Disables the separate folder attributes hide filter. When the folder attributes filter is disabled, the regular attributes filter will apply to both files and folders.</p> <p><i>Example: Set</i> <b>HIDEFILTERFOLDERATTR off</b></p>
HIDEFILTERFOLDER S	/O	(no value)	<p>Clears the folders hide filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder Options</a> dialog is <i>Filters / Hide Filter / Folders</i>.</p> <p><i>Example: Set</i> <b>HIDEFILTERFOLDERS</b></p>
		<pattern>	<p>Sets the folders hide filter in the source file display to the specified <a href="#">wildcard pattern</a>. Folders whose name matches the pattern will be hidden from the display of the current folder.</p> <p>The supplied pattern can be prefixed with <b>regex:</b> to specify the pattern is a regular expression.</p>

			<p>If the specified pattern is already set as the filter, the name filter will be cleared, making the command automatically work as a toggle.</p> <p><i>Example: Set</i> <b>HIDEFILTERFOLDERS .svn</b></p>
HIDESYSTEMFILES	/K	on	<p>Turns on the <b>Hide protected operating system files</b> option on the <a href="#">Folders / Folder Display</a> page in Preferences. This option causes all files and folders with both the H (hidden) and S (system) attributes to be hidden.</p> <p><i>Example: Set</i> <b>HIDESYSTEMFILES on</b></p>
		off	<p>Turns off the <b>Hide protected operating system files</b> option.</p> <p><i>Example: Set</i> <b>HIDESYSTEMFILES=off</b></p>
		toggle	<p>Toggles the <b>Hide protected operating system files</b> option on or off.</p> <p><i>Example: Set</i> <b>HIDESYSTEMFILES=toggle</b></p>
ICONMODESORTHEADER	/K	on	<p>Turns on the display of the column header (for sorting) in the icon modes (large icon, thumbnail, etc). This controls the <i>Show sort header in icon modes</i> on the <a href="#">File Displays / Options</a> page in Preferences.</p> <p><i>Example: Set</i> <b>ICONMODESORTHEADER=on</b></p>
		off	<p>Turns off the column header in the icon modes.</p> <p><i>Example: Set</i> <b>ICONMODESORTHEADER=off</b></p>

		<b>toggle</b>	Toggles the icon mode column header on and off.  <i>Example: Set <b>ICONMODESORTHEADER=toggle</b></i>
ICONS	/K	<b>on</b>	Enables the display of icons in power and details view modes in the current file display. This overrides the setting on the appropriate page in Preferences ( <a href="#">File Display Modes / Details</a> or <a href="#">File Display Modes / Power Mode</a> ).  <i>Example: Set <b>ICONS=on</b></i>
		<b>off</b>	Turns off the display of icons in power and details modes for the current file display.  <i>Example: Set <b>ICONS=off</b></i>
		<b>toggle</b>	Toggles the display of icons in power and details modes.  <i>Example: Set <b>ICONS=toggle</b></i>
		<b>reset</b>	Resets the icon display to the current Preferences settings.  <i>Example: Set <b>ICONS=reset</b></i>
INVERT	/S	<i>(no value)</i>	Inverts the appearance of toolbar buttons that appear highlighted (or checked) when the <b>Set</b> option they control is currently on.  For example, a command like <b>Set TREE=toggle</b> will appear highlighted on the toolbar when the folder tree is displayed. Changing the command to <b>Set TREE=toggle INVERT</b> would cause the toolbar button to appear highlighted when the tree is not displayed.  <i>Example: Set <b>HIDESYSTEMFILES=toggle INVERT</b></i>
JOBSBAR	/K	<b>on</b>	Displays the <a href="#">jobs bar</a> in the current Lister.  <i>Example: Set <b>JOBSBAR=on</b></i>

		<b>off</b>	Hides the <a href="#">jobs bar</a> in the current Lister.  <i>Example: Set <b>JOBSBAR=off</b></i>
		<b>toggle</b>	Toggles the <a href="#">jobs bar</a> on and off.  <i>Example: Set <b>JOBSBAR=toggle</b></i>
KEEPFOLDERSALPHA A	/K	<b>on</b>	Turns the <b>Keep folders sorted alphabetically</b> option on in the source file display. This modifies the <a href="#">Folder Options</a> for the current folder.  <i>Example: Set <b>KEEPFOLDERSALPHA=on</b></i>
		<b>off</b>	Turns the <b>Keep folders sorted alphabetically</b> option off.  <i>Example: Set <b>KEEPFOLDERSALPHA off</b></i>
		<b>toggle</b>	Toggles the state of the <b>Keep folders sorted alphabetically</b> option.  <i>Example: Set <b>KEEPFOLDERSALPHA=toggle</b></i>
LAYOUT	/K	<b>remember</b>	Remembers the current layout and appearance of the active Lister.  <i>Example: Set <b>LAYOUT=remember</b></i>
		<b>restore</b>	Restores the previously remembered layout and appearance of the active Lister. For example, if you run the <b>Set LAYOUT=remember</b> command, and then make changes to the Lister like closing the tree, opening the viewer pane, or selecting a new <a href="#">style</a> , the <b>Set LAYOUT=restore</b> command would restore the Lister to its original state.  <i>Example: Set <b>LAYOUT=restore</b></i>
LISTERCMD	/K	<b>minimize</b>	Minimizes the currently active Lister window.  <i>Example: Set <b>LISTERCMD=minimize</b></i>
		<b>maximize</b>	Maximizes the currently active Lister window.  <i>Example: Set <b>LISTERCMD=maximize</b></i>

	<b>restore</b>	Restores the original size and position of the window (before it was either minimized or maximized).  <i>Example: Set LISTERCMD=restore</i>
	<b>togglemaximize</b>	If the current Lister window is not maximized, it will be maximized, otherwise it will be restored. You could use this to add a hotkey that switches a Lister in and out of "full-screen" mode.  <i>Example: Set LISTERCMD=togglemaximize</i>
	<b>showall</b>	Makes all currently open Listers visible. Minimized windows will be restored, and all Lister windows will come to the front.  <i>Example: Set LISTERCMD=showall</i>
	<b>minimizeall</b>	Minimizes all currently open Listers.  <i>Example: Set LISTERCMD=minimizeall</i>
	<b>tileh</b>	Tiles all currently open Listers horizontally across the screen.  <i>Example: Set LISTERCMD=tileh</i>
	<b>tilev</b>	Tiles all Listers vertically across the screen.  <i>Example: Set LISTERCMD=tilev</i>
	<b>cascade</b>	Cascades all Lister windows. All windows are made the same size and positioned staggered diagonally down and across the screen.  <i>Example: Set LISTERCMD=cascade</i>
	<b>toggleminimizeall</b>	Minimizes all currently open Lister windows. If all windows are already minimized they will all be restored.  <i>Example: Set LISTERCMD=toggleminimizeall</i>
	<b>ontopon</b>	Sets the Lister's "on top" state to on (so it appears above all non-topmost windows).

			<p><i>Example: Set</i> <b>LISTERCMD=ontopon</b></p>
		<b>ontopoff</b>	<p>Sets the Lister's "on top" state to off (so it can go behind normal windows).</p> <p><i>Example: Set</i> <b>LISTERCMD=ontopoff</b></p>
		<b>tofront</b>	<p>Brings the Lister window to the front. This command is most useful when run from a <a href="#">script</a>.</p> <p><i>Example: Set</i> <b>LISTERCMD=tofront</b></p>
LISTERPOS	/K	<x>,<y>	<p>Sets the position of the active Lister to the specified x and y coordinates. You can also specify a delta to modify the current position.</p> <p><i>Example: Set</i> <b>LISTERPOS=500,200</b> <i>Example: Set</i> <b>LISTERPOS=+100,+50</b></p>
LISTERSIZE	/K	<w>,<h>	<p>Sets the size of the currently active Lister to the specified width and height. You can also specify a delta to modify the current size.</p> <p><i>Example: Set</i> <b>LISTERSIZE=1024,768</b> <i>Example: Set</i> <b>LISTERSIZE=+50,+50</b></p>
		<b>auto</b>	<p>Automatically resizes the Lister horizontally (as much as possible) to exactly fit the currently displayed columns (only in Details or Power mode).</p> <p><i>Example: Set</i> <b>LISTERSIZE=auto</b></p>
LISTERTITLE	/O	(no value)	<p>Resets the title of the current Lister back to its default.</p> <p><i>Example: Set</i> <b>LISTERTITLE</b></p>
		<custom title>	<p>Sets a custom title for the currently active Lister. You can use several special "tokens" in the title string to insert various pieces of information:</p> <p><b>%P</b> - full path of the current (source) folder <b>%N</b> - name of the current (source) folder</p>

			<p><b>%R</b> - drive root of the current (source) folder  <b>%D</b> - full path of the destination folder  <b>%M</b> - name of the destination folder  <b>%G</b> - target if the folder is a junction or softlink  <b>%1</b> - full path in the left file display  <b>%2</b> - full path in the right file display  <b>%3</b> - folder name in the left file display  <b>%4</b> - folder name in the right file display  <b>%L</b> - name of the Layout the Lister came from (if any)  <b>%T</b> - complete original title (useful for simply adding a prefix or suffix)</p> <p><i>Example: Set LISTERTITLE "Directory Opus - %N"</i></p> <p>This command normally act as a toggle, such that the title will be cleared if you run the command when the specified title is already set. You can prevent this by prefixing the title with "notoggle:". This can be useful in event-driven scripts which may make redundant requests to set the title and would have to check its current value to avoid resetting it otherwise.</p> <p><i>Example: Set LISTERTITLE "notoggle:Directory Opus - %N"</i></p>
MANUALSORT	/K	<b>on</b>	<p>Turn on <a href="#">manual sorting</a> in the current file display. The default manual sort order will be used unless alternative manual sort names have been configured and you specify the name using the <code>&lt;name&gt;</code> parameter.</p> <p><i>Example: Set MANUALSORT=on,MySortOrder</i></p>

		<b>off</b>	Turn off manual sorting in the current source file display.  <i>Example: Set <b>MANUALSORT=off</b></i>
		<b>toggle</b>	Toggle the manual sort mode on or off in the current file display.  <i>Example: Set <b>MANUALSORT=toggle</b></i>
		<name>	Specifies an alternative name for the sort order to use, which must first have been configured using the <b>manual_sort_names</b> option on the <a href="#">Miscellaneous / Advanced</a> page in Preferences.  <i>Example: Set <b>MANUALSORT=MySortOrder,toggle</b></i>
MANUALSORTRESET	/O	(no value)	Resets the current manual sort order for the folder in the source file display. The file list will be resorted using the current (non-manual) sort method and your old manual sort order will be discarded.  <i>Example: Set <b>MANUALSORTRESET</b></i>
		<name>	Resets the named manual sort order for the current folder. The name must first have been configured using the <b>manual_sort_names</b> option on the <a href="#">Miscellaneous / Advanced</a> page in Preferences.  <i>Example: Set <b>MANUALSORTRESET=MySortOrder</b></i>
		<b>!default</b>	Resets the default manual sort order for the current folder.  <i>Example: Set <b>MANUALSORTRESET=!default</b></i>

		<b>!all</b>	Resets all manual sort orders (default and named) for the current folder.  <i>Example: Set</i> <b>MANUALSORTRESET=!all</b>
MANUALSORTSAVE	/S	(no value)	Saves the current manual sort order in the current folder. You would only need to use this command if you don't have automatic saving of manual sort orders enabled.  <i>Example: Set</i> <b>MANUALSORTSAVE</b>
METAPANE	/K	<b>on</b>	Turns the <a href="#">metadata pane</a> on in the currently active Lister.  <i>Example: Set</i> <b>METAPANE=on</b>
		<b>off</b>	Turns the metadata pane off in the active Lister.  <i>Example: Set</i> <b>METAPANE=off</b>
		<b>toggle</b>	Toggles the metadata pane on or off.  <i>Example: Set</i> <b>METAPANE=toggle</b>
		<b>horiz</b>	Forces the metadata pane to horizontal layout when it is opened.  <i>Example: Set</i> <b>METAPANE=toggle,horiz</b>
		<b>vert</b>	Specifies vertical layout for the metadata pane.  <i>Example: Set</i> <b>METAPANE=on,vert</b>
		<b>togglelayout</b>	Toggles the layout of the metadata pane between vertical and horizontal.  <i>Example: Set</i> <b>METAPANE=togglelayout</b>
METAPANESIZE	/K	<size>[,<size>]	Adjusts the size of the metadata pane in the active Lister. The size is given as a percentage of the total size of the Lister, and applies in the appropriate dimension based on the current layout of the metadata pane (so for example, when the pane is horizontal this affects its height).  It is also possible to specify two

			<p>separate sizes, and the command will toggle between them.</p> <p><i>Example: Set METAPANESIZE 25,50</i></p>
NAVLOCK	/K	<b>on</b>	<p>Turns <a href="#">navigation lock</a> on in the current Lister. This command is only available if the Lister is in <a href="#">dual-display</a> mode.</p> <p><i>Example: Set NAVLOCK=on</i></p>
		<b>off</b>	<p>Turns navigation lock off in the active Lister.</p> <p><i>Example: Set NAVLOCK=off</i></p>
		<b>toggle</b>	<p>Toggles navigation lock on or off in the active Lister.</p> <p><i>Example: Set NAVLOCK=toggle</i></p>
NOSCRIPT	/S	(no value)	<p>Use the <b>NOSCRIPT</b> argument to prevent <a href="#">script events</a> from firing in response to the Set command. Currently this only works with the <b>Set LISTERCMD=ToFront</b> command, to prevent the <a href="#">OnActivateLister</a> event from being triggered.</p> <p><i>Example: Set LISTERCMD=ToFront NOSCRIPT</i></p>
QUICKFILTER	/O	(no value)	<p>Clears the "quick filter" pattern in the current source file display. This is the same filter edited by the <a href="#">Filter Bar</a>. Note that this does not clear the <b>QUICKFILTERFLAGS</b> value, and so it's possible that files may remain filtered out even after the filter pattern is cleared. Use the <b>QUICKFILTERCLEAR</b> argument to clear the quick filter completely.</p> <p><i>Example: Set QUICKFILTER</i></p>
		<pattern>	<p>Sets the quick filter in the current source file display to the specified <a href="#">wildcard pattern</a>. This is the same filter edited by the <a href="#">Filter Bar</a>. Files that do not match the pattern will be hidden from the display.</p>

			<p>If the filter is already set to the specified pattern, it will be cleared, making the command work as a toggle automatically.</p> <p><i>Example: Set QUICKFILTER *.jpg</i></p>
		<b>!prev</b>	<p>Restores the previous quick filter in the current file display. By default the quick filter is cleared when changing folders (although this can be changed with the <b>Clear Quick filter automatically when changing folders</b> option on the <a href="#">File Displays / Filter Bar</a> page in Preferences). For example, you could assign a hotkey to restore the previous filter after having gone into a sub-directory and then back again to the parent.</p> <p><i>Example: Set QUICKFILTER=!prev</i></p>
QUICKFILTERCLEAR	/S	(no value)	<p>Clears the quick filter in the current source file display. Both the filter pattern and the flags are cleared. This does not affect filtering caused by folder options or the global filters in Preferences.</p> <p><i>Example: Set QUICKFILTERCLEAR</i></p>
QUICKFILTERFLAGS	/O	(no value)	<p>Clears the quick filter flags in the current source file display. The flags are as listed below. Clearing the flags does not clear the filter pattern, so it's possible that files may remain filtered out even after the flags are cleared. Use the <b>QUICKFILTERCLEAR</b> argument to clear the quick filter completely.</p> <p><i>Example: Set QUICKFILTERFLAGS</i></p>
		<b>showfiles</b>	<p>Shows all files, even if they are hidden by the filter pattern. This does not override folder format or global filters.</p> <p><i>Example: Set QUICKFILTERFLAGS=showfiles</i></p>

		<b>showdirs</b>	Shows all folders, even if they are hidden by the filter pattern.  <i>Example: Set</i> <b>QUICKFILTERFLAGS=showdirs</b> <b>QUICKFILTER a*</b>
		<b>hidefiles</b>	Hides all files.  <i>Example: Set</i> <b>QUICKFILTERFLAGS=hidefiles</b>
		<b>hidedirs</b>	Hides all folders.  <i>Example: Set</i> <b>QUICKFILTERFLAGS=hidedirs</b>
		<b>disable</b>	Temporarily disables the quick filter, leaving the filter pattern intact.  <i>Example: Set</i> <b>QUICKFILTERFLAGS=disable</b>
READONLY	/K	<b>on</b>	Makes the current file display read-only. Currently this is only supported by Zip archives. When the file display is marked as read-only, attempts to modify the contents of the current Zip archive will fail. This command has no affect when not viewing a Zip archive.  <i>Example: Set</i> <b>READONLY=on</b>
		<b>off</b>	Clears the read-only flag from the current file display. The <b>Open Zip files as read-only by default</b> option on the <a href="#">Zip &amp; Other Archives / Zip Files</a> page in Preferences can make Zip archives read-only by default, and you can then use this command to make them writeable.  <i>Example: Set</i> <b>READONLY=off</b>
		<b>toggle</b>	Toggles the read-only flag on or off in the current file display.  <i>Example: Set</i> <b>READONLY=TOGGLE</b>
RECYCLEBINEMPTY	/S	<i>(no value)</i>	This has no effect as a command - its only use is with the <b>@ifset</b> and <b>@icon command modifiers</b> . It lets you test if the recycle bin is currently empty.

			<p><i>Example:</i></p> <pre>@ifset:RECYCLEBINE MPTY @confirm The recycle bin is empty! @ifset:else Delete EMPTYRECYCLE</pre>
RELATIVEDATEGRAPHS	/K	<b>on</b>	<p>Turns on the <b>Show relative graphs behind modified date columns</b> option on the <a href="#">Folders / Folder Display</a> page in Preferences. As this is a global setting, all currently open Listers will be affected.</p> <p><i>Example: Set</i> <b>RELATIVEDATEGRAPHS=on</b></p>
		<b>off</b>	<p>Turns off the <b>Show relative date graphs behind modified date columns</b> option.</p> <p><i>Example: Set</i> <b>RELATIVEDATEGRAPHS=off</b></p>
		<b>toggle</b>	<p>Toggles the <b>Show relative date graphs behind modified date columns</b> option on or off.</p> <p><i>Example: Set</i> <b>RELATIVEDATEGRAPHS=toggle</b></p>
RELATIVESIZEGRAPHS	/K	<b>on</b>	<p>Turns on the <b>Show relative graphs behind size columns</b> option on the <a href="#">Folders / Folder Display</a> page in Preferences. As this is a global setting, all currently open Listers will be affected.</p> <p><i>Example: Set</i> <b>RELATIVESIZEGRAPHS=on</b></p>
		<b>off</b>	<p>Turns off the <b>Show relative date graphs behind size columns</b> option.</p>

			<p><i>Example: Set</i> <b>RELATIVESIZEGRAPHS=off</b></p>
		<b>toggle</b>	<p>Toggles the <b>Show relative date graphs behind size columns</b> option on or off.</p> <p><i>Example: Set</i> <b>RELATIVESIZEGRAPHS=toggle</b></p>
RELDIMENSIONOVERLAYS	/K	<b>on</b>	<p>Turns the display of thumbnails-mode relative dimension bars on. This modifies the <b>Overlay relative dimension bars</b> option on the <a href="#">File Display Modes / Thumbnails</a> page in Preferences. As this is a global setting, all currently open Listers will be affected.</p> <p><i>Example: Set</i> <b>RELDIMENSIONOVERLAYS=on</b></p>
		<b>off</b>	<p>Turns the display of thumbnails-mode relative dimension bars off.</p> <p><i>Example: Set</i> <b>RELDIMENSIONOVERLAYS=off</b></p>
		<b>toggle</b>	<p>Toggles the display of thumbnails-mode relative dimension bars on or off.</p> <p><i>Example: Set</i> <b>RELDIMENSIONOVERLAYS=toggle</b></p>
SAVEFORMAT	/O	<i>(no value)</i>	<p>Displays the <i>Save Folder Format</i> dialog, which lets you save the <a href="#">folder format</a> in the source file display.</p> <p><i>Example: Set</i> <b>SAVEFORMAT</b></p>
		<b>folder</b>	<p>Saves the folder format for the current folder (without displaying the <i>Save Folder Format</i> dialog). You can combine this with the <b>replace</b> and <b>subfolders</b> arguments.</p> <p><i>Example: Set</i> <b>SAVEFORMAT=folder</b></p>
		<b>all</b>	<p>Saves the current format for all folders (i.e. this makes it the new <b>User</b></p>

			<p><b>default</b> format). You can combine this with the <b>clear</b>, <b>replace</b> and <b>quiet</b> arguments.</p> <p><i>Example: Set SAVEFORMAT=all</i></p>
		<b>favorite</b>	<p>Saves the current format as a <b>Favorite Format</b>. You can specify the format name using the <b>FORMAT</b> argument.</p> <p><i>Example: Set SAVEFORMAT=favorite FORMAT "My Fave Format"</i></p>
		<b>subfolders</b>	<p>Use with the <b>folder</b> argument to save the folder format for the current folder and all sub-folders.</p> <p><i>Example: Set SAVEFORMAT=folder,subfolders</i></p>
		<b>replace</b>	<p>Use with the <b>folder</b> and <b>all</b> arguments to replace any existing folder formats with the new format.</p> <p><i>Example: Set SAVEFORMAT=all,replace</i></p>
		<b>clear</b>	<p>Use with the <b>all</b> argument to clear any existing saved folder formats (so that the new <b>User default</b> format will be used).</p> <p><i>Example: Set SAVEFORMAT=all,clear</i></p>
		<b>quiet</b>	<p>Use with the <b>all</b> argument to suppress the warning dialog before replacing all existing folder formats.</p> <p><i>Example: Set SAVEFORMAT=all,replace,quiet</i></p>
SCRIPTDISABLE	/K	<b>on</b>	<p>Turns on the global option to disable all script add-ins. This will disable any event scripts, command scripts and column scripts you have installed.</p>

			<i>Example: Set <b>SCRIPTDISABLE=on</b></i>
		<b>off</b>	Turns off the global script disable option.  <i>Example: Set <b>SCRIPTDISABLE=off</b></i>
		<b>toggle</b>	Toggles the global script disable option on or off.  <i>Example: Set <b>SCRIPTDISABLE=toggle</b></i>
SHOWCOMPATIBILITYFILES	/K	<b>on</b>	Turns the <b>Show compatibility files</b> option on in the source file display. This modifies the <a href="#">Folder Options</a> for the current folder. See the section on <a href="#">Compatibility Files</a> for more information about this concept.  <i>Example: Set <b>SHOWCOMPATIBILITYFILES=on</b></i>
		<b>off</b>	Turns the <b>Show compatibility files</b> option off in the source file display.  <i>Example: Set <b>SHOWCOMPATIBILITYFILES=off</b></i>
		<b>toggle</b>	Toggles the <b>Show compatibility files</b> option on or off.  <i>Example: Set <b>SHOWCOMPATIBILITYFILES=toggle</b></i>
SHOWEVERYTHING	/K	<b>on</b>	Turns the <a href="#">Show Everything</a> mode on in the source file display. This overrides any active filters and ensures that all files and folders are visible.  <i>Example: Set <b>SHOWEVERYTHING=on</b></i>
		<b>off</b>	Turns the <i>Show Everything</i> mode off in the source file display.  <i>Example: Set <b>SHOWEVERYTHING=off</b></i>

		<b>toggle</b>	<p>Toggles Show Everything mode on or off.</p> <p><i>Example: Set <b>SHOWEVERYTHING=toggle</b></i></p>
SHOWFILTERATTR	/O	(no value)	<p>Clears the attributes show filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder Options</a> dialog is <i>Filters / Show Filter / Attributes</i>.</p> <p><i>Example: Set <b>SHOWFILTERATTR</b></i></p>
		<attributes>	<p>Sets the attributes show filter in the source file display. Only files that have the specified attributes set will be displayed - all others will be hidden.</p> <p>The &lt;attributes&gt; value is one or more of the following letters: <b>R</b> (read-only), <b>A</b> (archive), <b>H</b> (hidden), <b>S</b> (system), <b>E</b> (encrypted), <b>C</b> (compressed), <b>O</b> (offline), <b>I</b> (non-indexed), <b>P</b> (pinned). See <a href="#">Changing Attributes</a> for detailed descriptions.</p> <p><i>Example: Set <b>SHOWFILTERATTRe</b></i></p> <p>If the specified attributes are already set as the filter, the filter will be cleared, making the command automatically act as a toggle. You can also specify two sets of attributes, and the command will alternate between them each time it is run.</p> <p><i>Example: Set <b>SHOWFILTERATTR e,ra</b></i></p>
SHOWFILTERFILENAME	/O	(no value)	<p>Clears the filename show filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder Options</a> dialog is <i>Filters / Show Filter / Filename</i>.</p>

			<p><i>Example: Set</i> <b>SHOWFILTERFILENAME</b></p> <p>Sets the filename show filter in the source file display to the specified <a href="#">wildcard pattern</a>. Only files that match the pattern will be shown - all other files will be hidden from the display.</p> <p>The supplied pattern can be prefixed with <b>regex:</b> to specify the pattern is a regular expression.</p> <p>If the specified pattern is already set as the filter, it will be cleared, making the command work as a toggle automatically.</p> <p><i>Example: Set</i> <b>SHOWFILTERFILENAME</b> <b>*.(doc xls)</b></p>
SHOWFILTERFOLDE RATTR	/O	(no value)	<p>Clears the folder attributes show filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder Options</a> dialog is <i>Filters / Show Filter / Folder Attributes</i>.</p> <p><i>Example: Set</i> <b>SHOWFILTERFOLDERATTR</b></p>
		<attributes>	<p>Sets the folder attributes show filter in the source file display. Only folders that have the specified attributes set will be displayed - all others will be hidden.</p> <p>The &lt;attributes&gt; value is one or more of the following letters: <b>R</b> (read-only), <b>A</b> (archive), <b>H</b> (hidden), <b>S</b> (system), <b>E</b> (encrypted), <b>C</b> (compressed), <b>O</b> (offline), <b>I</b> (non-indexed), <b>P</b> (pinned). See <a href="#">Changing Attributes</a> for detailed descriptions.</p> <p><i>Example: Set</i> <b>SHOWFILTERFOLDERATTR o</b></p>

			<p>If the specified attributes are already set as the filter, the filter will be cleared, making the command automatically act as a toggle. You can also specify two sets of attributes, and the command will alternate between them each time it is run.</p> <p><i>Example: Set</i>  <b>SHOWFILTERFOLDERATTR  o,off</b></p>
		<b>off</b>	<p>Disables the separate folder attributes show filter. When the folder attributes filter is disabled, the regular attributes filter will apply to both files and folders.</p> <p><i>Example: Set</i>  <b>SHOWFILTERFOLDERATTR off</b></p>
SHOWFILTERFOLDERS	/O	(no value)	<p>Clears the folders show filter in the source file display. This modifies the folder options for the current folder - the equivalent setting in the <a href="#">Folder Options</a> dialog is <i>Filters / Show Filter / Folders</i>.</p> <p><i>Example: Set</i>  <b>SHOWFILTERFOLDERS</b></p>
		<pattern>	<p>Sets the folders show filter in the source file display to the specified <a href="#">wildcard pattern</a>. Only folders whose name matches the pattern will be shown - all other folders will be hidden from the display.</p> <p>The supplied pattern can be prefixed with <b>regex</b>: to specify the pattern is a regular expression.</p> <p>If the specified pattern is already set as the filter, it will be cleared, making the command work as a toggle automatically.</p>

			<p><i>Example: Set</i>  <b>SHOWFILTERFOLDERS "*" Reports *</b></p>
SHOWMILLIS	/K	<b>on</b>	<p>Turns on the display of milliseconds in file time columns. This controls the <a href="#">Preferences / Folders / Folder Display / Show milliseconds</a> option. Note that seconds must also be displayed for this command to have any effect.</p> <p><i>Example: Set SHOWMILLIS=on</i></p>
		<b>off</b>	<p>Turns off the display of milliseconds in file time columns.</p> <p><i>Example: Set SHOWMILLIS=off</i></p>
		<b>toggle</b>	<p>Toggles the display of milliseconds on or off.</p> <p><i>Example: Set SHOWMILLIS=toggle</i></p>
SHOWSECONDS	/K	<b>on</b>	<p>Turns on the display of seconds in file time columns. This controls the <a href="#">Preferences / Folders / Folder Display / Show seconds</a> option.</p> <p><i>Example: Set SHOWSECONDS=on</i></p>
		<b>off</b>	<p>Turns off the display of seconds in file time columns.</p> <p><i>Example: Set SHOWSECONDS=off</i></p>
		<b>toggle</b>	<p>Toggles the display of seconds on or off.</p> <p><i>Example: Set</i>  <b>SHOWSECONDS=toggle</b></p>
SORTBY	/K	<column>, ...	<p><a href="#">Sorts</a> the current file display by the specified column. The value must be one of the valid <a href="#">column keywords</a>, and the column must also be displayed in the file display.</p> <p>As well as the column keywords, <b>SORTBY</b> recognizes the special keyword synonyms <b>accessed</b>, <b>created</b>, <b>date</b>, <b>disksize</b>, <b>modified</b>, <b>path</b> and <b>size</b>. This lets you sort by date, size or path without needing to know the exact column that is displayed (e.g. the column could be <b>size</b>, <b>sizekb</b> or</p>

			<p><b>sizerel</b> - but the sorting is the same in all cases, and <b>Set SORTBY=size</b> would work for any column).</p> <p>It is possible to sort the list by multiple columns, by specifying more than one comma-separated keyword. You can also specify that the sort order for a particular column should be reversed by prefixing its keyword with a hyphen.</p> <p>The <b>Set SORTBY</b> command can also be used to automatically add the specified columns to the file display (since a column needs to be displayed in the list in order to sort by it). Prefix the column with a + sign to enable this. If the column is not already in the list it will be added to the end of the existing columns. You can also specify the position where the column should be added if it doesn't already exist - see the description of the <b>Set COLUMNSADD</b> command for details on this.</p> <p><i>Example: Set SORTBY=picsize,-modified</i>  <i>Example: Set SORTBY=+datetaken</i></p>
		<b>sortlist</b>	<p>When used on a toolbar or menu, the command will turn into a dynamic list of available columns which can be sorted by.</p> <p><i>Example: Set SORTBY=sortlist</i></p>
SORTNAMEEXTSEPARATELY	/K	<b>on</b>	<p>Turns the <b>Sort name and extension separately</b> option on in the source file display. This modifies the <a href="#">Folder Options</a> for the current folder.</p> <p><i>Example: Set SORTNAMEEXTSEPARATELY=on</i></p>
		<b>off</b>	<p>Turns the <b>Sort name and extension separately</b> option off.</p> <p><i>Example: Set</i></p>

			<b>SORTNAMEEXTSEPARATELY=off</b>
		<b>toggle</b>	Toggles the <b>Sort name and extension separately</b> option on or off in the current file display.  <i>Example: Set <b>SORTNAMEEXTSEPARATELY=toggle</b></i>
<b>SORTNUMERIC</b>	<b>/K</b>	<b>on</b>	Turns the <b>Numeric order filename sorting</b> option on in the source file display. This modifies the <a href="#">Folder Options</a> for the current folder.  <i>Example: Set <b>SORTNUMERIC=on</b></i>
		<b>off</b>	Turns the <b>Numeric order filename sorting</b> option off.  <i>Example: Set <b>SORTNUMERIC=off</b></i>
		<b>toggle</b>	Toggles the <b>Numeric order filename sorting</b> option on or off in the current file display.  <i>Example: Set <b>SORTNUMERIC=toggle</b></i>
<b>SORTORDER</b>	<b>/K</b>	<b>folders</b>	Changes the sort order in the source file display so that folders are listed before files. This modifies the <a href="#">Folder Options</a> for the current folder.  <i>Example: Set <b>SORTORDER=folders</b></i>
		<b>files</b>	Lists files before folders in the current file display.  <i>Example: Set <b>SORTORDER=files</b></i>
		<b>mixed</b>	Sorts files and folders together in the current file display.  <i>Example: Set <b>SORTORDER=mixed</b></i>
		<b>cycle</b>	Cycles through the three different ordering options in the current file display.  <i>Example: Set <b>SORTORDER=cycle</b></i>
<b>SORTREVERSE</b>	<b>/K</b>	<b>on</b>	Reverses the sort order in the current file display. If the list is only sorted by one column, the direction of that column sort is reversed. If multiple

		<p>columns are selected for sorting, their directions are not altered but the overall result is reversed as the final step in the sorting.</p> <p>This modifies the <a href="#">Folder Options</a> for the current folder.</p> <p><i>Example: Set SORTREVERSE=on</i></p>
	<b>off</b>	<p>Turns the reverse sort flag off for the current folder, restoring the sort order to normal.</p> <p><i>Example: Set SORTREVERSE=off</i></p>
	<b>toggle</b>	<p>Toggles reverse sort on or off in the current folder.</p> <p><i>Example: Set SORTREVERSE=toggle</i></p>
	<b>togglesmart</b>	<p>Toggles reverse sort on or off in the current folder. If used with the <b>SORTBY</b> argument on the same command line, this argument operates slightly differently to <b>toggle</b>. If the column specified for <b>SORTBY</b> is already sorted, the sort order will be reversed, but if the specified column is not already sorted, the sort order will not be reversed.</p> <p>To illustrate this, imagine the list is currently sorted forwards by name, and the command <b>Set SORTBY=size SORTREVERSE=toggle</b> is run multiple times. The resulting sort orders after each iteration would be:</p> <ol style="list-style-type: none"> <li>1. Sorted by name, forwards</li> <li>2. Sorted by size, backwards</li> <li>3. Sorted by size, forwards</li> </ol> <p>Contrast this with the command <b>Set SORTBY=size SORTREVERSE=togglesmart</b>:</p> <ol style="list-style-type: none"> <li>1. Sorted by name, forwards</li> <li>2. Sorted by size, forwards</li> </ol>

			<p>3. Sorted by size, backwards</p> <p><i>Example: Set SORTBY=desc SORTREVERSE=togglesmart</i></p>
SORTWORDS	/K	on	<p>Turns the <b>Word sort</b> option on in the source file display. This modifies the <a href="#">Folder Options</a> for the current folder.</p> <p><i>Example: Set SORTWORDS=on</i></p>
		off	<p>Turns off the <b>Word sort</b> option in the source file display.</p> <p><i>Example: Set SORTWORDS=off</i></p>
		toggle	<p>Toggles the <b>Word sort</b> option in the current folder.</p> <p><i>Example: Set SORTWORDS=toggle</i></p>
SOUNDS	/K	on	<p>Turns on the <b>Enable Sound Events</b> option on the <a href="#">Miscellaneous / Sounds</a> page in Preferences.</p> <p><i>Example: Set SOUNDS=on</i></p>
		off	<p>Turns off the <b>Enable Sound Events</b> option.</p> <p><i>Example: Set SOUNDS=off</i></p>
		toggle	<p>Toggles the <b>Enable Sound Events</b> option on or off.</p> <p><i>Example: Set SOUNDS=toggle</i></p>
SOURCE	/K	left	<p>Sets the left (or top) file display in a dual-display Lister to be the source.</p> <p><i>Example: Set SOURCE=left</i></p>
		right	<p>Sets the right (or bottom) file display to be the source.</p> <p><i>Example: Set SOURCE=right</i></p>
		focus	<p>Sets the file display that currently has the input focus to be the source.</p> <p><i>Example: Set SOURCE=focus</i></p>
		toggle	<p>Toggles the state (source/destination) of the left and right file displays.</p> <p><i>Example: Set SOURCE=toggle</i></p>

STATE	/K	<b>source</b>	<p>Sets the currently active Lister to be the source. When a Lister becomes the source, the previous source (if any) becomes the destination, and the previous destination (if any) is turned off. In a dual-display Lister, this command has no effect, because the currently active file display is by definition already the source.</p> <p><i>Example: Set STATE=source</i></p>
		<b>dest</b>	<p>Sets the currently active Lister to be the destination. In a dual-display Lister, this is equivalent to <b>Set SOURCE=toggle</b> - the source will become the destination and vice versa.</p> <p><i>Example: Set STATE=dest</i></p>
		<b>lockoff</b>	<p>Locks the active Lister as off. When a Lister is locked off, clicking in it will not make it the source or destination - only another <b>Set STATE</b> command can unlock it. This command has no effect in a dual-display Lister.</p> <p><i>Example: Set STATE=lockoff</i></p>
STATUSBAR	/K	<b>on</b>	<p>Turns the <a href="#">status bar</a> on in the active Lister.</p> <p><i>Example: Set STATUSBAR=on</i></p>
		<b>off</b>	<p>Turns the status bar off in the active Lister.</p> <p><i>Example: Set STATUSBAR=off</i></p>
		<b>toggle</b>	<p>Toggles the status bar on or off in the active Lister.</p> <p><i>Example: Set STATUSBAR=toggle</i></p>
STATUSBARSTYLE	/K	<b>single</b>	<p>Sets the status bar style to one single status bar even in a dual-display Lister.</p> <p><i>Example: Set STATUSBARSTYLE=single</i></p>
		<b>dual</b>	<p>One single status bar, with a separate definition in dual-display mode.</p> <p><i>Example: Set STATUSBARSTYLE=dual</i></p>

		<b>independent</b>	<p>Separate status bars for left/right file displays, with separate definitions for left and right.</p> <p><i>Example: Set</i> <b>STATUSBARSTYLE=independent</b></p>
		<b>independentsame</b>	<p>Separate status bars for left/right file displays, with the same definition for both.</p> <p><i>Example: Set</i> <b>STATUSBARSTYLE=independentsame</b></p>
		<b>bottom</b>	<p>Puts the status bar at the bottom of the Lister rather than at the bottom of the file display.</p> <p><i>Example: Set</i> <b>STATUSBARSTYLE=dual,bottom</b></p>
		<b>nobottom</b>	<p>Puts the status bar at the bottom of the file display rather than the bottom of the Lister.</p> <p><i>Example: Set</i> <b>STATUSBARSTYLE=independentsame,nobottom</b></p>
		<b>glass</b>	<p>Enable glass when the status bar is at the bottom of the Lister.</p> <p><i>Example: Set</i> <b>STATUSBARSTYLE=bottom,glass</b></p>
		<b>noglass</b>	<p>Disable glass when the status bar is at the bottom of the Lister.</p> <p><i>Example: Set</i> <b>STATUSBARSTYLE=bottom,single,noglass</b></p>
TABPOSITION	/K	<b>above</b>	<p>Set folder tabs in the current Lister to display above the file displays. This overrides the default folder tab position as configured on the <a href="#">Folder Tabs / Options</a> page in Preferences.</p> <p><i>Example: Set</i> <b>TABPOSITION=above</b></p>
		<b>below</b>	<p>Folder tabs in the current Lister will display below the file displays.</p> <p><i>Example: Set</i> <b>TABPOSITION=below</b></p>

	<b>left</b>	<p>Folder tabs will display to the left of the file displays.</p> <p><i>Example:</i><b>Set TABPOSITION=left</b></p>
	<b>right</b>	<p>Folder tabs will display to the right of the file display.</p> <p><i>Example:</i><b>Set TABPOSITION=right</b></p>
	<b>together</b>	<p>In a dual file display Lister, the folder tabs for each file display will display together (e.g. when set to above or below, a horizontal dual-display Lister would have the two folder tab bars together between the two file displays). This overrides the default setting configured on the <a href="#">Folder Tabs / Options</a> page in Preferences.</p> <p>You can use this keyword by itself or in conjunction with one of the above positional arguments, to change both settings simultaneously.</p> <p><i>Example:</i><b>Set TABPOSITION=above,together</b></p>
	<b>apart</b>	<p>In a dual display Lister, folder tabs will be apart from each other (e.g. when set to above or below, a horizontal dual-display Lister would have tabs above the top file display and below the bottom file display).</p> <p><i>Example:</i><b>Set TABPOSITION=apart</b></p>
	<b>normal</b>	<p>In a dual display Lister the folder tab position will be as configured (e.g. when set to above, both tab bars would be above their respective displays).</p> <p><i>Example:</i><b>Set TABPOSITION=above,normal</b></p>
	<b>reset</b>	<p>Resets the folder tab position in this Lister to the defaults as configured in</p>

			<p>Preferences.</p> <p><i>Example: Set TABPOSITION=reset</i></p>
THUMBNAILLABELS	/K	<b>on</b>	<p>Turns the display of thumbnail labels on. This is a global setting - it modifies the state of the <b>Display labels</b> option on the <a href="#">File Display Modes / Thumbnails</a> page in Preferences.</p> <p><i>Example: Set THUMBNAILLABELS=on</i></p>
		<b>off</b>	<p>Turns thumbnail labels off.</p> <p><i>Example: Set THUMBNAILLABELS=off</i></p>
		<b>toggle</b>	<p>Toggles thumbnail labels on or off.</p> <p><i>Example: Set THUMBNAILLABELS=toggle</i></p>
THUMBNAILRATING S	/K	<b>on</b>	<p>Turns the thumbnail overlay of rating stars on or off. This is a global setting - it modifies the state of the <b>Overlay rating</b> option on the <a href="#">File Display Modes / Thumbnails</a> page in Preferences.</p> <p><i>Example: Set THUMBNAILRATINGS=on</i></p>
		<b>off</b>	<p>Turns the ratings overlay off.</p> <p><i>Example: Set THUMBNAILRATINGS=off</i></p>
		<b>toggle</b>	<p>Toggles the ratings overlay on or off.</p> <p><i>Example: Set THUMBNAILRATINGS=toggle</i></p>
TREE	/K	<b>on</b>	<p>Turns the folder tree on in the active Lister. In a dual-display Lister, the <b>Open second Folder Tree in dual display mode</b> option on the <a href="#">Folder Tree / Options</a> page in Preferences controls whether a second tree opens automatically - if that option is off, you can use the <b>dual</b> keyword to force a second tree to open as well.</p> <p><i>Example: Set TREE=on</i></p>

		<b>off</b>	Turns the folder tree off in the active Lister.  <i>Example: Set TREE=off</i>
		<b>toggle</b>	Toggles the folder tree on or off in the active Lister.  <i>Example: Set TREE=toggle</i>
		<b>left</b>	Controls the left (or top) folder tree in a dual-display Lister.  <i>Example: Set TREE=left,toggle</i>
		<b>right</b>	Controls the right (or bottom) folder tree in a dual-display Lister.  <i>Example: Set TREE=right,toggle</i>
		<b>dual</b>	In a dual-display Lister, controls both trees at once.  <i>Example: Set TREE=dual,toggle</i>
		<b>source</b>	Controls the folder tree that "belongs" to the source file display.  <i>Example: Set TREE=source,on</i>
		<b>dest</b>	Controls the folder tree that belongs to the destination file display.  <i>Example: Set TREE=dest,toggle</i>
TREELOCK	/K	<b>on</b>	Turns the folder tree lock on for the active Lister (or when there are two trees, for the source file display). This is equivalent to clicking the padlock icon in the folder tree's header, but can be used even if the tree header is turned off in Preferences. When the folder tree is locked it no longer changes selection automatically to follow the current source path.  <i>Example: Set TREELOCK=on</i>
		<b>off</b>	Turns the folder tree lock off for the active Lister.  <i>Example: Set TREELOCK=off</i>
		<b>toggle</b>	Toggles the folder tree lock on and off.  <i>Example: Set TREELOCK=toggle</i>

TREESHOWPATHTOSEL	/K	<b>on</b>	Turns the folder tree's <b>Highlight path to selected folder</b> option on. This is a global setting and so affects all Listers. When turned on, the additional options on the <a href="#">Folder Tree / Appearance</a> Preferences page apply.  <i>Example: Set</i> <b>TREESHOWPATHTOSEL=on</b>
		<b>off</b>	Turns tree path highlighting off.  <i>Example: Set</i> <b>TREESHOWPATHTOSEL=off</b>
		<b>toggle</b>	Toggles tree path highlighting on or off.  <i>Example: Set</i> <b>TREESHOWPATHTOSEL=toggle</b>
TREEROOT	/K	<location>	Rebuilds the folder tree for the current source folder so that it is rooted at the specified location. The folder you specify will appear at the top of the tree, and the file display will change to show the root folder (unless it's already displaying a folder underneath that root).  <i>Example: Set</i> <b>TREEROOT "C:\Program Files"</b>
		<b>reset</b>	Resets the root of the folder tree to the root specified by the <b>Start Folder Tree at</b> option the <a href="#">Folder Tree / Options</a> page in Preferences.  <i>Example: Set</i> <b>TREEROOT=reset</b>
UTILITY	/K	<b>find</b>	Displays the <a href="#">utility panel</a> in <a href="#">Find Files</a> mode.  <i>Example: Set</i> <b>UTILITY=find</b>
		<b>sync</b>	Displays the utility panel in <a href="#">Synchronize</a> mode.  <i>Example: Set</i> <b>UTILITY=sync</b>
		<b>dupe</b>	Displays the utility panel in <a href="#">Duplicate File Finder</a> mode.  <i>Example: Set</i> <b>UTILITY=dupe</b>
		<b>undo</b>	Displays the utility panel showing the undo list (file operations that can be

		undone).
	<b>filelog</b>	Displays the utility panel showing the <a href="#">file operations log</a> .  <i>Example: Set UTILITY=undo</i>
	<b>ftplog</b>	Displays the utility panel showing the <a href="#">FTP logs</a> .  <i>Example: Set UTILITY=filelog</i>
	<b>otherlog</b>	Displays the utility panel showing the "other logs" page.  <i>Example: Set UTILITY=ftplog</i>
	<b>email</b>	Displays the utility panel showing the outgoing email log.  <i>Example: Set UTILITY=otherlog</i>
	<b>on</b>	Turns the utility panel on in the active Lister.  <i>Example: Set UTILITY=email</i>
	<b>off</b>	Turns the utility panel off.  <i>Example: Set UTILITY=on</i>
	<b>toggle</b>	Toggles the utility panel on or off in the active Lister.  <i>Example: Set UTILITY=off</i>
	<b>focus</b>	Gives focus to the utility panel if it's open. If used with <b>toggle</b> , the utility panel will only be toggled closed if it has focus.  <i>Example: Set UTILITY=ftplog,toggle</i>
	<b>expand</b>	If the utility panel is in a shrunken state, this argument in conjunction with <b>toggle</b> will cause the panel to expand rather than close.  <i>Example: Set UTILITY=find,toggle,focus</i>
	<b>noexpand</b>	When used with <b>toggle</b> (or other keywords that turn the panel on),

			<p><b>noexpand</b> prevents the utility panel from being expanded if it was previously saved in a shrunken state. That is, it will turn on but remain shrunken.</p> <p><i>Example: Set</i>  <b>UTILITY=find,toggle,noexpand</b></p>
VIEW	/K	<mode>[,<mode>]	<p>Changes the <a href="#">view mode</a> in the current file display. The valid mode keywords are <b>largeicons</b>, <b>smallicons</b>, <b>list</b>, <b>details</b>, <b>power</b>, <b>thumbnails</b> and <b>tiles</b>.</p> <p>You can specify two different view modes to create a command that toggles from one mode to the other. With this usage, you can append an asterisk (*) to the view mode keyword to specify that the button should appear highlighted when in that mode.</p> <p>The <b>cycle</b> keyword can be used to cycle through more than two modes.</p> <p><i>Example: Set</i>  <b>VIEW=details,thumbnails*</b></p>
		<b>cycle</b>	<p>Cycles through the view modes. If used by itself, this will cycle through all the available view modes - otherwise, combine with the appropriate view mode keywords to create a command that cycles through specific modes.</p> <p><i>Example: Set</i>  <b>VIEW=largeicons,smallicons,details,cycle</b></p>
VIEWERTOOLBAR	/O	<name>	<p>This command lets you change which toolbar is used for the <a href="#">Viewer Toolbar</a>. If you don't specify a name the default Viewer Toolbar is selected.</p> <p><i>Example: Set</i> <b>VIEWERTOOLBAR "My Viewer Toolbar"</b></p>
VIEWPANE	/K	<b>on</b>	<p>Turns the <a href="#">viewer pane</a> on in the currently active Lister.</p>

			<i>Example: Set VIEWPANE=on</i>
		<b>off</b>	Turns the viewer pane off in the active Lister.  <i>Example: Set VIEWPANE=off</i>
		<b>toggle</b>	Toggles the viewer pane on or off in the active Lister.  <i>Example: Set VIEWPANE=toggle</i>
		<b>horiz</b>	Forces the viewer pane to horizontal layout when it is opened.  <i>Example: Set VIEWPANE=toggle,horiz</i>
		<b>vert</b>	Specifies vertical layout for the viewer pane.  <i>Example: Set VIEWPANE=on,vert</i>
		<b>togglelayout</b>	Toggles the layout of the viewer pane between vertical and horizontal.  <i>Example: Set VIEWPANE=togglelayout</i>
VIEWPANELOCK	/K	<b>on</b>	Turns on the <a href="#">viewer pane</a> lock in the current Lister. When the viewer pane lock is turned on, it will continue to display its current image even if the file selection is changed in the Lister.  <i>Example: Set VIEWPANELOCK=on</i>
		<b>off</b>	Turns off the viewer pane lock.  <i>Example: Set VIEWPANELOCK=off</i>
		<b>toggle</b>	Toggles the viewer pane lock on or off.  <i>Example: Set VIEWPANELOCK=toggle</i>
VIEWPANESHELLICONS	/K	<b>on</b>	Enables shell icons in the <a href="#">viewer pane</a> . See <a href="#">Preferences / Viewer / Viewer Pane / Display shell icons</a> .

			<p><i>Example: Set</i> <b>VIEWPANESHELLICONS=on</b></p>
		<b>off</b>	<p>Disables shell icons in the viewer pane.</p> <p><i>Example: Set</i> <b>VIEWPANESHELLICONS=off</b></p>
		<b>toggle</b>	<p>Toggles shell icons in the viewer pane on or off.</p> <p><i>Example: Set</i> <b>VIEWPANESHELLICONS=toggle</b></p>
VIEWPANESHELLTHUMBS	/K	<b>on</b>	<p>Enables shell thumbnails in the <a href="#">viewer pane</a>. See <a href="#">Preferences / Viewer / Viewer Pane / Display shell thumbnails</a>.</p> <p><i>Example: Set</i> <b>VIEWPANESHELLTHUMBS=on</b></p>
		<b>off</b>	<p>Disables shell thumbnails in the viewer pane.</p> <p><i>Example: Set</i> <b>VIEWPANESHELLTHUMBS=off</b></p>
		<b>toggle</b>	<p>Toggles shell thumbnails in the viewer pane on or off.</p> <p><i>Example: Set</i> <b>VIEWPANESHELLTHUMBS=toggle</b></p>
VIEWPANESIZE	/K	<size>[,<size>]	<p>Adjusts the size of the viewer pane in the active Lister. The size is given as a percentage of the total size of the Lister, and applies in the appropriate dimension based on the current layout of the viewer pane (so for example, when the pane is horizontal this affects its height).</p> <p>It is also possible to specify two separate sizes, and the command will toggle between them.</p> <p><i>Example: Set</i> <b>VIEWPANESIZE 25,50</b></p>

## SetAttr

The **SetAttr** internal command can be used to:

- Modify the attributes of files or folders
- Compress (by setting the **C** attribute) or encrypt files (by setting the **E** attribute), and set the default compression/encryption status for folders
- Modify the last modified and creation timestamps of files or folders
- Modify file metadata, either through a [user-interface](#) or [programmatically](#)
- Assign your own description to files and folders, and edit the internal [comment](#) for a Zip archive
- Modify attributes of files on remote FTP sites

### Command Arguments:

Argument	Type	Possible values	Description														
<i>(no argument)</i>	-	-	Displays the <a href="#">Change Attributes &amp; Times</a> dialog, which lets you modify the attributes and timestamps of selected files and folders.  <i>Example: SetAttr</i>														
ATTR	/K	<attributes>	Sets the specified attributes for selected files and folders, and clears all others. The <attributes> value must consist of one or more of the following letters: <table border="1"><tbody><tr><td><b>R</b></td><td>Read-only (files can not be deleted or overwritten)</td></tr><tr><td><b>A</b></td><td>Archive (file needs to be archived, used by backup tools)</td></tr><tr><td><b>H</b></td><td>Hidden (files can be marked as hidden to hide them from the display)</td></tr><tr><td><b>S</b></td><td>System (file is a system file - usually set in conjunction with <b>H</b>)</td></tr><tr><td><b>N</b></td><td>Normal (normal attributes, none of the other attributes set)</td></tr><tr><td><b>I</b></td><td>Non-content Indexed (contents will not be indexed by the system)</td></tr><tr><td><b>C</b></td><td>Compressed (on NTFS-formatted drives only)</td></tr></tbody></table>	<b>R</b>	Read-only (files can not be deleted or overwritten)	<b>A</b>	Archive (file needs to be archived, used by backup tools)	<b>H</b>	Hidden (files can be marked as hidden to hide them from the display)	<b>S</b>	System (file is a system file - usually set in conjunction with <b>H</b> )	<b>N</b>	Normal (normal attributes, none of the other attributes set)	<b>I</b>	Non-content Indexed (contents will not be indexed by the system)	<b>C</b>	Compressed (on NTFS-formatted drives only)
<b>R</b>	Read-only (files can not be deleted or overwritten)																
<b>A</b>	Archive (file needs to be archived, used by backup tools)																
<b>H</b>	Hidden (files can be marked as hidden to hide them from the display)																
<b>S</b>	System (file is a system file - usually set in conjunction with <b>H</b> )																
<b>N</b>	Normal (normal attributes, none of the other attributes set)																
<b>I</b>	Non-content Indexed (contents will not be indexed by the system)																
<b>C</b>	Compressed (on NTFS-formatted drives only)																

			<p><b>E</b> Encrypted (on NTFS-formatted drives only)</p> <p>Note that modifying the <b>C</b> or <b>E</b> attributes may take longer than normal, as the file data has to be (un)compressed or (un)encrypted. Setting these attributes for folders sets the default compression/encryption state for new files created in those folders. A file can be compressed, or encrypted, but not both at once.</p> <p><i>Example: SetAttr ATTR rca</i></p>
CHMOD	/K	<attribute mask>	<p>Sets the specified attributes for files on a remote FTP site. The &lt;attribute mask&gt; is specified using <a href="#">octal notation</a>.</p> <p><i>Example: SetAttr CHMOD 666</i></p>
CLEARATTR	/K	<attributes>	<p>Clears the specified attributes for selected files and folders. The specified attributes will be turned off, but other attributes will be untouched. See the description of the <b>ATTR</b> argument for a list of attribute values.</p> <p><i>Example: SetAttr CLEARATTR c</i></p>
CREATED	/K	<date/time>	<p>Sets the creation timestamp for selected files and folders. The value for this argument can be given as either a date only, a time only, or a date and time.</p> <p>The accepted formats for the date string are <b>YYYYMMDD</b> or <b>YYYY-MM-DD</b>, and the time string must be in the format <b>HH:MM:SS</b>.</p> <p>If you specify a time as well as a date the time must come after the date, separated by a space, and you must enclose the entire value in quotes (because of the embedded space). If only a time is provided, the file's current date will be preserved (and vice versa).</p> <p><i>Example: SetAttr CREATED "1973-09-22 3:35"</i></p>
		<b>now</b>	<p>Sets the creation timestamp to the current date and time.</p>

			<i>Example: SetAttr CREATED=now</i>
		<b>modified</b>	Copies the last modified timestamp of the file.  <i>Example: SetAttr CREATED=modified</i>
		<b>taken</b>	For image files, copies the <i>Date Taken</i> timestamp. If the file doesn't have an EXIF tag the creation timestamp won't be changed.  <i>Example: SetAttr CREATED=taken</i>
		<b>digitized</b>	For image files, copies the <i>Date Digitized</i> timestamp. If the file doesn't have an EXIF tag the creation timestamp won't be changed.  <i>Example: SetAttr CREATED=digitized</i>
		<b>parent</b>	Copies the creation timestamp from the parent folder.  <i>Example: SetAttr CREATED=parent</i>
		<b>doccreated</b>	For document files, copies the <i>Document Created</i> timestamp. If the file doesn't have this timestamp in its metadata the creation timestamp won't be changed.  <i>Example: SetAttr CREATED=doccreated</i>
		<b>docedited</b>	For document files, copies the <i>Document Last Edited</i> timestamp. If the file doesn't have this timestamp in its metadata the creation timestamp won't be changed.  <i>Example: SetAttr CREATED=docedited</i>
		<b>docsaved</b>	For document files, copies the <i>Document Last Saved</i> timestamp. If the file doesn't have this timestamp in its metadata the creation timestamp won't be changed.  <i>Example: SetAttr CREATED=docsaved</i>
DESCRIPTION	/O	(no value)	Displays the <a href="#">Set Description</a> dialog, which lets you assign your own description string to selected files and folders.

			<p><i>Example:</i><b>SetAttr DESCRIPTION</b></p>
		<description>	<p>Sets the description for selected files and folders to the specified string.</p> <p><i>Example:</i><b>SetAttr DESCRIPTION "Project Files for Keith"</b></p> <p>Note that an empty string will be treated the same as not passing a string at all, causing the <a href="#">Set Description</a> dialog to open. If you want an empty string to clear the description, use the <b>SETDESCRIPTION</b> argument instead.</p>
FILE	/M	<filename>, ...	<p>Specifies the name of the file or files to modify. If you don't provide this argument the command operates on all selected items in the source Lister. This is the default argument for the <b>SetAttr</b> command - you don't need to specify the <b>FILE</b> keyword.</p> <p>If you only specify the filename instead of the full path of the file or files, Opus will look in the current source folder. You can also specify a <a href="#">wildcard pattern</a>. Remember that if the filename contains spaces you need to enclose it in quotes.</p> <p><i>Example:</i><b>SetAttr *.xls DESCRIPTION "Annual Results for 2011" SETATTR r</b></p>
FILTER	/K	<filter>	<p>Applies the specified filter to the contents of selected folders. This must have previously been created from the <a href="#">File Operations / Filters</a> page in Preferences. You can also directly specify a <a href="#">simple wildcard pattern</a></p> <p><i>Example:</i> <b>SetAttr FILTER "temp files" ATTR n</b></p>
META	/O/M	(no value)	<p>Displays the <a href="#">Set Metadata</a> dialog, which lets you modify the metadata for selected files and folders.</p> <p><i>Example:</i><b>SetAttr META</b></p>
		keyword:<value>, ...	<p>Sets the specified metadata fields to the supplied values. Changes are made (where applicable) to all selected files.</p> <p>Each <i>keyword:&lt;value&gt;</i> pair must be enclosed in quotes if the value contains any</p>

			<p>spaces. This argument can accept multiple <i>keyword:&lt;value&gt;</i> pairs to make changes to more than one metadata field at once.</p> <p>See the section on <a href="#">programmatic setting of metadata</a> for more information.</p> <p><b>Example: SetAttr META "artist:Pink Floyd" "album:Dark Side of the Moon"</b></p>
MODIFIED	/K	<date/time>	<p>Sets the last modified timestamp for selected files and folders. The value for this argument can be given as either a date only, a time only, or a date and time.</p> <p>The accepted formats for the date string are <b>YYYYMMDD</b> or <b>YYYY-MM-DD</b>, and the time string must be in the format <b>HH:MM:SS</b>.</p> <p>If you specify a time as well as a date the time must come after the date, separated by a space, and you must enclose the entire value in quotes (because of the embedded space). If only a time is provided, the file's current date will be preserved (and vice versa).</p> <p><b>Example: SetAttr MODIFIED 20080110</b></p>
		<b>now</b>	<p>Sets the last modified timestamp to the current date and time.</p> <p><b>Example: SetAttr MODIFIED=now</b></p>
		<b>created</b>	<p>Copies the creation timestamp of the file.</p> <p><b>Example: SetAttr MODIFIED=created</b></p>
		<b>taken</b>	<p>For image files, copies the <i>Date Taken</i> timestamp. If the file doesn't have an EXIF tag the last modified timestamp won't be changed.</p> <p><b>Example: SetAttr MODIFIED=taken</b></p>
		<b>digitized</b>	<p>For image files, copies the <i>Date Digitized</i> timestamp. If the file doesn't have an EXIF tag the last modified timestamp won't be changed.</p> <p><b>Example: SetAttr MODIFIED=digitized</b></p>

		<b>parent</b>	<p>Copies the last modified timestamp from the parent folder.</p> <p><i>Example: SetAttr MODIFIED=parent</i></p>
		<b>doccreated</b>	<p>For document files, copies the <i>Document Created</i> timestamp. If the file doesn't have this timestamp in its metadata the last modified timestamp won't be changed.</p> <p><i>Example: SetAttr MODIFIED=doccreated</i></p>
		<b>docedited</b>	<p>For document files, copies the <i>Document Last Edited</i> timestamp. If the file doesn't have this timestamp in its metadata the last modified timestamp won't be changed.</p> <p><i>Example: SetAttr MODIFIED=docedited</i></p>
		<b>docsaved</b>	<p>For document files, copies the <i>Document Last Saved</i> timestamp. If the file doesn't have this timestamp in its metadata the last modified timestamp won't be changed.</p> <p><i>Example: SetAttr MODIFIED=docsaved</i></p>
RECURSE	/S	<i>(no value)</i>	<p>Changes made by this command will be recursively applied to files within selected folders. This does not affect the <b>META</b> argument - only attributes, timestamps and descriptions can be applied recursively.</p> <p><i>Example: SetAttr CLEARATTR hs RECURSE</i></p>
SETATTR	/K	<i>&lt;attributes&gt;</i>	<p>Sets the specified attributes for selected files and folders. The specified attributes will be turned on, but other attributes will be untouched. See the description of the <b>ATTR</b> argument for a list of attribute values.</p> <p><i>Example: SetAttr SETATTR r</i></p>
SETDESCRIPTION	/O	<i>(no value)</i>	<p>Clears the description for selected files and folders.</p> <p><i>Example: SetAttr SETDESCRIPTION</i></p> <p>The difference between the <b>DESCRIPTION</b> and <b>SETDESCRIPTION</b> arguments is what</p>

			they do when not given a value. <b>SETDESCRIPTION</b> clears the description while <b>DESCRIPTION</b> opens a dialog for you to type in a description.
		<description>	Sets the description for selected files and folders to the specified string.  <i>Example: SetAttr SETDESCRIPTION "Approved for release"</i>
TOGGLEATTR	/K	<attributes>	Toggles (inverts) the state of the specified attributes. If each attribute specified is currently set it will be cleared, and vice versa. Attributes that aren't specified will be unaffected. See the description of the <b>ATTR</b> argument for a list of attribute values.  <i>Example: SetAttr TOGGLEATTR h</i>
ZIPCOMMENT	/S	(no value)	Lets you edit the internal <a href="#">Zip comment</a> when viewing the contents of a Zip archive. The comment is stored inside the Zip file and can be displayed by other Zip tools.  <i>Example: SetAttr ZIPCOMMENT</i>

## Show

The **Show** internal command can be used to:

- Display selected image files in the [standalone image viewer](#)
- Display a slideshow of the contents of the current folder
- View and manipulate installed viewer plugins (third-party libraries that can extend the image handling abilities of Opus)
- Dynamically adjust the size of thumbnails displayed in the current file display
- Execute the internal commands of the standalone image viewer (allowing you to configure the viewer toolbar and hotkeys)

## Command Arguments:

Argument	Type	Possible values	Description
<i>(no argument)</i>	-	-	<p>Opens the <a href="#">standalone image viewer</a> to display selected files.</p> <p><i>Example: Show</i></p>
AUTOFILELIST	/S	<i>(no value)</i>	<p>Used to automatically populate the "next / previous" list in the viewer with other files visible in the folder tab which launched the command.</p> <p>Use <b>AUTOFILELIST</b> on its own, without <b>LISTSIBLINGS</b>, to make a command with similar behavior to launching the viewer by double-clicking a file. Specifically, it makes the command respect the <b>Generate next/previous list (when opened via double-click)</b> option on the <a href="#">Viewer Behavior</a> page in Preferences, even when not triggered by a double-click. If the option is off then <b>AUTOFILELIST</b> on its own may have no effect.</p> <p>If you combine <b>AUTOFILELIST</b> and <b>LISTSIBLINGS</b> together, the command will always populate the "next / previous" list, even if the Preferences option is off, unless more than one file is selected. (If two or more files are selected, and you run the command on them, then just those files will be in the "next / previous" list.)</p> <p>The "next / previous" list generated by <b>AUTOFILELIST</b> (when the Preferences option is on, or when combined with <b>LISTSIBLINGS</b>) will usually be the same as that generated by <b>LISTSIBLINGS</b> on its own. The two differ in situations where what's visible in the folder tab does not correspond to a real folder on disk. If the folder tab has a filter applied, or is showing something like Find Results or a mode like Flat View, then <b>AUTOFILELIST</b> gets you a list which corresponds to that filtered or multi-directory view; on the other hand, <b>LISTSIBLINGS</b> (on its own) gets you a list of files from the same directory as the file you start</p>

			with, which may be files that are not displayed in the folder tab at all.  <i>Example: Show <b>AUTOFILELIST LISTSIBLINGS</b></i>
FILE		<file>	Specifies the file or folder to show. If a folder is specified, all files directly below it will be queued to the viewer's "next / previous" list. If you don't provide a file or folder on the command line, all selected files from the folder tab the command was launched from will be used, if applicable. Remember to enclose paths in quotes if they contain spaces.  <i>Example: Show "C:\Pictures\Mum&amp;Dad.jpg"</i>
FULLSCREEN	/S	(no value)	Opens the image viewer in full screen mode.  <i>Example: Show <b>FULLSCREEN</b></i>
LISTSIBLINGS	/S	(no value)	When <b>LISTSIBLINGS</b> is used on its own, without <b>AUTOFILELIST</b> , it automatically populates the "next / previous" list in the viewer with other files from the same directory as the specified file (or single selected file).  If neither <b>LISTSIBLINGS</b> nor <b>AUTOFILELIST</b> are specified, or if the command is run against multiple selected files, only the specified or selected files will be included in the viewer's "next / previous" list.  See the description of <b>AUTOFILELIST</b> for what happens when both are specified, and the differences between the two.  <i>Example: Show <b>FULLSCREEN LISTSIBLINGS</b></i>
NOUSEEXISTING	/S	(no value)	Prevents the re-use of an existing viewer window - a new window will always be opened. This overrides the <b>Reuse existing viewer window</b> option on the <a href="#">Viewer Behavior</a> page in Preferences.  <i>Example: Show <b>NOUSEEXISTING</b></i>
PLUGIN	/K	<plugin name>	Forces the use of the specified plugin to display the files. Without this, Opus will automatically determine the best plugin to use (or, for a format that can be handled internally, no plugin will be used).

			<p>The <i>&lt;plugin name&gt;</i> can be either the name of the plugin DLL (including the <b>.dll</b> extension) or the "pretty name" of the plugin. Note that this command can't force a plugin to view a file it can't handle.</p> <p>This argument is also used in conjunction with <b>PLUGINDISABLE</b> to enable or disable the specified plugin.</p> <p><i>Example: Show PLUGIN "Animated GIF"</i></p>
PLUGINABOUT	/K	<i>&lt;plugin name&gt;</i>	<p>Displays the <b>About</b> dialog for the specified plugin. The <i>&lt;plugin name&gt;</i> can be either the name of the plugin DLL (including the <b>.dll</b> extension) or the "pretty name" of the plugin.</p> <p><i>Example: Show PLUGINABOUT text.dll</i></p>
PLUGINCONFIG	/K	<i>&lt;plugin name&gt;</i>	<p>Displays the configuration dialog for the specified plugin (if it has one).</p> <p><i>Example: Show PLUGINCONFIG dcwrap.dll</i></p>
PLUGINDISABLE	/O	<i>(no value)</i>	<p>Toggles the enable state of the specified plugin. If the plugin is currently enabled it will be disabled, and vice versa. The plugin must be specified using the <b>PLUGIN</b> argument.</p> <p><i>Example: Show PLUGINDISABLE PLUGIN gifanim.dll</i></p>
		<b>enable</b>	<p>Enables the plugin specified with the <b>PLUGIN</b> argument.</p> <p><i>Example: Show PLUGINDISABLE=enable PLUGIN text.dll</i></p>
		<b>disable</b>	<p>Disables the plugin specified with the <b>PLUGIN</b> argument.</p> <p><i>Example: Show PLUGIN audiotags.dll PLUGINDISABLE disable</i></p>
PLUGINLIST	/S	<i>(no value)</i>	<p>Displays a dynamic list of your installed viewer plugins (acts as a <a href="#">dynamic button</a>). The generated list contains a sub-menu for each viewer plugin, with commands to enable/disable, configure and show information about the plugin.</p> <p><i>Example: Show PLUGINLIST</i></p>
PLUGINMANAGER	/S	<i>(no value)</i>	<p>Displays the <a href="#">Viewer / Viewer Plugins</a> page in Preferences, which lets you see and manage your</p>

			<p>installed viewer plugins.</p> <p><i>Example: Show <b>PLUGINMANAGER</b></i></p>
POS	/K	<x>,<y>	<p>Overrides the default positioning of the standalone viewer, and opens its window at the specified coordinates. This could be used, for example, to always display the viewer on a particular monitor.</p> <p><i>Example: Show <b>POS 1920,0</b></i></p>
SIZE	/K	<width>,<height>	<p>Overrides the default size of the standalone viewer.</p> <p><i>Example: Show <b>POS 1920,0 SIZE 960,1080</b></i></p>
SLIDESHOW	/S	(no value)	<p>Initiates a slideshow of images. If any files are currently selected in the source file display, only those images will be shown in the slideshow - otherwise, all image files in the current folder will be displayed. Use with the <b>LISTSIBLINGS</b> argument to always include all files in the folder, irrespective of how many are currently selected.</p> <p>You can adjust the speed of the slideshow, and choose whether the order should be linear or random, from the <a href="#">Viewer Behavior</a> page in Preferences.</p> <p><i>Example: Show <b>SLIDESHOW LISTSIBLINGS FULLSCREEN</b></i></p>
THUMBNAILSIZE	/K	<size>	<p>Sets the size of thumbnails in the active Lister to the specified size in pixels. This overrides the global thumbnail size set in the <a href="#">File Display Modes / Thumbnails</a> page in Preferences.</p> <p><i>Example: Show <b>THUMBNAILSIZE 192</b></i></p>
		<width>,<height>	<p>Sets the width and height of thumbnails separately. This lets you have non-square thumbnail sizes if desired.</p> <p><i>Example: Show <b>THUMBNAILSIZE 92,64</b></i></p>
		<change>	<p>Adjusts the size of thumbnails in the active Lister by the specified delta value. You must specify either a leading + or - to make this a relative size change.</p> <p><i>Example: Show <b>THUMBNAILSIZE +32</b></i></p>

		<i>&lt;change w&gt;,&lt;change h&gt;</i>	Adjusts both the width and height of thumbnails by the specified deltas.  <i>Example: Show THUMBNAILSIZE +16,+24</i>
		<b>left</b>	Applies the size change to only the left (or top) file display in a <a href="#">dual-display</a> Lister.  <i>Example: Show THUMBNAILSIZE left,128</i>
		<b>right</b>	Applies the size change to only the right (or bottom) file display.  <i>Example: Show THUMBNAILSIZE +64,right</i>
		<b>source</b>	Applies the size change to only the source file display.  <i>Example: Show THUMBNAILSIZE source,64,80</i>
		<b>dest</b>	Applies the size change to only the destination file display.  <i>Example: Show THUMBNAILSIZE dest,256</i>
		<b>both</b>	Applies the size change to both visible file displays in a dual-display Lister, but only the active folder tabs.  <i>Example: Show THUMBNAILSIZE both,+32</i>
		<b>all</b>	Applies the size change to all file displays in the Lister, including both sides of a dual-display Lister and all folder tabs. This is the default behaviour.  <i>Example: Show THUMBNAILSIZE all,+32</i>
		<b>reset</b>	Resets the thumbnail size to the value set in Preferences.  <i>Example: Show THUMBNAILSIZE all,reset</i>
		<b>list</b>	Generates <a href="#">dynamic buttons</a> that provide a number of thumbnail size options, intelligently chosen to suit your system DPI settings.  <i>Example: Show THUMBNAILSIZE=list</i>
USEEXISTING	/S	<i>(no value)</i>	Forces the re-use of an existing viewer window - a new window will never be opened if there is an existing viewer currently open. This overrides the <b>Reuse existing viewer window</b> option on the <a href="#">Viewer Behavior</a> page in Preferences.

			<i>Example:</i> <b>Show USEEXISTING</b>
VIEWERCMD	/K	<command>	Only for use in the <a href="#">standalone viewer</a> , this command forms the basis of the default viewer toolbar, context menu and hotkeys. These commands can also be used from scripts that run within the context of the viewer.
		<b>alpha</b>	Toggle the <i>Hide Alpha Channel</i> option on and off.  <i>Example:</i> <b>Show VIEWERCMD=alpha</b>
		<b>close</b>	Closes the standalone viewer window.  <i>Example:</i> <b>Show VIEWERCMD=close</b>
		<b>cmdbar</b>	Displays a <a href="#">FAYT</a> -style command bar at the bottom of the viewer window, which lets you type in an ad-hoc command to run in the context of the viewer.  <i>Example:</i> <b>Show VIEWERCMD=cmdbar</b>
		<b>copy</b>	Copies the currently selected region of the image to the clipboard.  <i>Example:</i> <b>Show VIEWERCMD=copy</b>
		<b>copyto</b>	Prompts for a new filename to copy the currently viewed file to.  <i>Example:</i> <b>Show VIEWERCMD=copyto</b>
		<b>crop</b>	Crops the image to the currently selected region. The file on disk is not modified unless the image is saved.  <i>Example:</i> <b>Show VIEWERCMD=crop</b>
		<b>cut</b>	Cuts the currently viewed file to the clipboard. If you then perform a paste in a Lister the viewed file will be moved.  <i>Example:</i> <b>Show VIEWERCMD=cut</b>

	<b>delete</b>	<p>Deletes the currently viewed file (after prompting for confirmation).</p> <p><i>Example: Show VIEWERCMD=delete</i></p>
	<b>dragsel</b>	<p>Lets you trigger certain mouse actions from an <b>OnViewerEvent</b> script event. For example, when the script event is triggered to tell you the user clicked the left mouse button, you could trigger scrolling or expand mode depending on where on the window the mouse was clicked.</p> <p>The default behavior is to scroll; use the "select" keyword to trigger selection mode, and "expand" to trigger expand/scroll mode.</p> <p>By default the command assumes the left mouse button was used to trigger the event - use the "rclick" keyword for the right mouse button, and "mclick" keyword for the middle mouse button.</p> <p>You can also provide the coordinates of the click by passing "pos:x,y" on the command line - if not provided, the current mouse position is assumed.</p> <p><i>Example: Show VIEWERCMD=dragsel,expand,mclick,pos:100,100</i></p>
	<b>first</b>	<p>Goes back to view the first file in the list.</p> <p><i>Example: Show VIEWERCMD=first</i></p>
	<b>flip</b>	<p>Flips the currently viewed image. The file on disk is not modified unless the image is saved. Use with <b>horiz</b> to flip horizontally or <b>vert</b> to flip vertically.</p> <p><i>Example: Show VIEWERCMD=flip,horiz</i></p>

	<b>fullscreen</b>	Toggles full-screen mode on and off.  <i>Example: Show VIEWERCMD=fullscreen</i>
	<b>gamma</b>	Adjust the gamma value of the image display. The adjustment value can be one of the following:  +<value> - increase gamma by <value> -<value> - decrease gamma by <value> <value> - set absolute gamma value <b>0</b> -<value> - set absolute negative gamma value <b>reset</b> - reset gamma to default value  <i>Example: Show VIEWERCMD=gamma,+1</i> <i>Example: Show VIEWERCMD=gamma,reset</i>
	<b>goto</b>	Go to a specified file in the list. Must be used with a value indicating the image, where <b>0</b> is the first image, <b>1</b> is the second and so on. This command can also be used to jump forwards or backwards a specific number of files, by specifying a number preceded by + or -.  <i>Example: Show VIEWERCMD=goto,0</i> <i>Example: Show VIEWERCMD=goto,+10</i>
	<b>help</b>	Displays help about the image viewer.  <i>Example: Show VIEWERCMD=help</i>
	<b>hex</b>	Toggles the display in and out of hexadecimal mode.  <i>Example: Show VIEWERCMD=hex</i>
	<b>info</b>	Toggles the image information overlay on and off.  <i>Example: Show VIEWERCMD=info</i>
	<b>last</b>	Go to the last file in the list.  <i>Example: Show VIEWERCMD=last</i>

	<b>mark</b>	<p>This keyword is used to control image marking. It has many different functions, which are accessed by combining it with the following keywords:</p> <ul style="list-style-type: none"> <li>• <i>(no sub-keyword)</i>: Toggle mark state of current image (same as the <b>toggle</b> keyword).</li> <li>• <b>browse</b>: Browse marked pictures (opens the marked pictures collection in a new tab).</li> <li>• <b>clear</b>: Clear all marks.</li> <li>• <b>exchange</b>: Exchange the current image for the previously marked image (unmarks the previous one and marks this one in its place).</li> <li>• <b>first</b>: Jump to the first marked image.</li> <li>• <b>last</b>: Jump to the last marked image.</li> <li>• <b>next</b>: Jump to the next marked image.</li> <li>• <b>nohighlight</b>: Prevents a button being highlighted when the condition it describes is true (e.g. when an image is marked, stops a <b>mark,toggle</b> button from being highlighted). Use with <b>on, off, toggle</b> and <b>view</b>.</li> <li>• <b>off</b>: Unmarks the current image.</li> <li>• <b>on</b>: Marks the current image.</li> <li>• <b>prev</b>: Jump to the previously marked image.</li> <li>• <b>return</b>: Return from a jump to the image you were previously viewing (e.g. after <b>mark,first</b> to jump to the first marked image, <b>mark,return</b> would put you back where you were).</li> <li>• <b>toggle</b>: Toggle mark state of current image.</li> <li>• <b>view</b>: Toggle the marked pane on and off.</li> </ul> <p><i>Example: Show VIEWERCMD=mark,toggle,nohighlight</i>  <i>Example: Show VIEWERCMD=view</i></p>
	<b>meta</b>	<p>Toggles the embedded metadata panel on or off, and controls its width. Combine with the following keywords:</p> <ul style="list-style-type: none"> <li>• <i>&lt;width&gt;</i>: Specify the width of the metadata panel in pixels.</li> <li>• <b>grow</b>: When opening the metadata panel, the viewer window will grow as much as possible to accommodate it. When closing the panel, provided the window has not been manually resized, its original width will be restored.</li> <li>• <b>nofocus</b>: Prevents the metadata editor from gaining input focus when it opens.</li> <li>• <b>off</b>: Turns the metadata panel off.</li> </ul>

		<ul style="list-style-type: none"> <li>• <b>on</b>: Turns the metadata panel on.</li> <li>• <b>toggle</b>: Toggles the metadata panel on and off.</li> </ul> <p><i>Example: Show VIEWERCMD=meta,toggle,grow,400</i></p>
	<b>minwidth</b>	<p>Save the width of the current viewer window as the new minimum width. When new viewers open they won't automatically size themselves narrower than this width.</p> <p><i>Example: Show VIEWERCMD=minwidth</i></p>
	<b>moveto</b>	<p>Prompts for a new filename to move the currently viewed file to.</p> <p><i>Example: Show VIEWERCMD=moveto</i></p>
	<b>next</b>	<p>Go to and view the next file in the list.</p> <p><i>Example: Show VIEWERCMD=next</i></p>
	<b>nextlist</b>	<p>Generates a list of the subsequent files in the image list (acts as a <a href="#">dynamic button</a> - designed for use on the drop-down attached to the <i>Next</i> button).</p> <p><i>Example: Show VIEWERCMD=nextlist</i></p>
	<b>notfullscreen</b>	<p>Hides this button when the viewer is not in full-screen mode. You can combine this with any other keyword.</p> <p><i>Example: Show VIEWERCMD=delete,notfullscreen</i></p>
	<b>onlyfullscreen</b>	<p>Hides this button unless the viewer is in full-screen mode. You can combine this with any other keyword.</p> <p><i>Example: Show VIEWERCMD=close,onlyfullscreen</i></p>
	<b>open</b>	<p>Open and view a new file. By default this will prompt for the file to open, but you can provide a filename on the command line (e.g. from a script).</p>

			<p><i>Example: Show VIEWERCMD=open</i></p> <p><i>Example: Show VIEWERCMD "open,c:\my pictures\image.jpg"</i></p>
		<b>pluginabout</b>	<p>Displays the <i>About</i> dialog for the current viewer plugin.</p> <p><i>Example: Show VIEWERCMD=pluginabout</i></p>
		<b>plugincfg</b>	<p>Displays the configuration dialog for the current viewer plugin (if it provides one).</p> <p><i>Example: Show VIEWERCMD=plugincfg</i></p>
		<b>plugincmd</b>	<p>Trigger a command provided by the current viewer plugin (if it provides any). The command is specified as a number where <b>0</b> means the first command, <b>1</b> means the second and so on.</p> <p><i>Example: Show VIEWERCMD=plugincmd,0</i></p>
		<b>plugincmds</b>	<p>Generates a list of the commands provided by the current viewer plugin (if it provides any). This acts as a <a href="#">dynamic button</a>.</p> <p><i>Example: Show VIEWERCMD=plugincmds</i></p>
		<b>prev</b>	<p>Go to and view the previous file in the list.</p> <p><i>Example: Show VIEWERCMD=prev</i></p>
		<b>prevlist</b>	<p>Generates a list of the previous files in the image list (acts as a <a href="#">dynamic button</a> - designed for use on the drop-down attached to the <i>Previous</i> button).</p> <p><i>Example: Show VIEWERCMD=prevlist</i></p>
		<b>print</b>	<p>Print the currently viewed file.</p> <p><i>Example: Show VIEWERCMD=print</i></p>
		<b>refresh</b>	<p>Refresh the currently viewed file. The file will be reloaded from disk.</p> <p><i>Example: Show VIEWERCMD=refresh</i></p>
		<b>reselect</b>	<p>Reselect the previous selection. This lets you easily crop more than one image to the same area</p>

			(i.e. select a region, crop and save this image, move to the next image, reselect the previous region, crop and save, ...).  <i>Example: Show VIEWERCMD=reselect</i>
		<b>restore</b>	Undoes the previous crop operation.  <i>Example: Show VIEWERCMD=restore</i>
		<b>rotate</b>	Rotate the display of the current image. The file on disk is not modified unless you save the image. The amount to rotate by can be specified as follows:  <ul style="list-style-type: none"> <li>+&lt;value&gt; - rotate clockwise by &lt;value&gt; degrees</li> <li>-&lt;value&gt; - rotate anti-clockwise by &lt;value&gt; degrees</li> <li>&lt;value&gt; - set rotation to an absolute value</li> <li><b>reset</b> - reset the rotation</li> </ul> <i>Example: Show VIEWERCMD=rotate,+90</i>
		<b>save</b>	Save any changes you have made to the current image. Use with the <b>quiet</b> option to replace the existing file silently. This command is unavailable if the image is not in one of the formats that Opus is able to save (currently PNG, JPG, BMP and GIF) - in that case, you can use <b>saveas</b> to save the image in one of those formats.  <i>Example: Show VIEWERCMD=save,quiet</i>
		<b>saveas</b>	Prompts for a new filename to save the image to. The <i>Save Picture</i> dialog also lets you choose the image format to save in. You can optionally provide a filename to save to on the command line (e.g. from a script).  <i>Example: Show VIEWERCMD=saveas</i> <i>Example: Show VIEWERCMD "saveas,c:\my pictures\image.png"</i>
		<b>scroll</b>	Scrolls the current image. You must specify either <b>horiz</b> or <b>vert</b> to indicate the dimension you want to scroll, combined with another

		<p>keyword to indicate how far to scroll. The keywords are:</p> <ul style="list-style-type: none"> <li>• <b>bottom</b>: Scroll to the bottom (vertical) or far right (horizontal).</li> <li>• <b>down</b>: Scroll down (vertical) or right (horizontal).</li> <li>• <b>horiz</b>: Scroll horizontally.</li> <li>• <b>pagedown</b>: Scroll down a page (vertical) or right a page (horizontal).</li> <li>• <b>pageup</b>: Scroll up a page (vertical) or left a page (horizontal).</li> <li>• <b>top</b>: Scroll to the top (vertical) or far left (horizontal).</li> <li>• <b>up</b>: Scroll up (vertical) or left (horizontal).</li> <li>• <b>vert</b>: Scroll vertically.</li> <li>• <b>center</b>: Scroll to the center. Can optionally be combined with <b>horiz</b> or <b>vert</b>.</li> </ul> <p><i>Example: Show VIEWERCMD=scroll,vert,pagedown</i></p>
	<b>selaspect</b>	<p>Fixes the aspect ratio of the selection marquee. You can specify an aspect ratio (16:9, 3/2, etc), or <b>reset</b> to remove the restriction.</p> <p><i>Example: Show VIEWERCMD=selaspect,16:9</i>  <i>Example: Show VIEWERCMD=selaspect,reset</i></p>
	<b>selectall</b>	<p>Select the entire image.</p> <p><i>Example: Show VIEWERCMD=selectall</i></p>
	<b>selectfile</b>	<p>Select the currently displayed file in the folder tab the viewer was launched from.</p> <p><i>Example: Show VIEWERCMD=selectfile</i></p>
	<b>shortcutbar</b>	<p>Toggle the display of the shortcut bar on and off.</p> <p><i>Example: Show VIEWERCMD=shortcutbar</i></p>
	<b>slideshow</b>	<p>Toggle slideshow mode on and off.</p> <p><i>Example: Show VIEWERCMD=slideshow</i></p>
	<b>statusbar</b>	<p>Toggle the status bar on and off.</p>

			<i>Example: Show VIEWERCMD=statusbar</i>
		<b>toolbar</b>	Toggle display of the toolbar on and off.  <i>Example: Show VIEWERCMD=toolbar</i>
		<b>wallpaper</b>	Sets the currently displayed image as your Windows desktop wallpaper. By default the wallpaper will be set to <i>center</i> mode, but you can specify a mode by providing an additional keyword: <ul style="list-style-type: none"> <li>• <b>center</b>: The image will be centered on the desktop with a solid color fill surrounding it (this is the default).</li> <li>• <b>fill</b>: The image will be expanded or shrunk to fill the desktop (may leave black bars at the top and bottom or sides, to preserve the aspect ratio).</li> <li>• <b>fit</b>: The image will be expanded or shrunk to fill the desktop (the top and bottom or sides of the image may be cropped to preserve the aspect ratio).</li> <li>• <b>span</b>: The image will be spanned across multiple monitors.</li> <li>• <b>stretch</b>: The image will be stretched to fit the monitor, ignoring its aspect ratio.</li> <li>• <b>tile</b>: A small image will be tiled multiple times horizontally and vertically to fit the monitor.</li> </ul> <i>Example: Show VIEWERCMD=wallpaper,span</i>
		<b>zoom</b>	Adjusts the zoom level of the current image display. You must provide an additional keyword to indicate the desired zoom level: <ul style="list-style-type: none"> <li>• <b>+</b> : zoom in.</li> <li>• <b>-</b> : zoom out.</li> <li>• <b>+&lt;value&gt;</b> : Zoom in a specified amount (percentage).</li> <li>• <b>-&lt;value&gt;</b> : Zoom out a specified amount (percentage).</li> <li>• <b>&lt;value&gt;</b> : Zoom to an absolute percentage of the original size.</li> <li>• <b>fit</b>: Set zoom mode to "fit to page" - the image will be shrunk to fit the page if it's too large to fit all at once.</li> <li>• <b>grow</b>: Set zoom mode to "grow to page" - the image will be shrunk to fit if it's larger, and expanded to fill the page if it's smaller.</li> </ul>

			<ul style="list-style-type: none"> <li>• <b>tile</b>: Set zoom mode to "tile" - a small image will be tiled multiple times horizontally and vertically to fill the page.</li> <li>• <b>reset</b>: Reset the zoom level.</li> </ul> <p><i>Example: Show VIEWERCMD=zoom,+</i></p>
VIEWPANECMD	/K	<command>	<p>Sends commands to the viewer pane (if it's open) that are the equivalent of clicking the buttons in the viewer pane's toolbar. For example, you may want to assign these commands to a hotkey so you can trigger actions in the viewer pane from the keyboard.</p> <p>The available commands are: <b>prev</b> (previous file), <b>next</b> (next file), <b>rotateleft</b>, <b>rotateright</b>, <b>zoomfit</b>, <b>zoomgrow</b>, <b>zoomin</b>, <b>zoomout</b>, <b>zoomreset</b>.</p> <p><i>Example: Show VIEWPANECMD=zoomin</i></p>

## Split

The Split internal command can be used to:

- Split a file into two or more smaller parts
- UUEncode the split files for transmission via email/usenet

## Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Displays the <a href="#">Split File</a> dialog, which lets you split the selected files into parts of configurable size. The split files are written to the destination folder by default, and will have a numerical prefix appended to their names to indicate their order.  <b>Example: Split</b>
FROM	/M	<filename>, ...	Specify the file or files to be split. If this argument is not provided, all selected files in the current source file display will be used. Remember to enclose the filename with quotes if it contains a space. This is the default argument for the <b>Split</b> command and so you don't need to specify the <b>FROM</b> keyword.  <b>Example: Split "C:\Video Files\Video.avi"</b>
HERE	/S	(no value)	Split files will be written to the current source folder instead of the destination folder.  <b>Example: Split HERE</b>
SIZE	/K	<part size>	Specify the size of the split parts. This argument lets you automate the split function. The part size is given in bytes, kilobytes (if followed by the <b>KB</b> suffix), megabytes (if followed by the <b>MB</b> suffix) or gigabytes (if followed by the <b>GB</b> suffix).  <b>Example: Split HERE SIZE 1.44MB</b>
TO	/K	<destination>	Split files will be written to the specified folder instead of the current destination file display.  <b>Example: Split SIZE 1000000 TO C:\SplitOutput</b>
UUENCODE	/S	(no value)	Uuencodes the output files. Uuencoding is sometimes used when transmitting files across UseNet or another communications medium that doesn't support 8-bit data.  <b>Example: Split SIZE 1MB UUENCODE</b>

## Toolbar

The **Toolbar** internal command can be used to:

- Open or close toolbars or toolbar sets
- Open or close a floating toolbar, control its appearance and position on screen
- Save a new set or modify the Default Toolbar Set
- Import a toolbar into your configuration
- Reset your toolbars to the defaults

## Command Arguments:

Argument	Type	Possible values	Description
APPBAR	/K	<b>left</b>	When opening a floating toolbar, specify this argument to dock the toolbar with the left edge of the screen. Combine with <b>POS</b> to control which monitor it docks on.  <i>Example: <b>Toolbar my_tools STATE=float APPBAR=left TOGGLE</b></i>
		<b>top</b>	Docks the toolbar with the top of the screen.  <i>Example: <b>Toolbar my_tools STATE=floatactive APPBAR=top</b></i>
		<b>right</b>	Docks the toolbar with the right of the screen.  <i>Example: <b>Toolbar my_tools FLOAT POS=1920,0 APPBAR=right</b></i>
		<b>bottom</b>	Docks the toolbar with the bottom of the screen.  <i>Example: <b>Toolbar my_tools STATE=float APPBAR=bottom</b></i>
		<b>off</b>	Undocks the toolbar if it is already docked.  <i>Example: <b>Toolbar my_tools FLOAT APPBAR=off POS=50,100</b></i>
AUTOCLOSE	/S	(no value)	Add this argument when opening a floating toolbar to make the toolbar close automatically after running a command. This lets you have a toolbar that can pop open (by key press for example), run a command and close automatically again, and saves having to add the <b>Toolbar CLOSE=*this</b> command to all the buttons on the toolbar.  <i>Example: <b>Toolbar my_tools STATE=float AUTOCLOSE</b></i>
CLOSE	/O	(no value)	Closes the toolbar specified with the <b>NAME</b> argument. Note that <b>NAME</b> is the default argument for this command, the <b>NAME</b> keyword itself does not need to appear. The toolbar will be closed only in the current Lister unless <b>LOCAL=no</b> is also specified to close it globally.  <i>Example: <b>Toolbar Operations CLOSE</b></i>

		<b>all</b>	Closes all toolbars in the current Lister.  <i>Example: Toolbar CLOSE=all</i>
FLOAT	/O	(no value)	Opens the toolbar as a floating toolbar. This is the equivalent of specifying <b>STATE=float</b> . Using the various arguments provides more control over the floating toolbar. If combined with the <b>UPDATE</b> argument you can make changes to already-open toolbars.  <i>Example: Toolbar my_tools FLOAT TOGGLE</i>
		<b>default</b>	Opens the floating toolbar with the default appearance. If not specified (and no other arguments are specified), Opus will remember the toolbar's settings from the last time it was floated.  <i>Example: Toolbar my_tools FLOAT=default</i>
		<b>active</b>	Activates the newly opened toolbar (equivalent to <b>STATE=floatactive</b> ).  <i>Example: Toolbar my_tools FLOAT=active</i>
		<b>vertical</b>	Lay the floating toolbar out vertically instead of horizontally.  <i>Example: Toolbar my_tools FLOAT=vertical</i>
		<b>locked</b>	Lock the floating toolbar from being resized or moved.  <i>Example: Toolbar my_tools FLOAT=vertical,locked POS=100,100</i>
		<b>autohide</b>	Set the floating toolbar to auto-hide when it is docked.  <i>Example: Toolbar my_tools FLOAT=autohide APPBAR=bottom</i>
		<b>roundededges</b>	Display the toolbar with rounded edges.  <i>Example: Toolbar my_tools FLOAT=roundededges</i>
		<b>grid</b>	Make all buttons in the floating toolbar the same size.  <i>Example: Toolbar my_tools FLOAT=vertical,grid</i>

		<b>frame</b>	Display the floating toolbar with a frame.  <i>Example: Toolbar my_tools FLOAT=frame,roundededges</i>
		<b>noframe</b>	Display the floating toolbar with no frame.  <i>Example: Toolbar my_tools FLOAT=noframe TOGGLE</i>
		<b>transparent</b>	Make the floating toolbar transparent.  <i>Example: Toolbar my_tools FLOAT=noframe,transparent</i>
		<b>taskbar</b>	Display the floating toolbar with the system taskbar theme.  <i>Example: Toolbar my_tools FLOAT=taskbar,autohide</i>
		<b>glass</b>	Display the floating toolbar using glass.  <i>Example: Toolbar my_tools FLOAT=glass</i>
		<b>topopus</b>	Set the floating toolbar to appear in top of other Opus windows.  <i>Example: Toolbar my_tools FLOAT=topopus</i>
		<b>toplevel</b>	Set the floating toolbar to appear on top of all other windows.  <i>Example: Toolbar my_tools FLOAT=toplevel</i>
		<b>hotkeys</b>	Enable hotkeys in the floating toolbar.  <i>Example: Toolbar my_tools FLOAT=hotkeys</i>
IMPORT	/S	<i>(no value)</i>	Imports the toolbar specified with the <b>NAME</b> argument into your Opus configuration. For this command, <b>NAME</b> must specify the full path of the toolbar file to import. This would be most useful on the context menu for <b>.dop</b> (toolbar) files.  <i>Example: Toolbar NAME {f} IMPORT</i>
LINE	/K	<i>&lt;line&gt;</i>	When opening a toolbar, specifies the toolbar line it is to appear on. The line number is given relative to the specified toolbar state. For example, when <b>STATE</b> is bottom, the line given is relative to the group of toolbars at the bottom of the Lister. <i>&lt;line&gt;</i> is counted

			<p>from 0.</p> <p><i>Example: <b>Toolbar Operations LINE 1</b></i></p>
		<line>,<position>	<p>Specifies both the line and the position of the new toolbar. This lets you open a toolbar on the same line as an existing toolbar.</p> <p>&lt;position&gt; is given as the number of pixels from the left (or top, for vertical toolbars) edge of the Lister window.</p> <p><i>Example: <b>Toolbar Images LINE 1,500</b></i></p>
LIST	/O	(no value)	<p>Displays a generated list of all your toolbars (acts as a <a href="#">dynamic button</a>). This lets you turn toolbars on or off from a drop-down menu.</p> <p><i>Example: <b>Toolbar LIST</b></i></p>
		sets	<p>Displays a list of <a href="#">toolbar sets</a> and lets you switch between them.</p> <p><i>Example: <b>Toolbar LIST=sets</b></i></p>
		add	<p>When a toolbar set is chosen from the generated list, it will be loaded in "add" mode - meaning the toolbars within it will be added to any currently open toolbars. This overrides the state of the option saved within the set itself.</p> <p><i>Example: <b>Toolbar LIST=sets,add</b></i></p>
		replace	<p>When a toolbar set is chosen from the generated list, it will be loaded in "replace" mode - meaning the toolbars within it will replace any currently open toolbars.</p> <p><i>Example: <b>Toolbar LIST=sets,replace</b></i></p>
		toggle	<p>When a toolbar set is chosen from the generated list, it will turn off any previously loaded set, but leave the default toolbars unaffected. Selecting the set again will turn it off.</p> <p><i>Example: <b>Toolbar LIST=sets,toggle</b></i></p>
		usekeys	<p>When a toolbar set is chosen from the generated list, Opus will check the state of the <b>Shift</b> and <b>Ctrl</b> keys to determine the behaviour. Holding the <b>Shift</b> key will select "add" mode, and the <b>Ctrl</b> key will select "replace" mode.</p> <p><i>Example: <b>Toolbar LIST=sets,usekeys</b></i></p>

		<b>nocontext</b>	Prevents the toolbar sets in the generated list from appearing "checked" when Opus considers they are currently active.  <i>Example: <b>Toolbar LIST=sets,add,nocontext</b></i>
LOADSET	/O	(no value)	Loads the set specified by the NAME argument. The default behaviour specified in the set ("add", "replace" or "toggle") will be used unless overridden.  <i>Example: <b>Toolbar my_set LOADSET</b></i>
		<b>add</b>	Adds the toolbars in the specified set to any currently open toolbars.  <i>Example: <b>Toolbar my_set LOADSET=add</b></i>
		<b>replace</b>	Replaces any currently open toolbars with those in the specified set.  <i>Example: <b>Toolbar my_set LOADSET=replace</b></i>
		<b>toggle</b>	Turns off any previously loaded set, but leaves the default toolbars unaffected. If the specified set is already open it will be closed (so running the command twice has the effect of toggling the set on and off).  <i>Example: <b>Toolbar my_set LOADSET=toggle</b></i>
		<b>usekeys</b>	Uses the state of the <b>Shift</b> and <b>Ctrl</b> keys to determine the behaviour when loading the set. Holding the <b>Shift</b> key will select "add" mode, and the <b>Ctrl</b> key will select "replace" mode.  <i>Example: <b>Toolbar my_set LOADSET=usekeys</b></i>
LOCAL	/O	(no value)	Opens a toolbar local to the current Lister. Note that as this is the default behaviour, specifying <b>LOCAL</b> or <b>LOCAL=yes</b> has no effect.  <i>Example: <b>Toolbar Images LOCAL</b></i>
		<b>no</b>	Opens the toolbar globally in all Listers.  <i>Example: <b>Toolbar Images LOCAL=no</b></i>
NAME		<toolbar name>	Specifies the name of the toolbar - used in conjunction with the other arguments to open or close the specified toolbar. This is the

			<p>default argument for the <b>Toolbar</b> command and so the <b>NAME</b> keyword does not need to be used. If no other arguments to the command are provided, the named toolbar will be turned on.</p> <p><i>Example: <b>Toolbar Images</b></i></p>
		<b>*this</b>	<p>Makes the command apply to the current toolbar. When used from a toolbar button, the <b>Toolbar</b> command will apply to the toolbar the button is contained within.</p> <p><i>Example: <b>Toolbar CLOSE *this</b></i></p>
POS	/K	<x>, <y>	<p>Specifies the on-screen position when opening a floating toolbar. The position is given relative to the top-left corner of the primary display monitor. This can also be used to control which monitor a docked toolbar appears on (by combining it with the <b>APPBAR</b> argument).</p> <p><i>Example: <b>Toolbar Applications STATE=float POS 1200,80</b></i></p>
		<b>mouse</b>	<p>The floating toolbar will appear centered over the current position of the mouse pointer.</p> <p><i>Example: <b>Toolbar LauncherBar STATE=float POS=mouse</b></i></p>
		<b>mousel</b>	<p>The floating toolbar is left-aligned with the position of the mouse pointer.</p> <p><i>Example: <b>Toolbar Images STATE=float POS=mousel TOGGLE</b></i></p>
		<b>mouser</b>	<p>The floating toolbar is right-aligned with the mouse pointer.</p> <p><i>Example: <b>Toolbar Applications STATE=float POS=mouser</b></i></p>
RESETDEFAULTS	/K	(no value)	<p>Resets the toolbars to the factory defaults. All existing toolbars will be closed, the default toolbars will be reset, and then only those toolbars will be opened.</p> <p>If you have made changes to the default toolbars those changes will be lost, but none of your other toolbars will be affected (other than being turned off if they were open when the command was run).</p>

			<i>Example: Toolbar</i> <b>RESETDEFAULTS</b>
		<b>stateonly</b>	Doesn't modify the toolbars themselves - instead, only the current set of toolbars is reset to the defaults. Any toolbars you have created yourself will be closed and the default set of toolbars displayed, but any modifications you have made to the default toolbars will be unaffected.  <i>Example: Toolbar</i> <b>RESETDEFAULTS=stateonly</b>
		<b>quiet</b>	Suppresses the confirmation prompt before resetting the toolbars.  <i>Example: Toolbar</i> <b>RESETDEFAULTS=quiet</b>
SAVESET	/O	(no value)	Saves the current toolbars as a <a href="#">toolbar set</a> . By default the new set will be set to "replace" mode. Opus will prompt for a name for the set unless you specify one with the <b>NAME</b> argument.  <i>Example: Toolbar</i> <b>SAVESET</b>
		<b>add</b>	Saves the toolbar set in "add" mode.  <i>Example: Toolbar</i> <b>SAVESET=add</b>
		<b>toggle</b>	Saves the toolbar set in "toggle" mode.  <i>Example: Toolbar</i> <b>SAVESET=toggle</b>
		<b>quiet</b>	Suppresses the confirmation prompt when saving over an existing toolbar set.  <i>Example: Toolbar</i> <b>my_set SAVESET=toggle,quiet</b>
SETDEFAULT	/O	(no value)	Saves the toolbars in the current Lister as the Default Toolbar Set.  <i>Example: Toolbar</i> <b>SETDEFAULT</b>
		<b>quiet</b>	Suppresses the confirmation prompt before saving the set.  <i>Example: Toolbar</i> <b>SETDEFAULT=quiet</b>
STATE	/K	<b>top</b>	Opens the specified toolbar at the top of the Lister. This is the default behaviour for this command.

			<i>Example: Toolbar Applications</i> <b>STATE=top</b>
		<b>bottom</b>	Opens the toolbar at the bottom of the Lister. These toolbars appear at the very bottom of the window, underneath all other Lister elements.  <i>Example: Toolbar Images</i> <b>STATE=bottom TOGGLE</b>
		<b>fdbottom</b>	Opens the toolbar at the bottom of the file display(s). Similar to <b>bottom</b> but will not extend left under the leftmost folder tree.  <i>Example: Toolbar Images</i> <b>STATE=fdbottom TOGGLE</b>
		<b>left</b>	Opens the toolbar at the left of the Lister. These toolbars appear vertically at the very left edge of the window.  <i>Example: Toolbar Drives</i> <b>STATE=left</b>
		<b>right</b>	Opens the toolbar at the right of the Lister. These toolbars appear vertically at the very right edge of the window.  <i>Example: Toolbar Drives</i> <b>STATE=right LINE=1 TOGGLE</b>
		<b>center</b>	Opens the toolbar in the center of the Lister. These toolbars appear between the two file displays in a <a href="#">dual-display</a> Lister - they will be either horizontal or vertical depending on the file display arrangement.  <i>Example: Toolbar Drives</i> <b>STATE=center TOGGLE</b>
		<b>viewpane</b>	Opens the toolbar between the file displays and the viewer pane, when the viewer pane is at the right of the Lister. If the viewer pane is not open, or is at the bottom of the window, this value behaves the same as <b>right</b> .  <i>Example: Toolbar Applications</i> <b>STATE=viewpane</b>
		<b>tree</b>	Opens the toolbar between the left file display and the folder tree.  <i>Example: Toolbar Drives</i> <b>STATE=tree</b>
		<b>float</b>	Floats the toolbar. You can specify the position of the floating toolbar with the <b>POS</b> argument.

			<i>Example: Toolbar Applications</i> <b>STATE=float TOGGLE</b>
		<b>floatactive</b>	Floats the toolbar and makes it active. This can be used to open a toolbar and have it take input focus so you can open its drop-down menus from the keyboard.  <i>Example: Toolbar Tools</i> <b>STATE=floatactive POS=0,0</b>
		<b>fdb</b>	Changes which toolbar is used for the <a href="#">File Display border</a> .  <i>Example: Toolbar my_fdb</i> <b>STATE=fdb</b>
TOGGLE	/S	(no value)	Toggles the toolbar specified with the <b>NAME</b> argument. If the toolbar is not currently opened it will be opened, otherwise it will be closed.  <i>Example: Toolbar Operations</i> <b>TOGGLE LINE=1</b>
UPDATE	/S	(no value)	Updates an existing floating toolbar with new appearance and position values. If the toolbar is not currently opened it will be opened using the defined appearance.  <i>Example: Toolbar my_tools</i> <b>FLOAT=glass POS=50,50 UPDATE</b>

## Undo

The **Undo** internal command can be used to:

- Undo the last undoable file operation (undo delete to recycle bin, etc)
- Undo any one of the previous operations via a menu
- Display a list of undoable operations and undo all or individual operations

### Command Arguments:

Argument	Type	Possible values	Description
(no argument)	-	-	Undoes the most recent undoable file operation. Not all file operations are undoable - for example, files deleted from network or removable drives can not be undone.  <i>Example: <b>Undo</b></i>
ITEM	/K	<index>	Undoes the specified action from the undo list. The <index> indicates the operation to undo, as displayed in the undo list.  <i>Example: <b>Undo ITEM 3</b></i>
LIST	/O	(no value)	Displays a list of undoable actions (acts as a <a href="#">dynamic button</a> ). Any operation from the list can be undone simply by selecting it from the list.  <i>Example: <b>Undo LIST</b></i>
		<b>nokeys</b>	Disables the automatically-assigned hotkeys that are normally added to the generated list.  <i>Example: <b>Undo LIST=nokeys</b></i>
		<b>menu</b>	Displays the undo list in a sub-menu.  <i>Example: <b>Undo LIST=menu,nokeys</b></i>
PAGE	/S	(no value)	Displays the <b>Undo Log</b> page of the <a href="#">utility panel</a> . This displays a list of undoable actions, and lets you undo individual operations or all of them.  <i>Example: <b>Undo PAGE</b></i>

## External control codes

The external control codes are used to pass information like the names of selected files to external programs. For example, the command **notepad.exe {f}** would launch Notepad, and pass the name of the first selected file as an argument on its command line.

External control codes can (despite the name) also be used with the internal commands. For example, the command **CreateFolder {date|yyyyMMdd}** will create a new folder automatically named after the current date.

Most control codes have both a short and a long form. The short form requires less typing and produces a smaller function definition, while the long form is more descriptive and may make the function more understandable. Note that, unlike the arguments for internal commands, the external control codes are case-sensitive.

See the page on [passing files to external programs](#) for more information about the external control codes.

### Codes for passing filenames

The following codes are used to pass the names of selected files to external programs. The filename codes available offer all possible combinations of the following five criteria:

- **Full paths or filename only** - whether the full path and name of selected items are passed (e.g. *C:\Program Files\GPSsoftware\Directory Opus\dopus.exe*) or just the filename (e.g. *dopus.exe*).
- **One at a time or all at once** - when multiple files are selected, this determines whether a command is repeated once for each selected file, or run once only, with all selected filenames passed on the same command line (separated by spaces).
- **Selected items required or not required** - when selected items are required and none are selected, the command will not be run at all. When selected items are not required, the command will still be run and it will act as if the filename code was not present at all.
- **Long filenames or short filenames** - short filenames are useful for running 16 bit programs or other legacy software that can't handle long filenames.
- **Source or destination file display** - in a dual-display Lister, you can pass selected files from the destination file display as well as from the source.

The long form of each code is built from a combination of keywords that reflect the five criteria. If a long code ends in a \$ sign it is the "selected items required" form of the code - it is said to "need" selected items.

Long form	Short form	Description
-----------	------------	-------------

{filepath}	{f!}	<p>Passes the full path and filename of each selected item. Files are passed one at a time - a command that uses this code will be repeated once for each additional selected item. If no files are selected the command will still be run, passing an empty string for this code.</p> <p><i>Full paths, one at a time, selected items not required, long filenames, source file display</i></p>
{filepath\$}	{f}	<p>The "need" form of <b>{filepath}</b> (requires at least one selected item in the file list).</p> <p><i>Full paths, one at a time, selected items required, long filenames, source file display</i></p>
{file}	{o!}	<p>Passes the filename only of each selected item (without its path). Multiple files are passed one at a time.</p> <p><i>Filenames only, one at a time, selected items not required, long filenames, source file display</i></p>
{file\$}	{o}	<p>The "need" form of <b>{file}</b>.</p> <p><i>Filenames only, one at a time, selected items required, long filenames, source file display</i></p>
{allfilepath}	{F!}	<p>Passes the full path and filename of all selected items at once. The command will only be run once, no matter how many items are selected - the names of all selected items will be passed on the command line separated by spaces. You can use the <b>sep=</b> modifier (described below) to specify an alternative separator character.</p> <p><i>Full paths, all at once, selected items not required, long filenames, source file display</i></p>
{allfilepath\$}	{F}	<p>The "need" form of <b>{allfilepath}</b>.</p> <p><i>Full paths, all at once, selected items required, long filenames, source file display</i></p>
{allfile}	{O!}	<p>Passes the filename only of all selected items at once. File paths are not passed.</p> <p><i>Filenames only, all at once, selected items not required, long filenames, source file display</i></p>
{allfile\$}	{O}	<p>The "need" form of <b>{allfile}</b>.</p> <p><i>Filenames only, all at once, selected items required, long filenames, source file display</i></p>
{filepathshort}	{fs!}	<p>Passes the full path and filename of each selected item, in short (8.3) format. This is useful for running 16 bit programs or other legacy software that can't handle long filenames.</p> <p><i>Full paths, one at a time, selected items not required, short filenames, source file display</i></p>
{filepathshort\$}	{fs}	<p>The "need" form of <b>{filepathshort}</b>.</p> <p><i>Full paths, one at a time, selected items required, short filenames, source file display</i></p>

{fileshort}	{os!}	<p>Passes the filename only (no paths) of each selected item, in short (8.3) format.</p> <p><i>Filenames only, one at a time, selected items not required, short filenames, source file display</i></p>
{fileshort\$}	{os}	<p>The "need" form of <b>{fileshort}</b>.</p> <p><i>Filenames only, one at a time, selected items required, short filenames, source file display</i></p>
{allfilepathshort}	{Fs!}	<p>Passes the short (8.3) form of the full path and filename of all selected items at once.</p> <p><i>Full paths, all at once, selected items not required, short filenames, source file display</i></p>
{allfilepathshort\$}	{Fs}	<p>The "need" form of <b>{allfilepathshort}</b>.</p> <p><i>Full paths, all at once, selected items required, short filenames, source file display</i></p>
{allfileshort}	{Os!}	<p>Passes the filename only of all selected items at once, in short (8.3) format.</p> <p><i>Filenames only, all at once, selected items not required, short filenames, source file display</i></p>
{allfileshort\$}	{Os}	<p>The "need" form of <b>{allfileshort}</b>.</p> <p><i>Filenames only, all at once, selected items required, short filenames, source file display</i></p>
{filepathdest}	{fd!}	<p>Passes the full path and filename of each selected item in the destination file display. Files are passed one at a time - a command that uses this code will be repeated once for each additional selected item. If no files are selected in the destination, or there is currently no destination, the command will still be run, passing an empty string for this code.</p> <p><i>Full paths, one at a time, selected items not required, long filenames, destination file display</i></p>
{filepathdest\$}	{fd}	<p>The "need" form of <b>{filepathdest}</b>. If there are no files selected in the destination, or there is currently no destination, the command will not be executed.</p> <p><i>Full paths, one at a time, selected items required, long filenames, destination file display</i></p>
{filedest}	{od!}	<p>Passes the filename only of each selected item from the destination file display (without its path). Multiple files are passed one at a time.</p> <p><i>Filenames only, one at a time, selected items not required, long filenames, destination file display</i></p>
{filedest\$}	{od}	<p>The "need" form of <b>{filedest}</b>.</p> <p><i>Filenames only, one at a time, selected items required, long filenames, destination file display</i></p>
{allfilepathdest}	{Fd!}	<p>Passes the full path and filename of all selected items, from the destination file display, all at once. The command will only be run once, no matter how many items are selected - the names of all selected items will be passed on the command line separated by spaces.</p>

		<i>Full paths, all at once, selected items not required, long filenames, destination file display</i>
{allfilepathdest\$}	{Fd}	The "need" form of <b>{allfilepathdest}</b> .  <i>Full paths, all at once, selected items required, long filenames, destination file display</i>
{allfiledest}	{Od!}	Passes the filename only of all selected items at once, from the destination file display. File paths are not passed.  <i>Filenames only, all at once, selected items not required, long filenames, destination file display</i>
{allfiledest\$}	{Od}	The "need" form of <b>{allfiledest}</b> .  <i>Filenames only, all at once, selected items required, long filenames, destination file display</i>
{filepathshortdest}	{fsd!}	Passes the full path and filename of each selected item, in short (8.3) format, from the destination file display. This is useful for running 16 bit programs or other legacy software that can't handle long filenames.  <i>Full paths, one at a time, selected items not required, short filenames, destination file display</i>
{filepathshortdest\$}	{fsd}	The "need" form of <b>{filepathshortdest}</b> .  <i>Full paths, one at a time, selected items required, short filenames, destination file display</i>
{fileshortdest}	{osd!}	Passes the filename only (no paths) of each selected item in the destination file display, in short (8.3) format.  <i>Filenames only, one at a time, selected items not required, short filenames, destination file display</i>
{fileshortdest\$}	{osd}	The "need" form of <b>{fileshortdest}</b> .  <i>Filenames only, one at a time, selected items required, short filenames, destination file display</i>
{allfilepathshortdest}	{Fsd!}	Passes the short (8.3) form of the full path and filename of all selected items at once, from the destination file display.  <i>Full paths, all at once, selected items not required, short filenames, destination file display</i>
{allfilepathshortdest\$}	{Fsd}	The "need" form of <b>{allfilepathshortdest}</b> .  <i>Full paths, all at once, selected items required, short filenames, destination file display</i>
{allfileshortdest}	{Osd!}	Passes the filename only of all selected items at once, in short (8.3) format, from the destination file display.  <i>Filenames only, all at once, selected items not required, short filenames, destination file display</i>
{allfileshortdest\$}	{Osd}	The "need" form of <b>{allfileshortdest}</b> .  <i>Filenames only, all at once, selected items required, short filenames, destination file display</i>

The filename codes all support the following modifiers. These can be added to the control code sequence using a vertical bar (|) to separate the filename code from the modifier - for example, **{file|noext}** strips the file extension from the returned filename. Multiple modifiers can be provided using additional vertical bars to separate them - for example **{file|noext|escwild}** combines both **noext** and **escwild** together.

Modifier	Description
<b>noext</b>	Strips the filename extension from filenames. For example, if <b>{file}</b> returned <i>cat_photo.jpg</i> then <b>{file noext}</b> would return <i>cat_photo</i> . Note that if the item is a folder then its name will not be changed, even if it contains a dot, as folders are not considered to have file extensions.
<b>ext</b>	Returns only the filename extension (the opposite of <b>noext</b> ). For example, if <b>{file}</b> returned <i>cat_photo.jpg</i> then <b>{file ext}</b> would return <i>.jpg</i> . Note that if the item is a folder then nothing is returned, even the folder's name contains a dot.
<b>ext2</b>	Returns the filename extension without the dot. For example, if <b>{file ext}</b> returned <i>.jpg</i> then <b>{file ext2}</b> would return <i>jpg</i> .
<b>ext=&lt;ext&gt;</b>	Replaces the original filename extension with the specified one. For example, <b>{file ext=tmp}</b> would return the name of each selected file with the extension changed to <i>.tmp</i> . Folder names are not changed, even if they contain dots.
<b>escbackslash</b>	<p>Automatically escapes any occurrences of \ in the filename, turning them into \\. </p> <p>This can be necessary when using codes like <b>{filepath}</b> to insert paths into strings which will have sequences like \n and \\ interpreted specially.</p> <p>For example, a command to add the selected file from the source and the selected file from the destination to the clipboard on separate lines:</p> <p><b><a href="#">Clipboard</a> EXPANDNEWLINES SET {filepath escbackslash}\n{filepathdest escbackslash}.</b></p> <p>In this example, if the source file was <i>C:\New Folder\test.txt</i> then it would be converted to <i>C:\\New Folder\\test.txt</i>. If the \N sequence had not been escaped, it would have removed the letter N from the path and put <i>ew Folder\test.txt</i> on a separate line. (There are more subtle cases when dealing with folder paths, but they are less important with file paths.)</p>
<b>escnl</b>	<p>Automatically escapes any occurrences of \n in the filename, turning them into \\n.</p> <p>(This is a weaker version of <b>escbackslash</b>, above. You should probably use <b>escbackslash</b> instead.)</p> <p>Using <b>escnl</b> helps when using codes like <b>{filepath}</b> as the message text for a dialog code like <b>{dlgstring}</b>. For example, the path <i>C:\New Folder\test.txt</i> would be converted to <i>C:\\New Folder\test.txt</i>. If the \N sequence wasn't</p>

	escaped, it would have been converted into a line-break in the dialog message text.
<b>escregexp</b>	Automatically escapes any regular expression characters in the filename. Used when passing filenames to internal commands that understand <a href="#">regular expressions</a> , in case a filename contains characters like \, ., ( and ) which have special meaning to some functions. For example, <b>{file escregexp noext}</b> would turn <i>Accounts (2010).xls</i> into <i>Accounts \ (2010\)</i> .
<b>escwild</b>	Automatically escapes any wildcard characters in the filename. Used when passing filenames to internal commands that understand <a href="#">standard wildcards</a> , in case a filename contains characters like ( and ) which have special meaning to some functions. For example, <b>{file escwild noext}</b> would turn <i>Accounts (2010).xls</i> into <i>Accounts '(2010)'</i> .
<b>file</b>	Redirects the filenames to a temporary text file. This is useful with external programs that can accept a list of files from a text file rather than on the command line. (In turn, that is useful because command lines have a maximum length which may limit the number of filenames you can pass directly.) The name of the temporary file is passed on the command line in place of the filenames themselves. In the text file, each filename is separated by a space. If a filename or its path contains an embedded space it will be surrounded by quotes. For example, <b>{allfilepath file}</b> .
<b>filem</b>	The same as <b>file</b> except that each filename is written to a separate line in the text file. For example, <b>{allfilepath filem}</b> .
<b>fileq</b>	The same as <b>file</b> except that each filename is always surrounded by quotes, whether it contains an embedded space or not. For example, <b>{allfilepath fileq}</b> .
<b>filemq</b>	The same as <b>filem</b> except that each filename is always surrounded by quotes. For example, <b>{allfilepath filemq}</b> .
<b>utf8</b>	Forces the file written by <b>file</b> , <b>filem</b> or <b>fileq</b> to use UTF-8 format. For example, <b>{allfilepath filem utf8}</b> .
<b>ucsle</b>	Forces the file written by <b>file</b> , <b>filem</b> or <b>fileq</b> to use UCS-16LE format. For example, <b>{allfilepath filem ucsle}</b> .
<b>ucsbe</b>	Forces the file written by <b>file</b> , <b>filem</b> or <b>fileq</b> to use UCS-16BE format. For example, <b>{allfilepath filem ucsbe}</b> .
<b>nobom</b>	Prevents a BOM from being written that identifies the file type. For example, <b>{allfilepath filem utf8 nobom}</b> .
<b>cronly</b>	When used with <b>filem</b> or <b>filemq</b> , makes it so the file has only CR (carriage return) characters between each line. The default is CR,LF pairs as is standard on Windows. For example, <b>{allfilepath filem cronly}</b> .
<b>lfonly</b>	When used with <b>filem</b> or <b>filemq</b> , makes it so the file has only LF (line feed) characters between each line. The default is CR,LF pairs as is standard on Windows. For example, <b>{allfilepath filem lfonly}</b> .
<b>nopath</b>	Returns only the final component of the path. For example, if <b>{filepath}</b> returns <i>C:\Program Files\GPSsoftware\Directory Opus\</i> , <b>{filepath nopath}</b> would return <i>Directory Opus\</i> . You would normally use something like <b>{file}</b> to get just the name of something, but using <b>{filepath nopath}</b> has the subtle difference of including the \ on the end of the name when the item is a folder.

<b>noroot</b>	Removes the root (i.e. the first component) from the file's path. For example, if <b>{filepath}</b> returns <i>C:\Windows\notepad.exe</i> , <b>{filepath noroot}</b> would return <i>Windows\notepad.exe</i> . With UNC network paths, the result is relative to the share, with the computer and share itself removed. With FTP paths, the result is relative to the site's root and excludes the site itself. You can combine <b>noroot</b> with <b>..</b> to get a parent path with the root removed. For example, <b>{filepath .. noroot}</b>
<b>noshort</b>	Suppresses automatic shortening of file paths which are longer than 260 characters. Opus itself can handle very long paths but many Windows programs cannot. When sending very long paths to external programs, Opus normally uses the short (8.3) versions to reduce problems. Use <b>noshort</b> to prevent this when you know the target program can cope. For example, <b>{filepath noshort}</b> .
<b>noterm</b>	Removes the trailing path separator from folder paths. For example, <b>{filepath}</b> might return <i>C:\Program Files\</i> whereas <b>{filepath noterm}</b> would return <i>C:\Program Files</i> .
<b>subdir</b>	Replaces characters in the file's path so that the full path can be added below another path. For example, <b>{filepath subdir}</b> might return <i>C;\Windows\notepad.exe</i> . Note that a semi-colon has replaced the colon, allowing the path to be used relative to another directory, say if you wanted to create a backup of the file called <i>D:\Backups\C;\Windows\notepad.exe</i> .
<b>sep=</b>	Lets you specify an alternative separator for codes that insert multiple filenames on the one line. For example, <b>{allfilepath}</b> normally inserts all selected filepaths separated by spaces. You could use <b>{allfilepath sep=,}</b> to have the filepaths comma-separated instead.
<b>..</b>	Modifies the code to refer to the selected item's parent folder rather than the item itself. You can pass multiple <b>..</b> modifiers separated by <b>\</b> to return folders higher up the tree. For example <b>{filepath ..\..}</b> would return the name of the item's parent's parent.
<b>\</b>	Modifies the code to refer to the root of the drive of the selected item. For example <b>{filepath }</b> might return <i>C:\</i> .

In addition to the above codes, the following control codes are supported to provide compatibility with Windows Explorer. They can not be used in MS-DOS batch functions.

Modifier	Equivalent To
%1	{filepath}
%2	{filepath}
%L	{filepath}
%*	{allfilepath}

### Codes for passing paths

The following codes are used to pass various paths to external programs. The path codes available offer all possible combinations of the following three criteria:

- **Which path** - either the current source or destination path, or in a dual-display Lister, the left (top) or right (bottom) paths irrespective of their source/destination status.
- **Path required or not required** - when a path is required, the command will not be run at all if that path is not valid - for example, when there is no valid destination folder. When the path is not required, the command will still be run and it will act as if the path code was not present at all.
- **Long filenames or short filenames** - short filenames are useful for running 16 bit programs or other legacy software that can't handle long filenames.

The long form of each code is built from a combination of keywords that reflect the three criteria. If a long code ends in a \$ sign it is the "path required" form of the code - it is said to "need" a valid path. Note that the **left** and **right** paths do not support the "required" forms.

Long form	Short form	Description
{sourcepath}	{s!}	Source path, long filenames, not required.
{sourcepath\$}	{s}	Source path, long filenames, required.
{destpath}	{d!}	Destination path, long filenames, not required.
{destpath\$}	{d}	Destination path, long filenames, required.
{sourcepathshort}	{ss!}	Source path, short filenames, not required.
{sourcepathshort\$}	{ss}	Source path, short filenames, required.
{destpathshort}	{ds!}	Destination path, short filenames, not required.
{destpathshort\$}	{ds}	Destination path, short filenames, required.
{leftpath}	{l}	Left (top) path, long filenames.
{rightpath}	{r}	Right (bottom) path, long filenames.
{leftpathshort}	{ls}	Left path, short filenames.

{rightpathshort}	{rs}	Right path, long filenames.
------------------	------	-----------------------------

In addition to the path codes related to current Lister paths, the following codes allow you to pass the values of certain system or application-specific paths.

Long form	Short form	Description
{apppath}	{p}	Returns the path of an installed application, as listed in the <i>App Paths</i> key in the registry. For example, <b>{apppath winword.exe}</b> would insert the install path of Microsoft Word.
{apppathshort}	{ps}	The path of an installed application, in short (8.3) format.
{alias}	{A}	Returns the path of a <a href="#">folder alias</a> . For example, <b>{alias mydocuments}</b> would return the path to your documents folder.  You can use <b>{alias libraries}</b> to resolve the path of a library folder; for example, <b>{alias libraries}/Documents</b> would return the real (default) path of the <i>Documents</i> library.
{aliasshort}	{As}	The path of a folder alias, in short (8.3) format.

The path codes support the following modifiers. These can be added to the control code sequence using a vertical bar (|) to separate the filename code from the modifier - for example, **{sourcepath|noterm}** strips the trailing termination character from the path. Multiple modifiers can be provided using additional vertical bars to separate them - for example **{sourcepath|nopath|noterm}** combines both **nopath** and **noterm** together.

Modifier	Description
<b>escbackslash</b>	Automatically escapes any occurrences of \ in the path, turning them into \\.  This can be necessary when using codes like <b>{sourcepath}</b> to insert paths into strings which will have sequences like \n and \\ interpreted specially.  For example, a command to add the source and destination paths to the clipboard on separate lines:  <a href="#">Clipboard</a> <b>EXPANDNEWLINES SET</b> <b>{sourcepath escbackslash}\n{destpath escbackslash}</b> .  In this example, if the source path was <i>C:\New Folder\</i> then it would be

	<p>converted to <code>C:\\New Folder\\</code>. If the <code>\N</code> sequence had not been escaped, it would have removed the letter <code>N</code> from the path and put <code>ew Folder\</code> on a separate line. More subtly, if the <code>\</code> on the end of the source path was not also escaped, it would have interfered with the <code>\n</code> which comes between the two paths in the example command.</p>
<b>escnl</b>	<p>Automatically escapes any occurrences of <code>\n</code> in the path, turning them into <code>\\n</code>.</p> <p>(This is a weaker version of <b>escbackslash</b>, above. You should probably use <b>escbackslash</b> instead.)</p> <p>Using <b>escnl</b> helps when using codes like <b>{sourcepath}</b> as the message text for a dialog code like <b>{dlgstring}</b>. For example, the path <code>C:\New Folder\</code> would be converted to <code>C:\\New Folder\</code>. If the <code>\n</code> sequence weren't escaped, it would have been converted into a line-break in the dialog message text.</p>
<b>escregexp</b>	<p>Automatically escapes any regular expression characters in the path. Used when passing paths to internal commands that understand <a href="#">regular expressions</a>, in case a path contains characters like <code>\</code>, <code>.</code>, <code>(</code> and <code>)</code> which have special meaning to some functions. For example, <b>{sourcepath escregexp noterm}</b> would turn <code>C:\Accounts (2010)\</code> into <code>C:\\Accounts \ (2010\)</code>.</p>
<b>escwild</b>	<p>Automatically escapes any wildcard characters in the path. Used when passing paths to internal commands that understand <a href="#">standard wildcards</a>, in case a path contains characters like <code>(</code> and <code>)</code> which have special meaning to some functions. For example, <b>{sourcepath escwild noterm}</b> would turn <code>C:\Accounts (2010)\</code> into <code>C:\Accounts '(2010)'</code>.</p>
<b>nopath</b>	<p>Returns only the final component of the path. For example, if <b>{sourcepath}</b> returns <code>C:\Program Files\GPSSoftware\Directory Opus\</code>, <b>{sourcepath nopath}</b> would return <code>Directory Opus\</code>.</p>
<b>noroot</b>	<p>Returns the path without the root, i.e. without the first component. For example, if <b>{sourcepath}</b> returns <code>C:\Program Files\GPSSoftware\Directory Opus\</code>, <b>{sourcepath noroot}</b> would return <code>Program Files\GPSSoftware\Directory Opus\</code>. With UNC network paths, the result is relative to the share, with the computer and share itself removed. With FTP paths, the result is relative to the site's root and excludes the site itself. You can combine <b>noroot</b> with <code>..</code> to get a parent path with the root removed. For example, <b>{sourcepath .. noroot}</b></p>
<b>noshort</b>	<p>Suppresses automatic shortening of paths which are longer than 260 characters. Opus itself can handle very long paths but many Windows programs cannot. When sending very long paths to external programs, Opus normally uses the short (8.3) versions to reduce problems. Use <b>noshort</b> to prevent this when you know the target program can cope. For example, <b>{sourcepath noshort}</b>.</p>
<b>noterm</b>	<p>Strips the trailing path separator from the returned path. For example, <b>{sourcepath noterm}</b> might return <code>C:\Program Files\GPSSoftware\Directory Opus</code>.</p>
<b>subdir</b>	<p>Replaces characters in the path so that it can be added below another path. For example, <b>{sourcepath subdir}</b> might return <code>C;\Program Files</code>. Note that a semi-colon has replaced the colon, allowing the path to be used as a</p>

	sub-directory, say if you wanted to backup the files from there into <i>D:\Backups\C;\Program Files</i> .
..	Modifies the code to refer to the path's parent rather than the path itself. You can pass multiple .. modifiers separated by \ to return folders higher up the tree. For example <b>{destpath ..\..}</b> would return the parent of the parent of the current destination path.
\	Modifies the code to refer to the root of the drive of the specified path. For example <b>{sourcepath }</b> might return <b>C:\</b> .

## Codes to display dialogs

The following codes can be used to display various simple dialogs when the command is run. This lets you provide information (like selecting a file or a folder, or entering a string) at "run-time" rather than incorporating the value into the command itself. For more flexibility, you can use a [script dialog](#).

Long form	Short form	Description
{dlgopen}	{Rf}	Displays an <b>Open File</b> dialog, which lets you select an existing filename to pass to the command.
{dlgmulti}	{Rm}	Displays an <b>Open File</b> dialog in multiple selection mode, which lets you select one or more existing files.
{dlgsave}	{RF}	Displays a <b>Save File</b> dialog, which lets you enter a new filename to pass to the command.
{dlgfolder}	{Rd}	Displays a <b>Select Folder</b> dialog, which lets you select a folder to pass to the command.
{dlgstring}	{Rs}	Displays a dialog that lets you enter an arbitrary string.
{dlgstringS}	{RS}	The same as <b>{dlgstring}</b> except the contents of the string field are automatically selected (only applies if a default value is specified - see below).
{dlgpassword}	{Rp}	The same as <b>{dlgstring}</b> except the contents of the string field are obscured - useful for entering passwords.
{dlgchoose}	{Rc}	Displays a drop-down list that lets you choose from a list of values.
{dlgchooseS}	{RC}	The same as <b>{dlgchoose}</b> except the value list is automatically sorted.

You will almost certainly want to use the following parameters to control the behavior of these codes.

- **{dlgopen}, {dlgmulti}, {dlgsave}**

The template for these codes is `{code}<title>|<default>`. `<title>` specifies the title of the dialog box (a default title will be used if not provided), and `<default>` specifies the default filename.

*For example, {dlgopen/Select filename/dopus.txt} would set the title to **Select filename** and the default filename to **dopus.txt**.*

For **{dlgopen}** and **{dlgmulti}** you can specify a wildcard for `<default>` to set the filter type of the file dialog.

*For example, {dlgopen/\* .jpg;\*.bmp;\*.gif} would leave the dialog title as its default, and set the dialog to filter on **.jpg**, **.bmp** and **.gif** files.*

For **{dlgsave}** you can specify an additional parameter to populate the "Save as Type" dropdown list in the save dialog.

*For example, {dlgsave/Title/Default Name.txt/type=#Text Files!\*.txt!Doc Files!\*.doc}.*

The # following the **type=** causes the default "All files" items to be added to the drop-down - take out the # if you don't want this. Following that are one or more pairs of strings, separated by exclamation marks (!). The first string of each pair is the plain text string shown in the drop-down, and the second string of each pair is the actual file extension. You can also specify multiple extensions for the one type by separating them with semicolons - for example, **type=JPEG Files!\*.jpg;\*.jpeg**.

These codes also support the modifiers listed under [Codes for passing filenames](#). To use these modifiers, you must also include the `<title>` and `<default>` parameters (they can be left blank if desired).

*For example {dlgopen///noext} would strip the extension from the selected filename.*

- **{dlgfolder}**

The template for this codes is `{dlgfolder}<title>|<default>`. `<title>` specifies the title of the dialog box (a default title will be used if not provided), and `<default>` specifies the default filename.

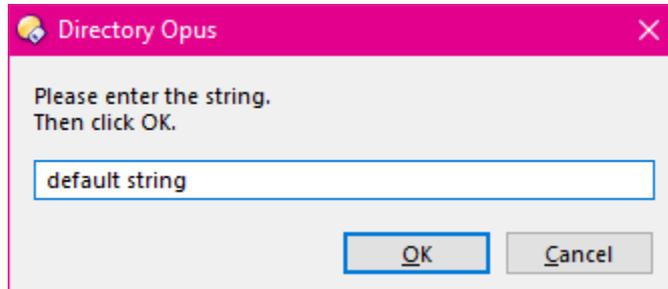
*For example, {dlgfolder/Select folder/C:\Users} would set the title to **Select folder** and the initial selected folder would be **C:\Users**.*

This code supports the modifiers listed under [Codes for passing paths](#). To use these modifiers, you must also include the `<title>` and `<default>` parameters (they can be left blank if desired). An additional modifier, `expand`, is also supported. This causes the initially selected folder to be automatically expanded.

*For example {dlgfolder//C:\Program Files/expand} would set **C:\Program Files** as the default folder and automatically expand it in the displayed dialog.*

- **{dlgstring}, {dlgstringS}, {dlgpassword}**

The template for these codes is `{code}<message>|<default>}`. `<message>` specifies the message displayed in the dialog box, and `<default>` specifies the default value of the string field. You can include line-breaks in the message text using the special code `\n` (and if you need to include a literal `\n` sequence in the text you must escape the `\` character, as in `\\n`).



For example, `{dlgstring|Please enter the string.\nThen click OK.|default string}`.

**{dlgpassword}** also accepts an optional **confirm** modifier, which must follow the `<default>` value parameter in the code. If this is specified, the dialog will display two password fields, and you must enter the same value in both fields (as a confirmation) before the OK button will be enabled.

For example, `{dlgpassword|Please enter your password twice.||confirm}`.

- **{dlgchoose}, {dlgchooseS}**

The template for these codes is `{code}<message>|<option 1>[=<value 1>][+<option 2>[=<value 2>]...]}`.

`<message>` specifies the message displayed in the dialog box. You can include line-breaks in the message text using the special code `\n` (and if you need to include a literal `\n` sequence in the text you must escape the `\` character, as in `\\n`).

`<option 1>` is the text displayed for the first option in the drop-down list, `<option 2>` is the second option, and so on. Multiple options are separated with a `+` sign.

`<value 1>` is an optional value associated with the first option. If provided, the `option` is displayed in the drop-down list, and the `value` is what is passed through on the command line. If `values` aren't provided, the `option` text is used for both.



For example, `{dlgchoose|Select encoding quality|High=320+Medium=256+Low=128}` would add the options **High**, **Medium** and **Low** to the drop-down list, and pass the values **320**, **256** and **128** (respectively) through on the command line.

## Codes for date and time

The following codes are used to pass formatted date and time strings to external programs. They are also quite often used to pass date strings to the internal commands - for example, to create a folder named after the current date, you might use a command like **CreateFolder** **{date|yyyyMMdd}**.

Code	Description
<b>{date &lt;format&gt;}</b>	Current date (local time).
<b>{dateu &lt;format&gt;}</b>	Current date (UTC).
<b>{time &lt;format&gt;}</b>	Current time (local time).
<b>{timeu &lt;format&gt;}</b>	Current time (UTC).

The *<format>* value is a string consisting of various *tokens* that are used to format the date and time strings. If no format is specified, your default system date and time format is used.

As an example, **{date|yyyy-MM-dd}** would format the date like *2016-09-22*, and **{time|HHmmss}** would format the time like *084450*.

**Locales:** Your system's language settings (locale) affect date and time formats. For example, the MMM and MMMM date codes will produce month names in your spoken language. The **tt** time code outputs your locale's versions of "AM" and "PM", which may even be completely missing in places which do not use them (e.g. France). You can override the locale and switch to the **system invariant locale** by placing an **I** (capital-i) as the first letter of the format string. The system invariant locale is similar to the North American locale and produces identical results on all machines and setups. For example, running **{date|Idd-MMM} {time|Ihh:mm tt}** on June 23rd at 10:52 PM will always output "23-Jun 10:52 PM".

The **date codes** use the following tokens - note that these tokens are case sensitive! The *ISO week* and *ISO year* tokens refer to the [ISO week date system](#).

Date token	Description
d	Day of month as a number, with no leading zero for single-digit days.
dd	Day of month as a number, with a leading zero for single-digit days.
ddd	Day of week as a three-letter abbreviation (e.g. <i>Wed</i> ).
dddd	Day of week as its full name (e.g. <i>Wednesday</i> ).
w	ISO week number, no leading zero.

ww	ISO week number, leading zero.
W	Simple week number, no leading zero.
WW	Simple week number, leading zero.
M	Month as a number, no leading zero.
MM	Month as a number, leading zero.
MMM	Month as a three-letter abbreviation (e.g. <i>Jan</i> ).
MMMM	Month as its full name (e.g. <i>January</i> ).
y	Year as last two digits, but with no leading zero for years less than 10 (e.g. <i>2009</i> -> <i>9</i> ).
yy	Year as last two digits, with a leading zero (e.g. <i>2009</i> -> <i>09</i> ).
yyyy	Year as a four digit number.
Y	ISO year as last two digits, no leading zero.
YY	ISO year as last two digits, leading zero.
YYYY	ISO year as four digit number.
gg	Period/era string - ignored if the date to be formatted does not have an associated era.

The **time codes** use the following tokens - these are also case sensitive.

Time token	Description
h	Hours with no leading zero for single-digit hours, 12 hour clock.
hh	Hours with leading zero for single-digit hours, 12 hour clock.
H	Hours with no leading zero, 24 hour clock.
HH	Hours with leading zero, 24 hour clock.
m	Minutes with no leading zero.
mm	Minutes with leading zero.
s	Seconds with no leading zero.
ss	Seconds with leading zero.
t	One-character AM/PM string (e.g. <i>A</i> or <i>P</i> ). See note about locales, above.
tt	Multiple-character AM/PM string. See note about locales, above.

### Other Codes

The **{clip}** code allow you to pass the contents of the clipboard to an external program.

The **{dpi}** code lets you use DPI related information.

The `{$}` code lets you insert the value of a variable that you have previously set with the `@set` [command modifier](#).

Long form	Short form	Description
<code>{clip}</code>	<code>{c}</code>	Passes the contents of the clipboard (only if the clipboard contains text data).
<code>{dpi}</code>	-	<p>The <code>{dpi}</code> control code lets you use DPI-sensitive values with simple commands. This can be useful if you have buttons which specify column or window sizes and you want consistent results from the same button in different DPIs.</p> <ul style="list-style-type: none"> <li>• <code>{dpi}</code> on its own will insert the current DPI. <b>96</b> at standard DPI, <b>192</b> at 200% DPI, and so on.</li> <li>• <code>{dpi %}</code> will report the insert DPI scale factor. <b>100</b> at standard DPI, <b>200</b> at 200% DPI, and so on.</li> <li>• <code>{dpi &lt;number&gt;}</code> will convert a standard 96 DPI pixel width to the current DPI. For example, if you are at 200% DPI, <code>{dpi 25}</code> will output <b>50</b>.</li> <li>• <code>{dpi /&lt;number&gt;}</code> will convert from the current DPI back to standard 96 DPI pixels. For example, if you are at 200% DPI, <code>{dpi /50}</code> will output <b>25</b>.</li> </ul>
<code>{\$&lt;variable&gt;}</code>	-	<p>Inserts the value of the named variable. This must have been previously set using the <code>@set</code> modifier.</p> <p>The <code>@set</code> modifier can be used to assign the value of another external code to a variable. For example, if you want to ask the user to enter a string with the <code>{dlgstring}</code> code, you could assign that to a variable which would then let you use that string more than once in the function:</p> <pre>@set name {dlgstring Enter new folder name} CreateFolder "\${\$name}" Go "\${\$name}" NEWTAB</pre> <p>The <b>FileType NEW</b> command automatically sets a value called <b>newfile</b> to the name of the newly created file.</p>

## Command modifier reference

The following is a list of the various [command modifiers](#) supported in toolbar buttons and hotkeys.

Modifier	Description
@admin	<p>The function requires administrator permissions under Vista and above. This modifier will result in a UAC prompt appearing (unless the Lister is already in <a href="#">administrator mode</a>), and any external programs run by this function will be elevated.</p> <p>If used by itself, this modifier affects the entire function. It can also be used to elevate a specific command rather than the entire button.</p> <p><b>@admin</b> - <i>elevate the entire function</i> <b>@admin:notepad.exe {f}</b> - <i>only elevates Notepad</i></p>
@async	<p>The command following this modifier will be run asynchronously - Opus will not wait for it to exit before running the next command in the function (or before running this command again for the next selected file).</p> <p>The default behaviour is for a function that consists of a single command to run that command asynchronously for each selected file; a function that consists of multiple commands will run each command synchronously. The default behaviour can be changed with the <b>function_default_async</b> option on the <a href="#">Miscellaneous / Advanced</a> page in Preferences, or it can be overridden on a per-command basis with this modifier.</p> <p><b>@async:notepad.exe {f}</b> - <i>runs Notepad asynchronously</i></p>
@codepage	<p>Changes the code page of an <a href="#">MS-DOS batch command</a>. If not specified the default code page is 1252 (<a href="#">Windows-1252</a>).</p>

	<p><b>@codepage:1258</b> - sets the code page to Vietnamese</p>
@confirm	<p>Displays a confirmation dialog. If the user clicks the cancel button, the function is aborted at this point.</p> <p>The template for this modifier is: <b>@confirm:</b>&lt;message&gt; &lt;positive text&gt; &lt;negative text&gt;</p> <p>&lt;message&gt; is the text shown in the dialog, &lt;positive text&gt; is the text shown on the "OK" button, and &lt;negative text&gt; is the text shown on the "Cancel" button. These three strings are all optional - if not provided, default strings will be used. If you provide &lt;positive text&gt; but not &lt;negative text&gt; then the dialog will have an "OK" button but no "Cancel" button at all.</p> <p>You can include line-breaks in the message text using the special code \n (and if you need to include a literal \n sequence in the text you must escape the \ character, as in \\n).</p> <p><b>@confirm:Really copy files?</b> - uses default text for the OK and Cancel buttons</p> <p><b>@confirm:Really proceed? Oui! Non!</b> - specifies custom text for the two buttons</p> <p><b>@confirm:All finished Acknowledged</b> - custom text for the OK button; no Cancel button at all</p>
@dirsonly	<p>The function will operate only on selected folders, ignoring any files.</p> <p><b>@dirsonly</b> - operate only on directories</p>
@disableif	<p>The button will be disabled if a command clause test is false. This is similar to the conditional testing offered by the @if modifier described below.</p> <p><b>@disableif:Set DUAL=toggle</b> - button will be disabled if the Lister is in dual-display mode</p>

	<p><b>@disableif:!Set DUAL=toggle</b> - button will be disabled if the Lister is NOT in dual-display mode</p>
<p>@disableifpath @disableifpathr</p>	<p>The button will be disabled if the current source path doesn't match the specified pattern. You can provide either an absolute path to test against, or a wildcard pattern (use <b>@disableifpath</b> for <a href="#">standard wildcards</a>, and <b>@disableifpathr</b> for <a href="#">regular expressions</a>).</p> <p><b>@disableifpath:^C:\\</b> - disable command if source path is on the C: drive</p> <p><b>@disableifpath:!*\Work\*</b> - disable unless source path is underneath a folder called <i>Work</i></p> <p>Paths may use aliases, environment variables, {apppath} codes, and so on:</p> <p><b>@disableifpath:!/desktop</b> - disable unless in the Desktop folder</p> <p><b>@disableifpath:!\%SystemRoot%\System32</b> - disable unless in <i>C:\Windows\System32</i> (on a typical system)</p> <p><b>@disableifpath:{apppath\dopus.exe}</b> - disable if in the folder where Opus is installed</p>
<p>@disablenosel</p>	<p>The command will be disabled when no files or folders are selected, or optionally when no files of a certain type are selected. This is done automatically for certain standard commands and the modifier lets you make your own commands behave in a similar way. This is similar to <b>@hidenosel</b>.</p> <p><b>@disablenosel</b> - <i>disable button when nothing is selected</i></p> <p><b>@disablenosel:files</b> - <i>disable button when no files are selected</i></p> <p><b>@disablenosel:dirs</b> - <i>disable button when no folders are selected</i></p>

For lister toolbars, you can use the **minfiles**, **maxfiles** and **numfiles** keywords to specify a minimum, maximum, or exact number of files which must be selected for the button to be enabled. Similarly, **mindirs**, **maxdirs** and **numdirs** can limit the number of selected folders.

**@disablenosel:numfiles=2** - *disable button unless exactly 2 files are selected*

**@disablenosel:maxfiles=5** - *disable button unless 5 files or fewer are selected*

**@disablenosel:mindirs=3,maxdirs=5** - *disable button unless 3, 4 or 5 folders are selected*

Using the **type** keyword you can configure a button to be disabled unless at least one file is selected whose name matches the supplied wildcard pattern. (It is not required that all selected files to match the pattern; only one.) You can also combine this with **dirs** for a button that works if a folder is selected or if a file of a certain type is selected. If you use **type**, it must be the last thing on the line, since everything after the = will be considered part of the pattern (allowing the pattern to include things which would otherwise be confused with other keywords and parameters).

**@disablenosel:type=\*.jpg** - *disable button when no files ending with ".jpg" are selected*

**@disablenosel:type=old\*** - *disable button when no files beginning with "old" are selected*

**@disablenosel:dirs,type=\*.png** - *disable unless any folders, or any files ending with ".png", are selected*

You can also use a **!** character to negate the test. It must be the first thing after the **:**. For example,

**@disablenosel:!type=\*.jpg** - *disable button if files ending with ".jpg" are selected*

@externalonly

Ignores any Opus [internal commands](#) specified in the function, and treats all commands as external. This lets you configure an MS-DOS

	<p>batch function using commands like <b>copy</b> which would normally be overridden by the internal command set.</p> <p><b>@externalonly</b>      - <i>external commands only</i></p>
@filesfromdroponly	<p>Accepts files and folders only via drag-and-drop. A button with this modifier will ignore any selected files or folders in the Lister - only files dropped onto the button will be used by a command within it.</p> <p><b>@filesfromdroponly</b>      - <i>accept files only via drag-and-drop</i></p>
@filesonly	<p>The function will operate only on selected files, ignoring any folders.</p> <p><b>@filesonly</b>      - <i>operate only on files</i></p>
@firstfileonly	<p>The function will operate on only the first selected file, irrespective of how many items are selected or dropped onto the button.</p> <p><b>@firstfileonly</b>      - <i>operate on only the first selected file</i></p>
@functype	<p>This directive is only used within an <a href="#">embedded function</a>. It lets you specify the type of the embedded function (if not specified, the embedded function is assumed to be a standard function - Opus or external command). For example, the following command would open a new Lister with an embedded script function that runs in the context of the new Lister:</p> <pre> Go NEW [ @functype:script @script:vbscript ' script function here ]</pre>

	<p><b>@functype:script</b> - the embedded function is a script</p> <p><b>@functype:msdos</b> - the embedded function is an MS-DOS batch function</p>
@hideif	<p>The button will be hidden if a command clause test is false. This is similar to the conditional testing offered by the <b>@if</b> modifier described below.</p> <p><b>@hideif:Set DUAL=toggle</b> - button will be hidden if the Lister is in dual-display mode</p> <p><b>@hideif:!Set DUAL=toggle</b> - button will be hidden if the Lister is NOT in dual-display mode</p>
@hideifpath @hideifpathr	<p>The button will be hidden if the current source path doesn't match the specified pattern. You can provide either an absolute path to test against, or a wildcard pattern (use <b>@hideifpath</b> for <a href="#">standard wildcards</a>, and <b>@hideifpathr</b> for <a href="#">regular expressions</a>).</p> <p><b>@hideifpath:^C:\\</b> - hide command if source path is on the C: drive</p> <p><b>@hideifpath:!*\\Work\\*</b> - hide unless source path is underneath a folder called <i>Work</i></p> <p>Paths may use aliases, environment variables, {apppath} codes, and so on:</p> <p><b>@hideifpath:!/desktop</b> - hide unless in the Desktop folder</p> <p><b>@hideifpath:!%SystemRoot%\System32</b> - hide unless in <i>C:\Windows\System32</i> (on a typical system)</p> <p><b>@hideifpath:{apppath dopus.exe}</b> - hide if in the folder where Opus is installed</p>

@hidenosel

The button will be hidden when no files or folders are selected, or optionally when no files of a certain type are selected. This is similar to **@disablenosel**.

**@hidenosel** - *hide button when nothing is selected*

**@hidenosel:files** - *hide button when no files are selected*

**@hidenosel:dirs** - *hide button when no folders are selected*

For lister toolbars, you can use the **minfiles**, **maxfiles** and **numfiles** keywords to specify a minimum, maximum, or exact number of files which must be selected for the button to be visible. Similarly, **mindirs**, **maxdirs** and **numdirs** can limit the number of selected folders.

**@hidenosel:numfiles=2** - *hide button unless exactly 2 files are selected*

**@hidenosel:maxfiles=5** - *hide button unless 5 files or fewer are selected*

**@hidenosel:mindirs=3,maxdirs=5** - *hide button unless 3, 4 or 5 folders are selected*

Using the **type** keyword you can configure a button to be hidden unless at least one file is selected whose name matches the supplied wildcard pattern. (It is not required that all selected files to match the pattern; only one.) You can also combine this with **dirs** for a button that is visible if a folder is selected or if a file of a certain type is selected. If you use **type**, it must be the last thing on the line, since everything after the = will be considered part of the pattern (allowing the pattern to include things which would otherwise be confused with other keywords and parameters).

**@hidenosel:type=\*.jpg** - *hide button when no files ending with ".jpg" are selected*

**@hidenosel:type=old\*** - *hide button when no files beginning with "old" are selected*

**@hidenosel:dirs,type=\*.png** - *hide unless any folders, or any files ending with ".png", are selected*

You can also use a **!** character to negate the test. It must be the first thing after the **:**. For example,

**@hidenosel:!type=\*.jpg** - *hide button if files ending with ".jpg" are selected*

@icon  
@icon!  
@iconp

The **@icon** directive lets you create buttons that dynamically change their icon based on the test of a command clause, typically via the **Set** command (similar to **@if** and **@ifset**). The default *View Mode Cycle* button uses this to change its icon to reflect the current view mode.

```
Set VIEW=Cycle
@icon:largeicons,Set VIEW=LargeIcons
@icon:smallicons,Set VIEW=SmallIcons
@icon:listmode,Set VIEW=List
@icon:powermode,Set VIEW=Power
@icon:thumbnails,Set VIEW=Thumbnails
@icon:tile,Set VIEW=Tiles
@icon:detailmode,Set VIEW=Details
```

Each **@icon** directive in the button specifies the icon to use, and the command directive to test for. For example, if **Set VIEW=LargeIcons** tests as true, the icon called **largeicons** will be displayed on the button. If none of the **@icon** tests match then the icon set in the button itself will be used as the default image.

In the special case of a three-button button, **@iconp:** and **@icon!:** directives can also be used in any of the child buttons to change the icon of the parent. **@iconp:** used in one of the child buttons will set the icon of the parent (but do nothing to the child button). **icon!:** used in one of the child buttons will set the icon of the parent as well as of that child button.

If you are testing based on the **Set** command, the command name is technically optional and can be omitted. For example, **VIEW=LargeIcons** on its own is the same as **Set VIEW=LargeIcons**. This ability is there to avoid breaking old buttons and, with new buttons, we recommend you include the command name in all cases to avoid problems in the future.

In the command editor, you can **Ctrl**+click the **@icon:** string to select the icon directly.

**@icon** can use the **RECYCLEBINEMPTY** argument to test if the recycle bin is empty or not. For example,

```
@icon:c:\icons\empty.png,RECYCLEBINEM  
PTY
```

Toolbar buttons that use **@icon** with **RECYCLEBINEMPTY** will refresh themselves automatically when the recycle bin state changes (this lets you have a button whose icon reflects the state of the recycle bin).

**@if**

**@ifset**

Allows simple conditional behaviour based on tests for various commands.

**@if** is a general test which works with any command, and **@ifset** is a specific test which only works with the [Set](#) command conditions. For example, **@ifset:DUAL=on** is equivalent to **@if:Set DUAL=on**.

The test **@if** actually performs is to evaluate whether a toolbar button with the specified command would appear highlighted or not. For example, if the Lister is in dual-display mode, a button with **Set DUAL=toggle** as the command would appear highlighted. Therefore, the **@if:Set DUAL=toggle** modifier would evaluate to true.

You can also evaluate whether a toolbar button would be enabled or not, using **@if:enabled**. For example, [Navigation Lock](#) is only enabled when a Lister is in dual-display mode. Therefore, the modifier **@if:enabled Set NAVLOCK=Toggle** would evaluate to true when the Lister was in dual-display mode.

You can also test if a variable has been set by the **@set** directive. Additionally, a button on the [File Display toolbar](#) can test if it's tied to the left or right file display.

For example, you could configure a button to read a folder into the right file display in a dual-display Lister, or into the current file display otherwise.

```
@if:Set DUAL=on           // if DUAL
mode is on
Set FOCUS=right         // give focus to
the right file display
Go /mypictures OPENINRIGHT // read
directory into that file display
@if:else               // otherwise
Go /mypictures         // read directory
into current file display
@if:common           // common
commands (always executed)
Set VIEW=thumbnails   // set view to
thumbnails mode
```

As another example, **@ifset** can use the **RECYCLEBINEMPTY** argument to test if the recycle bin is empty or not:

```
@ifset:RECYCLEBINEMPTY
@confirm The recycle bin is empty!
@ifset:else
Delete EMPTYRECYCLE
```

Conditional testing can also be used on buttons in the [standalone viewer](#). The following function will toggle the display between 100% zoom and *Grow To Page* modes, by using **@if** to test the current zoom mode:

```
@if:Show VIEWERCMD=zoom, reset
Show VIEWERCMD=zoom, grow
@if:else
Show VIEWERCMD=zoom, reset
```

The possible forms of this modifier are:

**@if:<command line>** - test if a specific command condition is true

**@ifset:<Set argument>** - test for a specific Set command condition

**@if:enabled <command>** - test if a command button would be enabled (rather than highlighted)

**@if:\$foo** - test if a variable called "foo"

	<p><i>has been set</i></p> <p><b>@if:\$glob:bar</b> - test if a global variable called "bar" has been set</p> <p><b>@if:SIDE=left</b> - test if the toolbar is tied to the left file display</p> <p><b>@if:SIDE=right</b> - test if the toolbar is tied to the right file display</p> <p><b>@if:else</b> - an "else" clause that is executed if nothing else matches</p> <p><b>@if:common</b> - common instructions that are always executed</p> <p>You can negate the condition by prefixing it with a ! character. For example, all four of these are equivalent:</p> <pre>@if:!Set DUAL=on @if:Set DUAL=off @ifset:!DUAL=on @ifset:DUAL=off</pre>
<p>@ifexists</p>	<p>Allows simple conditional behaviour based on whether the specified drive or path exists.</p> <p><b>@ifexists:&lt;drive or path&gt;</b> - test if drive or path exists</p> <p><b>@ifexists:!<i>&lt;drive or path&gt;</i></b> - test if drive or path does not exist</p> <p><b>@ifexists:else</b> - an "else" clause that is executed if the path doesn't exist</p> <p><b>@ifexists:common</b> - common instructions that are always executed</p>
<p>@ifpath @ifpathr</p>	<p>Allows simple conditional behaviour based on the current source path. You can provide either an absolute path to test against, or a wildcard pattern (use <b>@ifpath</b> for <a href="#">standard wildcards</a>, and <b>@ifpathr</b> for <a href="#">regular expressions</a>).</p> <p>For example, you could use this to open different programs when double-clicking on an image file, depending on where the file is stored. If the following command was added to the <a href="#">Left double-click event</a> for the <b>Images file type group</b>, pictures located on the network will be opened in the Opus viewer when they are double-clicked, whereas files located on a local drive would open in Photoshop.</p> <pre><b>@ifpath:\\*</b> // if path begins with \\ <b>Show</b> // invoke the</pre>

*Opus viewer*  
**@ifpath:else** // otherwise  
**Photoshop.exe {f}** // open in  
*Photoshop (the full path would*  
*generally*  
*need to be specified - this is just*  
*an example)*

The possible forms of this modifier are:

**@ifpath:**<path or wildcard> - test for a path (using standard pattern matching)  
**@ifpathr:**<regular expression> - test for a path (using regular expressions)  
**@ifpath:else** - an "else" clause that is executed if nothing else matches  
**@ifpath:common** - common instructions that are always executed

You can negate the condition by prefixing it with a ! character.

Examples:

**@ifpath:C:\Program Files** - Tests you are in C:\Program Files  
**@ifpath:!C:\Program Files** - Tests you are anywhere but C:\Program Files  
**@ifpath:C:\Program Files\\*** - Tests you're in a folder below C:\Program Files (does not include Program Files itself)  
**@ifpath:!C:\Program Files\\*** - Tests you are not in a folder below C:\Program Files

@ifrunning

Allows simple conditional behaviour based on whether a specified process is currently running. You can use wildcards or (by prefixing the pattern with **regex:**) regular expressions.

**@ifrunning:notepad.exe** - test if Notepad is running  
**@ifrunning:!note\*** - test if a process starting with "note" is not running  
**@ifrunning:else** - an "else" clause that is executed if the process doesn't exist  
**@ifrunning:common** - common instructions that are always executed

@keydown

Allows simple conditional behaviour based on whether various qualifier keys are held down when the function is run.

The qualifiers to test for are specified using one or more keywords (**shift**, **ctrl** and **alt**) which represent the **Shift**, **Control** and **Alt** keys. For example, you could configure a button to select all files, then copy, move or create shortcuts to them, depending on which keys were held down.

```
Select ALL // select all items (always
executed)
@keydown:none // if no qualifiers are down
Copy // copy files to destination
@keydown:shift // else if shift key is down
Copy MOVE // move files to destination
@keydown:ctrlshift // else if control and shift
keys are down
Copy MAKELINK // make shortcuts in
destination
```

The possible forms of this modifier are:

```
@keydown:<qualifiers> - test for a qualifier key or
combination of keys
@keydown:!<qualifiers> - test for a qualifier key or
combination of keys NOT being down
@keydown:any - tests if any qualifier keys
are held down
@keydown:none - instructions that are
executed if no qualifiers are held down
@keydown:common - common instructions that
are always executed
```

@leavedoswindowopen

The console window opened for an [MS-DOS batch function](#) will remain open when the function has completed. Without this modifier the window will close when the function finishes, which may make it hard to read the output of any command-line programs.

Leaving the DOS window opens relies on passing a particular argument (**/K**) to the system command interpreter, **cmd.exe**. If you have modified the default command processor on your system by setting the **%ComSpec%** environment variable, Opus will not pass **/K** as there is no way for it to know if this argument would be understood by the new command processor. Instead, you can set the **%ComSpecLeaveOpenArg%** environment variable, and this will be

	<p>used instead of <b>%ComSpec%</b> whenever <b>@leavedoswindowopen</b> is used.</p> <p><b>@leavedoswindowopen</b>     - <i>leave DOS prompt open when function finishes</i></p>
@nocall	<p>An <a href="#">MS-DOS batch function</a> function that invokes an external <b>.bat</b> file will run it directly, rather than using <i>call</i> semantics. This means that control will not return to the parent function once the external <b>.bat</b> file has finished. The default behaviour is to use the DOS <b>call</b> instruction which returns control to the parent function at the end.</p> <p><b>@nocall:</b><i>&lt;batch file&gt;</i>     - <i>invoke external batch file, do not return control to this function</i></p>
@nodeselect	<p>Files and folders will remain selected at the end of the function. This lets you override the <b>Deselect files used in functions</b> option on the <a href="#">File Operations / Options</a> page in Preferences, on a per-function basis.</p> <p><b>@nodeselect</b>     - <i>do not deselect items used by this function</i></p>
@noexpandenv	<p>Prevent the expansion of any environment variables in the function. Normally environment variables like <b>%USERPROFILE%</b> are expanded during the function parsing phase - this modifier causes variable names to be left intact.</p> <p><b>@noexpandenv</b>     - <i>do not expand environment variables</i></p>
@nofilenamequoting	<p>Disables the automatic quoting that Opus performs when <a href="#">external codes</a> like <b>{filepath}</b> are used to pass the name of a file that contains spaces. By default Opus will wrap names with embedded spaces in quotes, but occasionally you may want to disable this - some external programs may not need or understand quotes on their command line, and you may also want to provide explicit quotes in a function in case</p>

	<p>the automatic quoting gets confused by complicated command structures.</p> <p><b>@nofilenamequoting</b>      - <i>do not automatically quote file names and paths</i></p>
@nolocalizefiles	<p>Prevents Opus from automatically downloading or extracting non-filesystem files when passing their paths to external programs. For example, you may want to use <b>{filepath}</b> to pass the <b>ftp://</b> path of a file on a remote FTP server to an external program. By default Opus will download the file to a temporary file, and pass the name of the temporary file to the program - with this modifier, Opus will instead pass the original file path.</p> <p><b>@nolocalizefiles</b>      - <i>do not automatically localize (download) remote files</i></p>
@noprogess	<p>Disables the automatic display of a progress indicator for this function. You should use this with care as without a progress indicator there is no way to abort the command or monitor its progress. Normally only script functions should use this when they want to provide and control their own progress indicator.</p> <p><b>@noprogess</b>      - <i>do not display an automatic progress dialog for this function</i></p>
@norunbatch	<p>When an <a href="#">MS-DOS batch function</a> combines internal and external commands, the default behaviour is for Opus to split the function at every internal command, and execute all the external commands that came before it. For example, imagine the following (rather pointless) function:</p> <pre> <b>echo one</b> <b>Help ABOUT</b> <b>echo two</b> <b>@leavedoswindowopen</b> </pre> <p>The default behaviour of this function would be to open a DOS window and print the string "one", then display the Opus <b>About</b></p>

	<p>dialog, and then open another DOS window and print the string "two". If, however, the function were changed as follows:</p> <pre><b>@norunbatch</b> <b>echo one</b> <b>Help ABOUT</b> <b>echo two</b> <b>@leavedoswindowopen</b></pre> <p>The new behaviour would be to first display the <b>About</b> dialog, and then open a single DOS window and print both the strings "one" and "two". The <b>@norunbatch</b> modifier causes all Opus internal commands to be executed first, followed by all external commands.</p> <pre><b>@norunbatch</b>           - <i>do not split batch functions</i>                         <i>because of internal commands</i></pre>
<p><b>@resovelinks</b></p>	<p>Shortcuts or links passed to commands in this function will be resolved to their targets. Without this modifier, a selected <b>.lnk</b> file would be passed as-is.</p> <pre><b>@resovelinks</b>           - <i>resolve shortcut targets</i>                         <i>when passing filenames to commands</i></pre>
<p><b>@runbatch</b></p>	<p>Force the execution of an MS-DOS batch function at a certain point. The default behaviour is for such functions to be executed at the very end of the function - using this modifier you can force a break in the function at any line. For example:</p> <pre><b>echo one</b> <b>@runbatch</b> <b>echo two</b> <b>@leavedoswindowopen</b></pre> <p>This will cause two separate DOS windows to open, one displaying the text "one" and the other displaying "two". Without the <b>@runbatch</b> directive, the two commands would have been executed in the one window.</p>

	<p><b>@runbatch</b>                    <i>- force a DOS batch function to execute at a certain point</i></p>
@runmode	<p>Modifies the "run" state of external programs launched by the function - that is, how its main window should appear. This is equivalent to the <b>Run</b> drop-down on the <a href="#">simple command editor</a>. Not all programs will support this setting, so you will need to use trial and error to some extent. You should only set the mode to <b>hide</b> if you are sure of what you are doing - it is most useful for hiding the otherwise brief flash of a DOS window when running a DOS program.</p> <p><b>@runmode:min</b>                    <i>- minimize the program's main window</i>  <b>@runmode:max</b>                    <i>- maximize the program's main window</i>  <b>@runmode:hide</b>                    <i>- hide the program's main window</i></p>
@runonce	<p>Forces the command following the modifier to only be run once per function, instead of once per file.</p> <p><b>@runonce:&lt;command&gt;</b>    <i>- the specified command will only be run once</i></p>
@script	<p>Specifies the scripting language used for a script. This is mainly used for <a href="#">embedded rename scripts</a>, because otherwise the script language is selected via a drop-down at the top of the script editor. The <b>@script</b> modifier must be followed by a keyword that specifies the scripting language (or in the case of <a href="#">script add-ins</a>, the file extension of the add-in itself specifies the language).</p> <p>For example,</p> <p><b>Rename PATTERN * TO *</b>  <b>@script:vbscript</b>  <b>Function ....</b></p>

	<p><b>@script:</b><i>&lt;language&gt;</i> - specifies scripting language</p>
<p>@set</p>	<p>Sets the value of a named variable that can be used by the remainder of the commands in the function. You can pass the value of a variable to internal and external commands using the <b>{}</b> <a href="#">control code</a>. Variables are most useful in the situation where the value is defined by a <b>{dlgstring}</b> code. For example,</p> <pre><b>@set dir={dlgstring Enter new folder name to copy files to} CreateFolder ".\{\$dir}" Copy TO ".\{\$dir}"</b></pre> <p>Normally, variables do not persist from one invocation of the function to another, and you can not refer to variables set in one function from another. However this can be accomplished by prefixing the variable name with a special scope marker that determines which scope the variable will live in.</p> <ul style="list-style-type: none"> <li><b>src:</b><i>&lt;name&gt;</i> - variable will be scoped to the source folder tab</li> <li><b>dst:</b><i>&lt;name&gt;</i> - variable will be scoped to the destination folder tab</li> <li><b>left:</b><i>&lt;name&gt;</i> - variable will be scoped to the left folder tab</li> <li><b>right:</b><i>&lt;name&gt;</i> - variable will be scoped to the right folder tab</li> <li><b>tab:</b><i>&lt;name&gt;</i> - variable will be scoped to the active folder tab (see below)</li> <li><b>lst:</b><i>&lt;name&gt;</i> - variable will be scoped to the Lister (the whole window)</li> <li><b>glob:</b><i>&lt;name&gt;</i> - variable will be scoped globally</li> </ul> <p>The <b>src:</b> and <b>tab:</b> scopes usually point to the same folder tab within commands. (The <b>tab:</b> scope was introduced so that <a href="#">status bar codes which use variables</a> can refer to the side that the status bar is on, without worrying if it is the source, destination, left or right. It is supported in commands for consistency.)</p>

	<p>You can also mark variables to be saved on disk - their values will be remembered from one Opus session to the next. To accomplish this, simply add an exclamation mark (!) to the scope marker. For example, <b>glob!:&lt;name&gt;</b> refers to a variable with global scope that will be saved on disk.</p> <p><b>@set &lt;name&gt;=&lt;value&gt;</b> - sets the named variable to the specified value</p> <p><b>@set&lt;name&gt;</b> - deletes the named variable</p> <p>If you have buttons or status bar codes which react to variable changes, you may need to run the special <b>@toggle:update</b> command to make them update their appearances after making changes to variables. See the <b>@toggle</b> documentation, below.</p>
<p>@sync</p>	<p>The command following this modifier will be run synchronously - Opus will wait for it to exit before running the next command in the function (or before running this command again for the next selected file).</p> <p>The default behaviour is for a function that consists of a single command to run that command asynchronously for each selected file; a function that consists of multiple commands will run each command synchronously. The default behaviour can be changed with the <b>function_default_async</b> option on the <a href="#">Miscellaneous / Advanced</a> page in Preferences, or it can be overridden on a per-command basis with this modifier.</p> <p><b>@sync:notepad.exe {f}</b> - runs Notepad synchronously</p>
<p>@toggle</p>	<p>For buttons that indicate or change state (e.g. <b>Set VIEW=details</b>), this modifier lets you change when the button appears highlighted.</p> <p>Depending on where the button is displayed, the highlight may appear as a checkmark (e.g. in a text-only menu), or as the icon or label</p>

background being drawn in a different color or style (similar to when the button is pushed).

Taking the **Set VIEW=details** command as an example, ordinarily the button would appear highlighted when the file display was in details mode, and not highlighted when it was in any other mode. The **@toggle** modifier can change this:

**@toggle:invert**               - *inverts the usual highlight state of the toolbar button*  
**@toggle:disable**           - *prevents the button from ever appearing highlighted*

The **@toggle** modifier can also control when a button appears highlighted by testing a command, similar to the way **@if** and **@ifset** test commands to decide what to run. You can use this to highlight a button based on a command which is different to, or in addition to, the commands which the button actually runs.

For example, this button with no **@toggle** line will switch you to *Details* mode when clicked, and will appear highlighted when you are in *Details* mode as well:

```
Set VIEW=Details
```

When **@toggle** is not given, the highlighting (if any) is based on the first command in the button.

Using **@toggle** you can do things like run one **Set** command while testing for another. As a simple, somewhat contrived example, the button below switches to *Details* mode when clicked but appears highlighted when in *Power* mode:

```
Set VIEW=Details
@toggle:if Set VIEW=Power
```

You can make the **@toggle** test in addition to the first command, rather than instead of it, by putting an "&" character after the "if":

```
Set VIEW=Details
Set COLUMNSADD=thumbnail(0,96)
@toggle:if&Set
COLUMNSTOGGLE=thumbnail
```

Without the **@toggle** line, the button above would be highlighted when the view mode was *Details* (due to the command) and the highlight would not be affected by the *Thumbnail* column (highlighting does not consider the second command). With the **@toggle** line added, the button will only be highlighted if the view mode is set to *Details* and the *Thumbnail* column is present.

You can omit the "Set" command name from the **@toggle** line; it will assume you mean the **Set** command if no name is specified. However, this allowance is to avoid breaking old buttons and we recommend you always specify the command when making new ones.

You can test for multiple commands at once by separating them with semi-colons. For example, you could leave out the ampersand and still test for both *Details* mode and the *Thumbnail* column using the following directive (and this will work regardless of any other commands in the button, or even as the only line in the button):

```
@toggle:if Set VIEW=Details;Set
COLUMNSTOGGLE=thumbnail
```

You can also negate the result of a test with the exclamation mark in

front of the **Set** clause. For example, to make the button appear checked in any mode other than Details:

```
@toggle:if !Set VIEW=Details
```

The **@toggle** directive can also test for the existence of a variable (one set via the **@set** directive). For this to work, the directive must be the very first line of the button, and the variable you are testing must have tab, Lister or global scope. You can use this to create your own custom toggle buttons that keep track of their state using scoped variables. For example:

```
@toggle:if $glob:TestVar
@if:$glob:TestVar
@set glob:TestVar
@if:else
@set glob:TestVar=on
```

This button tests for the existence of a global variable called *TestVar*. If the variable exists, the button will appear highlighted. Clicking the button evaluates the **@ifset:** directives and will reverse the state of the variable: if the variable is set, it will be deleted; otherwise, it will be set. The end result is a button that performs no function except to toggle a variable and its visible state each time it is clicked. (This can be useful if the variable controls other things, such as the behaviors of other commands or scripts, or the visibility of [status bar elements](#).)

A button that begins with **@toggle:if** in this way will automatically update its appearance (to reflect a change in the state of the variable) whenever **@set** is used within it. If you use **@set** in a separate button or hotkey to change the variable, you need to force an update of the toggle button's appearance using the special command **@toggle:update**.

For example, if you have one button which highlights based on a variable but does not change that variable:

```
@toggle:if $glob:TestVar
```

And if you have another button which changes that variable but does not highlight based on it, then you will need to add the `@toggle:update` line to ensure the first button's highlight is updated when you click the second button:

```
@if:$glob:TestVar  
@set glob:TestVar  
@if:else  
@set glob:TestVar=on  
@if:common  
@toggle:update
```

The **@toggle:update** command will also force all **status bars** to refresh, in case they are [using variables](#) to display or hide information.

@useactivelister

The function will operate on the active Lister rather than the current source Lister. Normally these will be the same thing, but this could be useful for a hotkey that you want to have operate on whatever Lister window is currently active, without having to make sure that Lister is set to source first.

**@useactivelister**      - *use active Lister instead of source lister*

# Scripting Reference

The Directory Opus scripting interface is built around ActiveX. The advantage of this is that Opus scripting is *language neutral* - you can use any ActiveX scripting language you have installed on your machine. Windows ships with **VBScript** and **JScript** interpreters, and you can add additional languages from third-party providers (e.g. Perlscript, Python, etc).

The scripting interface is presented as a number of [Objects](#) and [Events](#).

- *Objects* are created by a script, or provided to a script by Opus, and have defined methods and properties that allow you to interact with Opus.
- *Events* are functions provided by a script, that Opus will invoke under certain conditions. For example, the *OnBeforeFolderChange* event is fired before the folder is changed in a Lister.

## Scripting Objects

Scripts can interact with Opus using a number of different objects. For example, the **Lister** object represents a physical [Lister](#) window, providing information about that Lister to the script and letting the script modify or control that Lister.

An object has two main types of interface that you can call on:

- *Properties* let the script query (and sometimes set) a simple value. For example, the **Lister** object has a **title** property that lets the script retrieve the Lister window's title string.
- *Methods* are functions an object provides that the script can invoke to perform a task. For example, the **DOpus** object has an **Output** method that lets the script output a text string to the script log.

The main distinction between properties and methods is that methods can (but don't always) accept arguments that modify their behavior, and don't have to (but often do) return a result.

Properties and methods that return a result can return a number of different types of variables, and methods often take one or more parameters that can also be a number of types. ActiveX scripting languages are generally *typeless* - that is, a variable's type does not have to be defined in advance, and conversion from one type to another is often (but not always) automatic. Opus scripting mainly uses variables of the following types:

- *Int* - a whole number
- *Currency* - this is a standard variant type, and is used by Opus in just a couple of cases that require numbers larger than an *Int* can hold (unfortunately ActiveX scripting does not support a 64 bit integer type).
- *String* - a text string
- *Bool* - a Boolean (either **True** or **False**)
- *Date* - a date/time value
- *Collection* - a collection of multiple objects of (generally) the same type. Collections can be easily enumerated in some languages (e.g. in VBScript, using the *For Each* construct). Collections are only returned by Opus - unlike an array (or a Vector object), they are never created or modified directly (although some script methods can be used to modify them in certain cases). For example, the **Tab** object has a property called **selected** that represents all currently selected items in that tab - it is a *collection* of items.
- *Object* - an Opus script object with defined methods and properties (one of the object types listed below). Some objects can be both an object and a collection - that is, they have methods and properties, but can also be enumerated like a collection. Some objects also have a *default value* - that is, simply using the object's name without any method or property will return a value of its own. For example, the **Metadata** object's default value is a string indicating the primary type of metadata available. You can also refer to an object's default value using the special **def\_value** property name.
- *Variant* - a variable of any type (this is used with a few objects - for example, a **Var** object can store a variant)

There are two objects that are provided to scripts as global variables when they are invoked by Opus:

- [DOpus](#): This object is available to *all* scripts. It provides various helper methods, and collections that let you access things like Listers and toolbars.
- [Script](#): This object is provided to [script add-ins](#) when their various event handlers are invoked. It provides information relating to the script itself.

There are also a number of objects that Opus provides as parameters to methods within a script. For example, when a [script function](#) is invoked (e.g. when the button is clicked), Opus calls its **OnClick** method, passing it a **ClickData** object.

- [AboutData](#): This object is provided to the [OnAboutScript](#) method, which is called when the user clicks the *About* button for a script in the [Toolbars / Scripts](#) Preferences page.
- [ActivateListerData](#): This object is provided to the [OnActivateLister](#) method, which is called whenever a Lister window is activated or deactivated.
- [ActivateTabData](#): This object is provided to the [OnActivateTab](#) method, which is called whenever a tab is activated.

- [AddCmdData](#): This object is provided to the [OnAddCommands](#) method, which allows a script to [add new internal commands](#).
- [AddColData](#): This object is provided to the [OnAddColumns](#) method, which allows a script to [add new information columns](#).
- [AfterFolderChangeData](#): This object is provided to the [OnAfterFolderChange](#) method, which is called after a new folder has been read.
- [BeforeFolderChangeData](#): This object is provided to the [OnBeforeFolderChange](#) method, which is called before a new folder is read.
- [ClickData](#): This object is provided to the [OnClick](#) method, which is called whenever a [script function](#) is invoked (e.g. when the button is clicked or hotkey pressed).
- [CloseListerData](#): This object is provided to the [OnCloseLister](#) method, which is called before a Lister closes.
- [CloseTabData](#): This object is provided to the [OnCloseTab](#) method, which is called before a tab closes.
- [ConfigChangeData](#): This object is provided to the [OnScriptConfigChange](#) method, which notifies a script when the user edits the script's configuration.
- [DisplayModeChangeData](#): This object is provided to the [OnDisplayModeChange](#) method, which is called when the display mode changes in a tab.
- [DoubleClickData](#): This object is provided to the [OnDoubleClick](#) method, which is called when a file or folder is double-clicked.
- [FlatViewChangeData](#): This object is provided to the [OnFlatViewChange](#) method, which is called when the Flat View mode changes in a tab.
- [GetCopyQueueNameData](#): This object is provided to the [OnGetCopyQueueName](#) event, which is called whenever a copy operation begins that uses automatically-managed copy queues.
- [GetCustomFieldData](#): This object is provided to the [OnGetCustomFields](#) event, which lets a [rename script](#) add its own fields to the *Rename* dialog.
- [GetHelpContentData](#): This object is provided to the [OnGetHelpContent](#) event, which lets a script add its own content to the Opus F1 help.
- [GetNewNameData](#): This object is provided to the [OnGetNewName](#) method, which is one of the supported methods a [rename script](#) can provide.
- [ListerResizeData](#): This object is provided to the [OnListerResize](#) event, which is called whenever a Lister window is resized.
- [ListerUIChangeData](#): This object is provided to the [OnListerUIChange](#) method, which is called when various user interface elements (tree, viewer, etc) are open or closed in a Lister.
- [OpenListerData](#): This object is provided to the [OnOpenLister](#) method, which is called when a new Lister is opened.
- [OpenTabData](#): This object is provided to the [OnOpenTab](#) method, which is called when a new tab is opened.
- [ScriptColumnData](#): This object is provided to the [script-specified method](#) used to add new columns to Opus.
- [ScriptCommandData](#): This object is provided to the [script-specified method](#) used to [add new internal commands](#) to Opus.
- [ScriptInitData](#): This object is provided to the [OnInit](#) method, which is called once to initialize each script in the *Script Addins* folder.
- [ShutdownData](#): This object is provided to the [OnShutdown](#) method, which is called before Opus shuts down.
- [SourceDestData](#): This object is provided to the [OnSourceDestChange](#) method, which is called when the source or destination state of a tab changes.

- [StartupData](#): This object is provided to the [OnStartup](#) method, which is called when Opus starts up.
- [StyleSelectedData](#): This object is provided to the [OnStyleSelected](#) method, which is called when a new [style](#) is chosen in a Lister.
- [TabClickData](#): This object is provided to the [OnTabClick](#) event, which is called whenever a tab is clicked with a qualifier key held down.
- [ViewerEventData](#): This object is provided to the [OnViewerEvent](#) event, which is called whenever certain events occur in a standalone [image viewer](#).

The remaining objects are all obtained using methods or properties provided by the objects listed above. For example, a **Lister** object is obtained using the **DOpus.listeners** collection property, and a **Vector** object is obtained using the **DOpusFactory.Vector** method.

- [Alias](#): This object represents a [folder alias](#), and is retrieved using the **Aliases** object.
- [Aliases](#): This object is a collection of all defined folder aliases. It is retrieved using the **DOpus.aliases** collection property.
- [Args](#): This object represents the arguments supplied on the command line for [script-defined internal commands](#). It is retrieved from the **ScriptCommandData.Func.args** property.
- [AudioMeta](#): This object provides metadata properties relating to audio files. It is obtained from the **Metadata** object.
- [AudioCoverArt](#): This object provides access to an audio file's embedded cover art. It is obtained from the **AudioMeta.coverart** property.
- [Blob](#): This object provides a simple interface for dealing with binary data. It is obtained from the **DOpusFactory.Blob** method and also returned by the **AudioCoverArt.data** property.
- [BusyIndicator](#): A **BusyIndicator** object lets you control the breadcrumbs bar busy indicator from your script.
- [Column](#): This object represents a column that has been added to the display in a tab. A collection of columns can be obtained from the **Format** object.
- [Command](#): This object is used to run Opus commands. It is obtained from the **ScriptCommandData.func** or **ClickData.func** properties, and can also be created by the **DOpusFactory.Command** method.
- [Control](#): The **Control** object represents a control on a [script dialog](#); it lets you read and modify a control's value (and contents).
- [CustomFieldData](#): The **CustomFieldData** object is provided to a [rename script](#) via the [GetNewNameData.custom](#) property. It provides access to the value of any [custom fields](#) that your script added to the *Rename* dialog.
- [Date](#): This object is provided to make it easier to deal with variables representing dates. It is obtained from the **DOpusFactory.Date** method as well as various properties in other objects.
- [Dialog](#): This object is used to display dialogs or popup menus. It is obtained from the **Func.Dlg**, **Command.Dlg** or **DOpus.Dlg** methods.
- [DialogListColumn](#): The **DialogListColumn** object represents a column in a *Details* mode *list view* control in a [script dialog](#). It's obtained by enumerating the [DialogListColumns](#) object.
- [DialogListColumns](#): The **DialogListColumns** object lets you query or modify the columns in a *Details* mode *list view* control in a [script dialog](#). Use the [Control.columns](#) property to obtain a **DialogListColumns** object.

- [DialogListItem](#): The **DialogListItem** object represents an item in a *combo box* or *list box* control in a [script dialog](#). It's returned by the [Control.GetItemAt](#) and [Control.GetItemByName](#) methods.
- [DialogOption](#): This object is used in conjunction with the **Dialog** object. It lets you specify a checkbox option that is added to the dialog.
- [Dock](#): This object represents a floating toolbar. The **Toolbar** object provides a collection that represents all instances of that toolbar that are currently floating.
- [DocMeta](#): This object provides metadata properties relating to document files. It is obtained from the **Metadata** object.
- [DOpusFactory](#): This object is a helper object that you can use to create various other objects like **Map** and **Vector**. It is obtained from the **DOpus.Create** method.
- [DPI](#): The **DPI** object is a helper object that provides a number of methods and properties relating to the system DPI setting. It's returned via the [DOpus.DPI](#) property.
- [Drive](#): The **Drive** object provides information about a drive (hard drive, CD ROM, etc) on your system. A **Vector** of drives on your system can be obtained from the [FSUtil.Drives](#) method.
- [ExeMeta](#): This object provides metadata properties relating to executable (program) files. It is obtained from the **Metadata** object.
- [Favorite](#): The **Favorite** object represents a [favorite folder](#). It is retrieved by enumerating or indexing the [Favorites](#) object.
- [Favorites](#): The **Favorites** object holds a collection of all the defined [favorite folders](#). It is retrieved from the [DOpus.favorites](#) method.
- [File](#): This object lets you read or write binary data from or to a file. It is obtained from the **FSUtil.OpenFile** and **Item.Open** methods.
- [FileAttr](#): This object represents file attributes (like read only, archived, etc). It used by the **Item** and **Format** objects, and can be created by the [FSUtil.NewFileAttr](#) method.
- [FileGroup](#): This object exposes information about a file group (when a **Tab** is set to group by a particular column). It is used by the **Item** and **Tab** objects.
- [FileSize](#): This object is used to represent a size in bytes (mainly because ActiveX scripting doesn't have proper support for 64 bit integers). It is used by the **Item** and **TabStats** objects.
- [FiletypeGroup](#): This object represents a file type group (as configured in the [File Type Groups](#) section of the [file type editor](#)).
- [Filters](#): This object lets you access information about the global filter settings (configured on the [Folders / Global Filters](#) page in Preferences).
- [FolderEnum](#): This object lets a script enumerate the contents of a folder. It is obtained using the **FSUtil.ReadDir** method.
- [FontMeta](#): This object provides metadata properties relating to font files. It is obtained from the **Metadata** object.
- [Format](#): This object provides information about the display format in a tab. It is obtained from the **Tab.format** property.
- [FSUtil](#): This object provides various utility methods relating to file system activity. It is obtained from the **DOpus.FSUtil** property.
- [Func](#): This object is passed to a [script function](#) (via **ClickData.func**) or [script-defined internal command](#) (via **ScriptCommandData.func**). It provides information relating to the function invocation (source and destination tabs, arguments, etc).
- [ImageMeta](#): This object provides metadata properties relating to image files. It is obtained from the **Metadata** object.
- [Item](#): This object represents a file or a folder. It can be returned from various methods of the **Tab** object, when enumerating a folder using the **FSUtil.ReadDir** method, and is used to provide files for a command to act on using the **Command** object.

- [Lister](#): This object represents a [Lister](#) window.
- [Listers](#): The [Listers](#) object is a collection of all currently open Lister windows (each one represented by a [Lister](#) object). It can be obtained from the **DOpus.listers** property.
- [Map](#): This object is similar to an array or vector (e.g. [Vector](#)) in that it can store one or more objects, but has the advantage of using a dictionary system to locate objects rather than numeric indexes. It is obtained from the **DOpusFactory.Map** method.
- [Metadata](#): This object represents a file or folder's metadata. It can be obtained from the **Item.metadata** property, as well as the **FSUtil.GetMetadata** method.
- [Msg](#): The **Msg** object represents a [script dialog](#) input event message. It's returned by the **Dialog.GetMsg** method which you call when running the message loop for a [detached dialog](#).
- [OtherMeta](#): This object provides general metadata properties relating to files and folders. It is obtained from the **Metadata** object.
- [Path](#): This object represents a file system path. It contains several methods to manipulate the path. Path objects are returned by several properties and can be created by the **FSUtil.NewPath** method.
- [Progress](#): This object represents a progress dialog, that lets you visually indicate to the user the progress of your script function. It is obtained from the **Command.progress** property.
- [QuickFilter](#): This object provides information about the state of the quick filter in a tab. It's obtained from the **Tab.quickfilter** property.
- [Rect](#): The **Rect** object represents a rectangle.
- [Results](#): This object represents the results of a command (the error code in the case of failure, plus any new tabs or Listers created by the command). It is obtained from the **Command.results** property.
- [ScriptColumn](#): This object represents a [script-defined column](#). It is obtained from the **ScriptInitData.AddColumn** method, while processing the **OnInit** event.
- [ScriptCommand](#): This object represents a [script-defined internal command](#). It is obtained from the **ScriptInitData.AddCommand** method, while processing the **OnInit** event.
- [ScriptConfig](#): This object represents script-defined configuration data that Opus stores for each script. The configuration items are initialised via the **ScriptInitData.config** property, and are then available to the script via the **Script.config** property.
- [ScriptStrings](#): The **ScriptStrings** object is returned by the **DOpus.strings** property. It lets you access any strings defined via [string resources](#).
- [ShellProperty](#): The **ShellProperty** object represents a shell property - an item of metadata for a file or folder that comes from Windows or third-party extensions. The **FSUtil.GetShellPropertyList** method lets you retrieve a list of available shell properties.
- [SmartFavorite](#): A **SmartFavorite** object represents an entry for a folder in the [SmartFavorites](#) table. It is retrieved by enumerating or indexing the [SmartFavorites](#) object.
- [SmartFavorites](#): The **SmartFavorites** object lets you query the contents of the [SmartFavorites](#) table. It is retrieved from the **DOpus.smartfavorites** property.
- [SortOrder](#): The **SortOrder** object is returned by the **Format.manual\_sort\_order** property if [manual sort mode](#) is active. It lets you query and modify the sort order.
- [StringTools](#): This object provides utility functions for string encoding and decoding. It is obtained from the **DOpusFactory.StringTools** method.
- [StringSet](#): This object is similar to an array or vector (e.g. [Vector](#)) of strings, but has the advantage of using a dictionary system to locate strings rather than numeric indexes. It is obtained from the **DOpusFactory.StringSet** and **StringSetI** methods.
- [SysInfo](#): The **SysInfo** object is created by the **DOpusFactory.SysInfo** method. It lets scripts access miscellaneous system information that may not be otherwise easy to obtain from a script.

- [Tab](#): This object represents a folder tab in a Lister. A Lister's tabs are available via various Lister object properties (e.g. **Lister.activetab**) and also used to specify the source/destination of a command (e.g. **Command.sourcetab**).
- [TabStats](#): This object provides various statistics about a folder tab (the number of selected files, total number of items, etc). It is obtained from the **Tab.stats** and **Tab.selstats** properties.
- [Toolbar](#): This object represents a toolbar. It is obtained with the **DOpus.toolbars** and **Lister.toolbars** properties.
- [Toolbars](#): The **Toolbars** object lets you enumerate all the defined toolbars in your Directory Opus configuration (whether currently turned on or not).
- [Var](#): This object represents a variable. Toolbar buttons, hotkeys and scripts can read and store variables, and variables can be saved from one session of Opus to another. The **Var** object is obtained from the **Vars** collection.
- [Vars](#): This object represents a collection of variables. Depending on the variables' scope it can be obtained from the **DOpus.vars**, **Lister.vars**, **Tab.vars**, **Command.vars** or **Script.vars** properties.
- [Vector](#): This object is similar to an array - it can store an unlimited number of elements of any type. Several properties and methods in the Opus scripting interface use Vectors, and you can use them interchangeably with arrays in most cases. The Vector is provided because some scripting languages only offer incomplete or incompatible arrays - using Vectors means the object can be used consistently across any ActiveX scripting language. A **Vector** is created by the **DOpusFactory.Vector** method.
- [Version](#): This object represents information about the current Opus version. It is obtained from the **DOpus.Version** property.
- [VideoMeta](#): This object provides metadata properties relating to movie files. It is obtained from the **Metadata** object.
- [Viewer](#): The **Viewer** object represents a standalone [image viewer](#).
- [Viewers](#): The **Viewers** object is a collection of all currently open standalone [image viewers](#). It can be obtained via the **DOpus.viewers** property
- [Wild](#): This object allows a script to access the in-built pattern matching functions in Opus. It is obtained from the **FSUtil.NewWild** method.
- [WinVer](#): This object represents information about the current Windows version. It is obtained from the **Version.winver** property.

## AboutData

If a [script add-in](#) provides an [OnAboutScript](#) method, it is passed an **AboutData** object when invoked via the user clicking the *About* button in Preferences.

Property Name	Return Type	Description
window	<i>int</i>	This is a handle to the parent window that the script should use if displaying a dialog via the <a href="#">Dialog</a> object. Even though this is not a <a href="#">Lister</a> or <a href="#">Tab</a> , it can still be assigned to the <b>Dialog.window</b> property to set the parent window of the dialog.

## ActivateListerData

If a [script add-in](#) implements the [OnActivateLister](#) event, the method receives an **ActivateListerData** whenever the window activation state of a Lister changes.

Property Name	Return Type	Description
active	<i>bool</i>	Returns <b>True</b> if this Lister is activating, <b>False</b> if deactivating. Note that if the activation moves from one Lister straight to another the script will be called twice.
lister	<i>object:</i> <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the Lister that is closing.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .

## ActivateTabData

If a [script add-in](#) implements the [OnActivateTab](#) event, the method receives an **ActivateTabData** object whenever the activation state of a tab changes.

Property Name	Return Type	Description
newtab	<i>object:</i> <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object representing the tab that has become active.
oldtab	<i>object:</i> <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object representing the tab that has gone inactive.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .

## AddCmdData

The **AddCmdData** object is passed to the [OnAddCommands](#) event in a [script add-in](#). The script can use this to [add internal commands](#) using the **AddCommand** method.

Method Name	Arguments	Return Type	Description
AddCommand	<i>none</i>	<i>object:</i> <a href="#">ScriptCommand</a>	Adds a new internal command to Opus. The returned <a href="#">ScriptCommand</a> object must be properly initialized. A script add-in can add as many internal commands as it likes to the Opus internal command set.

## AddColData

The **AddColData** object is passed to the **OnAddColumns** event in a [script add-in](#). The script can use this to [add columns](#) using the **AddColumn** method.

Method Name	Arguments	Return Type	Description
-------------	-----------	-------------	-------------

AddColumn	<i>none</i>	<i>object:</i> <a href="#">ScriptColumn</a>	Adds a new information column to Opus. The returned <a href="#">ScriptColumn</a> object must be properly initialized. A script add-in can add as many columns as it likes, and these will be available in file displays, infotips and the <a href="#">Advanced Find</a> function.
-----------	-------------	---------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

### **AfterFolderChangeData**

If a [script add-in](#) implements the [OnAfterFolderChange](#) event, the method receives an **AfterFolderChangeData** object once the folder read is complete.

<b>Property Name</b>	<b>Return Type</b>	<b>Description</b>
action	<i>string</i>	Returns a string indicating the action that triggered the folder read. The string will be one of the following: <i>normal, refresh, parent, root, back, forward, dblclk.</i>
path	<i>object:</i> <a href="#">Path</a>	<i>If the read failed, this will return a <a href="#">Path</a> object representing the path that Opus tried to read.</i>  <i>If the read was successful, this property is not provided - instead, the <b>tab</b> property provides access to this information.</i>  Use the <b>result</b> property to know if the read was a success.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift, ctrl, alt, lwin, rwin.</i>
result	<i>bool</i>	Returns <b>True</b> if the folder was read successfully, or <b>False</b> on failure.
tab	<i>object:</i> <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object representing the tab that read the folder.

## Alias

An **Alias** object represents a defined [folder alias](#). It is retrieved by enumerating or indexing the [Aliases](#) object.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the name of the alias.
path	<i>object:Path</i>	Returns the target of the alias as a <a href="#">Path</a> object.
system	<i>bool</i>	<b>True</b> if the object is a system-defined alias, <b>False</b> if it is user defined.

## Aliases

The **Aliases** object holds a collection of all the defined [folder aliases](#). It is retrieved from the [DOpus.aliases](#) method.

Property Name	Return Type	Description
<default value>	<i>collection:Alias</i>	You can enumerate the <b>Aliases</b> object, or query the value of an individual alias by name (e.g. <code>DOpus.Output(DOpus.aliases("desktop").path);</code> )

Method Name	Arguments	Return Type	Description
Add	<string:name> <string:path>	<i>none</i>	Adds a new alias to the system with the specified name and path. Note that you should <b>not</b> provide the leading forward-slash (/) in the alias name.
Delete	<string:name>	<i>none</i>	Deletes the specified alias.
Update	<i>none</i>	<i>none</i>	Updates the state of this object. When the <b>Aliases</b> object is first retrieved via <b>DOpus.aliases</b> , a snapshot is taken of the aliases at that time. If you make changes via the object it will reflect them but any changes made outside the script (e.g. via the <b>Favorites ADD=alias</b> command) will not be detected unless you call the <b>Update</b> method.

## Args

The **Args** object is passed to a script when it is invoked via a command, via the [Func.args](#) property. It is used when a command added by a script has a [command line template](#) and provides access to any arguments provided on the command line.

Property Name	Return Type	Description
<argument name>	variant	<p>The <b>Args</b> object will have one property corresponding to each of the arguments in the command line template.</p> <p>For example, if the command line template is <b>NAME/K,SIZE/N</b>, the <b>Args</b> object would have two properties, called <b>name</b> and <b>size</b>.</p> <p>The type returned by each property is also defined by the template. In the above example, <b>name</b> would return a <i>string</i> and <b>size</b> an <i>int</i>.</p> <p>A <b>/S</b> argument returns a <i>bool</i>, a <b>/N</b> argument returns an <i>int</i>, and all other argument types return a <i>string</i>. Note that a <b>/O</b> argument will also return a <i>bool</i> if no string value is provided on the command line.</p> <p>If an argument is marked in the template as <b>/M</b> (multiple) then it returns a <a href="#">Vector</a> containing elements of the appropriate type.</p> <p>If an argument was not provided on the command line by the user, its property will either return <i>bool</i> (for a <b>/S</b> or <b>/O</b> argument), or an empty variant otherwise.</p>
<b>got_arg</b>	object	<p>The <b>got_arg</b> property returns an object with a <i>bool</i> child property for each argument in the template. It lets you test if a particular argument was provided on the command line, before you actually query for the value of the argument. For example, <i>If Args.got_arg.size Then...</i></p>

## AudioCoverArt

The **AudioCoverArt** object provides access to an audio file's embedded cover art. It is obtained through the [AudioMeta.coverart](#) property.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns a string indicating the intended use for this cover art. Possible values are <i>artist</i> , <i>back</i> , <i>band</i> , <i>bandlogo</i> , <i>colorfulfish</i> (this is unfortunately part of the ID3 specification), <i>composer</i> , <i>conductor</i> , <i>front</i> , <i>icon</i> , <i>illustration</i> , <i>leadartist</i> , <i>leaflet</i> , <i>location</i> , <i>lyricist</i> , <i>media</i> , <i>other</i> , <i>otherfileicon</i> , <i>performance</i> , <i>publisherlogo</i> , <i>recording</i> , <i>vidcap</i> .
data	<i>object:Blob</i>	Returns a <a href="#">Blob</a> object representing the actual image data.
depth	<i>int</i>	Returns the bit depth of this image.
desc	<i>string</i>	Returns the description of this image (if any).
ext	<i>string</i>	Returns the default file extension for this image, if it can be determined.
height	<i>int</i>	Returns the height of this image, in pixels.
mime	<i>string</i>	Returns the image's MIME type, if specified in the file.
size	<i>object:FileSize</i>	Returns a <a href="#">FileSize</a> object representing the size of the image data.
type	<i>string</i>	Returns a "pretty" form of the intended use string (i.e. the <i>default value</i> ), translated to the current Opus user interface language.
width	<i>int</i>	Returns the width of this image, in pixels.

## AudioMeta

The **AudioMeta** object is retrieved from the [Metadata.audio](#) or [Metadata.audio\\_text](#) properties. It provides access to metadata relating to sound files.

Property Name	Return Type	Description
<column keyword>	variant	Returns the value of the specified column, as listed in the <i>Music</i> section of the <a href="#">Keywords for Columns</a> page.
coverart	collection: <a href="#">AudioCoverArt</a>	Returns a collection of <a href="#">AudioCoverArt</a> objects representing any cover art imagery stored in the audio file.  The default value of this property returns the number of cover art images - for performance reasons, you should check whether this is greater than 0 before enumerating or accessing individual items in the collection.

## BeforeFolderChangeData

If a [script add-in](#) implements the [OnBeforeFolderChange](#) event, the method receives a **BeforeFolderChangeData** object before the new folder is read.

Property Name	Return Type	Description
action	string	Returns a string indicating the action that triggered the folder read. The string will be one of the following: <i>normal, refresh, parent, root, back, forward, dblclk</i> .
initial	bool	Returns <b>True</b> if this is the first path to be read into this tab (i.e. previously the tab was empty).
path	object: <a href="#">Path</a>	Returns a <a href="#">Path</a> object representing the new path that is to be read.
qualifiers	string	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift, ctrl, alt, lwin, rwin</i> .

tab	<i>object:</i> <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object representing the tab that is changing folder.
-----	------------------------------------	------------------------------------------------------------------------------------

## **Blob**

A **Blob** object represents a chunk of binary data. Scripting languages like *VBScript* and *JScript* have no built-in support for binary data - this object can be used to allocate a chunk of memory and manipulate it in a similar way to low-level languages like C. You can use **Blob** objects in conjunction with the [File](#) object to read or write binary data from or to disk files.

**Blob** objects are convertible to and from two types of ActiveX arrays - a *SAFEARRAY* of type *VT\_UI1* (known as an **array**) and a *SAFEARRAY* of type *VT\_VARIANT*, with each variant holding a *VT\_UI1* (known as a **VB array**). You can initialize a **Blob** with either of these two types of array (either when creating it via the **DOpusFactory.Blob** method or with the **Blob.CopyFrom** method), and you can also convert a **Blob** back to an array with the **ToArray** and **ToVBAArray** methods. Support for these array types is dependent on the scripting language.

You can read and write individual bytes within the **Blob** by indexing the byte offset starting from 0. For example, *my\_blob(5) = 128* would set the value of the sixth byte in the blob to 128.

Property Name	Return Type	Description
size	<i>object:</i> <a href="#">FileSize</a>	Returns a <a href="#">FileSize</a> object representing the size of this <b>Blob</b> in bytes.

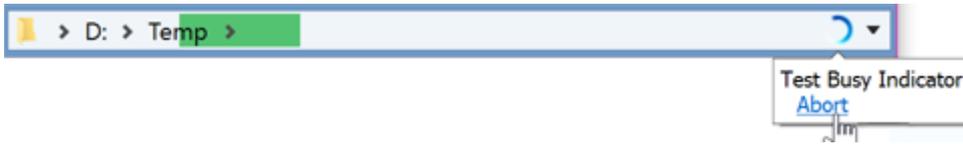
Method Name	Arguments	Return Type	Description
Compare	<Blob:source> <int:to> <int:from> <int:size>	<i>int</i>	Compares the contents of this <b>Blob</b> against another <b>Blob</b> (or array). By default the entire contents of the two blobs are compared. The optional parameters that let you configure the operation are:  to - specifies the byte offset within this <b>Blob</b> to compare against. Defaults to 0. from - specifies the byte offset within the source <b>Blob</b> to compare with. Defaults to 0. size - specifies the number of bytes to compare. Defaults to the full size of the source <b>Blob</b> .  The return value is <b>0</b> if the two blobs are the

			<p>same. A value of <b>-1</b> indicates this blob is less than the other blob, and <b>1</b> indicates this blob is greater than the other blob.</p>
CopyFrom	<p>&lt;Blob:source&gt; &lt;int:to&gt; &lt;int:from&gt; &lt;int:size&gt;</p> <p><i>or</i></p> <p>&lt;string&gt; &lt;type&gt;</p>	<i>none</i>	<p>Copies data from the source <b>Blob</b> (or array) into this <b>Blob</b>. By default the entire contents of the source <b>Blob</b> will be copied over the top of this one. The optional parameters that let you configure the operation are:</p> <p>to - specifies the byte offset within this <b>Blob</b> to copy to. Defaults to 0.  from - specifies the byte offset within the source <b>Blob</b> to copy from. Defaults to 0.  size - specifies the number of bytes to copy. Defaults to the full size of the source <b>Blob</b>.</p> <p>As well as copying from another <b>Blob</b>, you can use this method to initialise a <b>Blob</b> from a string. By default the <b>Blob</b> will be set to the Unicode form of the string; if you pass "<b>utf8</b>" as the second parameter it will initialise the <b>Blob</b> with the UTF8-encoded form of the string.</p> <p>If this <b>Blob</b> is not currently large enough to contain the copied data it will be resized automatically.</p>
Find	<p>&lt;Blob:search&gt; &lt;int:from&gt; &lt;int:size&gt;</p>	<i>object:FileSize</i>	<p>Searches the contents of this <b>Blob</b> for the data contained in another <b>Blob</b> (or array). By default the entire contents of this <b>Blob</b> are searched. The optional <b>from</b> parameter lets you specify the starting position for the search, and the optional <b>size</b> parameter lets you specify the length of data in this Blob to search through.</p> <p>The return value is -1 if the search data were not found, otherwise the offset from the start of the <b>Blob</b> data is returned.</p>
Free	<i>none</i>	<i>none</i>	Frees the memory associated with this <b>Blob</b> and resets its size to 0.
Init	<i>none</i>	<i>none</i>	Initialises the contents of the <b>Blob</b> (every byte within the blob will be set to 0). Equivalent to <i>Set(0)</i> .
Resize	<int:size>	<i>none</i>	Resizes the <b>Blob</b> to the specified number of bytes.

Reverse	<i>none</i>	<i>none</i>	Reverses the contents of the <b>Blob</b> .
Set	<byte:value> <int:to> <int:size>	<i>none</i>	Sets the contents of the <b>Blob</b> to the specified byte value (every byte within the blob will be set to that value). By default the whole <b>Blob</b> will be affected. The option <i>to</i> parameter lets you specify a byte offset to start at, and the optional <i>size</i> parameter lets you control the number of bytes affected.
ToArray	<int:from> <int:size>	SAFEARRAY of VT_UI1	Converts the contents of this <b>Blob</b> to a <i>SAFEARRAY</i> of type <i>VT_UI1</i> . By default the entire contents of the <b>Blob</b> will be copied to the array. The optional parameters that let you configure the operation are:  from - specifies the byte offset within the source <b>Blob</b> to copy from. Defaults to 0. size - specifies the number of bytes to copy. Defaults to the full size of the source <b>Blob</b> .
ToVBAArray	<int:from> <int:size>	SAFEARRAY of VT_VARIANT	Converts the contents of this <b>Blob</b> to a <i>SAFEARRAY</i> of type <i>VT_VARIANT</i> . Each variant in the array contains a <i>VT_UI1</i> . By default the entire contents of the <b>Blob</b> will be copied to the array. The optional parameters that let you configure the operation are:  from - specifies the byte offset within the source <b>Blob</b> to copy from. Defaults to 0. size - specifies the number of bytes to copy. Defaults to the full size of the source <b>Blob</b> .

## BusyIndicator

A **BusyIndicator** object lets you control the breadcrumbs bar busy indicator from your script.



By default a busy indicator simply indicates that something is happening; it can also be used to indicate the progress of a job (percentage complete from 0% to 100%). The user can click the spinning circle to see a description of the jobs that are running, and each job can optionally allow the user to abort it by displaying an *Abort* link they can click.

**BusyIndicator** objects are created using the [DOpusFactory.BusyIndicator](#) method. The basic steps for using one in your script are:

1. Create the object by calling **DOpus.Create.BusyIndicator()**.
2. Optionally set the **abort** property to **True** to enable user aborting.
3. Call the **Init** method to initialize and display the busy indicator.
4. Call the **Update** method when needed to update the progress or explanatory text.
5. Poll the **abort** property to check for user abort if desired.
6. Call the **Destroy** method to remove the busy indicator when the job is complete.

Property Name	Return Type	Description
abort	<i>bool</i>	<p>Before the <b>Init</b> method has been called, you can set this property to <b>True</b> to enable abort by the user (as shown above).</p> <p>After <b>Init</b> has been called, this property will return <b>True</b> if the user has clicked the <i>Abort</i> link.</p>

Method Name	Arguments	Return Type	Description
Destroy	<i>none</i>	<i>none</i>	Removes the busy indicator from display and destroys its internal data structures. The <b>BusyIndicator</b> object itself can be re-used by calling the <b>Init</b> method again.
Hide	<i>none</i>	<i>none</i>	Removes the busy indicator from display, but does not destroy its internal data. The indicator can be re-displayed by calling the <b>Show</b> method.
Init	<object:window> <string:description> <bool:visible>	<i>bool</i>	<p>Initializes a <b>BusyIndicator</b> object and optionally displays it. The <b>window</b> parameter specifies the Lister that the indicator is to be attached to - you can pass either a <a href="#">Lister</a> or a <a href="#">Tab</a> object.</p> <p>The optional <b>description</b> parameter lets you specify a text string that is displayed to the user when they click the spinning circle.</p> <p>The optional <b>visible</b> parameter lets you make the indicator visible immediately by passing <b>True</b>. Alternatively, call the <b>Show</b> method to make the indicator visible.</p>
Show	<i>none</i>	<i>none</i>	Displays the busy indicator.
Update	<string:description> <int:percentage>	<i>none</i>	Updates the busy indicator. The <b>description</b> parameter lets you specify a new description string, and the optional <b>percentage</b> parameter lets you specify a new percentage complete value from 0 to 100.

## ClickData

The **ClickData** object is passed to the [OnClick](#) method in a script function.

Property Name	Return Type	Description
func	<i>object:</i> <a href="#">Func</a>	Returns a <b>Func</b> object relating to this function. This provides access to information about the function's environment - (source and destination tabs, qualifier keys, etc).

## CloseListerData

If a [script add-in](#) implements the [OnCloseLister](#) event, the method receives a **CloseListerData** object before a Lister is closed.

Property Name	Return Type	Description
lister	<i>object</i> : <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the Lister that is closing.
prevent_save	<i>bool</i>	Set this to <b>True</b> to prevent the closing Lister from being saved as the new default Lister.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
shutdown	<i>bool</i>	Returns <b>True</b> if the Lister is closing because Opus is shutting down.

## CloseTabData

If a [script add-in](#) implements the [OnCloseTab](#) event, the method receives a **CloseTabData** object when a tab is closed.

Property Name	Return Type	Description
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
tab	<i>object</i> : <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object representing the tab that is closing.

## Column

The **Column** object represents an information column displayed in a tab (e.g. name, size, attributes, etc). A collection of **Column** objects can be retrieved from the [Format.columns](#) property.

Property Name	Return Type	Description
---------------	-------------	-------------

<default value>	<i>string</i>	Returns the name of the column.
autosize	<i>bool</i>	Returns <b>True</b> if the column width is set to <i>auto</i> .
collapse	<i>bool</i>	Returns <b>True</b> if the column width is set to <i>collapse</i> .
expand	<i>bool</i>	Returns <b>True</b> if the column width is set to <i>expand</i> .
fill	<i>bool</i>	Returns <b>True</b> if the column width is set to <i>fill</i> .
header	<i>string</i>	Returns the name of the column as displayed in the Lister column header.
label	<i>string</i>	Returns the name of the column as displayed in the <i>Columns</i> tab in the <i>Folder Options</i> dialog.
max	<i>int or string</i>	Returns the maximum width of the column in pixels, or the string "fill" if the maximum is set to <i>fill</i> .
name	<i>string</i>	Returns the name of the column.
reverse	<i>bool</i>	Returns <b>True</b> if the sort direction of the column is reversed.
sort	<i>int</i>	Returns the sort order of the column (e.g. 1 for the primary sort field, 2 for the secondary sort field, etc). Returns 0 if the display is not sorted by this column.
width	<i>int</i>	Returns the current display width of the column in pixels.

## Command

The **Command** object is used by a script to run Opus commands. Any command that you can run from a button or hotkey you can run from a script - a script can even run [commands added by other scripts](#). Fundamentally, using the **Command** object is similar to configuring an Opus button. You add one or more command lines to the object just the same as you add one or more command lines to a button's function. You can tell it which files and folders to act upon, and you can use the various methods and properties of the object to modify the behavior of the command. Once the object has been initialized you can use the **Run** or **RunCommand** methods to invoke the command.

A **Command** object can be created by the [DOpusFactory.Command](#) method. By default, this object has no source, destination, files to operate on, etc. - you must use the appropriate methods to configure the command before running it. You can also retrieve a **Command** object from the [Func.command](#) property, and in this case the source, destination, files to operate on and several other properties are pre-initialized for you.

Property Name	Return Type	Description
deselect	<i>bool</i>	Set this property to <b>False</b> to prevent files used by this command from being deselected, and <b>True</b> to deselect them once the function is finished. Note that files will

		only be deselected if they came from a <a href="#">Tab</a> object, and only then if the command is successful.
dest	<i>object:</i> <a href="#">Path</a>	Returns a <a href="#">Path</a> object that represents the destination folder of this command. If a destination tab is set, this will be the path in the tab. You can not set this property directly - instead, use either the <b>SetDest</b> or <b>SetDestTab</b> methods to change the destination folder.
desttab	<i>object:</i> <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object that represents the destination tab for this command (if it has one - not all commands require a destination). You can not set this property directly - instead, use the <b>SetDestTab</b> method to change the destination tab.
filecount	<i>int</i>	Returns the number of items in the <b>files</b> collection.
files	<i>collection:</i> <a href="#">Item</a>	Returns a collection of all <a href="#">Item</a> objects that represent the files and folders this command is to act upon. You can not modify this collection directly - instead you can use the various methods ( <b>ClearFiles</b> , <b>SetFiles</b> , <b>AddFile</b> , <b>RemoveFile</b> , etc.) to modify the list of items to act upon.
linecount	<i>int</i>	Returns the number of instruction lines added to the command.
progress	<i>object:</i> <a href="#">Progress</a>	Returns a <a href="#">Progress</a> object that you can use to display a progress indicator to the user.
results	<i>object:</i> <a href="#">Results</a>	After every command that is run with this object, a <b>Results</b> object is available from this property. This provides information about the outcome of the command.
source	<i>object:</i> <a href="#">Path</a>	Returns a <a href="#">Path</a> object that represents the source folder of this command. If a source tab is set, this will be the path in the tab. You can not set this property directly - instead, use either the <b>SetSource</b> or <b>SetSourceTab</b> methods to change the source folder.
sourcetab	<i>object:</i> <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object that represents the source tab for this command. You can not set this property directly - instead, use the <b>SetSourceTab</b> method to change the source tab.
vars	<i>object:</i> <a href="#">Vars</a>	This <a href="#">Vars</a> object represents all defined variables with <i>command scope</i> (that are scoped to this function - e.g. that were set using the <b>@set</b> directive).

Method Name	Arguments	Return Type	Description
AddFile	<string:path> or < <a href="#">Path</a> :path> or < <a href="#">Item</a> :item>	<i>int</i>	Adds the specified item to the collection of items this command is to act upon. You can pass the item's path as either a <i>string</i> or a <a href="#">Path</a> object, and

			<p>you can also pass an <a href="#">Item</a> object directly.</p> <p>This method returns the total number of items in the collection.</p>
AddFiles	<p><i>collection:</i> <a href="#">Item</a>  or <i>Vector:</i> <a href="#">Item</a>  or <i>Vector:</i> <a href="#">Path</a>  or <i>Vector:</i> <i>string</i></p>	<i>int</i>	<p>Adds the items in the specified collection to the list of items this command is to act upon. The return value is the new number of items in the collection.</p> <p>You can also pass a <a href="#">Vector</a> of <a href="#">Item</a> or <a href="#">Path</a> objects, or of strings (full paths), instead of a collection.</p>
AddFilesFromClipboard	<i>none</i>	<i>int</i>	<p>Adds the contents of the clipboard to the collection of items this command is to act upon. This method supports both files and file paths copied to the clipboard as text. The return value is the new number of items in the collection.</p>
AddFilesFromFile	<p>&lt;string:path&gt;  &lt;string:encoding&gt;</p>	<i>int</i>	<p>Reads file paths from the contents of the specified file and adds them to the item collection. You can provide the file's path as either a <i>string</i> or a <a href="#">Path</a> object. The file must consist of one absolute path per line.</p> <p>The encoding of the file is assumed to be ANSI, unless it has a BOM (byte-order-mark) at the start, or you specify the <i>encoding</i> argument. If you specify the encoding this must be a <i>string</i> equal to one of the following: <i>utf16be</i>, <i>utf16le</i>, <i>utf8</i>, <i>ansi</i> or <i>cp:XXXX</i> where <i>XXXX</i> specifies the code page number).</p> <p>The return value is the new number of items in the collection.</p>

AddFilesFromFolder	<string:path>	<i>int</i>	Adds the contents of the specified folder to the collection of items this command is to act upon. You can pass the folder's path as either a <i>string</i> or a <a href="#">Path</a> object. You can also append a <a href="#">wildcard pattern</a> to the path to only add files matching the specified pattern.
AddLine	<string:instruction >	<i>none</i>	Adds the specified instruction line to the command that this object will run. The <b>AddLine</b> method lets you build up complicated multiple line commands - add each line in turn and then run the command using the <b>Run</b> method. For a single line command it is simpler to use the <b>RunCommand</b> method.
Clear	<i>none</i>	<i>none</i>	Clears all instruction lines from the command.
ClearFailed	<i>none</i>	<i>none</i>	Clears the failure flags from the <a href="#">Item</a> collection. Any items that fail when a command is run will have their <b>failed</b> property set to <b>True</b> , and once this has happened the file will be skipped over by any subsequent commands. You can call this method to reset all the failure flags.
ClearFiles	<i>none</i>	<i>none</i>	Clears the collection of items this command is to act upon.
ClearModifier	<string:modifier>	<i>none</i>	Clears any modifiers that have been set for this command. The supported modifiers are a subset of the full list of <a href="#">command modifiers</a> - see the <b>SetModifier</b> method for a list of these. You can also pass * to clear all modifiers that have been set.
CommandList	<i>none</i> or <string:types>	<i>object: <a href="#">StringSet</a></i>	Returns a <a href="#">StringSet</a> containing the names of all the Opus commands. You can optionally filter this set by providing one or more of the following flags as an argument to the <b>CommandList</b> method:

			<p><i>i</i> - internal (built-in) commands  <i>s</i> - script commands  <i>u</i> - user commands</p>
Dlg	<i>none</i>	<i>object:Dialog</i>	Creates a new <a href="#">Dialog</a> object, that lets you display dialogs and popup menus. The dialog's <b>window</b> property will be automatically assigned to the source tab.
IsSet	<string:condition> [<string:command >]	<i>bool</i>	Returns <b>True</b> if the specified <a href="#">Set</a> command condition is true. This is the equivalent of the <a href="#">@ifset command modifiers</a> . The optional second parameter lets you test a condition based on a command other than <b>Set</b> - for example, <b>IsSet("VIEWERCMD=mark", "Show")</b> in the viewer to test if the current image is marked.
RemoveFile	<string:path> or < <a href="#">Path</a> :path> or < <a href="#">Item</a> :item> or <int:index>	<i>int</i>	Removes the specified file from the <a href="#">Item</a> collection. You can pass the file's path as either a <i>string</i> or a <a href="#">Path</a> object. You can also pass the <a href="#">Item</a> itself, or its index (starting from 0) within the collection. The return value is the new number of items in the collection.
Run	<i>none</i>	<i>int</i>	Runs the command that has been built up with this object. The return value indicates whether or not the command ran successfully. Zero indicates the command could not be run or was aborted; any other number indicates the command was run for at least some files. (Note that this is not the "exit code" for external commands. For external commands it only indicates whether or not Opus launched the command. If you need the exit code of an external command, use the <code>WScript.Shell Run</code> or <code>Exec</code> methods to run the command.) You can use the <b>Results</b>

			property to find out more information about the results of the command, and also discover which files (if any) failed using the <b>failed</b> property of each <a href="#">Item</a> in the <b>files</b> collection.
RunCommand	<string:instruction >	<i>int</i>	Runs the single line command given by the <i>instruction</i> argument. Calling this method is the equivalent of adding the single line with the <b>AddLine</b> method and then calling the <b>Run</b> method.
SetDest	<string:path>	<i>none</i>	Sets the command's destination to the specified path. You can provide the path as either a <i>string</i> or a <a href="#">Path</a> object. Calling this method clears the destination tab property from the command.
SetDestTab	< <a href="#">Tab</a> :tab>	<i>none</i>	Sets the command's destination to the specified tab. The destination path will be initialized from the tab automatically (so you don't need to call <b>SetDest</b> as well as <b>SetDestTab</b> ).
SetFiles	collection: <a href="#">Item</a>	<i>none</i>	Configures the command to use the files in the specified <a href="#">Item</a> collection as the items the command will act upon.  You can also pass a <a href="#">Vector</a> of <a href="#">Item</a> objects instead of a collection.
SetModifier	<string:modifier> <string:value>	<i>none</i>	Turns on a modifier for this command. The supported modifiers are a subset of the full list of <a href="#">command modifiers</a> :  <i>admin, async, codepage, externalonly, leavedoswindowopen, nodeselect, noexpandenv, nofilenamequoting, nocalizefiles, noprogess, norunbatch, resolvelinks, runmode</i>

			<p>Using this method is the equivalent of using the <b>AddLine</b> method to add the modifier to the command as an instruction; e.g.</p> <p><i>Command.SetModifier("async")</i> is the same as <i>Command.AddLine("@async")</i>.</p> <p>If the modifier requires a value it is passed as the second argument, e.g.</p> <p><i>Command.SetModifier("runmode", "hide")</i>.</p>
SetProgress	< <a href="#">Progress</a> :progress>	<i>none</i>	Lets you share the progress indicator from one command with another command. You can pass this method the value of <b>progress</b> property obtained from another <b>Command</b> object.
SetQualifiers	<string:qualifiers>	<i>none</i>	<p>This method lets you control which qualifier keys the command run by this object will consider to have been pressed when it was invoked. For example, several internal commands change their behavior when certain qualifier keys are held down - calling this method allows you to set which keys they will see.</p> <p>The <i>qualifiers</i> argument must consist of one or more of the following strings (comma-separated): <i>none, shift, ctrl, alt, lwin, rwin, win</i>.</p>
SetSource	<string:path>	<i>none</i>	Sets the command's source to the specified path. You can provide the path as either a <i>string</i> or a <a href="#">Path</a> object. Calling this method clears the source tab property from the command.
SetSourceTab	< <a href="#">Tab</a> :tab>	<i>none</i>	Sets the command's source to the specified tab. The source path will be initialized from the tab automatically (so you don't need to call <b>SetSource</b> as well as <b>SetSourceTab</b> ).

SetType	<string:type>	<i>none</i>	Sets the type of function that this command will run. This is equivalent to the drop-down control in the <a href="#">Advanced Command Editor</a> . The <i>type</i> argument must be one of the following strings: <i>std</i> , <i>msdos</i> , <i>script</i> , <i>wsl</i> . Standard ( <i>std</i> ) is the default if the type is not specifically set.
UpdateToggle	<i>none</i>	<i>none</i>	This method can be used to update the appearance of toolbar buttons that use <a href="#">@toggle:if</a> to set their selection state based on the existence of a <i>global</i> -, <i>tab</i> - or <i>Lister-scoped</i> variable. You would call this method if you have changed such a variable from a script to force buttons that use it to update their selection status.

### ConfigChangeData

If a [script add-in](#) implements the [OnScriptConfigChange](#) event, the method receives a **ConfigChangeData** object whenever the user modifies the script's configuration via the Preferences editor.

Property Name	Return Type	Description
changed	<a href="#">Vector</a> : <i>string</i>	Returns a <a href="#">Vector</a> containing the names of the configuration items that were modified.

### Control

The **Control** object represents a control on a [script dialog](#); it lets you read and modify a control's value (and contents). Use the [Dialog.Control](#) method to obtain a **Control** object.

Property Name	Return Type	Description
bg	<i>string</i>	Set or query the color used for the background (fill) of this control. This is in the format <i>#RRGGBB</i>

		(hexadecimal) or <i>RRR,GGG,BBB</i> (decimal).  Currently only <i>static text</i> controls are supported for this property.
columns	<i>object</i> : <a href="#">DialogListColumns</a>	For a <i>list view</i> control, returns a <a href="#">DialogListColumns</a> object that lets you query or modify the columns in <i>Details</i> mode.
count	<i>int</i>	Returns the number of items contained in the control (e.g. in a <i>combo box</i> , <i>list box</i> or <i>list view</i> , returns the number of items in the list).
cx	<i>int</i>	Set or query the width of the control, in pixels.
cy	<i>int</i>	Set or query the height of the control, in pixels.
enabled	<i>bool</i>	Set or query the enabled state of the control. Returns <b>True</b> if the control is enabled, <b>False</b> if it's disabled. You can set this property to change the state.
fg	<i>string</i>	Set or query the color used for the text (foreground) of this control. This is in the format <i>#RRGGBB</i> (hexadecimal) or <i>RRR,GGG,BBB</i> (decimal).  Currently only <i>static text</i> controls are supported for this property.
focus	<i>bool</i>	Set or query the input focus state of the control. Returns <b>True</b> if the control currently has input focus, <b>False</b> if it doesn't. Set to <b>True</b> to give the control input focus.
label	<i>string</i>	Set or query the control's label. Not all controls have labels - this will have no effect on controls (like the <i>list view</i> ) that don't.  Note that for <i>combo box</i> controls, this property is only valid for an editable combo - that is, one that you can type your own text into. You can use this property to set or query the current value of the editable text.
mode	<i>string</i>	For a <i>list view</i> control, lets you change or query the current view mode. Valid values are <b>icon</b> , <b>details</b> , <b>smallicon</b> , <b>list</b> .
readonly	<i>bool</i>	Set or query the <i>read only</i> state of an <i>edit</i> control.
style	<i>string</i>	Set or query the font styles used to display this control's label. The string consists of zero or more characters; valid characters are <b>b</b> for bold and <b>i</b> for italics.  Currently only <i>static text</i> controls are supported for this property.

value	<i>string</i> or <i>bool</i> or <i>int</i> or  <i>object:DialogListItem</i> or  <i>object:Vector</i>	<p>Set or query the control's value. The meaning of this property depends on the type of the control:</p> <ul style="list-style-type: none"> <li>• <b>Edit control:</b> Returns or accepts a string representing the current contents of the edit control.</li> <li>• <b>Check box:</b> For a simple on/off check box, returns or accepts a <i>bool</i> - <b>True</b> for checked and <b>False</b> for unchecked. For a tri-state check box, returns or accepts an <i>int</i> - <b>0</b> for unchecked, <b>1</b> for checked and <b>2</b> for the indeterminate state.</li> <li>• <b>Radio button:</b> Returns or accepts a <i>bool</i> - <b>True</b> for checked and <b>False</b> for unchecked.</li> <li>• <b>Tab:</b> Returns or accepts an <i>int</i> indicating the currently selected page in the tab control.</li> <li>• <b>List box / combo box / list view:</b> Returns or accepts a <a href="#">DialogListItem</a> representing the selected item. When setting the value it also accepts an <i>int</i> representing the 0-based index of the selected item.</li> </ul> <p>Note that for a multiple-selection <i>list box</i> or <i>list view</i>, this value will return a <a href="#">Vector</a> of <a href="#">DialogListItem</a> objects, representing all currently selected items.</p>
visible	<i>bool</i>	Set or query the visible state of the control. Returns <b>True</b> if the control is visible and <b>False</b> if it's hidden. You can set this property to hide or show the control.
x	<i>int</i>	Set or query the left (x) position of the control, in pixels.
y	<i>int</i>	Set or query the top (y) position of the control, in pixels.

Method Name	Arguments	Return Type	Description
AddItem	<string:name> [<int:value>]  or  <object:item>	<i>int</i>	<p>Adds a new item to the control (<i>list box</i>, <i>combo box</i> or <i>list view</i>). The first parameter is the item's name, and the optional second parameter is a data value to associate with the item. The item is added to the end of the list.</p> <p>You can also pass a <a href="#">DialogListItem</a> object obtained from another control.</p>

			The return value indicates the position in the list of the new item.
DeselectItem	<int:position> or <object:item>	<i>int</i>	<p>This method is mainly for use with multiple-selection <i>list box</i> and <i>list view</i> controls. It lets you deselect individual items in the control while leaving other items selected (or unaffected).</p> <p>You can specify either the index of the item to select (0 means the first item, 1 means the second and so on) or a <a href="#">DialogListItem</a> object obtained from the <b>GetItemAt</b> or <b>GetItemByName</b> methods.</p> <p>You can also specify <b>-1</b> to deselect all items in the list box.</p>
GetItemAt	<int:position>	object: <a href="#">DialogListItem</a>	Returns a <a href="#">DialogListItem</a> object representing the item contained in the control at the specified index ( <i>list box</i> , <i>combo box</i> or <i>list view</i> ). Item 0 represents the first item in the list, item 1 the second, and so on.
GetItemByLabel GetItemByName	<string:name>	object: <a href="#">DialogListItem</a>	Returns a <a href="#">DialogListItem</a> object representing the item contained in the control with the specified name ( <i>list box</i> , <i>combo box</i> or <i>list view</i> ). This method has two names (... <i>Label</i> and ... <i>Name</i> ) for historical reasons, you can use either method name interchangeably).
InsertItemAt	<int:position> <string:name> [<int:value>] or  <int:position> <object:item>	<i>int</i>	Inserts a new item in the control ( <i>list box</i> , <i>combo box</i> or <i>list view</i> ). The first parameter is the position to insert the item at (0 means the beginning of the list, 1 means the second position and so on). The second parameter is the item's name, and the optional third parameter is a data value to associate with the item.

			<p>Instead of the <i>name</i> and <i>value</i> you can also pass a <a href="#">DialogListItem</a> object obtained from another control.</p> <p>The return value indicates the position in the list of the new item.</p>
MoveItem	<p>&lt;int:position&gt; or &lt;object:item&gt;</p> <p>&lt;int:newposition&gt;</p>	<i>int</i>	<p>Moves an existing item to a new location (<i>list box, combo box or list view</i>). The first parameter is the item to move (you can pass either its index or a <a href="#">DialogListItem</a> object), and the second parameter is the new position the item should be moved to.</p> <p>The return value indicates the position in the list of the moved item.</p>
RemoveItem	<p>&lt;int:position&gt; or &lt;object:item&gt;</p>	none	<p>Removes an item from the control (<i>list box, combo box or list view</i>). You can provide either the index of the item to remove (0 means the first item, 1 means the second and so on) or a <a href="#">DialogListItem</a> object obtained from the <b>GetItemAt</b> or <b>GetItemByName</b> methods.</p> <p>You can also specify <b>-1</b> to completely clear the contents of the control, removing all items at once.</p>
SelectItem	<p>&lt;int:position&gt; or &lt;object:item&gt;</p> <p>or</p> <p>&lt;string:tab&gt;</p>	<i>int</i>	<p>Selects an item in the control. For a <i>list box, combo box or list view</i>, you can specify either the index of the item to select (0 means the first item, 1 means the second and so on) or a <a href="#">DialogListItem</a> object obtained from the <b>GetItemAt</b> or</p>

			<p><b>GetItemByName</b> methods.</p> <p>For a multiple-selection <i>list box</i> or <i>list view</i> you can also specify <b>-1</b> to select all items in the control.</p> <p>For a <i>tab control</i>, you can change which page is visible by specifying the name of the page (i.e. the name of the child dialog) to show.</p> <p>The return value indicates the new selected index.</p>
SelectRange	<int:start> <int:end>  or  <object:item1> <object:item2>	object: <a href="#">Vector</a>	<p>Selects text within an <i>edit control</i> (or the edit field in a <i>combo box</i> control). The two parameters represent the start and end position of the desired selection. To select the entire contents, use <b>0</b> for the start and <b>-1</b> for the end.</p> <p>The return value is a <a href="#">Vector</a> with two members that provide the current start and end of the selection. To query the range without changing it, simply call the <b>SelectRange</b> method with no arguments.</p> <p>In a <i>list box</i> or <i>list view</i> control, this method selects a range of items.</p>
SetPos	<int:x> <int:y>	none	Sets the position of this control. The x and y coordinates are specified in pixels.
SetPosAndSize	<int:x> <int:y> <int:cx> <int:cy>	none	Sets the position and size of the control, in a single operation. All coordinates are specified in pixels.

SetSize	<int:cx> <int:cy>	none	Sets the size of this control. The cx (width) and cy (height) values are specified in pixels.
---------	----------------------	------	-----------------------------------------------------------------------------------------------

## CustomFieldData

The **CustomFieldData** object is provided to a [rename script](#) via the [GetNewNameData](#).**custom** property. It provides access to the value of any [custom fields](#) that your script added to the *Rename* dialog.

Property Name	Return Type	Description
<custom field property>	<i>variant</i>	<p>The properties of the <b>CustomFieldData</b> object are entirely determined by the script itself.</p> <p>In the <a href="#">OnGetCustomFields</a> method, assign the default values of any custom fields you want to the <a href="#">GetCustomFieldData</a>.<b>fields</b> property. The type of each default value controls the type of the property.</p> <p>The <i>Rename</i> dialog only supports certain types of variables for custom fields, so you must only assign properties of compatible types. Preferences supports:</p> <ul style="list-style-type: none"> <li>• Boolean options (<b>True</b> or <b>False</b>) - the variable type must be <i>bool</i></li> <li>• Numeric options - the variable type must be <i>int</i></li> <li>• String options - the variable type must be <i>string</i></li> <li>• Drop-down list - the variable type must be a <a href="#">Vector</a> with an <i>int</i> as the first element (to specify the default selection), and strings for the remaining elements.</li> </ul>

## Date

The **Date** object is provided to make it easier to deal with variables representing dates. It converts automatically to an ActiveX *VT\_DATE* value and so can function as a drop-in replacement for a scripting language's native date variables. The main advantage is that it retains milliseconds, unlike *VT\_DATE* which has a one second resolution. It also provides some utility methods to manipulate dates. The [Item](#) object has a number of properties that returns **Date** objects.

You can create a new **Date** object using the [DOpusFactory.Date](#) method.

Property Name	Return Type	Description
<default value>	<i>date</i>	Returns a <i>VT_DATE</i> representing the value of this <b>Date</b> object (excluding the milliseconds).
day	<i>int</i>	Get or set the <i>day</i> value of the date.
hour	<i>int</i>	Get or set the <i>hour</i> value of the date.
min	<i>int</i>	Get or set the <i>minute</i> value of the date.
month	<i>int</i>	Get or set the <i>month</i> value of the date.
ms	<i>int</i>	Get or set the <i>milliseconds</i> value of the date.
sec	<i>int</i>	Get or set the <i>seconds</i> value of the date.
wday	<i>int</i>	Get the <i>day-of-the-week</i> value of the date.
year	<i>int</i>	Get or set the <i>year</i> value of the date.

Method Name	Arguments	Return Type	Description
Add	<int:value> <string:type>	<i>none</i>	Adds the specified value to the date. The interpretation of the specified value is controlled by the <i>type</i> string:  <b>l</b> - milliseconds <b>s</b> - seconds <b>m</b> - minutes <b>h</b> - hours <b>d</b> - days <b>w</b> - weeks <b>M</b> - months <b>y</b> - years
Clone	<i>none</i>	<i>object:Date</i>	Returns a new <b>Date</b> object set to the same date as this one.
Compare	<date:other> [<string:type>] [<int:tolerance>]	<i>int</i>	Compares this date against the <i>other</i> date. The return value will be <b>0</b> (equal), <b>1</b> (greater) or <b>-1</b> (less).

			<p>The optional <i>type</i> string controls how the comparison is performed:</p> <p><b>s</b> - ignore seconds. If specified, the optional <i>tolerance</i> argument specifies the comparison tolerance in seconds.</p> <p><b>sD</b> - ignore seconds, and compensate automatically for daylight savings.</p> <p><b>t</b> - compare times only</p> <p><b>d</b> - compare dates only</p>
Format	[<string:format>] [<string:flags>]	<i>string</i>	<p>Returns a formatted date or time string. The <i>format</i> and <i>flags</i> arguments are both optional.</p> <p>If you do not give a <i>format</i>, the result will include both date and time, formatted the same as date-time columns in the file display.</p> <p>If you give a <i>format</i> of just "<b>d</b>" or "<b>t</b>" then the result will be just the date or time part, formatted the same as date or time columns in the file display.</p> <p>The file display's formats depend on the user's locale and Windows settings. You should use those options if you wish to present a date/time to the user in the way they expect them to look, but not if you need to store them in a specific format.</p> <p>When using the file display's format (that is, the <i>format</i> argument is empty, "<b>d</b>" or "<b>t</b>"), you can optionally pass one or more case-sensitive flags in the second <i>flags</i> argument to override a few settings:</p> <ul style="list-style-type: none"> <li>• <b>N</b> - Force day names on in dates within the last week. "Today", "Monday", etc.</li> <li>• <b>n</b> - Force day names off.</li> <li>• <b>S</b> - Force seconds on in times.</li> <li>• <b>s</b> - Force seconds off.</li> <li>• <b>M</b> - Force milliseconds on in times. (Milliseconds will be zero if the stored time does not have millisecond accuracy.)</li> <li>• <b>m</b> - Force milliseconds off.</li> <li>• <b>P</b> - Force time hours to be padded to two digits.</li> <li>• <b>p</b> - Do not force time hours to be padded.</li> </ul>

			<p>For example, to get just the date, using the user's locale, but with day names forced off:  <b>myDate.Format("d","n")</b>. To get the date and time, using the user's locale, but with day names forced on and seconds forced off:  <b>myDate.Format("", "Ns")</b>.</p> <p>The <i>format</i> can also use the syntax shown in <a href="#">Codes for date and time</a>, allowing for arbitrary formats. For example,  <b>myDate.Format("D#yyyy-MM-dd T#HH:mm:ss")</b> would return a string like <b>2016-07-28 15:45:26</b>.</p> <p>When explicitly specifying a format, the <i>flags</i> argument should not be used and will be ignored.</p>
FromUTC	<i>none</i>	<i>object:Date</i>	Returns a new <b>Date</b> object with the date converted from UTC (based on the local time zone).
Reset	<i>none</i>	<i>none</i>	Resets the date to the current local date/time.
Set	<date:newdate>	<i>none</i>	Sets the value of this <b>Date</b> object to the supplied date.
Sub	<int:value> <string:type>	<i>none</i>	Subtracts the specified value from the date. The parameters are the same as for the <b>Add</b> method.
ToUTC	<i>none</i>	<i>object:Date</i>	Returns a new <b>Date</b> object with the date converted to UTC (based on the local time zone).

## Dialog

The **Dialog** object allows you to display dialogs that prompt the user for confirmation, allow them to input text strings or passwords, and select checkbox options or choose from a drop-down list. You can also use this object to display a popup menu on screen.

You can create a **Dialog** object from the [DOpus.Dlg](#), [Lister.Dlg](#), [Tab.Dlg](#), [Func.Dlg](#) and [Command.Dlg](#) methods.

See the [Example Scripts](#) section for an example of its use.

There are two different ways to use the **Dialog** object. You can either:

- Use the one-shot methods (**Folder**, **GetString**, **Multi**, **Open**, **Request** or **Save**) to display a simple dialog of various types, or
- Configure the dialog first by setting the values of the various properties, and then call the **Show** function to display it. This method also lets you create and use [script dialogs](#).

The one-shot methods accept several parameters but are generally not as flexible as building up the dialog and then calling **Show**.

Property Name	Return Type	Description
buttons	<i>string</i>	<p>Specifies the buttons that are displayed at the bottom of the dialog. These buttons are used to close the dialog. The <b>Show</b> method returns a value indicating which button was chosen (and this value is also available in the <b>result</b> property).</p> <p>Multiple button strings must be separated with vertical bar characters ( ). If a button has more than one button then by definition the last one is the "cancel" button. For example:</p> <pre>dlg.buttons = "OK Retry Cancel"</pre> <p>To specify <i>accelerators</i> for the buttons prefix the desired key with an ampersand (&amp;) character. For example:</p> <pre>dlg.buttons = "&amp;OK &amp;Retry &amp;Cancel"</pre> <p>Buttons can also have drop-down menus attached</p>

		<p>to them, by separating the drop-down items with plus signs (+). For example:</p> <pre>dlg.buttons = "OK Retry+Retry All Cancel"</pre> <p>Within drop-down menus, you can specify that certain menu items can be accessed directly from the main button by holding Shift, Ctrl or Shift+Ctrl. This is done by adding an equals sign (=) and then the label the button should display when the key is held down (usually an abbreviated version of the menu item label, or a repetition of the label itself if it is already short enough). The keys are automatically assigned and you can only do this for at most three items. For example:</p> <pre>dlg.buttons = "OK Retry+Retry All=Retry All Skip+Skip if same modified time=Skip Same Time Cancel"</pre>
choices	<i>object: <a href="#">Vector</a>(string) or array(string)</i>	<p>This property uses either a <a href="#">Vector</a> or an array of strings to provide a list of multiple options that can be shown to the user. The list can be presented in one of three ways:</p> <ul style="list-style-type: none"> <li>• <b>Drop-down list:</b> By default, the dialog will display a drop-down list allowing the user to select one option. The index of the chosen selection is available via the <b>selection</b> property when the <b>Show</b> method returns.</li> <li>• <b>Checkbox list:</b> If the <b>list</b> property is also given the dialog will display a scrolling list of items, each with a checkbox allowing it to be turned on or off.</li> <li>• <b>Popup menu:</b> If the <b>menu</b> property is also given, a popup menu will be displayed at the current mouse coordinates. Use a single hyphen ("-") as a menu label to insert a separator.</li> </ul>
confirm	<i>bool</i>	<p>In a text entry dialog (i.e. the <b>max</b> property has been specified) setting <b>confirm</b> to <b>True</b> will require that the user types the entered text again (in a second text field) to confirm it (e.g. for a password).</p>
cx	<i>int</i>	<p>For <a href="#">script dialogs</a> marked as resizable, this property lets you override the width of the dialog defined in the resource - although note you can't resize a dialog smaller than its initial size.</p>
cy	<i>int</i>	<p>For <a href="#">script dialogs</a> marked as resizable, this property lets you override the height of the dialog</p>

		defined in the resource - although note you can't resize a dialog smaller than its initial size.
defvalue	<i>string</i>	In a text entry dialog (i.e. the <b>max</b> property has been specified) this property allows you to initialize the text field with a default value.  (Old scripts may use "default" instead of "defvalue"; this is deprecated because it does not work in JScript where "default" is a reserved keyword.)
defid	<i>int</i>	Allows you to change the default button (i.e. the action that will occur if the user hits enter) in the dialog. Normally the first button is the default - this has a <b>defid</b> of 1. The second button would have a <b>defid</b> of 2, and so on. If a dialog has more than one button then by definition the very last button is the "cancel" button, and this has a <b>defid</b> of 0.
detach	<i>bool</i>	Set to <b>True</b> if you want a <a href="#">script dialog</a> to run in "detached" mode, where your script provides its <a href="#">message loop</a> .
disable_window	<i>object: <a href="#">Lister</a> or object: <a href="#">Tab</a> or object: <b>Dialog</b> or int</i>	Use this to cause the dialog to automatically disable another window when it's displayed. The user will be unable to click or type in the disabled window until the dialog is closed. Normally if you use this you would set this to the same value as the <b>window</b> property.  You can provide either a <a href="#">Lister</a> or a <a href="#">Tab</a> object, or another <b>Dialog</b> . If you are showing this dialog in response to the <a href="#">OnAboutScript</a> event, you can also pass the value of the <a href="#">AboutData.window</a> property.
icon	<i>string</i>	Displays one of several standard icons in the top-left corner of the dialog, which can be used, for example, to indicate the severity of an error condition. The valid values for this property are <i>warning</i> , <i>error</i> , <i>info</i> and <i>question</i> .  When used with a <a href="#">script dialog</a> this property lets you control the icon shown in the dialog's title bar.
input	<i>string</i>	In a text entry dialog, this property returns the text string that the user entered (i.e. once the <b>Show</b> method has returned).
language	<i>string</i>	Set this property to create a <a href="#">script dialog</a> in a particular language (if one or more <a href="#">language overlays</a> have been provided), rather than the currently selected language.

list	<i>object: <a href="#">Vector</a>(bool) or <a href="#">array</a>(bool) or <i>int</i></i>	<p>In conjunction with the <b>choices</b> property, this will cause the choices to be presented as a checkbox list. You can initialize this <b>Vector</b> or array with the same number of items as the choices property, and set each one to <b>True</b> or <b>False</b> to control the default state of each checkbox. Or, simply set this value to <b>0</b> to activate the checkbox list without having to initialize the state of each checkbox.</p> <p>When the <b>Show</b> method returns, this property will return a <a href="#">Vector</a> of <i>bools</i> that provide the state of each checkbox as set by the user.</p>
max	<i>int</i>	<p>This property enables text entry in the dialog - a text field will be displayed allowing the user to enter a string. Set this property to the maximum length of the string you want the user to be able to enter (or <b>0</b> to have no limit).</p> <p>When the <b>Show</b> method returns the text the user entered will be available in the <b>input</b> property.</p>
menu	<i>object: <a href="#">Vector</a>(int) or <a href="#">array</a>(int) or <i>int</i></i>	<p>In conjunction with the <b>choices</b> property, this will cause the choices to be presented as a popup menu rather than in a dialog. The menu will be displayed at the current mouse coordinates.</p> <p>You can initialize this <b>Vector</b> or array with the same number of items as the choices property, and set each one to a value representing various flags that control the appearance of the menu item. The available flags are as follows - their values must be added together if you need to specify more than one flag per item.</p> <ul style="list-style-type: none"> <li><b>1</b> - bold (indicates the default item)</li> <li><b>2</b> - checked (a checkmark will appear next to the item)</li> <li><b>4</b> - radio (a radio button will appear next to the item)</li> <li><b>8</b> - disabled (the user will not be able to select the item)</li> </ul> <p>You can also simply set this value to <b>0</b> or <b>1</b> to activate the popup menu without having to provide flags for each item (if set to <b>1</b>, the top item in the menu will appear bolded).</p> <p>The <b>Show</b> method returns the index of the menu</p>

		item the user chose (with <b>1</b> being the first item), or <b>0</b> if the menu was cancelled.
message	<i>string</i>	Specifies the message text displayed in the dialog.
opacity	<i>int</i>	For <a href="#">script dialogs</a> this property retrieves or sets the current dialog opacity level, from <b>0</b> (totally transparent) to <b>255</b> (totally opaque).
options	<i>collection:</i> <a href="#">DialogOption</a>	<p>This is a collection of five options that will be displayed as checkboxes in the dialog. Unlike the <b>choices</b> / <b>list</b> scrolling checkbox list, these options are displayed as physical checkbox controls. By default the five checkboxes are uninitialized and won't be displayed, but if you assign a label to any of them they will be shown to the user.</p> <p>When the <b>Show</b> method returns you can obtain the state of the checkboxes using the <b>state</b> property of each <a href="#">DialogOption</a> object.</p>
password	<i>bool</i>	In a text entry dialog, set this property to <b>True</b> to make the text entry field a password field. In a password field the characters the user enters are not displayed.
position	<i>string</i>	<p>When used with a <a href="#">script dialog</a> this property lets you control the dialog's position on screen. Accepted values are:</p> <ul style="list-style-type: none"> <li><b>center</b> - center the dialog over the parent window (the default)</li> <li><b>absolute</b> - specify an absolute position using the <b>x</b> and <b>y</b> properties</li> <li><b>parent</b> - position relative to the parent window (using <b>x</b> and <b>y</b>)</li> <li><b>monitor</b> - position relative to the current monitor (using <b>x</b> and <b>y</b>)</li> </ul> <p>Except when set to "center" the <b>x</b> and <b>y</b> properties can be used to adjust the dialog's position.</p>
result	<i>int</i>	This property returns the index of the button chosen by the user to close the dialog. The left-most button is index <b>1</b> , the next button is index <b>2</b> , and so on. If a dialog has more than one button then by definition the last (right-most) button is the "cancel" button and so this will return index <b>0</b> .

		If any buttons have associated drop-down menus then the contents of the menus also contribute to the index value. For example, if button index 2 has an additional item in a drop-down menu, then that item would be index 3, and the next button would be index 4.
select	<i>bool</i>	In a text entry dialog, set this property to <b>True</b> to automatically select the contents of the input field (as specified by the <b>defvalue</b> property) when the dialog opens.
selection	<i>int</i>	In a drop-down list dialog (one with the <b>choices</b> property set without either <b>list</b> or <b>menu</b> ), this property returns the index of the item chosen from the drop-down list after the <b>Show</b> method returns.
sort	<i>bool</i>	Set this property to <b>True</b> if the list of choices given by the <b>choices</b> property should be sorted alphabetically.
template	<i>string</i>	Lets you create a <a href="#">script dialog</a> . The <b>template</b> property can be set to the name of the script dialog to display (as defined in your script resources), or a string that contains raw XML defining the dialog.
title	<i>string</i>	Specifies the title text of the dialog.
want_resize	<i>bool</i>	Set this property to <b>True</b> if you want the script dialog to generate <b>resize</b> events in your message loop when the user resizes the dialog.
window	<i>object:</i> <a href="#">Lister</a> or <i>object:</i> <a href="#">Tab</a> or <i>object:</i> <b>Dialog</b> or <i>int</i>	Use this to specify the parent window of the dialog. The dialog will appear centered over the top of the specified window. You can provide either a <a href="#">Lister</a> or a <a href="#">Tab</a> object, or another <b>Dialog</b> . If you are showing this dialog in response to the <a href="#">OnAboutScript</a> event, you can also pass the value of the <a href="#">AboutData.window</a> property.  You only need to set this property if you obtain the <b>Dialog</b> option from the <a href="#">DOpus.Dlg</a> method - if the <b>Dialog</b> object comes from one of the other objects (e.g. <a href="#">Tab.Dlg</a> ) then its parent window will be set automatically.
x	<i>int</i>	Specifies the x-position of a <a href="#">script dialog</a> . Use the <b>position</b> property to control how the position is interpreted. After the dialog has been displayed you can change this property to move the dialog around on-screen.
y	<i>int</i>	Specifies the y-position of a <a href="#">script dialog</a> . Use the <b>position</b> property to control how the position is interpreted. After the dialog has been

displayed you can change this property to move the dialog around on-screen.

Method Name	Arguments	Return Type	Description
AddHotkey	<string:name>  <string:key>	<i>none</i>	<p>Creates a hotkey (or keyboard accelerator) for the specified key combination. When the user presses this key combination in your dialog, a <b>hotkey</b> event will be triggered.</p> <p>The name parameter is a name you assign that lets you identify the hotkey. The key parameter specified the actual key combination; this can optionally combine the qualifiers <b>ctrl</b>, <b>shift</b> and <b>alt</b> with a character or name of a special key. For example, <b>ctrl+t</b> or <b>alt+shift+F7</b>.</p>
Create	<i>none</i>	<i>none</i>	<p>When creating a <a href="#">script dialog</a>, calling this method creates the underlying dialog but does not display it. This lets you create the dialog and then initialize its controls before it is shown to the user. Using the <b>Create</b> method implies a <a href="#">detached dialog</a> - that is, the <b>detach</b> property will be implicitly set to <b>True</b>.</p> <p>Once the dialog has been created and its controls initialized, you can call <b>Show</b> to make it visible to the user. It will also go visible at the first <b>GetMsg</b> call if it hasn't already been shown.</p>
Control	<string:name> [<string:dialog>] [<string:tab>]	<i>object:</i> <a href="#">Control</a>	<p>Returns a <a href="#">Control</a> object corresponding to one of the controls on a script dialog. The control is identified by its <i>name</i>, as defined in the script dialog resource.</p> <p>The optional second and third parameters are only used when the control is in a <i>tab control</i> (that is, when it's in a dialog that's a child of another dialog). The <i>dialog</i> parameter specifies the name of its parent dialog. The <i>tab</i> parameter specifies the name of the tab control hosting the child dialog. You would only need to specify the</p>

			name of the tab if you have multiple tab controls and the same dialog is hosted inside more than one of them (this would be quite a rare occurrence).
DelHotkey	<string:name>	<i>none</i>	Deletes a hotkey you previously created with the <b>AddHotkey</b> method.
EndDlg	<int:result>	<i>none</i>	Ends a <a href="#">script dialog</a> running in detached mode. Normally dialogs end automatically when the user clicks the close button or another button that has its <b>Close Dialog</b> property set to <b>True</b> . This method lets you end a dialog under script control. The optional parameter specifies the result code that the <b>Dialog.result</b> property will return.
Folder	<string:title> <string:default> <bool:expand> <object>window>	<i>object:Path</i>	Displays a "Browse for Folder" dialog letting the user select a folder. The optional parameters are:  <i>title</i> - specify title of the dialog <i>default</i> - specify the default path selected in the dialog <i>expand</i> - specify <b>True</b> to automatically expand the initial path <i>window</i> - specify parent window for the dialog (a <a href="#">Lister</a> or a <a href="#">Tab</a> ). You can also set the window property of the <b>Dialog</b> object before calling this method.  A <a href="#">Path</a> object is returned to indicate the folder chosen by the user. This object will have an additional <b>result</b> property that will be <b>False</b> if the user cancelled the dialog - the other normal <b>Path</b> properties will only be valid if <b>result</b> is <b>True</b> .
GetMsg	<i>none</i>	<i>object:Msg</i>	Returns a <a href="#">Msg</a> object representing the most recent input event in a <a href="#">script dialog</a> (only used in <a href="#">detached mode</a> ). The return value will evaluate to <b>False</b> when the dialog is closed, which is when you should exit your <a href="#">message loop</a> .
GetString	<string:message> <string:default> <string:max> <string:buttons> <string:title> <object>window> <byref string:result>	<i>string</i>	Displays a text entry dialog allowing the user to enter a string. The optional parameters are:  <i>message</i> - specify message string in the dialog <i>default</i> - specify default string value <i>max</i> - specify maximum string length <i>buttons</i> - specify button labels (in the same

			<p>format as the <b>buttons</b> property described above)</p> <p><i>title</i> - specify dialog window title</p> <p><i>window</i> - specify parent window for the dialog (a <a href="#">Lister</a> or a <a href="#">Tab</a>). You can also set the window property of the <b>Dialog</b> object before calling this method.</p> <p><i>result</i> - for scripting languages that support <i>ByRef</i> parameters, this can specify a variable to receive the string the user enters.</p> <p>The return value is the entered string, or an empty value if the dialog was cancelled. The index of the button selected by the user will be available via the <b>result</b> property once this method returns. The left-most button is index <b>1</b>, the next button is index <b>2</b>, and so on. If a dialog has more than one button then by definition the last (right-most) button is the "cancel" button and so this will return index <b>0</b>.</p>
KillTimer	<string:name>	<i>none</i>	Stops the specified timer. The timer must previously have been created by a call to the <b>SetTimer</b> method.
LoadPosition	<string:id> <string:type>	<i>none</i>	<p>Restores the previously saved position of a <a href="#">script dialog</a>. The position must have previously been saved by a call to the <b>SavePosition</b> method.</p> <p>The <b>id</b> string is a string that Opus can use to identify your dialog or the script it comes from. The template name of the dialog will be automatically appended to this. For example, you might specify <b>id</b> as "kundal" - Opus would then internally save the position of a dialog called "dialog1" as "kundal!dialog1". Make sure you pick a string that other script authors are unlikely to use as Opus has no other way of telling the saved positions apart.</p> <p>The optional type parameter lets you control which position elements are restored - specify "pos" to only restore the position, "size" to only restore the size, or "pos,size" to restore both (this is also the</p>

			default, so you can also omit the argument all together).
Multi	<string:title> <string:default> <object>window>	collection: <a href="#">Item</a>	<p>Displays a "Browse to Open File" dialog that lets the user select one or more files. The optional parameters are:</p> <p><i>title</i> - specify title of the dialog  <i>default</i> - specify the default file selected in the dialog (if a folder is specified this specifies the default location but no file will be selected)  <i>window</i> - specify parent window for the dialog (a <a href="#">Lister</a> or a <a href="#">Tab</a>). You can also set the window property of the <b>Dialog</b> object before calling this method.</p> <p>A collection of <a href="#">Item</a> objects is returned to indicate the files selected by the user. The returned object will have a <b>result</b> property that you should check first - the collection of items is only valid if <b>result</b> returns <b>True</b>. If it returns <b>False</b> it means the user cancelled the dialog.</p>
Open	<string:title> <string:default> <object>window>	object: <a href="#">Item</a>	<p>Displays a "Browse to Open File" dialog that lets the user select a single file. The optional parameters are:</p> <p><i>title</i> - specify title of the dialog  <i>default</i> - specify the default file selected in the dialog (if a folder is specified this specifies the default location but no file will be selected)  <i>window</i> - specify parent window for the dialog (a <a href="#">Lister</a> or a <a href="#">Tab</a>). You can also set the window property of the <b>Dialog</b> object before calling this method.</p> <p>A single <a href="#">Item</a> object is returned to indicate the file selected by the user. This object will have an additional <b>result</b> property that will be <b>False</b> if the user cancelled the dialog - the other normal <b>Item</b> properties will only be valid if <b>result</b> is <b>True</b>.</p>
Request	<string:message> <string:buttons> <string:title> <object>window>	<i>int</i>	<p>Displays a dialog with one or more buttons. The optional parameters are:</p> <p><i>message</i> - specify message string in the dialog</p>

			<p><i>buttons</i> - specify button labels (in the same format as the <b>buttons</b> property described above)</p> <p><i>title</i> - specify dialog window title</p> <p><i>window</i> - specify parent window for the dialog (a <a href="#">Lister</a> or a <a href="#">Tab</a>). You can also set the window property of the <b>Dialog</b> object before calling this method.</p> <p>The return value is the index of the button selected by the user, and this is also available in the <b>result</b> property once the method returns. The left-most button is index <b>1</b>, the next button is index <b>2</b>, and so on. If a dialog has more than one button then by definition the last (right-most) button is the "cancel" button and so this will return index <b>0</b>.</p>
RunDlg	<i>none</i>	<i>int</i>	<p>Turns a previously <a href="#">detached dialog</a> into a non-detached one, by taking over and running the default message loop. The <b>RunDlg</b> method won't return until the dialog has closed. You might use this if you created a dialog using the <b>Create</b> method in order to initialize its controls, but don't actually want to run an interactive message loop.</p> <p>The return value is the same available in the <b>result</b> property, and represents the index of the close button selected by the user.</p>
SavePosition	<string:id>	<i>none</i>	<p>Saves the position (and size) of the dialog to your Opus configuration. The position can then be restored later on by a call to <b>LoadPosition</b>.</p> <p>Normally you would call <b>LoadPosition</b> before displaying your dialog, and <b>SavePosition</b> after the dialog has been closed.</p> <p>The <b>id</b> string is a string that Opus can use to identify your dialog or the script it</p>

			comes from. The template name of the dialog will be automatically appended to this. For example, you might specify <b>id</b> as "kundal" - Opus would then internally save the position of a dialog called "dialog1" as "kundal!dialog1". Make sure you pick a string that other script authors are unlikely to use as Opus has no other way of telling the saved positions apart.
SetTimer	<int:period>  <string:name>	string	Creates a timer that will generate a periodic <b>timer</b> event for your script. The <b>period</b> must be specified in milliseconds (e.g. 1000 would equal one second).  You can optionally specify a <b>name</b> for the timer - if you don't provide a name, one will be generated automatically (and the name of the new timer will be returned).
Show	none	int	Displays the dialog that has been pre-configured using the various properties of this object. See the properties section above for a full description of these.  The return value is the index of the button selected by the user, and this is also available in the <b>result</b> property once the method returns. The left-most button is index <b>1</b> , the next button is index <b>2</b> , and so on. If a dialog has more than one button then by definition the last (right-most) button is the "cancel" button and so this will return index <b>0</b> .
Save	<string:title> <string:default> <object>window> <string:type>	object: <a href="#">Path</a>	Displays a "Browse to Save File" dialog that lets the user select a single file or enter a new filename to save. The optional parameters are:  <i>title</i> - specify title of the dialog <i>default</i> - specify the default file selected in the dialog (if a folder is specified this specifies the default location but no file will be selected)  <i>type</i> - specify a list of filetypes to populate the "Save as Type" dropdown in the save dialog.

			<p><i>window</i> - specify parent window for the dialog (a <a href="#">Lister</a> or a <a href="#">Tab</a>). You can also set the window property of the <b>Dialog</b> object before calling this method.</p> <p>The optional <i>type</i> parameter consists of one or more pairs of strings, separated by exclamation marks (!). The first string of each pair is the plain text string shown in the drop-down, and the second string of each pair is the actual file extension. You can also specify multiple extensions for the one type by separating them with semicolon. If you want the default "All files" item to be added to the list, add a # character at the start of the string. For example, <i>#Text Files!*.txt!Doc Files!*.doc</i>.</p> <p>A <a href="#">Path</a> object is returned to indicate the file chosen by the user. This object will have an additional <b>result</b> property that will be <b>False</b> if the user cancelled the dialog - the other normal <b>Path</b> properties will only be valid if <b>result</b> is <b>True</b>.</p>
Vars	<string:id>	<i>object:</i> <a href="#">Vars</a>	<p>Returns a <a href="#">Vars</a> object that represents the variables that are scoped to this particular dialog. This allows scripts to use variables that persist from one use of the dialog to another.</p> <p>The <b>id</b> string is a string that Opus can use to identify your dialog or the script it comes from. The template name of the dialog will be automatically appended to this. For example, you might specify <b>id</b> as <i>"kundal"</i> - Opus would then internally save these variables for a dialog called <i>"dialog1"</i> as <i>"kundal!dialog1"</i>. Make sure you pick a string that other script authors are unlikely to use as Opus has no other way of telling the saved variables apart.</p>

## DialogListColumn

The **DialogListColumn** object represents a column in a *Details mode list view* control in a [script dialog](#). Use the [Control.columns](#) property to obtain a [DialogListColumns](#) object, and then enumerate it to obtain individual **DialogListColumn** objects.

Property Name	Return Type	Description
name	<i>string</i>	Returns or sets the column's name.
resize	<i>bool</i>	Set this property to <b>True</b> if you want this column to automatically resize when the list view is resized horizontally. Only one column can be set to auto-resize at a time.
sort	<i>int</i>	Returns <b>1</b> if the list view is currently sorted forwards by this column, <b>-1</b> if it's currently sorted backwards by this column, or <b>0</b> otherwise. Settings this property will re-sort the list.
width	<i>int</i>	Returns or sets the column's width in pixels. Set it to <b>-1</b> to automatically size the column to fit its content. You can automatically resize all columns at once using the <a href="#">DialogListColumns.AutoSize</a> method.

## DialogListColumns

The **DialogListColumns** object lets you query or modify the columns in a *Details mode list view* control in a [script dialog](#). Use the [Control.columns](#) property to obtain a **DialogListColumns** object.

You can enumerate this object to query the current columns. Each column is represented by a [DialogListColumn](#) object.

Method Name	Arguments	Return Type	Description
AddColumn	<string:name>	<i>int</i>	Adds a new column to the list view, and returns the index of the new column.
AutoSize	<i>none</i>	<i>none</i>	Automatically sizes all columns in the list view to fit their content.
DeleteColumn	<int:index>	<i>none</i>	Deletes the specified column.

GetColumnAt	<int:index>	<i>object:</i> <a href="#">DialogListColumn</a>	Returns a <a href="#">DialogListColumn</a> object representing the column in the specified position.
InsertColumn	<string:name> <int:position>	<i>int</i>	Inserts a new column in the list view at the specified position, and returns the index of the new column.

## DialogListItem

The **DialogListItem** object represents an item in a *combo box* or *list box* control in a [script dialog](#). It's returned by the [Control.GetItemAt](#) and [Control.GetItemByName](#) methods.

Property Name	Return Type	Description
checked	<i>int</i>	For a <i>list view</i> control with checkboxes enabled, returns or sets the check state of the item.  Check states are <b>0</b> (unchecked), <b>1</b> (checked), <b>2</b> (indeterminate), <b>3</b> (unchecked/disabled), <b>4</b> (checked/disabled), <b>5</b> (indeterminate/disabled).
data	<i>int</i>	Returns or sets the optional data value associated with this item.
icon	<i>string</i>	For a <i>list view</i> control, returns or sets the icon associated with this item. You can specify the path of a file or folder to use its icon, or a file extension (e.g. ".txt") to use a generic filetype icon. You can also extract an icon from a DLL or EXE by providing the path of the file followed by a comma and then the icon index within the file.
index	<i>index</i>	Returns the 0-based index of this item within the control.  For a combo edit box, this will return <b>-1</b> if the user typed in a string rather than selecting one from the list. The string they entered can be retrieved from the <b>name</b> property.
name	<i>string</i>	Returns or sets the item's name.
selected	<i>bool</i>	Returns or sets the item's selection state. Mostly useful with multiple-selection <i>list box</i> controls.
subitems	<i>collection:string</i>	For a list view control in <i>Details</i> mode, returns a collection of strings that lets you query or change the text of the item's sub-items. There will be one string in

		the collection for each column in the list, excluding the first column.
--	--	-------------------------------------------------------------------------

### ***DialogOption***

The **DialogOption** object is used with the options property of the [Dialog](#) object. A collection of five of these is provided, and any you initialize will be displayed to the user as physical checkbox controls.

Property Name	Return Type	Description
label	<i>string</i>	Set this to the desired label of the checkbox.
state	<i>bool</i>	Set this to the desired initial state of the checkbox. When the <b>Dialog.Show</b> method returns, you can read this property to find out the state the user chose.

### ***DisplayModeChangeData***

If a [script add-in](#) implements the [OnDisplayModeChange](#) event, the method receives a **DisplayModeChangeData** object when the [display mode](#) is changed in a tab.

Property Name	Return Type	Description
mode	<i>string</i>	Returns a <i>string</i> indicating the new display mode. Will be one of <i>largeicons</i> , <i>smallicons</i> , <i>list</i> , <i>details</i> , <i>power</i> , <i>thumbnails</i> or <i>tiles</i> .
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
tab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the tab the display mode changed in.

## **Dock**

A collection of **Dock** objects is accessed from the [Toolbar.docks](#) property. It corresponds to a currently open floating toolbar.

<b>Property Name</b>	<b>Return Type</b>	<b>Description</b>
<i>&lt;default value&gt;</i>	<i>int</i>	This is a handle to the window of the floating toolbar. It is not particularly useful.

## **DocMeta**

The **DocMeta** object is retrieved from the [Metadata.doc](#) or [Metadata.doc\\_text](#) properties. It provides access to metadata relating to document files.

<b>Property Name</b>	<b>Return Type</b>	<b>Description</b>
<i>&lt;column keyword&gt;</i>	<i>variant</i>	Returns the value of the specified column, as listed in the <i>Documents</i> section of the <a href="#">Keywords for Columns</a> page.

## DOpus

The **DOpus** object is one of the two global script objects provided by Opus, and is available to *all* scripts. It provides various helper methods, and collections that let you access things like Listers and toolbars.

Property Name	Return Type	Description
aliases	<i>object:</i> <a href="#">Aliases</a>	The <a href="#">Aliases</a> object gives the script access to the defined <a href="#">folder aliases</a> .
favoriteformats	<i>collection:</i> <a href="#">Format</a>	Returns a collection of <a href="#">Format</a> objects representing the used-defined favorite formats.
favorites	<i>object:</i> <a href="#">Favorites</a>	Returns a <a href="#">Favorites</a> object which lets you query and modify the user-defined favorite folders.
filters	<i>object:</i> <a href="#">Filters</a>	Returns a <a href="#">Filters</a> object which lets you access information about the global filter settings (configured on the <a href="#">Folders / Global Filters</a> page in Preferences).
language	<i>string</i>	Returns a string representing the current user interface language.
listers	<i>object:</i> <a href="#">Listers</a>	Returns a <a href="#">Listers</a> object which represents any currently open Lister windows (each one is represented by a <a href="#">Lister</a> object).
smartfavorites	<i>object:</i> <a href="#">SmartFavorites</a>	Returns a <a href="#">SmartFavorites</a> object which lets you query the <a href="#">SmartFavorites</a> data.
strings	<i>object:</i> <a href="#">ScriptStrings</a>	Returns a <a href="#">ScriptStrings</a> object which lets your script access any strings defined as <a href="#">string resources</a> .
vars	<i>object:</i> <a href="#">Vars</a>	This <a href="#">Vars</a> object represents all defined variables with <i>global scope</i> .
version	<i>object:</i> <a href="#">Version</a>	The <a href="#">Version</a> object provides information about the current Opus program version.
viewers	<i>object:</i> <a href="#">Viewers</a>	Returns a <b>Viewers</b> object which represents any currently open <a href="#">standalone image viewers</a> (each one is represented by a <a href="#">Viewer</a> object).

Method Name	Arguments	Return Type	Description
ClearOutput	<i>none</i>	<i>none</i>	Clears the script output log.
Create	<i>none</i>	<i>object:</i> <a href="#">DOpusFactory</a>	Creates and returns a new <a href="#">DOpusFactory</a> object, which can be used to create various

			lightweight helper objects like <a href="#">Blob</a> , <a href="#">Map</a> and <a href="#">Vector</a> .
Delay	<int:time>	<i>none</i>	Delays for the specified number of milliseconds before returning.
Dlg	<i>none</i>	<i>object:</i> <a href="#">Dialog</a>	Creates a new <a href="#">Dialog</a> object, that lets you display dialogs and popup menus.
DPI	<i>none</i>	<i>object:</i> <a href="#">DPI</a>	Creates the <a href="#">DPI</a> helper object which assists when dealing with different system scaling settings (e.g. high-DPI monitors).
FSUtil	<i>none</i>	<i>object:</i> <a href="#">FSUtil</a>	Creates a new <a href="#">FSUtil</a> object, that provides helper methods for accessing the file system.
GetClip	<i>none</i> or <string:type>	<i>string</i> or <i>collection:</i> <a href="#">Item</a>	Retrieves the current contents of the system clipboard, if it contains either text or files.  You can control the returned type by passing either " <b>text</b> " or " <b>files</b> " for the <type> argument - Opus will convert to the requested type if possible.  If <type> is not specified the contents will be returned in their native format.
GetClipFormat	<i>none</i>	<i>string</i>	Returns a string indicating the native format of the clipboard contents - " <b>text</b> ", " <b>files</b> " or an empty string in any other case.
GetQualifiers	<i>none</i>	<i>string</i>	Returns a string indicating which qualifier keys are currently held down. If none are held down, the string will be " <b>none</b> ". Otherwise, the string can contain any or all of the following, separated by commas: " <b>shift</b> ", " <b>ctrl</b> ", " <b>alt</b> ", " <b>lwin</b> ", " <b>rwin</b> ".  Note that many events pass you a similar list of qualifiers. If you are passed a list of qualifiers, you should generally use that list rather than call <code>DOpus.GetQualifiers</code> .

			<p>For example, script commands are passed a <a href="#">Func</a> object with a <i>qualifiers</i> property. That property will tell you which keys were held down when the command was triggered, and that may be different to the keys held down a few seconds later. When the user clicks a button to run a command, they normally expect the command to use the keys they held when they clicked, not the keys they are touching later while waiting for it to finish.</p> <p>Similarly, events like <a href="#">OnBeforeFolderChange</a> will often pass you an object like <a href="#">BeforeFolderChangeData</a> containing a <i>qualifiers</i> property which indicates key state when the event was triggered. You should normally use that instead of calling <code>DOpus.GetQualifiers</code>.</p> <p>If you do call <code>DOpus.GetQualifiers</code>, you would normally want to call it as soon as possible and then store the result, so there is less time for the user to let go of a key after triggering your script.</p> <p>If you call <code>DOpus.GetQualifiers</code> more than once, you may get a different result each time, due to keys being pushed or released between calls. Call it once and store the result if you need to do multiple checks and need them to be consistent. This does not generally affect the <i>qualifiers</i> properties mentioned earlier, since they are usually stored snapshots of the key state.</p>
Output	<string:text> [<bool:error>] [<bool:timestamp>]	none	Prints the specified text string to the script output log (found in the <a href="#">Utility Panel</a> , the <a href="#">CLI</a> in script mode, the <i>Rename</i> dialog

			<p>and the <i>Command Editor</i> in script mode).</p> <p>If the second argument is provided and set to <b>True</b>, the message will be displayed as an error. This means the text will be displayed in red and if no log windows are currently open, a warning icon will flash in the Lister status bar to alert the user of an error condition.</p> <p>If the optional third argument is provided and set to <b>True</b> then the log message will have a timestamp prepended to it. Timestamps only appear in the utility panel, not in places like the Command Editor's output panel. Error messages always get timestamps so if the second argument is <b>True</b> then the third is ignored</p>
ReloadScript	<string:file>	<i>none</i>	Causes Opus to reload and reinitialize the specified script. You must provide the full pathname of the script on disk (if a script add-in wants to reload itself you can pass the value of the <a href="#">Script.file</a> property).
SetClip	<string:text> or <i>collection</i> : <a href="#">Item</a> or <i>none</i>	<i>none</i>	Places the specified text, or <a href="#">Item</a> collection (or <a href="#">Vector</a> of <a href="#">Item</a> objects) on the system clipboard. If called with no arguments the clipboard will be cleared.
Toolbars	<string:type>	object: <a href="#">Toolbars</a>	<p>Returns a <a href="#">Toolbars</a> object which lets you enumerate all defined toolbars (whether they are currently open or not).</p> <p>You can restrict this object to only return in-use toolbars by</p>

			specifying the optional <i>type</i> parameter - specify " <b>listers</b> " to only return toolbars currently turned on in a Lister, and " <b>docks</b> " to only return toolbars that are currently floating.
TypeOf	<i>any</i>	<i>string</i>	Returns a string indicating the type of an object or variable.

## ***DOpusFactory***

The **DOpusFactory** object is a helper object that you can use to create various other objects. Unlike the objects that represent existing *things* (e.g. [Lister](#) or [Tab](#)), the objects created by **DOpusFactory** are independent objects that you can instantiate whenever you need their functionality. The **DOpusFactory** object is obtained via the [DOpus.Create](#) method.

Method Name	Arguments	Return Type	Description
Blob	<i>none</i> or <int:size> or <byte, byte, ...> or < <a href="#">Blob</a> :source>	<i>object:</i> <a href="#">Blob</a>	Returns a new <a href="#">Blob</a> object, that lets you access and manipulate a chunk of binary data from a script. If no parameters are given the new <b>Blob</b> will be empty - you can set its size using the <b>resize</b> method - otherwise you can specify the initial size as a parameter.  You can also create a <b>Blob</b> pre-filled with data by specifying the actual byte values (e.g. <i>Blob(72,69,76,76,79)</i> ).  If another <b>Blob</b> (or an array - see the documentation on the <b>Blob</b> object for a discussion of this) is given then the new <b>Blob</b> will be created as a copy of the existing one.
BusyIndicator or	<i>none</i>	<i>object:</i> <a href="#">BusyIndicator</a> or	Creates a new <a href="#">BusyIndicator</a> object, that lets you control the breadcrumbs bar busy indicator from your script.
Command	<i>none</i>	<i>object:</i> <a href="#">Command</a>	Creates a new <a href="#">Command</a> object, that lets you run Opus commands from a script.
Date	<i>none</i> or <variant:date>	<i>object:</i> <a href="#">Date</a>	Creates a new <a href="#">Date</a> object. If an existing <b>Date</b> object or date value is specified the new object will be initialized to that value, otherwise the date will be set to the current local time.

Map	<i>none</i> or <variant:key>,  <variant:value >...	<i>object:</i> <a href="#">Map</a>	Creates a new <a href="#">Map</a> object. If no arguments are provided, the <b>Map</b> will be empty. Otherwise, the <b>Map</b> will be pre-initialized with the supplied key/value pairs. For example: <i>Map("firstname","fred","lastname","blogs");</i> . The individual keys and values can be different types.
StringSet	<i>none</i> or <string>, ...	<i>object:</i> <a href="#">StringSet</a>	Creates a new case-sensitive <a href="#">StringSet</a> object. If no arguments are provided, the <b>StringSet</b> will be empty. Otherwise it will be pre-initialized with the supplied strings; for example: <i>StringSet("dog","cat","pony");</i>  You can also pass an array of strings or <a href="#">Vector</a> object to initialise the set.
StringSetI	<i>none</i> or <string>, ...	<i>object:</i> <a href="#">StringSet</a>	Creates a new case-insensitive <a href="#">StringSet</a> object. If no arguments are provided, the <b>StringSet</b> will be empty. Otherwise it will be pre-initialized with the supplied strings.
StringTools	<i>none</i>	<i>object:</i> <a href="#">StringTools</a>	Creates a new <a href="#">StringTools</a> object, that provides helper functions for string encoding and decoding.
Vector	<i>none</i> or <int:elements> or <i>variants...</i>	<i>object:</i> <a href="#">Vector</a>	Creates a new <a href="#">Vector</a> object. If no arguments are provided, the <b>Vector</b> will be empty.  If a single integer argument is provided, the <b>Vector</b> will be pre-initialized to that number of elements.  If more than one argument is provided, the <b>Vector</b> will be pre-initialized with those elements; for example: <i>Vector("dog","cat","horse");</i> The individual elements can be different types.

### **DoubleClickData**

If a [script add-in](#) implements the [OnDoubleClick](#) event, the method receives a **DoubleClickData** object when the user double-clicks a file or folder.

Property Name	Return Type	Description
call	<i>bool</i>	Set this property to <b>False</b> to prevent the <a href="#">OnDoubleClick</a> event being called for any further files during this operation (this is only effective if more than one file was double-clicked). Any remaining files will be opened according to their default handlers.
cont	<i>bool</i>	Set this property to <b>False</b> to abort double-click processing altogether on any further files during this operation (this is only effective if more than one file was double-clicked).
early	<i>bool</i>	Returns <b>True</b> if your <a href="#">OnDoubleClick</a> event is being called with only a path (via the <b>path</b> property) and not a full <a href="#">Item</a> object. This will occur if you set the <a href="#">ScriptInitData.early_dbclick</a> property to <b>True</b> when initialising your script.  When early is <b>True</b> , you can set the <b>skipfull</b> to <b>True</b> to prevent the second call with a full <b>Item</b> object.
is_dir	<i>bool</i>	Returns <b>True</b> if the item double-clicked is a directory, <b>False</b> if it's a file.
item	<i>object:Item</i>	Returns a <a href="#">Item</a> object representing the item that was double-clicked. This property is only present if the <b>early</b> property is <b>False</b> .
mouse	<i>string</i>	Returns a string that indicates the mouse button that launched the double-click. The string can be one of the following: <i>left, middle, none</i> .
multiple	<i>bool</i>	This is set to <b>True</b> if multiple files were double-clicked.
path	<i>object:Path</i>	Returns a <a href="#">Path</a> object providing the full pathname of the item that was double-clicked.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift, ctrl, alt, lwin, rwin</i> .
skipfull	<i>bool</i>	When the early property is <b>True</b> , set <b>skipfull</b> to <b>True</b> to prevent your <a href="#">OnDoubleClick</a> event from being called a second time.
tab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the tab that the item was double-clicked in.

## DPI

The **DPI** object is a helper object that provides a number of methods and properties relating to the system DPI setting. For example, you can use it to convert a pixel width into one scaled for the current system DPI. The **DPI** object is returned via the [DOpus.DPI](#) property.

Property Name	Return Type	Description
dpi	<i>int</i>	Returns the system DPI setting as a “dpi value” (e.g. 96, 192).
factor	<i>int</i>	Returns the DPI settings as a “scale factor” (e.g. 100, 125, 200).

Method Name	Arguments	Return Type	Description
Divide	<int:value>	<i>int</i>	Divides the provided size by the system DPI; e.g. if the system DPI was set to 150%, <b>DPI.Divide(60)</b> would return <b>40</b> .
Scale	<int:value>	<i>int</i>	Scales the provided size by the system DPI; e.g. if the system DPI was set to 200%, <b>DPI.Scale(75)</b> would return <b>150</b> .

## Drive

The **Drive** object provides information about a drive (hard drive, CD ROM, etc) on your system. You can obtain a [Vector](#) of **Drive** objects, one for each drive on your system, from the [FSUtil.Drives](#) method.

Property Name	Return Type	Description
<i>default value</i>	<i>string</i>	Returns the root of the drive (e.g. <b>C:\</b> ).
avail	<i>object:FileSize</i>	Returns a <a href="#">FileSize</a> object indicating the available free space on the drive.
bpc	<i>int</i>	Returns the bytes-per-cluster value for the drive.
fileys	<i>string</i>	Returns a string representing the filesystem type.
flags	<i>int</i>	Returns a value representing filesystem flags for the drive.
free	<i>object:FileSize</i>	Returns a <a href="#">FileSize</a> object indicating the total free space on the drive.
label	<i>string</i>	Returns the drive's label.
total	<i>object:FileSize</i>	Returns a <a href="#">FileSize</a> object indicating the total size of the drive.
type	<i>string</i>	Returns a string indicating the drive type (removable, fixed, remote, cdrom, ramdisk).

## ExeMeta

The **ExeMeta** object is retrieved from the [Metadata.exe](#) or [Metadata.exe\\_text](#) properties. It provides access to metadata relating to executable (program) files.

Property Name	Return Type	Description
<column keyword>	variant	Returns the value of the specified column, as listed in the <i>Programs</i> section of the <a href="#">Keywords for Columns</a> page.

## Favorite

A **Favorite** object represents a [favorite folder](#). It is retrieved by enumerating or indexing the [Favorites](#) object.

Property Name	Return Type	Description
<default value>	string	Returns the name of the favorite folder or sub-folder.
folder	bool	Returns <b>True</b> if this is a sub-folder, <b>False</b> if it's a favorite folder or separator.  If this object is a sub-folder it also behaves like a <a href="#">Favorites</a> object as well as a <b>Favorite</b> object, and can be enumerated and have elements added and removed from it.
separator	bool	Returns <b>True</b> if this is a separator.
path	object: <a href="#">Path</a>	Returns the path this favorite folder refers to as a <a href="#">Path</a> object.

Method Name	Arguments	Return Type	Description
SetName	<string:name>	none	Changes the name of this favorite folder. Note that changes you make to the list are not saved until you call the <a href="#">Favorites.Save</a> method.

SetPath	<string:path> or <object:Path>	none	Changes the path this favorite folder refers to. Note that changes you make to the list are not saved until you call the <a href="#">Favorites.Save</a> method.
---------	--------------------------------------	------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------

## Favorites

The **Favorites** object holds a collection of all the defined [favorite folders](#). It is retrieved from the [DOpus.favorites](#) property.

Property Name	Return Type	Description
<default value>	collection: <a href="#">Favorite</a>	You can enumerate the <b>Favorites</b> object to retrieve individual <a href="#">Favorite</a> objects.

Method Name	Arguments	Return Type	Description
Add	<string:typeOrName> <string:path> <int:insertpos> or <object:Favorite>	object: <a href="#">Favorite</a> or object: <b>Favorites</b>	<p>Adds a new favorite folder to the favorites list. Note that changes you make to the list are not saved until you call the <b>Save</b> method.</p> <p>This method performs three separate functions; it can add a separator, a sub-folder or a favorite folder.</p> <ul style="list-style-type: none"> <li>To add a separator, the parameters should be the type string <b>sep</b>, optionally followed by the insertion position (see below).  For example, <b>Favorites.Add("sep");</b></li> <li>To add a folder, the first parameter should be the string <b>folder:</b> followed by the name of the folder (as a single parameter), optionally followed by the insertion position.  For example, <b>Favorites.Add("folder:Picture Locations");</b></li> <li>To add a new favorite, the first parameter can optionally be the name of the favorite, and the second parameter can</li> </ul>

			<p>be the path of the folder to add, or the name can be omitted and only the path can be provided. In either case you can optionally include the insertion position as the last parameter.</p> <p>For example, <b>Favorites.Add("myfave", "c:\folder\path");</b> or <b>Favorites.Add("c:\folder\path");</b></p> <p>In all three cases the new item is added to the end by default, but you can optionally specify a position to insert the item somewhere else. E.g. specifying <b>0</b> for the insertion position would add it at the top of the list. You can provide either a number or another <a href="#">Favorite</a> object.</p> <p>For example, <b>Favorites.Add("myfave", "c:\folder\path", 0);</b></p> <p>The return value is either a <a href="#">Favorite</a> or a <b>Favorites</b> object (depending on whether you added a sub-folder or a favorite folder).</p>
Delete	<object: <a href="#">Favorite</a> > or <object: <b>Favorites</b> >	<i>none</i>	Deletes the specified favorite or sub-folder. Note that changes you make to the list are not saved until you call the <b>Save</b> method.
Find	<string:name> <int:index>	<i>object:<b>Favorites</b></i>	<p>Lets you locate a sub-folder one or more levels below the current one. The <b>name</b> parameter is the name or path and name of the sub-folder to look for (e.g. "myfave", "pictures/local", etc).</p> <p>The optional <b>index</b> parameter lets you handle the case when there might be more than one sub-folder with the same name. <b>Favorites.Find("pictures", 1);</b> would find the second sub-folder called "pictures" below the current level.</p>
Save	<i>none</i>	<i>none</i>	Saves any changes you've made to the favorites list. Once you call this method changes you have made will be reflected in Preferences and the favorites list in Listers. Note that you can only call this method on the main "root" <b>Favorites</b>

			object obtained from the <a href="#">DOpus.favorites</a> property
SetName	<string:name>	<i>none</i>	Changes the name of this sub-folder. Note that changes you make to the list are not saved until you call the <b>Save</b> method. You can only call this method on <b>Favorites</b> objects that refer to sub-folders, and not the main "root" folder.

## File

The **File** object lets you read and write binary data from and to a file on disk (or in a Zip file, FTP site, etc). While the Microsoft *Scripting.FileSystemObject* object lets you read and write files already, it only supports text, not binary data. You can also use the **File** object to modify a file's attributes and timestamps.

You can obtain a **File** object using the [FSUtil.OpenFile](#) and [Item.Open](#) methods. You can open a file in one of three modes:

- *read mode* - you can read data from the file via the **Read** method. You cannot write to it or modify its attributes.
- *write mode* - you can write data to the file via the **Write** method, and you can also modify the file's attributes. You cannot read data from it.
- *modify mode* - you can modify the file's attributes and timestamps, but you cannot read or write data.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the full pathname of the file.
error	<i>int</i>	Returns a Win32 error code that indicates the success or failure of the last operation. If the previous operation succeeded this will generally be <b>0</b> .  For example, if you try to open a non-existing file for reading using <b>FSUtil.OpenFile</b> , a valid <b>File</b> object will be returned - but the file itself would not be open. You can check if <b>error</b> returns <b>0</b> before proceeding to use the <b>File</b> object.
size	<i>object:FileSize</i>	Returns a <a href="#">FileSize</a> object representing the size of this file, in bytes.
tell	<i>object:FileSize</i>	Returns a <a href="#">FileSize</a> object representing the current position of the read or write cursor within this file, in bytes.

Method Name	Arguments	Return Type	Description
Close	<i>none</i>	<i>none</i>	<p>Closes the underlying file handle. After this call the <b>File</b> object is still valid but it can no longer read or write data.</p> <p>If you want to use the <b>SetAttr</b> method to modify the attributes of a file you have created, you may want to call <b>Close</b> first otherwise the file system will set the <i>A</i> (archive) attribute on the file whether you want it set or not.</p> <p>You may also want to close a file manually if you want to delete it, as some scripting languages (e.g. <i>JScript</i>) have lazy garbage collection and otherwise may keep the file handle open much longer than you intend.</p>
Read	<blob:target> <int:size>	<i>int</i> or <i>object:Blob</i>	<p>Reads data from the file. If you provide a <i>target</i> <a href="#">Blob</a> as the first parameter, the data will be stored in that <b>Blob</b>. Otherwise, a <b>Blob</b> will be created automatically.</p> <p>The optional <i>size</i> parameter specifies the number of bytes to read - the default behavior is to read the remaining contents of the file.</p> <p>If you provide a <b>Blob</b> then the return value indicates the number of bytes read successfully from the file. If a <b>Blob</b> isn't provided then the return value is the automatically created <b>Blob</b> - you can use its <b>size</b> property to discover the number of bytes that were read.</p>
Seek	<int:delta> <string:method>	<i>object:FileSize</i>	<p>Moves the read or write cursor within this file. The <i>delta</i> parameter specifies how many bytes to move - how this is interpreted depends on the optional <i>method</i> parameter:</p> <p><i>b</i> - move relative to the beginning of the file  <i>e</i> - move relative to the end of the file  <i>c</i> - move relative to the current position (this is the default method)</p>

			The return value is a <a href="#">FileSize</a> object indicating the new cursor position.
SetAttr	<p><i>object:</i> <a href="#">FileAttr</a></p> <p>or</p> <p>&lt;string:attributes&gt;</p>	<i>bool</i>	<p>Modifies the attributes of this file. You can either pass a string indicating the attributes to set, or a <a href="#">FileAttr</a> object. When using a string, valid attributes are:</p> <p><i>a</i> - archive  <i>c</i> - compressed  <i>e</i> - encrypted  <i>h</i> - hidden  <i>n</i> - normal  <i>r</i> - read-only  <i>s</i> - system  <p>p - pinned  i - non-content indexed</p> <p>Note that both <i>c</i> and <i>e</i> attributes cannot be set at the same time.</p> <p>When you pass a string you can also use + and - to turn some attributes on or off without affecting others. For example, <b>SetAttr("-r")</b> would turn off the read-only attribute.</p> <p>The return value is <b>True</b> if the operation was successful.</p> </p>
SetTime	<p>&lt;date:modify&gt;</p> <p>&lt;date:create&gt;</p> <p>&lt;date:access&gt;</p>	<i>bool</i>	<p>Modifies one or more of the file's timestamps. The <i>create</i> and <i>access</i> parameters are optional. If you wish to specify no change for a timestamp, specify <b>0</b>.</p> <p>Timestamps are specified as local time - use <b>SetTimeUTC</b> to specify them as UTC.</p> <p>The return value is <b>True</b> for success.</p>
SetTimeUTC	<p>&lt;date:modify&gt;</p> <p>&lt;date:create&gt;</p> <p>&lt;date:access&gt;</p>	<i>bool</i>	<p>Modifies one or more of the file's timestamps. The <i>create</i> and <i>access</i> parameters are optional. If you wish to specify no change for a timestamp, specify <b>0</b>.</p>

			Timestamps are specified as UTC time - use <b>SetTime</b> to specify them as local time.  The return value is <b>True</b> for success.
Truncate	<i>none</i>	<i>bool</i>	Truncates the file at the current position of the write cursor. You can use this in conjunction with the <b>Seek</b> method to pre-allocate a file's space on disk, for greater performance (i.e. seek to the final size of the file, truncate at that point, and then seek back to the start and write the data).  The return value is <b>True</b> for success.
Write	<blob:source> <int:from> <int:size>	int	Writes data from the specified <b>Blob</b> (or array) to the file. By default the entire contents of the <b>Blob</b> will be written - you can use the optional <i>from</i> parameter to specify the source byte offset, and the <i>size</i> parameter to specify the number of bytes to write.  The return value indicates the number of bytes successfully written to the file.

## FileAttr

The **FileAttr** object is provided to make it easier to deal with file attributes. Rather than dealing with attributes as a string of characters, or a number, it provides properties for each attribute that can be set or queried independently. You can create a new **FileAttr** object using the [FSUtil.NewFileAttr](#) method. **FileAttr** objects are also returned by properties of the [Format](#) and [Item](#) objects.

Each attribute is represented by two properties; a single character (e.g. **a**) and its full name (e.g. **archive**). Each property returns **True** if the attribute is set, and **False** if not. For **FileAttr** objects you create yourself, you can also set the value of these properties (and then, for example, apply the attributes to a file using the [File.SetAttr](#) method).

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns a string representing the attributes that are set (similar to the format displayed in the <i>Attr</i> column in the file display).
a archive	<i>bool</i>	A file or directory that is an archive file or directory. Applications typically use this attribute to mark files for backup or removal.

c compressed	<i>bool</i>	A file or directory that is compressed. For a file, all of the data in the file is compressed. For a directory, compression is the default for newly created files and subdirectories.
e encrypted	<i>bool</i>	A file or directory that is encrypted. For a file, all data streams in the file are encrypted. For a directory, encryption is the default for newly created files and subdirectories.
h hidden	<i>bool</i>	The file or directory is hidden. It is not included in an ordinary directory listing.
i nonindexed	<i>bool</i>	The file or directory is not to be indexed by the content indexing service.
o offline	<i>bool</i>	The data of a file is not available immediately. This attribute indicates that the file data is physically moved to offline storage. This attribute is used by Remote Storage, which is the hierarchical storage management software. Applications should not arbitrarily change this attribute.
p pinned	<i>bool</i>	The data of the file is to be kept available at all times; it should not be offloaded to offline storage.
r readonly	<i>bool</i>	A file that is read-only. Applications can read the file, but cannot write to it or delete it. This attribute is not honored on directories.
s system	<i>bool</i>	A file or directory that the operating system uses a part of, or uses exclusively.

<b>Method Name</b>	<b>Arguments</b>	<b>Return Type</b>	<b>Description</b>
Assign	<i>object:FileAttr</i> or <i>string</i>	none	Assigns a new set of attributes to this object. You can pass another <b>FileAttr</b> object, or a string (e.g. "hsr").
AttrName	<i>string</i>	<i>string</i>	Given a single character representing an attribute (e.g. "a") this method returns the name of the attribute in the user's current language (e.g. "Archive").
Clear	<i>object:FileAttr</i> or <i>string</i>	none	Clears (turns off) the specified attributes in this object. You can pass another <b>FileAttr</b> object, or a string representing the attributes to turn off.
Set	<i>object:FileAttr</i> or <i>string</i>	none	Sets (turns on) the specified attributes in this object. You can pass another <b>FileAttr</b> object, or a string representing the attributes to turn on.
ToString	none	none	Returns a string representing the attributes that are set (similar to the format displayed in the <i>Attr</i> column in the file display).

## FileGroup

The **FileGroup** object exposes information about a file group (when the file display is set to group by a particular column). The [Tab.filegroups](#) property returns a collection representing the current file groups in that tab, and the [Item.filegroup](#) property returns the current file group of a particular item.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the name of the group.
collapsed	<i>bool</i>	Returns <b>True</b> if the group is currently collapsed.
count	<i>int</i>	Returns the number of items in this group. Note that groups can be empty; empty groups are not displayed in the file display but will still be returned by the <b>Tab.filegroups</b> property.
id	<i>int</i>	Returns the id number of this group. Id numbers are arbitrary - you shouldn't place any meaning on the actual value, but you can compare the id fields as an easy way to tell if two items are in the same group.
members	<i>collection: <a href="#">Item</a></i>	Returns a collection of <a href="#">Item</a> objects that represents all the files and folders in this group.
type	<i>string</i>	Returns a string indicating the collation type of the group.

## FileSize

The **FileSize** object is provided to make it easier to deal with variables representing file sizes. It's very common these days for files to be larger than 4GB but unfortunately ActiveX scripting does not have proper support for the 64-bit integers needed to represent such large numbers.

Therefore, any time a file size or number representing a number of bytes is returned by the Opus scripting objects, it is as a **FileSize** object. For example, the [Item.size](#) property returns a **FileSize** representing the size of a particular file or folder.

You can create a new **FileSize** object using the [FSUtil.NewFileSize](#) method. A FileSize object normally represents an *unsigned* 64 bit integer but if you specify the "s" flag on creation, it will store a *signed* integer instead.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the number of bytes represented by this <b>FileSize</b> object as a <i>string</i> .
cy	<i>currency</i>	Returns the number of bytes as a <i>currency</i> value. This is a 64 bit data type but it is stored as a fractional value, so

		you must multiply the returned value by 10000 to obtain the actual byte size.
fmt	<i>string</i>	Returns the number of bytes as an automatically formatted string (e.g. if the <b>FileSize</b> value is 1024, the string <i>1 KB</i> would be returned).
high	<i>int</i>	Returns the highest (most significant) 32 bits of the file size.
low	<i>int</i>	Returns the lowest (least significant) 32 bits of the file size.

Method Name	Arguments	Return Type	Description
Add	<i>variant</i>	<i>none</i>	Adds the supplied value to the value of this <b>FileSize</b> object. You can pass a <i>string</i> , <i>int</i> or <i>currency</i> type, or another <b>FileSize</b> object.
Clone	<i>none</i>	<i>object:FileSize</i>	Clones this <b>FileSize</b> object and returns a new one set to the same value.
Compare	<i>variant</i>	<i>int</i>	Compares the supplied value with the value of this <b>FileSize</b> object. The return value will be <b>0</b> (equal), <b>1</b> (greater) or <b>-1</b> (less).
Div	<i>variant</i>	<i>none</i>	Divides the value of this <b>FileSize</b> object with the supplied value. You can pass a <i>string</i> , <i>int</i> or <i>currency</i> type, or another <b>FileSize</b> object.
Mult	<i>variant</i>	<i>none</i>	Multiplies the value of this <b>FileSize</b> object with the supplied value. You can pass a <i>string</i> , <i>int</i> or <i>currency</i> type, or another <b>FileSize</b> object.
Set	<i>variant</i>	<i>none</i>	Sets the <b>FileSize</b> to the supplied value. You can pass a <i>string</i> , <i>int</i> or <i>currency</i> type, or another <b>FileSize</b> object. You can also pass a <a href="#">Blob</a> consisting of exactly 1, 2, 4 or 8 bytes, in which case the data contained in the <a href="#">Blob</a> will be used to form the number.
Sub	<i>variant</i>	<i>none</i>	Subtracts the supplied value from the value of this <b>FileSize</b> object. You can pass a <i>string</i> , <i>int</i> or <i>currency</i> type, or another <b>FileSize</b> object. Note that the <b>FileSize</b> object is <i>unsigned</i> and so the value cannot go below zero.
ToBlob	<i>int</i>	<i>object:Blob</i>	Returns a <a href="#">Blob</a> containing the bytes that make up the current value. By default 8 bytes will be copied to the <a href="#">Blob</a> (the full 64 bit number) but you can pass an alternative number of bytes (1, 2 or 4) as a parameter to truncate the value.

## FiletypeGroup

The **FiletypeGroup** object represents a file type group (as configured in the [File Type Groups](#) section of the [file type editor](#)). You can find out which groups a file belongs to by querying the [Item.groups](#) property.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the internal name of this group.  The internal name is always the same in all languages.  Groups that come pre-defined when you install Opus have internal names like "Archives" and "Music" (which are also their English display names).  User-defined groups have internal names which are unique, automatically generated GUID strings like "{C4B716ED-2A9C-43C6-B325-7DADDEEFADA9}".
display_name	<i>string</i>	Returns the display name of this group.  The display name is what you see in the File Types editor. Display names may be translated differently in different languages.

## Filters

The **Filters** object provides information about the global file and folder filter settings (configured on the [Folders / Global Filters](#) page in Preferences). It is obtained from the [DOpus.filters](#) property.

Property Name	Return Type	Description
enable	<i>bool</i>	Returns <b>True</b> if the global wildcard filters are enabled.
file	<i>string</i>	Returns the global filename filter wildcard pattern. If the wildcard is configured to use regular expressions, it will have a <b>regex:</b> prefix in front of the pattern.
folder	<i>string</i>	Returns the global folder filter wildcard pattern. If the wildcard is configured to use regular expressions, it will have a <b>regex:</b> prefix in front of the pattern.
hidehidden	<i>bool</i>	Returns <b>True</b> if the global option to hide hidden files is on.

hidesystem	<i>bool</i>	Returns <b>True</b> if the global option to hide operating system files is on.
------------	-------------	--------------------------------------------------------------------------------

## FlatViewChangeData

If a [script add-in](#) implements the [OnFlatViewChange](#) event, the method receives a **FlatViewChangeData** object when the [Flat View](#) mode is changed in a tab.

Property Name	Return Type	Description
mode	<i>string</i>	Returns a <i>string</i> indicating the new Flat View mode. Will be one of <i>off</i> , <i>grouped</i> , <i>mixed</i> or <i>mixednofolders</i> .
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
tab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the tab the Flat View mode changed in.

## FolderEnum

The **FolderEnum** object is returned from the [FSUtil.ReadDir](#) method. It lets you enumerate (optionally, recursively) the contents of a folder.

Property Name	Return Type	Description
complete	<i>bool</i>	<b>True</b> if the enumeration is complete, otherwise <b>False</b> .
error	<i>int</i>	If an error occurs this will return the error code. It will return 0 on success.

Method Name	Arguments	Return Type	Description
Close	<i>none</i>	<i>none</i>	Closes the underlying file system handle used to perform the enumeration. You might call this method if you want to delete the folder you just enumerated. After this method is called the <b>complete</b> property will return <b>True</b> .
Next	<int:count> < <a href="#">Vector</a> :vector>	<i>object:Item</i> or <i>object:Vector</i>	Returns the next item in the enumeration.  By default (with no arguments provided) a single <a href="#">Item</a> object is returned. For higher performance, you can specify a number as the first argument to return more than one item at once - in this case, a

			<p><a href="#">Vector</a> of <a href="#">Item</a> objects is returned instead. Specify <b>1</b> to return all items in the folder in one call.</p> <p>You can also create your own <a href="#">Vector</a> and pass it as the second argument to stop Opus creating a new <a href="#">Vector</a> each time.</p>
--	--	--	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## FontMeta

The **FontMeta** object is retrieved from the [Metadata.font](#) property. It provides access to metadata relating to font files. The fields in this object correspond to those in the Windows SDK **LOGFONT** structure; further information about their meaning can be found on the [MSDN website](#).

Property Name	Return Type	Description
charset	<i>int</i>	The character set.
clipprecision	<i>int</i>	The clipping precision.
escapement	<i>int</i>	The angle, in tenths of degrees, between the escapement vector and the x-axis of the device.
fontname	<i>string</i>	The typeface name of the font.
height	<i>int</i>	The height, in logical units, of the font's character cell or character.
italic	<i>bool</i>	An italic font if set to <b>True</b> .
orientation	<i>int</i>	The angle, in tenths of degrees, between each character's base line and the x-axis of the device.
outprecision	<i>int</i>	The output precision.
pitchandfamily	<i>int</i>	The pitch and family of the font.
quality	<i>int</i>	The output quality.
strikeout	<i>bool</i>	A strikeout font if set to <b>True</b> .
underline	<i>bool</i>	An underlined font if set to <b>True</b> .
weight	<i>int</i>	The weight of the font in the range 0 through 1000.
width	<i>int</i>	The average width, in logical units, of characters in the font.

## Format

The **Format** object represents the current display format of a tab. It is obtained from the [Tab.format](#) property.

Property Name	Return Type	Description
alpha_folders	<i>bool</i>	Returns <b>True</b> if folders are always sorted alphabetically, <b>False</b> if otherwise.
autosize	<i>bool</i>	Returns <b>True</b> if column width auto-sizing is enabled, <b>False</b> if otherwise.
columns	<i>collection:Column</i>	Returns a collection of <a href="#">Column</a> objects that represent all the individual columns currently added to the display.
format_explain	<a href="#">Vector</a> : <i>string</i>	Returns a <a href="#">Vector</a> of strings representing the explanation of the current folder format (the same text visible when hovering the mouse over the format lock icon in the status bar).
getsizes	<i>string</i>	Returns a string that indicates the state of the option to automatically calculate folder sizes. The string returned will be one of <i>default</i> , <i>on</i> or <i>off</i> .
group_by	<i>string</i>	If grouping is enabled, returns the name of the column that the list is grouped by.
group_individual	<i>bool</i>	Returns <b>True</b> if the <b>Individual groups</b> option is enabled.
group_reverse	<i>bool</i>	Returns <b>True</b> if the groups are sorted in reverse order.
hide_attr	<i>object:FileAttr</i>	Returns a <a href="#">FileAttr</a> object indicating the file attributes that are hidden (any items with these attributes set will be hidden from the display).
hide_dirs	<i>string</i>	Returns the wildcard pattern of folders that are hidden from the display.
hide_dirs_regex	<i>bool</i>	Returns <b>True</b> if the current <b>hide_dirs</b> pattern is using regular expressions.
hide_ext	<i>bool</i>	Returns <b>True</b> if filename extensions are hidden, or <b>False</b> if they are displayed.
hide_files	<i>string</i>	Returns the wildcard pattern of files that are hidden from the display.
hide_files_regex	<i>bool</i>	Returns <b>True</b> if the current <b>hide_files</b> pattern is using regular expressions.
hide_folder_attr	<i>object:FileAttr</i> or <i>string</i>	Returns a <a href="#">FileAttr</a> object indicating the folder attributes that are hidden (any folders with these attributes set will be hidden from the display). If the separate folder attribute filter is disabled this property will return the string " <b>off</b> ".
ignore_prefix	<i>string</i>	Returns the filename prefixes that are ignored when sorting the list.
locked	<i>bool</i>	Returns <b>True</b> if the folder format is locked in the tab.
manual_sort	<i>bool</i>	Returns <b>True</b> if <a href="#">manual sorting</a> is enabled.
manual_sort_name	<i>string</i>	If manual sorting is active, returns the name of the current sort order (if it has one).

manual_sort_order	<i>object:</i> <a href="#">SortOrder</a>	If manual sort is active, returns a <a href="#">SortOrder</a> object which lets you query and change the sort order.
mix_type	<i>string</i>	Returns a string indicating the current file/folder mixing type. The string returned will be one of <i>mixed</i> , <i>files</i> (files first) or <i>dirs</i> (folders first).
name_ext	<i>bool</i>	Returns <b>True</b> if filenames and extensions are sorted separately.
numeric_name	<i>bool</i>	Returns <b>True</b> if numeric name sorting is enabled.
reverse_sort	<i>bool</i>	Returns <b>True</b> if the over-all sort order is reversed.
show_attr	<i>object:</i> <a href="#">FileAttr</a>	Returns a <a href="#">FileAttr</a> object indicating the file attributes that are shown (only items with these attributes set will be shown in the display).
show_dirs	<i>string</i>	Returns the wildcard pattern of folders that are shown (only folders matching this pattern will be shown).
show_dirs_regex	<i>bool</i>	Returns <b>True</b> if the current <b>show_dirs</b> pattern is using regular expressions.
show_files	<i>string</i>	Returns the wildcard pattern of files that are shown.
show_files_regex	<i>bool</i>	Returns <b>True</b> if the current <b>show_files</b> pattern is using regular expressions.
show_folder_attr	<i>object:</i> <a href="#">FileAttr</a> or <i>string</i>	Returns a <a href="#">FileAttr</a> object indicating the folder attributes that are shown (only folders with these attributes set will be shown in the display). If the separate folder attribute filter is disabled this property will return the string " <b>off</b> ".
sort_ext	<i>bool</i>	Returns <b>True</b> if the name column is sorted by filename extension rather than filename.
sort_field	<i>object:</i> <a href="#">Column</a>	Returns a <a href="#">Column</a> object representing the current sort field.
view	<i>string</i>	Returns the current <a href="#">view mode</a> as a string. The returned string will be one of <i>large_icons</i> , <i>small_icons</i> , <i>list</i> , <i>details</i> , <i>power</i> , <i>thumbnails</i> or <i>tile</i> .
word_sort	<i>bool</i>	Returns <b>True</b> if word sorting is enabled.

Method Name	Arguments	Return Type	Description
Update	<i>none</i>	<i>none</i>	The first time a script accesses a particular <b>Format</b> object, a snapshot is taken of the tab's format. If the script then makes changes to that tab (e.g. it changes the sort field, etc), these changes will not be reflected by the object. To re-synchronize the object with the tab, call the <b>Format.Update</b> method.

## **FSUtil**

The **FSUtil** object provides several utility methods for dealing with the file system. It is obtained using the **DOpus.FSUtil** method.

Method Name	Arguments	Return Type	Description
ComparePath	<string:path1> <string:path2> <string:flags>	<i>bool</i>	Compares the two provided path strings for equality - returns <b>True</b> if the two paths are equal, or <b>False</b> if otherwise.  The optional <i>flags</i> parameter lets you modify the comparison behavior. This parameter is a string containing one or more of the following letters:  <b>c</b> - case sensitive (makes the comparison case sensitive) <b>p</b> - parent (tests if <i>path2</i> is a parent of, or equal to, <i>path1</i> )
DisplayName	<string:path> <string:flags>	<i>string</i>	Retrieves the display name of a path. This is the form of a path that is intended to be displayed to the user, rather than used internally by Opus. For example, for a library path it will strip off the internal <i>?xxxxxxx</i> notation that Opus uses to identify library member folders.

			<p>The optional <i>flags</i> parameter lets you modify the behavior. This parameter is a string containing one or more of the following letters:</p> <p><b>e</b> - for editing (returns a string designed for editing rather than for display)  <b>f</b> - file part (returns the display filename rather than the entire path)  <b>r</b> - resolve (resolves library paths to their underlying file system folder)</p>
Drives	<i>none</i>	<b>Vector:</b> <a href="#">Drive</a>	Returns a <a href="#">Vector</a> of <a href="#">Drive</a> objects, one for each drive on the system.
Exists	<string:path>	<i>bool</i>	Returns <b>True</b> if the specified file, folder or device exists, or <b>False</b> otherwise.
GetErrorMsg	<int:error>	<i>string</i>	Returns the localized text description for a system error code.
GetItem	<string:path>	<i>object:</i> <a href="#">Item</a>	Creates an <a href="#">Item</a> object for the specified file path.
GetMetadata	<string:path>	<i>object:</i> <a href="#">Metadata</a>	Returns a <a href="#">Metadata</a> object representing the metadata for the specified file.
GetShellProperty	<string:path>  <string:property> or < <a href="#">Map</a> :properties>  <string:type>	<i>variant</i>	<p>Returns the value of one or more shell properties for the specified file. The file path must be provided as the first parameter. If the second parameter is the name (or PKEY) of the property to retrieve, the value of the property will be returned as the return value from this method.</p> <p>Alternatively, the second argument can be a <a href="#">Map</a> object which lets you retrieve multiple properties at once. Each property you want to retrieve should be added to</p>

			<p>the <a href="#">Map</a> with its name as a key, with an empty string as its value. The values in the <b>Map</b> will be replaced by the property values.</p> <p>The optional <i>type</i> argument lets you control how the properties are looked up by name. If the value of <i>type</i> is "R" then the first property whose raw name matches the supplied name will be used. If the value is "D" then the first property whose display name matches the supplied name will be used. If <i>type</i> is omitted then both raw and display names can match.</p> <p>Note that if a shell property is returned by the system as a SAFEARRAY type, it will be converted automatically to a <a href="#">Vector</a> object.</p>
GetShellPropertyList	<string:pattern> <string:type>	<i>object:</i> <a href="#">ShellProperty</a>	Returns a <a href="#">Vector</a> of <a href="#">ShellProperty</a> objects which represents all the possible shell properties available on the system. You can optionally provide a wildcard <i>pattern</i> as the first argument - if you do, only properties whose names match the supplied pattern will be returned. <p>The optional second argument lets you restrict the list of properties further. If the value of <i>type</i> is "R" then only properties whose raw name match the pattern will be returned. If the value is "D" then only properties whose display names match</p>

			the pattern will be returned. If <i>type</i> is omitted then both raw and display names will be compared.
GetType	<string:path> <string:flags>	<i>string</i>	Returns a string indicating the type of the specified file path. The string will be either <b>file</b> , <b>dir</b> or <b>invalid</b> if the path doesn't exist. The optional flags argument is used to control the behavior with archives - normally an archive will be reported as <b>dir</b> , but if you specify "a" for the flags parameter it will be reported as <b>file</b> .
Hash	<string:path> or <object: <a href="#">Blob</a> >  <string:type>	<i>string</i> or <i>object:</i> <a href="#">Vector</a>	Calculates a checksum for the specified file or <a href="#">Blob</a> . By default the MD5 checksum is calculated, but you can use the optional type parameter to change the checksum algorithm - valid values are "sha1", "sha256" and "sha512".  You can also specify multiple types (e.g. "md5,sha1,sha256") as the <b>type</b> , in which case then all checksums will be calculated at the same time, and the result will be returned as a <a href="#">Vector</a> of <i>strings</i> (in the same order as you requested them).
NewFileAttr	<attributes>	<i>object:</i> <a href="#">FileAttr</a>	Creates a new <a href="#">FileAttr</a> object, which represents file attributes. You can initialize the new object by passing either a string representing the attributes to turn on (e.g. "hsr") or another <a href="#">FileAttr</a> object. If you don't pass a value the new object will default to all attributes turned off.
NewFileSize	<string:"s"> <size>	<i>object:</i> <a href="#">FileSize</a>	Creates a new <a href="#">FileSize</a> object, which makes it easier to handle 64 bit file

			<p>sizes. You can initialize this with a number of data types (<i>int</i>, <i>string</i>, <i>currency</i>, another <a href="#">FileSize</a> object, or a <a href="#">Blob</a> containing exactly 1, 2, 4 or 8 bytes).</p> <p>If the optional "s" flag is specified, the FileSize object will use a <i>signed</i> 64 bit value rather than <i>unsigned</i>.</p>
NewPath	<string:path>	<i>object:</i> <a href="#">Path</a>	Creates a new <a href="#">Path</a> object initialised to the provided path string.
NewWild	<string:pattern> <string:flags>	<i>object:</i> <a href="#">Wild</a>	<p>Creates a new <a href="#">Wild</a> object. If a pattern and flags are provided the pattern will be parsed automatically, otherwise you must call the <b>Parse</b> method to parse a pattern before using the object.</p> <p>See the description of the <a href="#">Wild.Parse</a> method for a list of the valid flags.</p>
OpenFile	<string:path> or <object: <a href="#">Blob</a> >  <string:mode>  <object>window> or <string:elevation>	<i>object:</i> <a href="#">File</a>	<p>Opens or creates a file and returns a <a href="#">File</a> object that lets you access its contents as binary data. You can pass a string or <a href="#">Path</a> object representing a file to open, and you can also pass an existing <a href="#">Blob</a> object to create a <a href="#">File</a> object that gives you read/write stream access to a chunk of memory.</p> <p>By default the file will be opened in <i>read mode</i> - specify "w" for the optional <i>mode</i> parameter to open the file in <i>write mode</i>. Note that you cannot both read and write with the same <b>File</b> object, unless it was opened on a <a href="#">Blob</a>.</p>

When opening in write mode, you can also specify optional flags that control how the file is opened:

*wc* - create a new file, only if it doesn't already exist.

*wa* - create a new file, always. If the file already exists it will be overwritten. (This is the default.)

*we* - open existing file. The file will not be created if it doesn't already exist.

*wo* - open existing file. If the file doesn't exist it will be created.

*wt* - truncate existing file. If the file exists it will be truncated. The file will not be created if it doesn't already exist.

When using write mode, you may add *f* (force) to any of the above mode strings to tell Opus to clear the read-only file attribute if it blocks modifying an existing file; otherwise, attempting to open a read-only file for writing will fail. For example, "*wof*" is like "*wo*" mode but also clears the read-only attribute.

If you only want to make changes to a file's attributes without modifying its data you can also specify "*m*" to open it in *modify mode*.

The optional third parameter lets you associate the **File** object with a [Tab](#) or a [Lister](#), which will be used if Opus needs to display any dialogs (e.g. a UAC elevation dialog). You may also

			<p>specify the string "NoElevate" for the third parameter to prevent UAC elevation entirely, or "ElevateNoAsk" to prevent UAC prompts while still gaining elevation if something else has already performed it.</p> <p>A <b>File</b> object is always returned, even if the file could not be opened. Check <b>File.error</b> on the returned object immediately after creating it to see if opening the file succeeded. Even if the file was not be opened, some of the object's methods may still work. For example, if a file doesn't exist then you can't open it or set its attributes, but permissions on an existing file may allow you to set its attributes while blocking you from modifying it or vice versa.</p>
ReadDir	<string:path> [<bool:recurse>]  [<bool:shell>]	<i>object:</i> <a href="#">FolderEnum</a>	Returns a <a href="#">FolderEnum</a> object that lets you enumerate the contents of the specified folder. <p>Pass <b>True</b> for the optional <i>recurse</i> flag to recursively enumerate the folder (i.e. to enumerate the contents of all its sub-folders, and their sub-folders, and so on).</p> <p>If the optional third parameter is set to <b>True</b> you can use <b>ReadDir</b> to enumerate shell (non-filesystem) folders; for example, the Quick Access</p>

			folder on Windows 10 could be enumerated by passing "/quickaccess" for the path and <b>True</b> for the third argument.
Resolve	<string:path> <string:flags>	<i>object:</i> <a href="#">Path</a>	Resolves the specified library or file collection path to its underlying file system path. This method can also be used to resolve a <a href="#">folder alias</a> , as well as application paths in the form <b>{apppath appname}</b> .  The optional flags are:  <i>j</i> - resolve junctions and links to their target folder
SameDrive	<string:path> <string:flags>	<i>bool</i>	Returns <b>True</b> if this path refers to the same drive or partition as the supplied path. The optional flags are:  <i>c</i> - treat CD burning staging area as the CD itself <i>m</i> - consider mount points <i>r</i> - real paths only <i>u</i> - compare user for FTP drives <i>z</i> - consider zip folders <i>Z</i> - consider zip folders (target)

## Func

The **Func** object is passed to a script when it is invoked via a command. In a [script function](#), it is passed to the [OnClick](#) method as the [ClickData.func](#) property, and in a script add-in that [adds an internal command](#), via the [ScriptCommandData.func](#) property. The **Func** object provides information about the default source and destination of the command, as well as details about how it was invoked.

Property Name	Return Type	Description
args	<i>object:</i> <a href="#">Args</a>	Returns an <a href="#">Args</a> object that provides access to any arguments given on the command line that invoked this script. This is used when the script has added an <a href="#">internal</a>

		<p><a href="#">command</a> to Opus. A command line template can be provided when the command is added, and any arguments the user provides on the command line for the script command will be available via this object.</p> <p>For most use the <b>argsmap</b> property may be an easier way to access your command's arguments.</p>
argsmap	<i>object:</i> <a href="#">Map</a>	Returns a <a href="#">Map</a> object that provides keyword lookup for each of the arguments given on the command line. An argument will only be present in the <a href="#">Map</a> if it was used on the command line, so you can easily check which arguments are present using the <a href="#">Map.exists()</a> method.
command	<i>object:</i> <a href="#">Command</a>	This property returns a pre-filled <a href="#">Command</a> object that can be used to run commands against the source and destination tabs. Using this object is the equivalent of calling <a href="#">DOpusFactory.Command</a> and setting the source and destination tabs manually.
desttab	<i>object:</i> <a href="#">Tab</a>	This object represents the default destination tab for the function.
fromdrop	<i>bool</i>	Returns <b>True</b> if the command was invoked via a drag-and-drop operation.
fromkey	<i>bool</i>	Returns <b>True</b> if the command was invoked via the keyboard (i.e. via a hotkey rather than a button).
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the command was invoked. The string can contain any or all of the following: <i>shift, ctrl, alt, lwin, rwin</i> .
sourcetab	<i>object:</i> <a href="#">Tab</a>	This object represents the default source tab for the function.
viewer	<i>object:</i> <a href="#">Viewer</a>	If this button was run from the <a href="#">standalone image viewer</a> , this object represents the viewer window.

Method Name	Arguments	Return Type	Description
Dlg	<i>none</i>	<i>object:</i> <a href="#">Dialog</a>	Creates a new <a href="#">Dialog</a> object, that lets you display dialogs and popup menus. The dialog's <b>window</b> property will be automatically assigned to the source tab.

### **GetCopyQueueNameData**

If a [script add-in](#) implements the [OnGetCopyQueueName](#) event, the method receives a **GetCopyQueueNameData** object whenever a copy operation begins that uses automatically-

managed copy queues (i.e. the *Automatically manage file copy queues* option on the [File Operations / Copy Options](#) page in Preferences is turned on).

Property Name	Return Type	Description
dest	<i>object:Path</i>	Returns a <a href="#">Path</a> object representing the destination path of the copy operation.
desttab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the destination folder tab.
dest_drives	<i>string</i>	Returns a binary string indicating the physical drive indices that the destination path is located on (if any). For example, 001000000000000000000000 indicates that drive C: is the destination drive.
move	<i>bool</i>	Returns <b>True</b> if the operation is a move instead of a copy.
name	<i>string</i>	Returns the default queue name for this operation.
source	<i>object:Path</i>	Returns a <a href="#">Path</a> object representing the source path of the copy operation.
sourcetab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the source folder tab.
source_drives	<i>string</i>	Returns a binary string indicating the physical drive indices that the source path is located on (if any). For example, 0000100000000000000000000000 indicates that drive E: is the source drive.

### GetCustomFieldData

If a [rename script](#) implements the [OnGetCustomFields](#) event, the method receives a **GetCustomFieldData** object whenever the Rename dialog runs the script.

Property Name	Return Type	Description
fields	<i>object:CustomFieldData</i>	Returns a <a href="#">CustomFieldData</a> object, that the script can use to add custom fields to the <i>Rename</i> dialog. Each property added to the object in this method will be create a new field in the dialog, allowing the user to supply additional information to your rename script.
field_labels	<i>object:Map</i>	This lets you assign labels to your script's custom fields, that are shown to the user in the <i>Rename</i> dialog. To do this, set this property to a <a href="#">Map</a> created via the <a href="#">DOpusFactory.Map</a> method, filled with name/label string pairs.
field_tips	<i>object:Map</i>	This lets you assign "cue banners" to any edit fields created by your script. A cue banner is displayed

		inside an empty edit field to prompt the user what sort of data the field expects. To use this, set this property to a <a href="#">Map</a> created via the <a href="#">DOpusFactory.Map</a> method, filled with name/banner string pairs.
focus	<i>string</i>	You can use this field to specify which control gets the input focus by default when your fields appear for the first time. Set it to the name of the desired control. You can also specify <b>!oldname</b> or <b>!newname</b> to assign focus to the standard old and new name fields.

## GetHelpContentData

If a script implements the [OnGetHelpContent](#) event, it receives a **GetHelpContentData** object when the method is called. You can use the methods of this object to add your help content to the Directory Opus F1 help system.

Method Name	Arguments	Return Type	Description
AddHelpImage	<string:name> < <a href="#">Blob</a> :image>	<i>none</i>	<p>Adds a PNG or JPG image and makes it available for your help pages. You can use any name you like for your images, although they must have either a <b>.png</b> or a <b>.jpg</b> suffix. Your help content can then refer to images by name, e.g. if you add an image and call it <b>myimage.jpg</b>, your html content could show it using:</p> <pre>&lt;img src="myimage.jpg"&gt;</pre> <p>If your script is bundled as a <a href="#">script package</a> you can include image files in a sub-directory of the package called <b>help</b>, and then load them easily using the <a href="#">Script.LoadHelpImage</a> method.</p>
AddHelpPage	<string:name> <string:title> <string:body>	<i>none</i>	<p>Adds a page of help content for your script to the F1 help file. You can call this method as many times as you like. If you add more than one page of help the first page will become the topic header and all subsequent pages will appear underneath it in the index.</p> <p>The <b>name</b> parameter is optional; if not supplied, a default name will be assigned automatically to each page you add. You can use the <a href="#">Script.ShowHelp</a> method to trigger the display of a specific page of</p>

		<p>your help by name.</p> <p>The <b>title</b> parameter specifies the page title; this is shown in the help index and should be descriptive enough that the user can recognise it as referring to your script. The <b>body</b> parameter specifies the actual HTML content for the help page. This should be everything you would normally find <i>between</i> the <b>&lt;body&gt;...&lt;/body&gt;</b> tags of an HTML page.</p> <p>If your script is bundled as a <a href="#">script package</a> you can include HTML files in a sub-directory of the package called <b>help</b>, and then load them easily using the <a href="#">Script.LoadHelpFile</a> method.</p>
--	--	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

### GetNewNameData

If a [rename script](#) is implemented using the [OnGetNewName](#) method, it receives a **GetNewNameData** object for each item being renamed.

Property Name	Return Type	Description
custom	object: <a href="#">CustomFieldData</a>	Returns a <a href="#">CustomFieldData</a> object which provides the values of any <a href="#">custom fields</a> your script added to the <i>Rename</i> dialog.
item	object: <a href="#">Item</a>	Returns an <a href="#">Item</a> object representing the file or folder being renamed.
newname	<i>string</i>	Returns the proposed new name of the item. This will be the result of the application of any selected standard options in the <a href="#">rename dialog</a> (numbering, capitalization, etc).
newname_ext	<i>string</i>	Returns the file extension of the proposed new name. Does not take multi-part extensions into account (e.g. will return ".rar" rather than ".part1.rar").
newname_ext_m	<i>string</i>	Returns the file extension of the proposed new name, taking multi-part extensions into account (e.g. will return ".part1.rar" rather than ".rar").
newname_field	<i>string</i>	Returns the contents of the <i>New Name</i> field (that is, not the calculated new name after all the options have been applied, but the actual text contents of the field as entered by the user).

newname_stem	<i>string</i>	Returns the file stem of the proposed new name. Does not take multi-part extensions into account (e.g. will return "catpictures.part1" rather than "catpictures").
newname_stem_m	<i>string</i>	Returns the file stem of the proposed new name, taking multi-part extensions into account (e.g. will return "catpictures" rather than "catpictures.part1").
oldname_field	<i>string</i>	Returns the "old name" pattern as entered by the user in the <a href="#">rename dialog</a> .

## ImageMeta

The **ImageMeta** object is retrieved from the [Metadata.image](#) or [Metadata.image\\_text](#) properties. It provides access to metadata relating to picture files.

Property Name	Return Type	Description
<column keyword>	<i>variant</i>	Returns the value of the specified column, as listed in the <i>Pictures</i> section of the <a href="#">Keywords for Columns</a> page.

## Item

The **Item** object represents a file or folder displayed in a tab. You can obtain a collection of **Item** objects from the various methods of the [Tab](#) object. A collection of **Item** objects is also used with the [Command](#) object to specify the files or folders a command should operate on. Using the [FSUtil.ReadDir](#) method to enumerate the contents of a folder also returns an **Item** object. You can create an **Item** from a file path using the [FSUtil.GetItem](#) method.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the full pathname of the item (i.e. path plus filename).
access	<i>date</i>	Returns the "last accessed" date, in local time.
access_utc	<i>date</i>	Returns the "last accessed" date, in UTC.
attr	<i>int</i>	Returns the item attributes. This value is a series of flags that are logically OR'd together. The attributes supported by Opus are:  1 - read only 2 - hidden 4 - system 32 - archive

		<p>2048 - compressed  4096 - offline storage  8192 - not content-indexed  16384 - encrypted  524288 - pinned</p> <p>Using the <b>fileattr</b> property, which returns a <a href="#">FileAttr</a> object, may be easier than dealing with the raw attribute flags.</p>
attr_text	<i>string</i>	Returns the item attributes as a string, as displayed in the file display.
checked	<i>bool</i>	Returns <b>True</b> if the item was checked (in <a href="#">checkbox mode</a> ), or <b>False</b> otherwise.
create	<i>date</i>	Returns the "creation" date, in local time.
create_utc	<i>date</i>	Returns the "creation" date, in UTC.
current	<i>bool</i>	This is only present for <b>Item</b> objects obtained from a <a href="#">Viewer</a> . It will be <b>True</b> if the item represents the currently displayed image.
display_name	<i>string</i>	Returns the display name of the item. Only a few items have a display name that is different to their actual name - some examples are certain system folders (like <i>C:\Users</i> which might have a translated display name in non-English locales).
ext	<i>string</i>	Returns the filename extension.
ext_m	<i>string</i>	Returns the filename extension, taking multi-part extensions into account. For example, a file called "file.part1.rar" might return ".rar" for <b>ext</b> but ".part1.rar" for <b>ext_m</b> .
failed	<i>bool</i>	Returns <b>True</b> if the item failed when used by a command. This is only meaningful in conjunction with the <a href="#">Command.files</a> collection - once the command has returned, this property will indicate success or failure on a per-file basis.
fileattr	<i>object:</i> <a href="#">FileAttr</a>	Returns a <a href="#">FileAttr</a> object that represents the item's attributes.
filegroup	<i>object:</i> <a href="#">FileGroup</a>	If the file display this item came from is grouped by a particular column, this property returns a <a href="#">FileGroup</a> object representing the group the item is in. If the item has no group this will return an empty string.
got_size	<i>bool</i>	Returns <b>True</b> for folder items if their size has been calculated by, for example, the <a href="#">GetSizes</a> command. If <b>False</b> , the <b>size</b> property will be unreliable for folders.
groups	<a href="#">Vector:</a> <a href="#">FiletypeGroup</a>	Returns a <a href="#">Vector</a> of <a href="#">FiletypeGroup</a> objects representing any and all file type groups that this file is a member of.

		If you only want to check membership of a particular file type group, see the <b>InGroup</b> method in the section below.
id	<i>int</i>	This is a unique ID for the item; it is used internally by Opus.
is_dir	<i>bool</i>	Returns <b>True</b> if the item represents a folder, and <b>False</b> for a file.
metadata	<i>object:</i> <a href="#">Metadata</a>	Returns a <a href="#">Metadata</a> object that provides access to the item's metadata.
modify	<i>date</i>	Returns the "last modified" date, in local time.
modify_utc	<i>date</i>	Returns the "last modified" date, in UTC.
name	<i>string</i>	Returns the name of the item.
name_stem	<i>string</i>	Returns the filename "stem" of the item. This is the name of the item with the filename extension removed. It will be the same as the <b>name</b> for folders.
name_stem_m	<i>string</i>	Returns the filename "stem" of the item, taking multi-part extensions into account. For example, a file called "file.part1.rar" might return "file.part1" for <b>name_stem</b> but "file" for <b>name_stem_m</b> .
path	<i>object:</i> <a href="#">Path</a>	Returns the path of the item's parent folder. This does not include the name of the item itself, which can be obtained via the <b>name</b> property.
realpath	<i>object:</i> <a href="#">Path</a>	Returns the "real" path of the item. For items located in <a href="#">virtual folders</a> like <a href="#">Libraries</a> or <a href="#">Collections</a> , this lets you access the item's underlying path in the real file system. The <b>realpath</b> property includes the full path to the item, including its own name.
selected	<i>bool</i>	Returns <b>True</b> if the item was selected, or <b>False</b> otherwise.
shortpath	<i>object:</i> <a href="#">Path</a>	Returns the short path of the item, if it has one. Note that short paths are disabled by default in Windows 10.
size	<i>object:</i> <a href="#">FileSize</a>	Returns the size of the item as a <a href="#">FileSize</a> object.

Method Name	Arguments	Return Type	Description
InGroup	<string:group>	<i>bool</i>	<p>Tests the file for membership of the specified file type group.</p> <p>Each file type group has two names: An internal name which is always the same in all languages, and a display name which may be translated differently for each language. The display name is what you see in the File Types editor. Groups that come pre-defined when you install Opus</p>

			<p>have internal names like <i>"Archives"</i> and <i>"Music"</i> (which are also their English display names). User-defined groups have internal names which are unique, automatically generated GUID strings like <i>"{C4B716ED-2A9C-43C6-B325-7DADDEEFADA9}"</i>.</p> <p>The <i>group</i> argument should be the name of the group you wish to test against, e.g. <i>"Music"</i>.</p> <p>By default, both the internal name and the display name are checked, and a match on either will return true. Prefix the <i>group</i> argument with <i>"name:"</i> to restrict the search to internal names, or with <i>"disp:"</i> to restrict the search to display names.</p> <p>To get a list of all file type groups which the file matches, use the <b>groups</b> property instead (see the section above).</p>
Labels	<pre>&lt;string:category&gt; &lt;string:flags&gt;</pre>	<a href="#">Vector</a> :string	<p>This method returns a <a href="#">Vector</a> of strings representing any <a href="#">labels</a> that have been assigned to the item.</p> <p>Both arguments are optional. The first is a <a href="#">wildcard pattern</a> that lets you filter the returned labels based on their category. For example, pass <i>"Status"</i> to only retrieve a list of status icons assigned to a file.</p> <p>The second optional argument contains flags keywords that control how the labels are returned. The only defined flag is <i>"explicit"</i> - if specified, wildcard and label filters will not be considered - only explicitly assigned labels will be returned. Note that if you want to provide the second argument but don't want to filter by category you should pass <i>"*"</i> for the first argument to match all categories.</p>
Open	<pre>&lt;string:mode&gt; &lt;object&gt;window&gt;</pre>	object: <a href="#">File</a>	<p>Opens this file and returns a <a href="#">File</a> object that lets you access its contents as binary data.</p> <p>By default the file will be opened in <i>read mode</i> - specify <i>"w"</i> for the optional <i>mode</i> parameter to open the file in <i>write mode</i>. Note that you cannot both read and write with the same <b>File</b> object.</p>

When opening in write mode, you can also specify optional flags that control how the file is opened:

*wc* - create a new file, only if it doesn't already exist.

*wa* - create a new file, always. If the file already exists it will be overwritten. (This is the default.)

*we* - open existing file. The file will not be created if it doesn't already exist.

*wo* - open existing file. If the file doesn't exist it will be created.

*wt* - truncate existing file. If the file exists it will be truncated. The file will not be created if it doesn't already exist.

When using write mode, you may add *f* (force) to any of the above mode strings to tell Opus to clear the read-only file attribute if it blocks modifying an existing file; otherwise, attempting to open a read-only file for writing will fail. For example, "*wof*" is like "*wo*" mode but also clears the read-only attribute.

If you only want to make changes to the file's attributes without modifying its data you can also specify "*m*" to open it in *modify mode*.

The optional *window* parameter lets you associate the **File** object with a [Tab](#) or a [Lister](#), which will be used if Opus needs to display any dialogs (e.g. a UAC elevation dialog). You may also specify the string "NoElevate" to prevent UAC elevation entirely, or "ElevateNoAsk" to prevent UAC prompts while still gaining elevation if something else has already performed it.

A **File** object is always returned, even if the file could not be opened. Check **File.error** on the returned object immediately after creating it to see if opening the file succeeded. Even if the file was not be opened, some of the object's methods may still work. For example, if a file doesn't exist then you can't open it or set its attributes, but permissions on an existing file may allow you to set its attributes while blocking you from modifying it or vice versa.

ShellProp	<string:property> <string:type>	<i>variant</i>	<p>Returns the value of the specified shell property for the item. The property argument can be the property's PKEY or its name.</p> <p>If you provide a name then the optional second argument lets you control how the properties are looked up by name. If the value of <i>type</i> is "R" then the first property whose raw name matches the supplied name will be used. If the value is "D" then the first property whose display name matches the supplied name will be used. If <i>type</i> is omitted then both raw and display names can match.</p> <p>Note that if a shell property is returned by the system as a SAFEARRAY type, it will be converted automatically to a <a href="#">Vector</a> object.</p>
Update	<i>none</i>	<i>none</i>	<p>Updates the <b>Item</b> object from the file on disk. You might use this if you had run a command to change an item's timestamp or attributes, and wanted to retrieve the new information.</p>

## Lister

The **Lister** object represents an open [Lister](#) window. A collection of currently open Lister objects is available from the [DOpus.listers](#) property, and if a command results in a new Lister being opened, the [Results](#) object.

Property Name	Return Type	Description
activetab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the currently active (source) tab.
bottom	<i>int</i>	Lister window bottom-edge coordinate.
custom_title	<i>string</i>	Returns the custom title of the Lister (if any) as set by the <b>Set LISTERTITLE</b> command. This may be an empty string. The <i>title</i> property returns the actual window title.
desttab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the current destination tab (in a dual-display Lister).
dual	<i>int</i>	Indicates whether the Lister is in <a href="#">dual-display mode</a> or not. Possible values are: 0 - single-display mode 1 - dual-display, vertical layout 2 - dual-display, horizontal layout
dualsize	<i>int</i>	Returns the current split percentage of the dual displays (e.g. <b>50</b> indicates they are evenly sized).
foreground	<i>bool</i>	Returns <b>True</b> if this Lister is currently the foreground (active) window.
lastactive	<i>bool</i>	Returns <b>True</b> if this Lister is currently the active Lister (foreground window), or was the most recently active Lister.
layout	<i>string</i>	Provides the name of the <a href="#">Lister layout</a> that this Lister came from (if any).
left	<i>int</i>	Lister window left-edge coordinate.
metapane	<i>int</i>	Indicates whether the <a href="#">metadata pane</a> is currently open or not. Possible values are: 0 - metadata pane is not open 1 - metadata pane is open, vertical layout 2 - metadata pane is open, horizontal layout
right	<i>int</i>	Lister window right-edge coordinate.
state	<i>string</i>	Returns the state of a single-display mode Lister: 0 - off 1 - source 2 - destination 4 - locked

tabs	<i>collection:</i> <a href="#">Tab</a>	Returns a collection of <a href="#">Tab</a> objects that represent all tabs in this Lister. In a dual-display Lister this includes tabs in both the left and right file displays.
tabsleft	<i>collection:</i> <a href="#">Tab</a>	Returns a collection of <a href="#">Tab</a> objects that represent the tabs in the left/top side of a dual-display Lister. In a single-display Lister this is equivalent to all the tabs in the Lister.
tabsright	<i>collection:</i> <a href="#">Tab</a>	Returns a collection of <a href="#">Tab</a> objects that represent the tabs in the right/bottom side of a dual-display Lister. In a single-display Lister this will return an empty collection.
title	<i>string</i>	Returns the current title of the Lister window.
toolbars	<i>collection:</i> <a href="#">Toolbar</a>	Returns a collection of <a href="#">Toolbar</a> objects representing all currently open toolbars in this Lister.
top	<i>int</i>	Lister window top-edge coordinate;
tree	<i>int</i>	Indicates whether or not the <a href="#">folder tree</a> is currently open. Possible values are: 0 - folder tree is not open 1 - a single tree is open, at the left of the Lister 2 - a single tree is open, at the right of the Lister 3 - two folder trees are open (in a dual-display Lister)
utilpage	<i>string</i>	If the <a href="#">utility panel</a> is currently open, returns a string indicating the currently selected utility page. Possible values are <b>find</b> , <b>sync</b> , <b>dupe</b> , <b>undo</b> , <b>filelog</b> , <b>ftplog</b> , <b>otherlog</b> , <b>email</b> .
utilpane	<i>int</i>	Indicates whether or not the <a href="#">utility panel</a> is currently open. Possible values are: 0 - utility panel is not open 1 - utility panel is open
vars	<i>object:</i> <a href="#">Vars</a>	This <a href="#">Vars</a> object represents all defined variables with <i>Lister scope</i> (that are scoped to this Lister).
viewpane	<i>int</i>	Indicates whether or not the <a href="#">viewer pane</a> is currently open. Possible values are: 0 - viewer pane is not open 1 - viewer pane is open, vertical layout 2 - viewer pane is open, horizontal layout

Method Name	Arguments	Return Type	Description
Dlg	<i>none</i>	<i>object:</i> <a href="#">Dialog</a>	Creates a new <a href="#">Dialog</a> object, that lets you display dialogs and popup menus. The dialog's <b>window</b> property will be automatically assigned to this Lister.
Update	<i>none</i>	<i>none</i>	The first time a script accesses a particular <b>Lister</b> object, a snapshot is taken of the Lister state. If the script then makes changes to that Lister (e.g. it opens a new tab, or moves the window), these changes will

not be reflected by the object. To re-synchronize the object with the Lister, call the **Lister.Update** method.

## Listers

The **Listers** object is a collection of all currently open [Listers](#). It can be obtained via the [DOpus.listers](#) property.

Property Name	Return Type	Description
<default value>	collection: <a href="#">Lister</a>	Lets you enumerate the currently open Listers.
lastactive	object: <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the most recently active Lister window.

Method Name	Arguments	Return Type	Description
Update	<i>none</i>	<i>none</i>	The first time a script accesses the <b>DOpus.listers</b> property, a snapshot is taken of all currently open Listers. If the script then opens or closes Listers itself, these changes will not be reflected by this collection. To re-synchronize the collection, call the <b>Update</b> method.

## ListerResizeData

If a [script add-in](#) implements the [OnListerResize](#) event, the method receives a **ListerResizeData** object whenever a Lister window is resized.

Property Name	Return Type	Description
action	<i>string</i>	Returns a <i>string</i> indicating the resize action that occurred. This will be one of the following strings: <i>resize, minimize, maximize, restore</i> .
width	int	Returns the new width of the Lister in pixels.
height	int	Returns the new height of the Lister in pixels.
lister	object: <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the Lister that was resized.

## ListerUIChangeData

If a [script add-in](#) implements the [OnListerUIChange](#) event, the method receives a **ListerUIChangeData** object when various user interface elements are opened or closed in a Lister.

Property Name	Return Type	Description
change	<i>string</i>	Returns a <i>string</i> indicating which UI elements changed. This will contain one or more of the following strings: <i>dual, tree, metapane, viewer, utility, duallayout, metapanelayout, viewerlayout, toolbars, toolbarset, toolbarsauto, minmax.</i>
lister	<i>object:</i> <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the Lister that is changing.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift, ctrl, alt, lwin, rwin.</i>

## Map

The **Map** object is an [associative container](#). It is similar to an array or vector (e.g. [Vector](#)) in that it can store one or more objects, but has the advantage of using a dictionary system to locate objects rather than numeric indexes. You can therefore insert or lookup objects using an arbitrary value (string, integer, date, etc.) as the key (e.g. **Map("foo")** to reference an element by the key "foo").

You can create a new **Map** using the [DOpusFactory.Map](#) method. The keys in a map can be enumerated, and are automatically kept sorted.

The two examples below both output the following:

```
count: 4

bird -> tweet

cat -> meow

dog -> woof

snake -> hiss
```

## JScript Example:

```
var map = DOpus.Create.Map();

map("cat") = "meow";
map("dog") = "woof";
map("bird") = "tweet";
map("snake") = "hiss";

DOpus.Output("count: " + map.count);

for (var e = new Enumerator(map); !e.atEnd(); e.moveNext())
{
    var key = e.item();
    var value = map(key);
    DOpus.Output(key + " -> " + value);
}
```

## VBScript Example:

```
Dim map, key, value

Set map = DOpus.Create.Map

map("cat") = "meow"
map("dog") = "woof"
map("bird") = "tweet"
map("snake") = "hiss"

DOpus.Output "count: " & map.count
```

```

For Each key In map
    value = map(key)
    DOpus.Output(key & " -> " & value)
Next

```

Property Name	Return Type	Description
count	<i>int</i>	Returns the number of elements the <b>Map</b> currently holds.
empty	<i>bool</i>	Returns <b>True</b> if the <b>Map</b> is empty, <b>False</b> if not.
length	<i>int</i>	A synonym for <b>count</b> .
size	<i>int</i>	A synonym for <b>count</b> .

Method Name	Arguments	Return Type	Description
assign	< <b>Map</b> :from>	<i>none</i>	Copies the contents of another <b>Map</b> to this one.
clear	<i>none</i>	<i>none</i>	Clears the contents of the <b>Map</b> .
erase	<variant:key>	<i>none</i>	Erases the element matching the specified key, if it exists in the map.
exists	<variant:key>	<i>bool</i>	Returns <b>True</b> if the specified key exists in the map.
merge	< <b>Map</b> :from>	<i>none</i>	Merges the contents of another <b>Map</b> with this one.

## Metadata

The **Metadata** object provides metadata information about a file or folder (metadata are things like the track number of a music file, the dimensions of an image, the author of a document, etc). You can obtain a **Metadata** object from the [Item.metadata](#) property if you have an **Item** object, and if not you can obtain it using the path of the item using the [FSUtil.GetMetadata](#) method.

The **Metadata** object provides different sub-objects as properties that group the available metadata into a number of categories, broadly corresponding to the categories listed on the [Keywords for Columns](#) page. You can determine the primary, or main type of metadata available using the *default value* of the **Metadata** object.

Property Name	Return Type	Description
< <i>default value</i> >	<i>string</i>	Returns a string indicating the primary type of metadata available in this object. The string will be one of the following: <i>none, video, audio, image, font, exe, doc</i> ,

		<p><i>other.</i></p> <p>Note that sometimes more than one type of metadata will be available. For example, author is a document field (and so found under the <b>doc</b> property), but pictures can have authors as well. In this instance, the <b>Metadata</b> object would provide both <b>ImageMeta</b> and <b>DocMeta</b> objects.</p> <p>If the returned string is <i>none</i> it means that no metadata is available for the file, and you should not attempt to access any of the other properties.</p>
audio	<i>object:</i> <a href="#">AudioMeta</a>	Returns an <a href="#">AudioMeta</a> object providing access to audio metadata. The properties of this object are generally returned as their appropriate underlying type (e.g. a numeric field like "track number" will be returned as an <i>int</i> ).
audio_text	<i>object:</i> <a href="#">AudioMeta</a>	Returns an <a href="#">AudioMeta</a> object that provides access to the unmodified text form of the audio metadata. This provides access to the same text as displayed in a Lister. For example, a numeric field like "track number" would be returned as a <i>string</i> rather than an <i>int</i> .
doc	<i>object:</i> <a href="#">DocMeta</a>	Returns a <a href="#">DocMeta</a> object providing access to document metadata.
doc_text	<i>object:</i> <a href="#">DocMeta</a>	Returns a <a href="#">DocMeta</a> object that provides access to the unmodified text form of the document metadata.
exe	<i>object:</i> <a href="#">ExeMeta</a>	Returns an <a href="#">ExeMeta</a> object providing access to executable (program) metadata.
exe_text	<i>object:</i> <a href="#">ExeMeta</a>	Returns an <a href="#">ExeMeta</a> object that provides access to the unmodified text form of the program metadata.
font	<i>object:</i> <a href="#">FontMeta</a>	Returns a <a href="#">FontMeta</a> object providing access to font file metadata.
image	<i>object:</i> <a href="#">ImageMeta</a>	Returns an <a href="#">ImageMeta</a> object providing access to picture metadata.
image_text	<i>object:</i> <a href="#">ImageMeta</a>	Returns an <a href="#">ImageMeta</a> object that provides access to the unmodified text form of the picture metadata.
other	<i>object:</i> <a href="#">OtherMeta</a>	Returns an <a href="#">OtherMeta</a> object that provides access to miscellaneous metadata.
tags	<i>collection:</i> <i>string</i>	Returns a collection of <i>strings</i> corresponding to the tags that are assigned to this item.
video	<i>object:</i> <a href="#">VideoMeta</a>	Returns a <a href="#">VideoMeta</a> object providing access to video metadata.
video_text	<i>object:</i> <a href="#">VideoMeta</a>	Returns a <a href="#">VideoMeta</a> object that provides access to the unmodified text form of the video metadata.

## Msg

The **Msg** object represents a [script dialog](#) input event message. It's returned by the [Dialog.GetMsg](#) method which you call when running the message loop for a [detached dialog](#).

Property Name	Return Type	Description
<default value>	<i>bool</i>	Returns <b>True</b> if the message is valid, or <b>False</b> if the dialog has been closed (which means you should exit your message loop).
checked	<i>int</i>	<p>If the event type is <b>checked</b>, this indicates the check state of the item. If checkboxes are used in automatic mode, this will be the <b>new</b> check state of the item. In manual mode, this will indicate the <b>existing</b> state and it's up to you to change the state if desired.</p> <p>Check states are <b>0</b> (unchecked), <b>1</b> (checked), <b>2</b> (indeterminate), <b>3</b> (unchecked/disabled), <b>4</b> (checked/disabled), <b>5</b> (indeterminate/disabled).</p>
control	<i>string</i>	<p>Returns the name of the control involved in the event. You can get a <a href="#">Control</a> object representing the control by passing this string to the <a href="#">Dialog.Control</a> method.</p> <p>For a <b>timer</b> event this returns the name of the timer that was triggered. For a <b>hotkey</b> event this returns the name of the hotkey. For a <b>drop</b> event this returns the name of the control that the files were dropped on.</p>
cx	<i>int</i>	For <b>resize</b> events, this property returns the new width of the dialog.
cy	<i>int</i>	For <b>resize</b> events, this property returns the new height of the dialog.
data	<i>int or bool</i>	<p>If the event type is <b>focus</b>, indicates the new focus state of the control - <b>True</b> if the control has gained the focus, or <b>False</b> if it's lost it.</p> <p>For a <i>combo box</i> or <i>list box</i> control: If the event type is <b>selchange</b> or <b>dblclk</b>, returns the data value associated with the selected item.</p> <p>For a two-state <i>check box</i> control or <i>radio button</i>: If the event type is <b>click</b>, returns a <i>bool</i> indicating the current check state.</p>

		<p>For a three-state <i>check box</i>: If the event type is <b>click</b>, returns an <i>int</i> representing the current state.</p> <p>If the event type is <b>timer</b>, this value indicates the number of milliseconds that have elapsed since the last time this timer was triggered.</p>
dialog	<i>string</i>	Returns the name of the parent dialog.
event	<i>string</i>	<p>Returns a string indicating the event that occurred. Currently defined events are:</p> <ul style="list-style-type: none"> <li>• <b>invalid</b>: The dialog has been closed.</li> <li>• <b>checked</b>: For a listview control with the <b>Checkboxes</b> property enabled, indicates that the checkbox of a list item has been clicked.</li> <li>• <b>click</b>: The control was clicked (e.g. a button, check box or radio button).</li> <li>• <b>dblclk</b>: An item in the list was double-clicked (list box, combo box or list view).</li> <li>• <b>rclick</b>: An item in the list was right-clicked (list box, list view).</li> <li>• <b>drop</b>: Files were dropped onto your dialog. The dialog template must have its <i>Accept Drops</i> property set to <b>True</b> to enable drag &amp; drop support.</li> <li>• <b>editchange</b>: The contents of an edit field were modified. For a list view this event indicates that the label of a list item was edited.</li> <li>• <b>focus</b>: The control gained or lost focus.</li> <li>• <b>hotkey</b>: A key combination added as a hotkey with the <a href="#">Dialog.AddHotkey</a> method has been pressed.</li> <li>• <b>resize</b>: The dialog was resized by the user. Only generated if the <a href="#">Dialog.want_resize</a> property has been set to <b>True</b>. Don't mix manual and automatic resizing with the same control: If you move or resize a control in response to this event, the control should not have any of the <b>resize</b> flags set in the dialog editor.</li> <li>• <b>selchange</b>: The selection was changed (list box, combo box, list view or tab).</li> <li>• <b>timer</b>: A periodic timer created with the <a href="#">Dialog.SetTimer</a> method has elapsed.</li> </ul>
focus	<i>bool</i>	Returns <b>True</b> if the control had focus when the message was generated.
index	<i>int</i>	Returns the current selection index for a <i>combo box</i> , <i>list box</i> or <i>tab control</i> .
mousex	<i>int</i>	Returns the horizontal position of the mouse cursor when the message was generated.
mousey	<i>int</i>	Returns the vertical position of the mouse cursor when the message was generated.

object	<i>variant</i>	For a <b>drop</b> event, this property returns a <a href="#">Vector</a> of <a href="#">Item</a> objects, representing the files that were dropped onto your dialog.
result	<i>bool</i>	Returns <b>True</b> if the message is valid, or <b>False</b> if the dialog has been closed.
tab	<i>string</i>	Returns the name of the parent tab (if the control is on a dialog that's inside a tab control).
value	<i>string</i>	For the <b>dblclk</b> , <b>editchange</b> and <b>selchange</b> events, returns the current contents of the edit field (or selected item label).

### ***OpenListerData***

If a [script add-in](#) implements the [OnOpenLister](#) event, the method receives an **OpenListerData** object when invoked when a new Lister opens.

Property Name	Return Type	Description
after	<i>bool</i>	Initially this is set to <b>False</b> , indicating that the event has been called before any tabs have been created. If you return <b>True</b> from the <a href="#">OnOpenLister</a> event, it will be called again and <b>after</b> will be set to <b>True</b> to indicate all tabs have been created.
lister	<i>object:</i> <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the newly opened Lister.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .

### ***OpenTabData***

If a [script add-in](#) implements the [OnOpenTab](#) event, the method receives an **OpenTabData** object when invoked when a new tab is opened.

Property Name	Return Type	Description
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
tab	<i>object:</i> <a href="#">Tab</a>	Returns a <a href="#">Tab</a> object representing the newly opened tab.

## OtherMeta

The **OtherMeta** object is retrieved from the [Metadata.other](#) property. It provides access to miscellaneous information about the file or folder.

Property Name	Return Type	Description
autodesc	<i>string</i>	An automatically generated description string for the item. This is the same string that is shown in the Description column in a Lister. Opus automatically generates the description for various types of files using the other metadata in ways that make the most sense.
dircount	<i>int</i>	For a folder, the size of which has been calculated via <a href="#">GetSizes</a> or similar, this provides the number of sub-folders directly underneath the folder.
dircounttotal	<i>int</i>	Similar to <b>dircount</b> , this provides the total number of sub-folders underneath the folder (this is a recursive count - it includes sub-sub-folders, sub-sub-sub-folders, etc.)
filecount	<i>int</i>	For a folder, the size of which has been calculated via <a href="#">GetSizes</a> or similar, this provides the number of files directly located in that folder.
filecounttotal	<i>int</i>	Similar to <b>filecount</b> , this provides the total number of files in the folder and all its sub-folders, sub-sub-folders, etc.
foldercontents	<i>string</i>	For a folder, the size of which has been calculated via <a href="#">GetSizes</a> or similar, this returns a string giving a summary of the contents of the folder.
nsdesc	<i>string</i>	A description automatically generated for the item by its parent virtual file system.
rating	<i>int</i>	Returns the user-assigned rating for this file or folder.
target	<i>object:Path</i>	Returns a <a href="#">Path</a> object representing the target path of shortcuts and links.
target_type	<i>string</i>	Returns a <i>string</i> indicating the type of the link ( <i>unknown, linkfile, dosfile, url, junction, softlink</i> ).
usercomment	<i>string</i>	Returns the user-assigned description for the file or folder.

## Path

The **Path** object represents a file or folder path. Many objects have properties that return a **Path** - for example, [Tab.path](#) returns the current folder in a tab as a **Path** object. You can create a new **Path** object from a string (or another **Path**) using the [DOpus.FSUtil.NewPath](#) method.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the full path as a string.
components	<i>int</i>	Returns the number of components in the path.
disks	<a href="#">Vector</a> : <i>int</i>	Returns a <a href="#">Vector</a> of <i>ints</i> representing the physical disk drive or drives that this path resides on.
drive	<i>int</i>	Returns the drive number the path refers to (1=A, 2=B, etc.) or 0 if the path does not specify a drive.
ext	<i>string</i>	Returns the filename extension of the path (the substring extending from the last . in the final component to the end of the string). This method does not check if the path actually refers to a file.  You can also change a path's file extension by setting this property (and strip the extension altogether by setting it to an empty string).
ext_m	<i>string</i>	Returns the filename extension of the path, taking multi-part extensions into account. For example, <b>ext</b> might return ".rar" whereas <b>ext_m</b> would return ".part1.rar".  You can't change the extension using <b>ext_m</b> , only <b>ext</b> .
filepart	<i>string</i>	Returns the filename part of the path (the last component).
longpath	<i>object:Path</i>	If this object represents a short pathname, this property returns the "long" equivalent.
pathpart	<i>string</i>	Returns the path minus the last component.
shortpath	<i>object:Path</i>	If this object represents a long pathname, this property returns the "short" equivalent, if it has one. Note that short paths are disabled by default in Windows 10.
stem	<i>string</i>	Returns the filename stem of the path (i.e. <b>filepart</b> minus <b>ext</b> ).
stem_m	<i>string</i>	Returns the filename stem taking multi-part extensions into account. For example, <b>stem</b> might return "pictures.part1" whereas <b>stem_m</b> would return "pictures".
test_parent	<i>bool</i>	Returns <b>True</b> if a call to the <b>Parent</b> method would succeed.
test_root	<i>bool</i>	Returns <b>True</b> if a call to the <b>Root</b> method would succeed.

Method Name	Arguments	Return Type	Description
Add	<string:name>	<i>none</i>	Adds the specified name to the path (it will become the last component).

Parent	<i>none</i>	<i>bool</i>	Removes the last component of the path. Returns <b>False</b> if the path does not have a valid parent.
Root	<i>none</i>	<i>bool</i>	Strips off all but the first component of the path. Returns <b>False</b> if the path is already at the root.
Set	<string:path> or < <b>Path</b> :path>	<i>none</i>	Sets the path represented by the <b>Path</b> object to the specified <i>string</i> . You can also set one <b>Path</b> object to the value of another.

## Progress

The **Progress** object lets you display and control a standard Directory Opus progress indicator dialog. This object can be obtained from the [Command.progress](#) property. The basic steps for using a **Progress** object are:

1. Initialize the fundamental properties.
2. Call the **Init** method to create the dialog (this creates it but does not show it to the user).
3. When ready, call the **Show** method to make the dialog visible.
4. Call the appropriate methods to initialize the state of the progress bars (by setting the total number of files, total byte size, etc).
5. As your operation progresses, advance the progress bars using methods like **StepFiles** and **StepBytes**.
6. If appropriate, poll the state of the *Abort* and other control buttons using **GetAbortState**.
7. Call the **Hide** method and destroy the object when your operation is finished.

Property Name	Return Type	Description
abort	<i>bool</i>	Before calling <b>Init</b> , set to <b>True</b> if the <i>Abort</i> button should be available, or <b>False</b> to disable it.
bytes	<i>bool</i>	Before calling <b>Init</b> , set to <b>True</b> if the dialog should show progress in bytes rather than whole files.
delay	<i>bool</i>	Before calling <b>Init</b> , set to <b>True</b> if the dialog should delay before appearing after the <b>Show</b> method is called. The delay is configured by the user in Preferences.
full	<i>bool</i>	Before calling <b>Init</b> , set to <b>True</b> to enable a "full size" progress indicator with two separate progress bars (one for files and one for bytes).
owned	<i>bool</i>	Before calling <b>Init</b> , set to <b>True</b> if the dialog should be owned by its parent window (the parent is given later, when the dialog is created via the <b>Init</b> method).
pause	<i>bool</i>	Before calling <b>Init</b> , set to <b>True</b> if the <i>Pause</i> button should be available.
skip	<i>bool</i>	Before calling <b>Init</b> , set to <b>True</b> if the <i>Skip</i> button should be available. (This just makes it so the <i>Skip</i> button can be enabled. You must still call <b>EnableSkip</b> later to actually enable it; usually once per file.)

Method Name	Arguments	Return Type	Description
AddFiles	<int:count> < <a href="#">FileSize</a> :bytes>	<i>none</i>	Adds the specified number of files to the operation total. The <i>bytes</i> argument is optional - in a "full size" progress indicator this lets you add to the total byte size of the operation.
ClearAbortState	<i>none</i>	<i>none</i>	<p>Clears the state of the three "control" buttons (<i>Abort</i> / <i>Pause</i> / <i>Skip</i>) so they no longer register as being clicked when <b>GetAbortState</b> is called.</p> <p>If you only want to clear the <i>Skip</i> state, you should normally do that as part of a call to <b>EnableSkip</b> instead. That way you avoid accidentally clearing one of the other states if they become set between you calling <b>GetAbortState</b> and <b>ClearAbortState</b>. In fact, there are very few situations where you should call <b>ClearAbortState</b>.</p>
EnableSkip	<bool:enable> <bool:delay> <bool:clear>	<i>none</i>	<p>Enables the progress dialog's <i>Skip</i> button. For <b>EnableSkip</b> to work, you must have set the <b>skip</b> property to <b>True</b> before the progress dialog was created by the <b>Init</b> method.</p> <p><b>enable</b>: If <b>True</b>, the <i>Skip</i> button should be enabled; otherwise, it should be disabled.</p> <p><b>delay</b> (optional, <b>True</b> by default): If <b>True</b>, there will be a short delay before the <i>Skip</i> button is enabled, with it temporarily disabled during the delay; otherwise, the change is instant. See below for why a delay is usually a good idea.</p> <p><b>clear</b> (optional, <b>True</b> by default): If <b>True</b>, any record of the user pushing the <i>Skip</i> will be cleared, such that <b>GetAbortState</b> no longer returns "s". You usually want this if the progress dialog just moved to a new item.</p> <p>If you support the <i>Skip</i> button, you should normally call <b>EnableSkip</b> once per file, just after you call <b>SetName</b> and similar methods. When used that way, you'll usually</p>

			want <b>delay</b> and <b>clear</b> to both be <b>True</b> , otherwise clicks of the <i>Skip</i> button intended for one file could affect the file(s) that come after it. For example, if a file takes a long time but then finishes just as the user gets tired of waiting and clicks <i>Skip</i> , the delay and cleared state ensure the unwanted click is harmless.
FinishFile	<i>none</i>	<i>none</i>	Finish the current file. If the byte size of the current file has been set the total progress will be advanced by any remaining bytes.
GetAbortState	<bool:autoPause> <string:wanted> <bool:simple>	<i>string</i>	<p>Polls the state of the three "control" buttons. This returns a <i>string</i> that indicates which, if any, of the three buttons have been clicked by the user. The button states are represented by the following letters in the returned string:</p> <p><b>a</b> - Abort <b>p</b> - Pause <b>s</b> - Skip</p> <p>If none of the states apply, an empty string is returned.</p> <p><b>autoPause</b> (optional, <b>False</b> by default): If <b>True</b>, pausing is handled for you automatically. Calls to <b>GetAbortState(True)</b> block while paused and don't return until unpaused; the "<b>p</b>" state is never returned. (Note that clicking <i>Skip</i> or <i>Abort</i> will implicitly unpause the operation.)</p> <p><b>wanted</b> (optional): If you only want to check one or two of the states, pass a string with their letters. For example, <b>GetAbortState(True,"ap")</b> will test for the <i>Abort</i> and <i>Pause</i> states, but not the <i>Skip</i> one. All states will be checked if the argument is an empty string or not given at all.</p> <p><b>simple</b> (optional, <b>True</b> by default): If <b>True</b>, the result string will have at most one letter, indicating the most important state. If <b>False</b>, it is possible for multiple states to be indicated at once. For example if <i>Skip</i> and then <i>Pause</i> are clicked, in that order, without the script clearing the <i>Skip</i> state, then <b>GetAbortState(False,"",False)</b> would</p>

			<p>return <b>"ps"</b> while <b>GetAbortState(False)</b> would return just <b>"p"</b>.</p> <p>To clear the state of the three buttons, call the <b>ClearAbortState</b> method. To clear just the <i>Skip</i> button's state, use the <b>EnableSkip</b> method.</p>
Hide	<i>none</i>	<i>none</i>	Hides the progress indicator dialog. The dialog object itself remains valid, and can be redisplayed with the <b>Show</b> method if desired.
HideFileByteCounts	<bool:show>	<i>none</i>	Hides or shows the <i>"XX bytes / YY bytes"</i> string in the progress dialog. You can use this to hide the string if the progress does not indicate a number of bytes (e.g. when it indicates a percentage). Pass <b>True</b> for the <i>show</i> argument to show the string and <b>False</b> to hide it.
Init	< <a href="#">Tab</a> :parent> or < <a href="#">Lister</a> :parent>  <string:title>	<i>none</i>	<p>Initializes the dialog. This method causes the actual dialog to be created, although it will not be displayed until the <b>Show</b> method is called. The fundamental properties shown above must be set before this method is called - once the dialog has been created they can not be altered.</p> <p>The <i>parent</i> parameter can be either a <a href="#">Tab</a> or a <a href="#">Lister</a> - this controls which window the dialog is centered over, and if the <b>owned</b> property is set to <b>True</b> which window it is owned by (always appears on top of). If no parent is provided the dialog will not be associated with any particular window.</p> <p>The <i>title</i> parameter specifies the window title of the dialog.</p>
InitFileSize	<i>none</i>	<i>none</i>	Resets the byte count for the current file to zero.
Restart	<i>none</i>	<i>none</i>	Resets the total completed file and byte counts to zero.
SetBytesProgress	< <a href="#">FileSize</a> :bytes>	<i>none</i>	Sets the total completed byte count.
SetFileSize	< <a href="#">FileSize</a> :bytes>	<i>none</i>	Sets the size of the current file.
SetFiles	<int:count>	<i>none</i>	Sets the total number of files.
SetFilesProgress	<int:count>	<i>none</i>	Sets the total completed file count.
SetFromTo	<string:header> <string:from> <string:to>	<i>none</i>	Sets the text at the top of the dialog that indicates the source and destination of an operation. The <i>header</i> argument refers to the string that normally says <i>From:</i> - this allows

			<p>you to change it in case that term is not applicable to your action. The <i>from</i> argument is the source path, and the <i>to</i> argument (if there is one) is the destination path. Note that if you specify a destination path this always has a <i>To:</i> header appended to it.</p> <p>If you omit the <i>to</i> argument entirely (not just passing an empty string), the destination line will become blank, including the <i>To:</i> header. Use that if you want the second line to be used sometimes but not always. If you never want anything on the second line, use the <b>SetStatus</b> method instead as it will not add space for the extra line.</p>
SetName	<string:name>	none	Sets the name of the current file.
SetPercentProgress	<int:percent>	none	Sets the current progress as a percentage (from 0 to 100).
SetStatus	<string:status>	none	<p>Sets the text displayed in the status line at the top of the dialog.</p> <p>This sets a single-line status message, while <b>SetFromTo</b> can be used to indicate source and destination paths on two lines.</p>
SetTitle	<string:title>	none	Sets the title of the dialog.
SetType	<string:type>	none	Sets the type of the current item - either <i>file</i> or <i>dir</i> .
Show	none	none	Displays the progress indicator dialog. Call this once you have created the dialog using the <b>Init</b> method.
SkipFile	<bool:complete>	none	Skips over the current file. Set the <i>complete</i> argument to <b>True</b> to have the file counted as "complete", or <b>False</b> to count it as "skipped".
StepBytes	< <a href="#">FileSize</a> :bytes>	none	Step the byte progress indicator the specified number of bytes.
StepFiles	<int:count>	none	Step the file progress indicator the specified number of files.

## QuickFilter

The **QuickFilter** object provides access to information about the quick filter state in a tab. It's returned by the [Tab.quickfilter](#) property.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the current filter string, if any.
autoclear	<i>bool</i>	Returns <b>True</b> if the auto-clear mode is set in Preferences.
autostar	<i>bool</i>	Returns <b>True</b> if the auto-star mode is set in Preferences.
disable	<i>bool</i>	Returns <b>True</b> if the filter is disabled.
easymode	<i>bool</i>	Returns <b>True</b> if easy mode is selected.
filter	<i>string</i>	Returns the current filter string.
hidealldirs	<i>bool</i>	Returns <b>True</b> if all folders are being hidden.
hideallfiles	<i>bool</i>	Returns <b>True</b> if all files are being hidden.
overrideflatview	<i>bool</i>	Returns <b>True</b> if filtering in flatview is enabled.
partial	<i>bool</i>	Returns <b>True</b> if partial matching is enabled.
realtime	<i>bool</i>	Returns <b>True</b> if realtime filtering is enabled.
regex	<i>bool</i>	Returns <b>True</b> if regular expression mode is enabled.
showalldirs	<i>bool</i>	Returns <b>True</b> if all folders are being shown.
showallfiles	<i>bool</i>	Returns <b>True</b> if all files are being shown.
showeverything	<i>bool</i>	Returns <b>True</b> if <a href="#">Show Everything</a> mode is on, which overrides (almost) all filtering.

## Rect

The **Rect** object represents a rectangle.

Property Name	Return Type	Description
left	<i>int</i>	Returns the left edge of the rectangle.
top	<i>int</i>	Returns the top edge of the rectangle.
right	<i>int</i>	Returns the right edge of the rectangle (note that this value is actually 1 outside the right edge).
bottom	<i>int</i>	Returns the bottom edge of the rectangle (note that this value is actually 1 outside the bottom edge).
width	<i>int</i>	Returns the width of the rectangle.
height	<i>int</i>	Returns the height of the rectangle.

## Results

The **Results** object provides information about the outcome of a function run by the [Command](#) object. It is obtained from the **Command.results** property.

Property Name	Return Type	Description
result	<i>int</i>	Indicates whether or not the command ran successfully. Zero indicates the command could not be run or was aborted; any other number indicates the command was run for at least some files. (Note that this is not the "exit code" for external commands. For external commands it only indicates whether or not Opus launched the command. If you need the exit code of an external command, use the WScript.Shell Run or Exec methods to run the command.)
newtabs	<i>collection:</i> <a href="#">Tab</a>	This property returns a collection of <a href="#">Tab</a> objects representing any new tabs created by the command.
newlisters	<i>collection:</i> <a href="#">Lister</a>	This property returns a collection of <a href="#">Lister</a> objects representing any new Listers created by the command.

## Script

The **Script** object is one of the two global script objects provided by Opus. This object is provided to [script addins](#) when their various event handlers are invoked (other than for the [OnInit](#) event). It provides information relating to the script itself.

Property Name	Return Type	Description
config	<i>object:</i> <a href="#">ScriptConfig</a>	Returns a <a href="#">ScriptConfig</a> object representing the configuration values for this script. In the <a href="#">OnInit</a> method a script can define the properties that make up its configuration - the user can then edit these values in Preferences. The object returned by the <b>config</b> property represents the values that the user has chosen.
file	<i>string</i>	Returns the path and filename of this script.
vars	<i>object:</i> <a href="#">Vars</a>	Returns a <a href="#">Vars</a> object that represents the variables that are scoped to this particular script. This allows scripts to use variables that persist from one invocation of the script to another.

Method Name	Arguments	Return Type	Description
HttpHelpEnabled	<i>none</i>	<b>bool</b>	Returns <b>True</b> if local HTTP help is enabled (that is, if help is shown in the user's web browser), <b>False</b> if the old <i>HtmlHelp</i> -style help is enabled. If HTTP help is enabled, your script is able to add its own help pages via the <a href="#">OnGetHelpContent</a> event, and it can trigger the display of its own help pages using the <b>ShowHelp</b> method.
InitColumns	<i>none</i>	<i>none</i>	If your script implements the <a href="#">OnAddColumns</a> event, you can call the <b>InitColumns</b> method at any time to reinitialize your columns. You may want to do this, for example, in response to the user modifying your script's configuration.
InitCommands	<i>none</i>	<i>none</i>	If your script implements the <a href="#">OnAddCommands</a> event, you can call the <b>InitCommands</b> method at any time to reinitialize your commands. You may want to do this, for example, in response to the user modifying your script's configuration.
LoadHelpFile	<string:name>	<i>string</i>	Using the <a href="#">OnGetHelpContent</a> event your script can add its own content to the F1 help. If your script is bundled as a <a href="#">script package</a> you can include .html files in a sub-directory of the package called <b>help</b> , and then load them easily using this method. You can then pass the loaded data to the <a href="#">GetHelpContentData.AddHelpPage</a> method to add the page.
LoadHelpImage	<string:name>	<i>object:Blob</i>	If your script is bundled as a <a href="#">script package</a> you can include PNG and JPG image files in a sub-directory of the package called <b>help</b> , and then load them easily using this method. You can then pass the loaded data to the <a href="#">GetHelpContentData.AddHelpImage</a> method to add the image.
LoadResources	<string:name> or <string:XML>	<i>none</i>	Loads external <a href="#">script resources</a> and makes them available to the script. You can either provide a filename or a raw XML string. If your script is bundled as a <a href="#">script package</a> , the resource file must have a <b>.odxml</b> extension for <b>LoadResources</b> to be able to find it in the package.
RefreshColumn	<string:name>	<i>none</i>	If your script implements any custom columns, you can use this method to cause them to be regenerated if they are currently shown in any tabs. You may want to do this,

			for example, in response to the user modifying your script's configuration. Pass the name of the column you want to regenerate as the argument to this method.
ShowHelp	<string:page>	none	<p>If your script adds its own help pages via the <a href="#">OnGetHelpContent</a> event, and the user has http help enabled, you can call this method to display your help in the user's web browser. You might want to do this when the user clicks a <b>Help</b> button in your <a href="#">script dialog</a>, for example. You can use the <b>HttpHelpEnabled</b> method to check if http help is enabled before calling this function.</p> <p>The optional parameter must be the name of the desired page to show, which corresponds to the name you supplied when you added it in the <a href="#">OnGetHelpContent</a> event handler. If you omit this parameter then your first help page will be shown.</p>

## ScriptColumn

In a script's [OnInit](#) method it can call the [ScriptInitData.AddColumn](#) method to [add custom columns](#) to Opus. These columns can be displayed in file displays and infotips, and can be searched on using the [Advanced Find](#) function. Each call to **AddColumn** returns a **ScriptColumn** object that the script needs to initialize.

Property Name	Return Type	Description
autogroup	<i>bool</i>	If this is set to <b>True</b> (which is the default), and the file display is <a href="#">grouped</a> by this column, Opus will generate the groups automatically based on the column value. If you set this to <b>False</b> , Opus will expect you to provide grouping information in your <a href="#">OnScriptColumn</a> function.
autorefresh	<i>bool or int</i>	Set to <b>True</b> (or <b>1</b> ) to force Opus to update the value for this column when a file changes. You can also set this value to <b>2</b> to force Opus to update the value when the file's attributes change (normally it would only update if the file modification time or size changed).
defsort	<i>int</i>	This property lets you control the default sort behavior for your column. Normally when the user clicks the column header to sort by a column the column is initially sorted in ascending order, and then clicking again reverses the sort order. If you set <b>defsort</b> to <b>-1</b> , the first click on the column header will sort in descending

		order. Date and size fields have this behavior set by default.
defwidth	<i>int</i> or <i>string</i>	Specifies a default width for your column, which will be used unless the file display has auto-sizing enabled. If you specify a simple integer value this represents a width measured in average characters (e.g. <i>12</i> specifies 12 average characters wide). You can also specify an absolute number of pixels by adding the <i>px</i> suffix (e.g. " <i>150px</i> " specifies 150 pixels).
graph_colors	<i>object</i> : <a href="#">Vector</a>	<p>For graph columns, specifies the first graph color set. The graph will be displayed in these colors as long as its percentage is below the threshold.</p> <p>You can either specify a single color (in <i>r,g,b</i> or <i>#rrggbb</i> format), in which case the graph will be a flat solid color, or exactly five colors to configure the graph's gradient. In the second case, the five colors correspond to <i>outer bright</i>, <i>inner bright</i>, <i>inner dark</i>, <i>outer dark</i>, and <i>flat</i>. The first four control the gradient and the fifth (flat) is used when gradients are disabled.</p> <p>The <b>graph_colors</b> property returns a <a href="#">Vector</a>; you need to use the <b>push_back()</b> method to add your colors to it.</p>
graph_colors2	<i>object</i> : <a href="#">Vector</a>	Similar to <b>graph_colors</b> , this property lets you configure a second set of colors for a graph column that will be used when the graph value exceeds the threshold.
graph_threshold		For graph columns, specifies the percentage threshold at which the graph will switch from the first color set to the second (e.g. a blue graph goes red to indicate a drive is nearly full). Set the threshold to <b>-1</b> to disable the second color set altogether.
grouporder	<i>string</i>	If the <b>autogroup</b> property is set to <b>False</b> , the <b>grouporder</b> property lets you control the order your column's groups appear in. Each group should be listed in the string in the desired order, separated by a semi-colon (e.g. " <i>Never Modified;Modified</i> "). If not provided, groups will default to sorting alphabetically.
header	<i>string</i>	If this property is set, this defines the string that will be displayed in the column header when this column is added to a Lister. If not set, the <b>label</b> value will be used.
infotiponly	<i>bool</i>	Set this to <b>True</b> if you want your column to be only available for use in <a href="#">Info Tips</a> . You might want this if your column takes a significant amount of time to return a value, in which case the user would probably only want to use it in an Info Tip so they can see the value on demand. If set to <b>False</b> (the default) the column will be available everywhere.

justify	<i>string</i>	This field lets you control the justification of your column. If not specified, columns default to left justify. Acceptable values are <i>center</i> , <i>left</i> , <i>right</i> and <i>path</i> .
keyscroll	<i>bool</i>	If this is set to <b>True</b> , and the user has the <b>Sort-field specific key scrolling</b> Preferences option enabled, then your column will participate in this special mode.
label	<i>string</i>	Use this to set a label for the column. This is displayed in the column header when the column is added to a Details/Power mode file display (unless overridden by the <b>header</b> property), and in various column lists such as in the <a href="#">Folder Options</a> dialog.
match	<a href="#">Vector</a> : <i>string</i>	If you add strings to this <a href="#">Vector</a> (e.g. via the <b>push_back</b> method) it will be used to provide a drop-down list of possible values when searching on this column using the <a href="#">Advanced Find</a> function.
maxstars	<i>int</i>	If the column type is set to <i>stars</i> this property lets you specify the maximum number of stars that will be used. This is used to ensure the column is sized correctly.
method	<i>string</i>	This is the name of the method in your script that provides the actual values for your new column. This would typically be set to <i>OnXXXXX</i> where <i>XXXXX</i> is the name of the command, however any method name can be used.  When the method is invoked it is passed a single argument, a <a href="#">ScriptColumnData</a> object. Generically this method is referred to as <a href="#">OnScriptColumn</a> .
multicol	<i>bool</i>	If your script implements multiple columns that require common calculations to perform, you may wish to set the <b>multicol</b> property. If this is set to <b>True</b> then your column handler function has the option of returning data for multiple columns simultaneously, rather than just the specific column it is being invoked for.  When your handler is called, the <a href="#">ScriptColumnData</a> object won't contain the usual <b>group</b> , <b>sort</b> , <b>type</b> and <b>value</b> properties. Instead, it will have a <b>columns</b> property that points to a <a href="#">Map</a> that lets you set the values for one or more of your columns at once.  For example, you might set the value of a column called <i>MyColumn</i> like this:  <pre>scriptColData.columns("MyColumn").value = "My Column Value";</pre>
name	<i>string</i>	This is the raw name of the column. This determines the name that can be used to control the column programmatically (for example, the <b>Set COLUMNSTOGGLE</b> command can be used to toggle

		<p>a column on or off by name).</p> <p>The name of a custom column is built from a combination of the name of the script that provides the column and the raw name of the column itself, and is preceded by the prefix <i>scp</i>:. For example, if your script were called <i>My Script</i> and your column's name were <i>My Column</i>, you could toggle this column using the command <b>Set COLUMNSTOGGLE="scp:My Script/My Column"</b>. You can use the button editor menus to build the command automatically, if you are unsure of anything.</p>
namerefresh		Set to <b>True</b> to force Opus to update the value for this column when a file's name changes.
nogroup	<i>bool</i>	Set to <b>True</b> to prevent the file display being grouped by this column.
nosort	<i>bool</i>	Set to <b>True</b> to prevent the file display being sorted by this column.
type	<i>string</i>	<p>This field lets you set the default type of the column.</p> <p>If not specified, columns default to plain text.</p> <p>Acceptable values are:</p> <p><b>number:</b></p> <p style="padding-left: 40px;">The column displays integer numbers.</p> <p><b>double:</b></p> <p style="padding-left: 40px;">The column displays floating point (fractional) numbers.</p> <p><b>size:</b></p> <p style="padding-left: 40px;">The column displays file sizes (automatically displays bytes, KB, MB, etc.).</p> <p><b>zip:</b></p> <p style="padding-left: 40px;">The column displays file sizes (uses the settings for Zip file sizes).</p> <p><b>graph:</b></p> <p style="padding-left: 40px;">The column displays a bar graph (expects a value from 0 to 100).</p> <p><b>graphrel:</b></p>

The column displays a bar graph. Opus automatically keeps track of the minimum and maximum values provided and scales the graph accordingly.

**graphrel0:**

Similar to **graphrel** except the minimum value is always 0, and Opus keeps track of the maximum value.

**igraph:**

The column displays an inverted bar graph.

**igraphrel:**

Inverted relative bar graph.

**igraphrel0:**

Inverted bar graph relative to 0.

**percent:**

The column displays a percentage.

**percentrel:**

Relative percentage.

**percentrel0:**

Percentage relative to 0.

**date:**

The column displays a date.

**time:**

The column displays a time.

**datetime:**

The column displays both a date and a time.

**stars:**

The column displays stars (similar to the built-in *Rating* column).

For plain text columns, you can specify **numeric sort** or **nonnumeric sort** to override the *"numeric order filename sorting"* setting in Folder Options. Similarly, **word sort** or **noword sort** can be used to override the *"word sort*

		<p>(<i>special handling for hyphens, etc.</i>)" setting. You can also combine both options, e.g. <b>nonnumericssort,nowordsort</b> to request only basic sorting. Leave the type unset, or set it to an empty string, for plain text data which respects the Folder Options sort settings.</p> <p>For <i>date</i>, <i>time</i> and <i>datetime</i> columns, you can also specify <b>utc</b> to have the values automatically converted from UTC to local time (e.g. <b>datetime,utc</b>).</p> <p>For <i>number</i> and <i>double</i> columns, you can also specify <b>signed</b> to have the values treated as signed rather than unsigned (e.g. <b>number,signed</b>).</p> <p>For the graph columns, you can use <b>graph_colors</b>, <b>graph_colors2</b> and <b>graph_threshold</b> to configure the graph's appearance.</p> <p>Your <a href="#">OnScriptColumn</a> method can override the type on a per-file basis, however this field sets the default type and also controls the behavior of the <a href="#">Advanced Find</a> function when searching using your column.</p>
userdata	<i>variant</i>	Allows you to associate a data value with a column. The value will be passed to your column handler in the <a href="#">ScriptColumnData.userdata</a> property

## ScriptColumnData

The **ScriptColumnData** object is passed to the script-defined entry points for any [custom columns](#) added by a [script add-in](#). The method name for these events is defined by the script itself, but generically it's referred to as [OnScriptColumn](#). Note that the fields **group**, **sort**, **type** and **value** are settable and are the way your method returns values for your column.

Property Name	Return Type	Description
col	<i>string</i>	Provides the name of the column that Opus wants the script to return the value for. If you use the same <a href="#">OnScriptColumn</a> method to provide multiple columns you can use this to tell the columns apart.
columns	<i>object:Map</i>	If the <a href="#">ScriptColumn.multicol</a> value is set to <b>True</b> when the column is added, then this property provides a <a href="#">Map</a> that lets you return the values of one or more columns at once.

		<p>You may want to use this method if your script returns multiple columns that all share common calculations (e.g. reading the contents of a folder). That way, you can avoid repeating potentially time consuming operations when you're called for the second and subsequent columns.</p> <p>The <a href="#">Map</a> contains one member element for each of your columns. Each member element has <b>group</b>, <b>group_type</b>, <b>sort</b>, <b>type</b>, <b>userdata</b>, and <b>value</b> properties which are equivalent to the ones described below.</p> <p>For example, you might set the value of a column called <i>MyColumn</i> like this:</p> <pre><i>scriptColData.columns("MyColumn").value = "My Column Value";</i></pre> <p>You should check if a column exists in the map before populating data for it. In some situations, Opus will only request a some of the columns your add-in supports, not all of them.</p> <p>If you do not fill in data for a column which Opus still needs, Opus will call your method again to request it, with its name in the <b>col</b> property (but still in multi-column mode). You can take advantage of this if calculating one piece of data yields values for some of your columns but not all of them. You do not need to populate every column if you the data is not available, but you should at least populate the <b>col</b> column.</p> <p>As a consequence of the above, when using multi-column mode you should <i>always</i> set some kind of value for any column you have spent time calculating, even if the result of the calculation is that nothing should be shown in the column. If nothing should be shown, set the value to an empty string (this is fine even if the column normally displays numbers or another type of data). If you don't set any value at all, Opus will assume you haven't calculated that column yet and may call your script again to ask for it, which could cause you to waste time re-calculating it when you already know it is blank.</p>
group	string	<p>If the <a href="#">ScriptColumn.autogroup</a> value is set to <b>False</b> when the column is added, you should set this value to indicate the group that this file should be placed in when the list is grouped by your column. If you don't provide a group then this file will go into the <i>Unspecified</i> group.</p>

		<p>If <b>autogroup</b> is set to <b>True</b> this value is ignored.</p> <p>Note that if the <a href="#">ScriptColumn.multicol</a> value is set to <b>True</b> when the column is added then this property will be found inside the <b>columns Map</b>.</p>
group_type	<i>string</i>	<p>If the group is set via the <b>group</b> property, <b>group_type</b> lets you control the formatting of the group title using the same keywords as the <b>type</b> field (e.g. you can supply a number and have the group title formatted as a file size by setting <code>group_type="size"</code>).</p> <p>Note that if the <a href="#">ScriptColumn.multicol</a> value is set to <b>True</b> when the column is added then this property will be found inside the <b>columns Map</b>.</p>
item	<i>object: <a href="#">Item</a></i>	Returns an <a href="#">Item</a> object representing the file or folder that Opus wants the script to return the column value for.
sort	<i>variant</i>	<p>Lets you control the sort order of your column by providing a <i>sort key</i> that can be different to the <b>value</b>. If provided, and the list is sorted by your column, Opus will use the value of this field to position this item rather than the <b>value</b> value.</p> <p>Note that if the <a href="#">ScriptColumn.multicol</a> value is set to <b>True</b> when the column is added then this property will be found inside the <b>columns Map</b>.</p>
tab	<i>object: <a href="#">Tab</a></i>	Returns a <a href="#">Tab</a> object representing the tab that contains the item.
type	<i>string</i>	<p>Lets you override the default type of the column (set via <a href="#">ScriptColumn.type</a> when the column was added) on a per-file/folder basis.</p> <p>If not specified, and no default was specified either, then columns default to plain text.</p> <p>Note that if the <a href="#">ScriptColumn.multicol</a> value is set to <b>True</b> when the column is added then this property will be found inside the <b>columns Map</b>.</p> <p>Acceptable values are:</p> <p><b>number:</b></p> <p style="padding-left: 40px;">The column displays integer numbers.</p> <p><b>double:</b></p> <p style="padding-left: 40px;">The column displays floating point (fractional) numbers.</p>

**size:**

The column displays file sizes (automatically displays bytes, KB, MB, etc.).

**zip:**

The column displays file sizes (uses the settings for Zip file sizes).

**graph:**

The column displays a bar graph (expects a value from 0 to 100).

**igraph:**

The column displays an inverted bar graph.

**percent:**

The column displays a percentage.

**date:**

The column displays a date.

**time:**

The column displays a time.

**datetime:**

The column displays both a date and a time.

**stars:**

The column displays stars (similar to the built-in *Rating* column). The **value** should be in the form "x" or "x/y".

For *date*, *time* and *datetime* columns, you can also specify **utc** to have the values automatically converted from UTC to local time (e.g. **datetime,utc**).

For *number* and *double* columns, you can also specify **signed** to have the values treated as signed rather than unsigned (e.g. **number,signed**).

The options above are a subset of those you can specify via [ScriptColumn.type](#), since not all options make sense on a per-file/folder basis.

		Note that if you mix different types within the one column then the results you get when sorting by this column, or searching on your column using the <a href="#">Advanced Find</a> function, may be hard to predict.
value	<i>variant</i>	<p>This field is how your method returns the actual value for your column - that is, the information that is displayed to the user in this column for each file and folder.</p> <p>If the type for this column has been set (either by <b>ScriptColumnData.type</b> or <a href="#">ScriptColumn.type</a>) then Opus will try to convert the provided value to the specified type. If the type is not set then Opus will treat the value as a plain text string.</p> <p>If you don't provide a sort key via the <b>sort</b> field then Opus will also use this value to order the list when the list is sorted by this column.</p> <p>Note that if the <a href="#">ScriptColumn.multicol</a> value is set to <b>True</b> when the column is added then this property will be found inside the <b>columns Map</b>.</p>
userdata	<i>variant</i>	<p>This returns the value associated with this column via <a href="#">ScriptColumn.userdata</a> (if any) when the column was added.</p> <p>Note that if the <a href="#">ScriptColumn.multicol</a> value is set to <b>True</b> when the column is added then this property will be found inside the <b>columns Map</b>.</p>

## ScriptCommand

In a script's [OnInit](#) method it can call the [ScriptInitData.AddCommand](#) method to [add commands](#) to the Opus internal command set. Each call to **AddCommand** returns a **ScriptCommand** object that the script needs to initialize.

Property Name	Return Type	Description
desc	<i>string</i>	Use this to set a description for the command, that is displayed in the <a href="#">Customize</a> dialog when the user selects the command from the <a href="#">Commands tab</a> .
hide	<i>bool</i>	Set to <b>True</b> to hide this command from the drop-down command list shown in the <a href="#">command editor</a> . This lets you add commands that can still be used in buttons and hotkeys but won't clutter up the command list.
icon	<i>string</i>	Use this property to assign a default icon to this command. You can specify the name of an internal icon (if you want

		to specify an icon from a particular set, use <i>setname:iconname</i> - use this if you have bundled your script in a <a href="#">script package</a> with its own icon set) or the path of an external icon or image file.
label	<i>string</i>	Use this to set a label for the command. This is displayed in the <a href="#">Commands tab</a> of the <a href="#">Customize</a> dialog (under the <i>Script Commands</i> category), and will form the default label of the button created if the user drags that command out to a toolbar.  The actual name of the command (used to invoke the command) is assigned through the <b>name</b> property.
method	<i>string</i>	This is the name of the method that Opus will call in your script when the command is invoked. This would typically be set to <i>OnXXXXX</i> where <i>XXXXX</i> is the name of the command, however any method name can be used.  When the method is invoked it is passed a single argument, a <a href="#">ScriptCommandData</a> object. Generically this method is referred to as <a href="#">OnScriptCommand</a> .
name	<i>string</i>	This is the name of the command. This determines the name that will invoke the command when it is used in buttons and hotkeys.
template	<i>string</i>	This lets you specify an optional command line template for the command. This is a string in the form <i>ARGNAME1/MOD,ARGNAME2/MOD,ARGNAME3/MOD</i> , etc, where <i>ARGNAME</i> is the name of the argument and <i>/MOD</i> are one or more <a href="#">modifiers used to indicate the argument type</a> . The command line template can specify as many arguments as needed.  When your command is invoked and its <a href="#">OnScriptCommand</a> event is triggered, any arguments supplied on the command line are parsed according to this template and provided via the <a href="#">ScriptCommandData.func.args</a> property.

### ***ScriptCommandData***

The **ScriptCommandData** object is passed to the script-defined entry points for any [internal commands](#) added by a [script add-in](#). The method name for these events is defined by the script itself, but generically it's referred to as [OnScriptCommand](#).

Property Name	Return Type	Description
---------------	-------------	-------------

cmdline	<i>string</i>	This returns the original command line that invoked the command. If any arguments were provided on the command line they are available in parsed form from the <b>func.args</b> property.
func	<i>object:</i> <a href="#">Func</a>	Returns a <a href="#">Func</a> object relating to this function. This provides access to information about the function's environment (source and destination tabs, etc) as well as any variables and parsed command line arguments.

## ScriptConfig

The **ScriptConfig** object is accessed via the [ScriptInitData.config](#) and the [Script.config](#) properties. The [ScriptInitData.config](#) property allows a script (in its [OnInit](#) method) to specify what configuration properties it supports, and provide default values for them. The properties assigned in [OnInit](#) will then be available in Preferences for the user to edit, and the user-edited configuration can then be accessed by other script methods using [Script.config](#).

Property Name	Return Type	Description
<configuration property>	<i>variant</i>	<p>The properties of the <b>ScriptConfig</b> object are entirely determined by the script itself.</p> <p>In the <b>OnInit</b> method, assign the default values of any configuration properties you want to this object. The type of each default value controls the type of the property.</p> <p>The Preferences page only supports editing certain types of variables, so you must only assign properties of compatible types. Preferences supports:</p> <ul style="list-style-type: none"> <li>• Boolean options (<b>True</b> or <b>False</b>) - the variable type must be <i>bool</i></li> <li>• Numeric options - the variable type must be <i>int</i></li> <li>• String options - the variable type must be <i>string</i></li> <li>• Multi-line string options - the variable type must be <i>string</i> and must contain at least one <i>CR/LF</i> pair. Note that a trailing <i>CR/LF</i> will be removed from the default value.</li> <li>• Multiple string options - the variable type must be a <a href="#">Vector</a> of <i>strings</i></li> </ul>

- Drop-down list - the variable type must be a [Vector](#) with an *int* as the first element (to specify the default selection), and strings for the remaining elements.

## ScriptInitData

The **ScriptInitData** object is passed to the [OnInit](#) event in a [script add-in](#). The script should initialize the various properties to identify itself, and can optionally [add internal commands](#) using the **AddCommand** method, and [custom columns](#) using the **AddColumn** method, before returning.

Property Name	Return Type	Description
config	object: <a href="#">ScriptConfig</a>	Returns a <a href="#">ScriptConfig</a> object, that the script can use to initialize its default configuration. Properties added to the object in this method will be displayed to the user in Preferences, allowing them to change their value and thus configure the behavior of the script.
config_desc	object: <a href="#">Map</a>	This lets you assign descriptions for your script's configuration options that are shown to the user in the editor dialog. To do this, set this property to a <a href="#">Map</a> created via the <a href="#">DOpusFactory.Map</a> method, filled with name/description string pairs.
config_groups	object: <a href="#">Map</a>	This lets you organize your script's configuration options into groups when shown to the user in the editor dialog. The group names are arbitrary - configuration options with the same group name will appear grouped together. Set this property to a <a href="#">Map</a> created via the <a href="#">DOpusFactory.Map</a> method, filled with name/group string pairs.
copyright	string	Lets the script specify a copyright message that is displayed to the user in Preferences.
default_enable	bool	Set this to <b>True</b> if the script should be enabled by default, or <b>False</b> if it should be disabled by default. The user can enable or disable scripts using Preferences - this simply controls the default state.
desc	string	Lets the script specify a description message that is displayed to the user in Preferences.
early dblclk	bool	Set this to <b>True</b> if your script implements the <a href="#">OnDoubleClick</a> event and (for performance reasons) you want to be called with only a path to the double-

		clicked item rather than a full <a href="#">Item</a> object. See the <a href="#">OnDoubleClick</a> event documentation for more details.
file	<i>string</i>	Returns the path and filename of this script.
group	<i>string</i>	Lets you specify an arbitrary group for this script. If scripts specify a group they will be displayed in that group in the list in Preferences.
log_prefix	<i>string</i>	Lets the script specify a string that will be prepended to any log output it performs. If not set the name of the script is used by default.
min_version	<i>string</i>	Specifies the minimum Opus version required. If the current version is less than the specified version the script will be disabled. You can specify the major version only (e.g. "11"), a major and minor version (e.g. "11.3") or a specific beta version (e.g. "11.3.1" for 11.3 Beta 1).
name	<i>string</i>	Lets the script specify a display name for the script that is shown in Preferences.
startup	<i>bool</i>	The <b>OnInit</b> method is called in two different circumstances - once during Opus startup, and again if the script is installed or edited when Opus is already running. This property will return <b>True</b> if the <b>OnInit</b> method is being called during Opus startup, or <b>False</b> for any other time.
url	<i>string</i>	Lets you provide a URL where the user can go to find out more about your script (it's displayed to the user in Preferences).
vars	<i>object:Vars</i>	Returns a <a href="#">Vars</a> collection of user and script-defined variables that are local to this script. These variables are available to other methods in the script via the <a href="#">Script.vars</a> property.
version	<i>string</i>	Lets the script specify a version number string that is displayed to the user in Preferences.

Method Name	Arguments	Return Type	Description
AddColumn	<i>none</i>	<i>object:ScriptColumn</i>	<p>Adds a new information column to Opus. The returned <a href="#">ScriptColumn</a> object must be properly initialized. A script add-in can add as many columns as it likes, and these will be available in file displays, infotips and the <a href="#">Advanced Find</a> function.</p> <p>Instead of adding columns in <b>OnInit</b>, your script can implement the <a href="#">OnAddColumns</a> method. This is more</p>

			flexible as it allows you to access your script's configuration at the time you add columns, and columns can be dynamically added and removed while Opus is running. If <a href="#">OnAddColumns</a> is implemented then this method is unavailable in <b>OnInit</b> .
AddCommand	<i>none</i>	<i>object:</i> <a href="#">ScriptCommand</a>	<p>Adds a new internal command to Opus. The returned <a href="#">ScriptCommand</a> object must be properly initialized. A script add-in can add as many internal commands as it likes to the Opus internal command set.</p> <p>Instead of adding commands in <b>OnInit</b>, your script can implement the <a href="#">OnAddCommands</a> method. This is more flexible as it allows you to access your script's configuration at the time you add commands, and commands can be dynamically added and removed while Opus is running. If <a href="#">OnAddCommands</a> is implemented then this method is unavailable in <b>OnInit</b>.</p>

## ScriptStrings

The **ScriptStrings** object is returned by the [DOpus.strings](#) property. It lets you access any strings defined via [string resources](#).

Property Name	Return Type	Description
langs	<i>object:</i> <a href="#">Vector</a>	Returns a <a href="#">Vector</a> of strings representing the languages that strings have been defined for.

Method Name	Arguments	Return Type	Description
Get	<string:name> <string:language>	<i>string</i>	Returns the text of a string specified by name. The name must match the name used in the string

			resources.  Optionally you can provide a language name as the second parameter, to retrieve a string from a particular language. Otherwise, the string is returned in the current language.
HasLanguage	<string:language>	<i>bool</i>	Returns <b>True</b> if strings in the specified language are defined in the resources.

## ShellProperty

The **ShellProperty** object represents a shell property - an item of metadata for a file or folder that comes from Windows or third-party extensions (as opposed to metadata from Opus's native metadata system).

The [FSUtil.GetShellPropertyList](#) method lets you retrieve a list of available shell properties. You can then use [FSUtil.GetShellProperty](#) or [Item.ShellProp](#) to retrieve the value of a property for a particular file.

Property Name	Return Type	Description
<b>defwidth</b>	<i>int</i>	The default width in pixels a column displaying this property should use.
<b>display_name</b>	<i>string</i>	The display name of this property (the name that should be shown to users).
<b>justify</b>	<i>string</i>	The default column justification for this property ( <b>left</b> , <b>right</b> , <b>center</b> ).
<b>pkey</b>	<i>string</i>	The PKEY (property key) for this property. This is a property's unique ID and the canonical way to refer to a property. You can use the <b>raw_name</b> and <b>display_name</b> values to access properties as well, but they are potentially inaccurate (since it's possible to have two properties with the same name) and also slower as the property has to be looked up by name each time.
<b>raw_name</b>	<i>string</i>	An internal name used by the property provider.
<b>type</b>	<i>string</i>	The type of data this property returns; <b>string</b> , <b>number</b> , <b>datetime</b> are the only supported types currently.

## ShutdownData

If a [script add-in](#) implements the [OnShutdown](#) event, the method receives a **ShutdownData** object when invoked on Opus shutdown.

Property Name	Return Type	Description
endsession	<i>bool</i>	Returns <b>True</b> if the Windows session is ending (that is, if Opus is shutting down because the system is shutting down), or <b>False</b> if it's just Opus that is quitting.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .

## SmartFavorite

A **SmartFavorite** object represents an entry for a folder in the [SmartFavorites](#) table. It is retrieved by enumerating or indexing the [SmartFavorites](#) object.

Property Name	Return Type	Description
path	<i>object:Path</i>	Returns the path this entry represents, as a <a href="#">Path</a> object.
points	<i>int</i>	Returns the number of points this entry has as a source folder. The point score is used by Opus to determine which folders to display.
destpoints	<i>int</i>	Returns the number of points this entry has as a destination folder.

## SmartFavorites

The **SmartFavorites** object lets you query the contents of the [SmartFavorites](#) table. It is retrieved from the [DOpus.smartfavorites](#) property.

Property Name	Return Type	Description
<default value>	collection: <a href="#">SmartFavorite</a>	You can enumerate the <b>SmartFavorites</b> object to retrieve individual <a href="#">SmartFavorite</a> objects.
threshold	int	Returns the number of points an entry must have before it would be displayed in the <a href="#">SmartFavorites</a> list.
max	int	Returns the maximum number of entries that would be displayed in the <a href="#">SmartFavorites</a> list.

## SortOrder

The **SortOrder** object is returned by the [Format.manual\\_sort\\_order](#) property if [manual sort mode](#) is active. It lets you query and modify the sort order.

Method Name	Arguments	Return Type	Description
GetOrder	<string:name>	object: <a href="#">Vector</a>	Returns a <a href="#">Vector</a> of strings representing the current sort order of files in the folder. If multiple manual sort orders have been defined, you can provide the name of a specific sort order as an argument to this method. If called with no arguments it returns the current sort order by default.
SetOrder	< <a href="#">Vector</a> :order> <string:name>	none	You can pass this method a <a href="#">Vector</a> of strings to change the sort order of the current folder. You can optionally provide the name of a sort order as the second parameter if you've got more than one sort order defined.
ResetOrder	<string:name>	none	Resets the manual sort order to the currently selected sort order (e.g. if the file display header indicates that it is sorted by name, <b>ResetOrder</b> would reset to filename order). You can optionally provide the name of a sort order as the second parameter if you've got more than one sort order defined.

## SourceDestData

If a [script add-in](#) implements the [OnSourceDestChange](#) event, the method receives a **SourceDestData** object to indicate which tab's state changed.

Property Name	Return Type	Description
dest	<i>bool</i>	Returns <b>True</b> if the tab is now the destination.
source	<i>bool</i>	Returns <b>True</b> if the tab is now the source. If both <b>source</b> and <b>dest</b> return False it indicates that the tab is now "off".
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
tab	<i>object:Tab</i>	Returns a <a href="#">Tab</a> object representing the tab.

## StartupData

If a [script add-in](#) implements the [OnStartup](#) event, the method receives a **StartupData** object when Opus starts up.

*This object currently has no defined methods or properties.*

## StringSet

The **StringSet** object is container that stores one or more unique strings. It is similar to an array or vector (e.g. [Vector](#)) but has the advantage of using a dictionary system to locate strings rather than numeric indexes. You can therefore lookup strings much more quickly than by linearly searching a [Vector](#).

You can create a new **StringSet** using the [DOpusFactory](#) object. A **StringSet** can be either case-sensitive ("cat" and "CAT" would be treated as two different strings) or case-insensitive.

Property Name	Return Type	Description
---------------	-------------	-------------

count	<i>int</i>	Returns the number of elements the <b>StringSet</b> currently holds.
empty	<i>bool</i>	Returns <b>True</b> if the <b>StringSet</b> is empty, <b>False</b> if not.
length	<i>int</i>	A synonym for <b>count</b> .
size	<i>int</i>	A synonym for <b>count</b> .

Method Name	Arguments	Return Type	Description
assign	< <b>StringSet</b> :from>	<i>none</i>	Copies the contents of another <b>StringSet</b> to this one. You can also pass an array of strings or <a href="#">Vector</a> object.
clear	<i>none</i>	<i>none</i>	Clears the contents of the <b>StringSet</b> .
erase	<string>	<i>none</i>	Erases the string if it exists in the set.
exists	<string>	<i>bool</i>	Returns <b>True</b> if the specified string exists in the set.
insert	<string>	<i>bool</i>	Inserts the string into the set if it doesn't already exist. Returns <b>True</b> if successful.
merge	< <b>StringSet</b> :from>	<i>none</i>	Merges the contents of another <b>StringSet</b> with this one.

## StringTools

A **StringTools** object provides several utility methods for encoding and decoding strings. For example, you can use a **StringTools** object to Base64-encode a chunk of data, or decode a UTF-8 encoded message header.

You can obtain a **StringTools** object using the [DOpusFactory.StringTools](#) method.

Method Name	Arguments	Return Type	Description
Decode	< <a href="#">Blob</a> :source> or <string:source>  <string:format>	<i>string</i> or  <a href="#">Blob</a>	Decodes an encoded string or data.  You can provide either a <a href="#">Blob</a> object or a string as the <i>source</i> to decode. Depending on the value of the <i>format</i> argument, either a string or a <a href="#">Blob</a> is returned.  If <i>format</i> is specified as " <b>base64</b> " the source will be <i>Base64</i> -decoded, and a <a href="#">Blob</a> is returned.  If <i>format</i> is specified as " <b>quoted</b> " the source will be <i>Quoted-printable</i> -decoded, and a <a href="#">Blob</a> is returned.

			<p>If <i>format</i> is specified as <b>"auto"</b> or not supplied, special handling is invoked to decode a MIME-encoded email subject (e.g. one beginning with =?), and a string is returned. If <b>"auto"</b> is specified it will also detect UTF-8 encoded data if it has a BOM at the beginning.</p> <p>If decoding UTF-8 (via <b>"auto"</b> or <b>"utf-8"</b>), any byte-order-mark (BOM) will be skipped if one exists at the beginning of the input data.</p> <p>Otherwise, <i>format</i> must be set to a valid code-page name (e.g. <b>"utf-8"</b>, <b>"gb2312"</b> etc.), or a Windows code-page ID (e.g. <b>936</b>, <b>65001</b>). The source will be decoded using the specified code-page and a string is returned.</p>
Encode	<p>&lt;<a href="#">Blob</a>:source&gt; or &lt;string:source&gt;</p> <p>&lt;string:format&gt;</p>	<p><i>string</i> or <a href="#">Blob</a></p>	<p>Encodes a string or data.</p> <p>You can provide either a <a href="#">Blob</a> object or a string as the <i>source</i> to decode. Depending on the value of the <i>format</i> argument, either a string or a <a href="#">Blob</a> is returned.</p> <p>If <i>format</i> is specified as <b>"base64"</b> the source will be <i>Base64</i>-encoded, and a string is returned.</p> <p>If <i>format</i> is specified as <b>"quoted"</b> the source will be <i>Quoted-printable</i>-encoded, and a string is returned.</p> <p>If <i>format</i> is specified as <b>"utf-8 bom"</b>, the output data is encoded as UTF-8 with a byte-order-mark (BOM) at the start. Use <b>"utf-8"</b> if you want UTF-8 without the BOM.</p> <p>Otherwise, <i>format</i> must be set to a valid code-page name (e.g. <b>"utf-8"</b>, <b>"gb2312"</b> etc.), or a Windows code-page ID (e.g. <b>936</b>, <b>65001</b>). The source will be encoded using the specified code-page and a <a href="#">Blob</a> is returned.</p>
IsASCII	<string:input>	<i>bool</i>	<p>Tests the input string to see if it only contains characters that can be represented in ASCII.</p> <p>If the result is false, the string is not safe to save into a text file unless you use a Unicode format such as UTF-8.</p> <p>This check is not affected by locales or codepages. Instead, it tests whether the string consists of only 7-bit ASCII characters, such that no characters will be lost or modified if you save the string to a text file and then load it back on any other computer.</p>

## StyleSelectedData

If a [script add-in](#) implements the [OnStyleSelected](#) event, the method receives a **StyleSelectedData** object when the user chooses a new [Lister style](#).

Property Name	Return Type	Description
lister	<i>object</i> : <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the Lister that changing style.
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
style	<i>string</i>	Returns the name of the newly selected style.

## SysInfo

The **SysInfo** object is created by the [DOpusFactory.SysInfo](#) method. It lets scripts access miscellaneous system information that may not be otherwise easy to obtain from a script.

Method Name	Arguments	Return Type	Description
FindProcess	<i>string</i>	<i>int</i>	Allows you to test if a named process is currently running, and returns the process's ID if so. If the process isn't running <b>0</b> is returned. You can use wildcards or (by prefixing the pattern with <b>regex</b> ;) regular expressions.
Monitors	<i>none</i>	<a href="#">Vector</a> : <a href="#">Rect</a>	Returns a <a href="#">Vector</a> of <a href="#">Rect</a> objects which provide information about the position and size of the display monitors in the system.
MouseMonitor	<i>none</i>	<i>int</i>	Returns the index of the monitor the mouse pointer is currently positioned on.
MousePosX	<i>none</i>	<i>int</i>	Returns the current x-coordinate of the mouse pointer.
MousePosY	<i>none</i>	<i>int</i>	Returns the current y-coordinate of the mouse pointer.

## Tab

The **Tab** object represents a folder tab in a Lister (even if the tab control isn't currently displayed, a Lister always has at least one open tab). You can obtain a collection of **Tab** objects from a [Lister](#) object. **Tab** objects are also used with the [Command](#) and [Func](#) objects, and if a command results in new tabs being opened, the [Results](#) object.

Property Name	Return Type	Description
all	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the files and folders currently displayed in this tab.  Note: The first time a script accesses this property (and all the other properties that return an <a href="#">Item</a> collection), a snapshot is taken of all the appropriate items. If the script then makes changes to those items (e.g. by creating a new file, modifying the selection, etc), these changes will not be reflected by the collection. To re-synchronize the collection call the <b>Update</b> method on the collection.
backlist	<i>collection:</i> <a href="#">Path</a>	Returns a collection of <a href="#">Path</a> objects that represents the paths in the "backward" history list for this tab (i.e. the folders you would get to by clicking the <i>Back</i> button).
color	<i>string</i>	Returns the tab's assigned color (if one has been assigned via, for example, the <b>Go TABCOLOR</b> command). The color is returned as a string in R,G,B format.
crumbpath	<i>object:</i> <a href="#">Path</a>	Returns the current path from the tab's breadcrumb control (if it has one), including any ghost path.
dirs	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the folders currently displayed in this tab.
dirty	<i>bool</i>	Returns <b>True</b> if the tab is marked as dirty, indicating its list of contents may be out of date. This can happen if the tab is in the background and the user has turned off the <b>Preferences / Folder Tabs / Options / Process file changes in background tabs</b> option.
displayed_label	<i>string</i>	Returns the currently displayed label of this tab.
filegroups	<i>collection:</i> <a href="#">FileGroup</a>	Returns a collection of <a href="#">FileGroup</a> objects that represents all the file groups in the tab (when the tab is grouped). You can use the <b>format.group_by</b> property to test if the tab is grouped or not.
files	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the files currently displayed in this tab.
format	<i>object:</i> <a href="#">Format</a>	Returns a <a href="#">Format</a> object representing the current folder format in this tab.

forwardlist	<i>collection:</i> <a href="#">Path</a>	Returns a collection of <a href="#">Path</a> objects that represents the paths in the "forward" history list for this tab (i.e. the folders you would get to by clicking the <i>Forward</i> button).
hidden	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the files and folders currently hidden from this tab.
hidden_dirs	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the folders currently hidden from this tab.
hidden_files	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the files currently hidden from this tab
label	<i>string</i>	Returns the current assigned tab label. Note that this may be an empty string if no custom label has been assigned. The <b>displayed_label</b> property returns the currently displayed label in all cases.
linktab	<i>object:</i> <b>Tab</b>	If this tab is linked to another tab, returns a <b>Tab</b> object representing the linked tab. If this tab is not linked this property returns <b>0</b> .
lister	<i>object:</i> <a href="#">Lister</a>	Returns a <a href="#">Lister</a> object representing the parent Lister that owns this tab.
lock	<i>string</i>	Returns the current lock state of the tab; one of "off", "on", "changes", "reuse".
navlock	<i>bool</i>	Returns <b>True</b> if this tab is linked in Navigation Lock mode. This property does not exist if the tab is not linked, so make sure you check the value of <b>linktab</b> first.
path	<i>object:</i> <a href="#">Path</a>	Returns the current path shown in this tab.
quickfilter	<i>object:</i> <a href="#">QuickFilter</a>	Returns a <a href="#">QuickFilter</a> object providing information about the state of the quick filter in this tab.
right	<i>bool</i>	Returns <b>True</b> if this tab is currently on the right or bottom side of a dual-display Lister, and <b>False</b> otherwise.
selected	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the selected files and folders currently displayed in this tab. Note that if <a href="#">checkbox mode</a> is turned on in the tab, this will be a collection of checked items rather than selected.
selected_dirs	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the selected folders currently displayed in this tab.
selected_files	<i>collection:</i> <a href="#">Item</a>	Returns a collection of <a href="#">Item</a> objects that represents all the selected files currently displayed in this tab
selstats	<i>object:</i> <a href="#">TabStats</a>	Returns a <a href="#">TabStats</a> object that provides various information about the tab, including the number of files, number of selected files, total size of selected files, etc. The "selected" counts provided by this object take <a href="#">checkbox mode</a> into account (that is, if checkbox mode is currently turned on, the counts will be for checked files rather than for selected files).

source	<i>bool</i>	Returns <b>True</b> if this tab is currently the <a href="#">source</a> , and <b>False</b> otherwise.
stats	<i>object: <a href="#">TabStats</a></i>	Returns a <a href="#">TabStats</a> object that provides various information about the tab, including the number of files, number of selected files, total size of selected files, etc. Unlike <b>selstats</b> , this object does not take <a href="#">checkbox mode</a> into account (so the "selected" counts will refer to selected rather than checked files).
vars	<i>object: <a href="#">Vars</a></i>	This <a href="#">Vars</a> object represents all defined variables with <i>tab scope</i> (that are scoped to this tab).
visible	<i>bool</i>	Returns <b>True</b> if this tab is currently visible (i.e. it is the active tab in either file display), and <b>False</b> otherwise.

Method Name	Arguments	Return Type	Description
Dlg	<i>none</i>	<i>object: <a href="#">Dialog</a></i>	Creates a new <a href="#">Dialog</a> object, that lets you display dialogs and popup menus. The dialog's <b>window</b> property will be automatically assigned to this tab.
Update	<i>none</i>	<i>none</i>	The first time a script accesses a particular <b>Tab</b> object, a snapshot is taken of the tab state. If the script then makes changes to that tab (e.g. it selects files, creates a new folder, etc), these changes will not be reflected by the object. To re-synchronize the object with the tab, call the <b>Tab.Update</b> method.

### **TabClickData**

If a [script add-in](#) implements the [OnTabClick](#) event, the method receives a **TabClickData** object when a tab is clicked with a qualifier key held down.

Property Name	Return Type	Description
qualifiers	<i>string</i>	Returns a string indicating any qualifier keys that were held down by the user when the event was triggered. The string can contain any or all of the following: <i>shift</i> , <i>ctrl</i> , <i>alt</i> , <i>lwin</i> , <i>rwin</i> .
tab	<i>object: <a href="#">Tab</a></i>	Returns a <a href="#">Tab</a> object representing the tab that was clicked.

## TabStats

The **TabStats** object provides various information and statistics about the current folder displayed in a tab. Note that the [Tab](#) object provides two versions of this object. **Tab.selstats** takes [Checkbox Mode](#) into account, and in this mode the values of the "checked" and the "selected" properties will be the same. If the object was retrieved via **Tab.stats** and the file display is in [Checkbox Mode](#), the two sets of properties will be different.

Property Name	Return Type	Description
bigimage_h	<i>int</i>	Returns the width in pixels of the largest image in the folder.
bigimage_w	<i>int</i>	Returns the height in pixels of the largest image in the folder.
bytes	<i>object:FileSize</i>	Returns the total number of bytes in the folder as a <a href="#">FileSize</a> object.
checkbox_mode	<i>bool</i>	Returns <b>True</b> if the tab is currently in <a href="#">Checkbox Mode</a> .
checkedbytes	<i>object:FileSize</i>	Returns the total number of bytes in checked items as a <a href="#">FileSize</a> object.
checkeddirbytes	<i>object:FileSize</i>	Returns the total number of bytes in checked folders as a <a href="#">FileSize</a> object.
checkeddirs	<i>int</i>	Returns the total number of checked folders.
checkedfilebytes	<i>object:FileSize</i>	Returns the total number of bytes in checked files as a <a href="#">FileSize</a> object.
checkedfiles	<i>int</i>	Returns the total number of checked files.
checkeditems	<i>int</i>	Returns the total number of checked items.
checkedmusiclength	<i>int</i>	Returns the total length in seconds of all checked music files.
dirbytes	<i>object:FileSize</i>	Returns the total number of bytes in all folders as a <a href="#">FileSize</a> object.
dirs	<i>int</i>	Returns the total number of folders.
filebytes	<i>object:FileSize</i>	Returns the total number of bytes in all files as a <a href="#">FileSize</a> object.
filedate_max	<i>date</i>	Returns the latest (most recent) file date in the folder.
filedate_min	<i>date</i>	Returns the earliest (oldest) file date in the folder.
files	<i>int</i>	Returns the total number of files.
items	<i>int</i>	Returns the total number of items.
largestfile	<i>object:FileSize</i>	Returns the size of the largest file in the folder as a <a href="#">FileSize</a> object.
musiclength	<i>int</i>	Returns the total length in seconds of all music files.
selbytes	<i>object:FileSize</i>	Returns the total number of bytes in all selected items as a <a href="#">FileSize</a> object.
seldirbytes	<i>object:FileSize</i>	Returns the total number of bytes in all selected folders as a <a href="#">FileSize</a> object.
seldirs	<i>int</i>	Returns the number of selected folders.

selffilebytes	<i>object:</i> <a href="#">FileSize</a>	Returns the total number of bytes in all selected files as a <a href="#">FileSize</a> object.
selffiles	<i>int</i>	Returns the number of selected files.
selitems	<i>int</i>	Returns the number of selected items.
selmusiclength	<i>int</i>	Returns the total length in seconds of all selected music files.

Method Name	Arguments	Return Type	Description
Update	<i>none</i>	<i>none</i>	The first time a script accesses a particular <b>TabStats</b> object, a snapshot is taken of the tab state. If the script then makes changes to that tab (e.g. it selects files, creates a new folder, etc), these changes will not be reflected by the object. To re-synchronize the object with the tab, call the <b>TabStats.Update</b> method.

## Toolbar

The **Toolbar** object represents a toolbar. **Toolbar** objects can represent a specific instance of a toolbar (open in a specific Lister), and can also represent the toolbar itself, on disk, that doesn't have to currently be open at all.

- When retrieved from the [Toolbars](#) object (which in turn comes from the [DOpus.Toolbars](#) method), the object represents a toolbar on disk. You can find out where it is currently in use from its properties.
- When retrieved from the [Lister.toolbars](#) property, it represents an instance of a toolbar in that particular Lister.

Property Name	Return Type	Description
<default value>	<i>string</i>	Returns the name of the toolbar.
deftoolbar	<i>bool</i>	Returns <b>True</b> if this is a default (factory-provided) toolbar, or <b>False</b> if it was user-created.  (Old scripts may use "default" instead of "deftoolbar"; this is deprecated because it does not work in JScript where "default" is a reserved keyword.)
listers	<i>collection:</i> <a href="#">Lister</a>	Returns a collection of <a href="#">Lister</a> objects representing any and all Listers this toolbar is currently open in.
docks	<i>collection:</i> <a href="#">Dock</a>	Returns a collection of <a href="#">Dock</a> objects representing any currently floating instances of this toolbar.
group	<i>string</i>	Returns a <i>string</i> indicating the group (position) of a particular instance of this toolbar. The returned string

		will be one of <i>top</i> , <i>bottom</i> , <i>left</i> , <i>right</i> , <i>center</i> , <i>fdright</i> , <i>fdbottom</i> , <i>tree</i> .
line	<i>int</i>	Returns the line number within the toolbar's group that it resides on. For example, the first toolbar at the top of the Lister would have a line of 0.
pos	<i>int</i>	Returns the pixel position from the left/top of the toolbar's line. If there are two or more toolbars with the same <b>line</b> number, the <b>pos</b> value determines the order they appear in.

## Toolbars

The **Toolbars** object lets you enumerate all the defined toolbars in your Directory Opus configuration (whether currently turned on or not). It's retrieved using the [DOpus.Toolbars](#) method.

Property Name	Return Type	Description
<default value>	collection: <a href="#">Toolbar</a>	Returns a collection of <a href="#">Toolbar</a> objects that you can enumerate.
fdb	string	Returns the name of the currently selected File Display Toolbar. If the FDB toolbar is disabled, returns the string <b>!static</b> to indicate a static header.
viewer	string	Returns the name of the currently selected Viewer Toolbar.

## Var

The **Var** object represents an individual user or script-defined variable. Individual **Var** objects can be accessed or enumerated from the [Vars](#) object.

Property Name	Return Type	Description
<default value>	variant or string	<p>The default value of the <b>Var</b> object returns the value of the variable itself, with one exception. If the <b>Var</b> object is being accessed as part of an enumeration of the <a href="#">Vars</a> collection, the default value returns the variable name.</p> <p>So for instance,</p> <pre>For Each Var in DOpus.Vars     DOpus.Output("Variable name = " &amp; Var) Next</pre> <p>Versus:</p>

		<pre>Set Var = DOpus.Vars("myvar")  DOpus.Output("Variable value = " &amp; Var)</pre>
name	<i>string</i>	Returns the name of the variable. You cannot change the name of a variable once it has been assigned - instead, delete the variable from its collection and add a new one.
persist	<i>bool</i>	Returns <b>True</b> if the variable is persistent (saved) or <b>False</b> if not. You can set this property to change the persistence state.
value	<i>variant</i>	<p>Returns the value of the variable. You can set this property to change the value of the variable.</p> <p>You can store any type of variable in a <b>Var</b> object, although not all types can be saved to disk. If you want your variable to be persistent you should only use <i>bool</i>, <i>int</i>, <i>string</i>, <i>date</i>, <i>currency</i> or a <a href="#">Vector</a> of those types.</p>

Method Name	Arguments	Return Type	Description
Delete	<i>none</i>	<i>none</i>	Deletes this variable from its parent collection.

## Vars

The **Vars** object represents a collection of user and script-defined variables. There are a number of different sets of variables, with differing scopes. Some sets support persistent variables, that are saved and re-loaded from one session to the other.

Scope	Accessed by	Supports Persistence	Description
Global	<a href="#">DOpus.vars</a>	Yes	Variables that are available throughout Opus. They can be accessed by any function or script.
Lister	<a href="#">Lister.vars</a>	Yes	Variables that are local to a Lister. Persistent variables are saved on a per-Lister basis in Lister Layouts.
Tab	<a href="#">Tab.vars</a>	Yes	Variables that are local to a particular tab. Persistent variables are saved per-Tab in Lister Layouts.

Script	<a href="#">Script.vars</a> <a href="#">ScriptInitData.vars</a>	Yes	Variables that are local to a particular script add-in.
Dialog	<a href="#">Dialog.Vars</a>	Yes	Variables that are tied to a particular <a href="#">script dialog</a> .
Command	<a href="#">Command.vars</a>	No	Variables that are local to a particular function. They are not saved from one invocation of the function to another and do not support persistence.

Property Name	Return Type	Description
<default value>	collection: <a href="#">Var</a>	Returns a collection of the variables in the collection. You can enumerate the <a href="#">Var</a> elements or refer to a specific one by its index or by its name.

Method Name	Arguments	Return Type	Description
Delete	<string:name>	<i>none</i>	Deletes the named variable from the collection. You can also specify a <a href="#">wildcard pattern</a> to delete multiple variables (or * for all).
Exists	<string:name>	<i>bool</i>	Returns <b>True</b> if the named variable exists in the collection, or <b>False</b> if it doesn't exist.
Get	<string:name>	<i>variant</i>	Returns the value of the named variable. You can use this method as an alternative to indexing the collection.
Set	<string:name> <variant:value>	<i>none</i>	Sets the named value to the specified value. You can use this method as an alternative to indexing the collection.  You can store any type of variable in a <b>Vars</b> collection, although not all types can be saved to disk. If you want your variable to be persistent you should only use <i>bool</i> , <i>int</i> , <i>string</i> , <i>date</i> , <i>currency</i> or a <a href="#">Vector</a> of those types.

## Vector

The **Vector** object is provided to address some short-comings in ActiveX scripting's array support. Some languages have better support than others for arrays, but the languages aren't consistent and some (like *JScript*) have incompatible arrays that Opus is unable to access at all. Therefore, any Opus scripting objects that take or return an array-like variable will use (or prefer to use) a **Vector** rather than an array.

A **Vector** object is mostly able to be used as a straight drop-in replacement for an arrays. They are *collections* and so can be enumerated, or accessed via index (e.g. **Vector(4)** to access the fifth element). They also have a number of helper methods to make manipulating them easier than arrays often are.

You can create a new **Vector** using the [DOpusFactory.Vector](#) method.

Property Name	Return Type	Description
capacity	<i>int</i>	Returns the capacity of the <b>Vector</b> (the number of elements it can hold without having to reallocate memory). This is not the same as the number of elements it currently holds, which can be 0 even if the capacity is something larger.
count	<i>int</i>	Returns the number of elements the <b>Vector</b> currently holds.
empty	<i>bool</i>	Returns <b>True</b> if the <b>Vector</b> is empty, <b>False</b> if not.
length	<i>int</i>	A synonym for <b>count</b> .
size	<i>int</i>	A synonym for <b>count</b> .

Method Name	Arguments	Return Type	Description
assign	< <b>Vector</b> :from> <int:start> <int:end>	<i>none</i>	Copies the value of another <b>Vector</b> to this one. If <i>start</i> and <i>end</i> are not provided, the entire <b>Vector</b> is copied - otherwise, only the specified elements are copied.  Instead of a <b>Vector</b> object you can also pass a <i>collection</i> to this method and the contents of the collection will be copied to the <b>Vector</b> .
back	<i>none</i>	<i>variant</i>	Returns the last element in the <b>Vector</b> .
clear	<i>none</i>	<i>none</i>	Clears the contents of the <b>Vector</b> .
erase	<int:index>	<i>none</i>	Erases the element at the specified index.
exchange	<int:index1> <int:index2>	<i>none</i>	Exchanges the positions of the two specified elements.

front	<i>none</i>	<i>variant</i>	Returns the first element in the <b>Vector</b> .
insert	<int:index> <variant:value>	<i>none</i>	Inserts the provided value at the specified position.
pop_back	<i>none</i>	<i>none</i>	Removes the last element of the <b>Vector</b> .
push_back	<variant:value>	<i>none</i>	Adds the provided value to the end of the <b>Vector</b> .
reserve	<int:capacity>	<i>none</i>	Reserves space in the <b>Vector</b> for the specified number of elements (increases its <b>capacity</b> , although the <b>count</b> of elements remains unchanged).  Note that <b>Vectors</b> grow dynamically - you don't have to specifically reserve or resize them. However if you want to add a large number of elements to a <b>Vector</b> it can be more efficient to reserve space for them first.
resize	<int:size>	<i>none</i>	Resizes the <b>Vector</b> to the specified number of elements. Any existing elements past the new size of the <b>Vector</b> will be erased.
shrink_to_fit	<i>none</i>	<i>none</i>	Reduces the capacity of the <b>Vector</b> to the number of elements it currently holds.
sort	<i>none</i>	<i>none</i>	Sorts the contents of the <b>Vector</b> . Strings and numbers are sorted alphabetically and numerically - other elements are grouped by type but not specifically sorted in any particular order.

## Version

The **Version** object is retrieved from the [DOpus.version](#) property. It provides information about the current version of Directory Opus.

Property Name	Return Type	Description
<default value>	<i>string</i>	Full version string (as shown in the <i>About</i> dialog).
build	<i>int</i>	The current build number.
module	<i>string</i>	The current module version (the version of <b>dopus.exe</b> itself). You can also enumerate or index this as a <i>collection:int</i> to retrieve the individual four digits of the module version.
product	<i>string</i>	The current product version (the release version of Directory Opus as a whole). You can also enumerate or index this as a <i>collection:int</i> to retrieve the individual four digits of the product version.
winver	<i>object:WinVer</i>	Returns a <a href="#">WinVer</a> object which provides information about the current version of Windows.

Method Name	Arguments	Return Type	Description
AtLeast	<string:version>	<i>bool</i>	Returns <b>True</b> if the current version of Opus is the specified version or greater. You can specify the major version only (e.g. "11"), a major and minor version (e.g. "11.3") or a specific beta version (e.g. "11.3.1").

## VideoMeta

The **VideoMeta** object is retrieved from the [Metadata.video](#) or [Metadata.video\\_text](#) properties. It provides access to metadata relating to movie files.

Property Name	Return Type	Description
<column keyword>	<i>variant</i>	Returns the value of the specified column, as listed in the <i>Movies</i> section of the <a href="#">Keywords for Columns</a> page.

## Viewer

The **Viewer** object represents a standalone [image viewer](#). A collection of **Viewer** objects is returned by the [Viewers](#) object, which is obtainable via the [DOpus.viewers](#) property. For functions launched from within a viewer (e.g. from its toolbar), the current **Viewer** object is provided by the [ClickData.func.viewer](#) property. You can use a **Viewer** object as the parameter for the [Command.SetSourceTab](#) method, which lets you run commands against an arbitrary viewer window.

Property Name	Return Type	Description
bottom	<i>int</i>	Returns the bottom coordinate of the viewer window.
current	<i>object:Item</i>	Returns an <a href="#">Item</a> object representing the currently displayed image.
files	<i>collection:Item</i>	Returns a collection of <a href="#">Item</a> objects representing the images in the viewer's list.
foreground	<i>bool</i>	Returns <b>True</b> if the viewer is currently the foreground (active) window in the system.
index	<i>int</i>	Returns the index of the currently viewed image within the viewer's list of files.

lastactive	<i>bool</i>	Returns <b>True</b> if the viewer is the most recently active viewer.
left	<i>int</i>	Returns the left coordinate of the viewer window.
parenttab	<i>object: <a href="#">Tab</a></i>	Returns a <a href="#">Tab</a> object representing the tab that launched the viewer (if there was one, and if it still exists).
right	<i>int</i>	Returns the right coordinate of the viewer window.
title	<i>string</i>	<p>Returns or sets the title bar string for the viewer window.</p> <p>You can use several special "tokens" in the title string to insert various pieces of information:</p> <ul style="list-style-type: none"> <li><b>%P</b> - full path of the currently viewed image</li> <li><b>%N</b> - name of the current displayed image</li> <li><b>%R</b> - drive root of the current image</li> <li><b>%E</b> - displays * if the image's metadata has been modified and not saved</li> <li><b>%I</b> - current image's index (number) in the list of images</li> <li><b>%O</b> - total number of images in the list</li> <li><b>%W</b> - width of the current image</li> <li><b>%H</b> - height of the current image</li> <li><b>%D</b> - depth of the current image (bits per pixel)</li> <li><b>%M</b> - current image's dimensions</li> <li><b>%S</b> - file size on disk</li> <li><b>%F</b> - folder name</li> <li><b>%C</b> - collection name if current image is <a href="#">marked</a></li> <li><b>%L</b> - any <a href="#">labels</a> assigned to the current image</li> <li><b>%T</b> - original title (useful for simply adding a prefix or suffix to the title)</li> <li><b>%%</b> - insert a literal % character</li> </ul>
top	<i>int</i>	Returns the top coordinate of the viewer window.

Method Name	Arguments	Return Type	Description
AddFile	<string:filepath> <int:index>	<i>none</i>	Adds the specified file to the viewer's current list of files. You can either pass a string or a <a href="#">Path</a> object to indicate the file to add to the list. By default the file will be added to the end of the list, unless you specify a 0-based index as the second argument.
Command	<string:command> or < <a href="#">Command</a> :command>	<i>none</i>	Runs a command in the context of this viewer window. You can either pass a string or a <a href="#">Command</a> object.  If the argument you pass is a string then it can only be a viewer command argument as documented for the <a href="#">Show VIEWERCMD</a> command. For example, <b>Command("next")</b> would run the <b>Show VIEWERCMD=next</b> command in the context of this viewer.  If you pass a <a href="#">Command</a> object then all commands (internal or external) can be used.
RemoveFile	<int:index> or <string:filepath>	<i>none</i>	Removes the specified file from the viewer's current list of files. You can either pass the 0-based index of the file to remove, or the filepath (either as a string or a <a href="#">Path</a> object).

## Viewers

The **Viewers** object is a collection of all currently open standalone [image viewers](#). It can be obtained via the [DOpus.viewers](#) property.

Property Name	Return Type	Description
<default value>	<i>collection:</i> <a href="#">Viewer</a>	Lets you enumerate the currently open viewers.
lastactive	<i>object:</i> <a href="#">Viewer</a>	Returns a <a href="#">Viewer</a> object representing the most recently active viewer window.

## ViewerEventData

If a script implements the [OnViewerEvent](#) event, it receives a **ViewerEventData** object whenever certain events occur in a standalone [image viewer](#).

Property Name	Return Type	Description
event	<i>string</i>	Returns a string indicating the event that occurred. The events currently defined are: <ul style="list-style-type: none"><li>• <b>create</b>: A new viewer has been created.</li><li>• <b>destroy</b>: A viewer window has been destroyed.</li><li>• <b>load</b>: A new image has been loaded in a viewer. The <b>item</b> property can be used to find out which file was loaded.</li><li>• <b>setfocus</b>: The viewer window has received focus (gone active).</li><li>• <b>killfocus</b>: The viewer window has lost focus (gone inactive).</li><li>• <b>click</b>: The left button was clicked on the image (requires mouse buttons to be set to trigger <i>Script event</i> in <b>Preferences / Viewer / Mouse Buttons</b>).</li><li>• <b>dblclk</b>: The left button was double-clicked on the image.</li><li>• <b>mclick</b>: The middle button was clicked on the image.</li></ul>
item	<i>object</i> : <a href="#">Item</a>	For the <b>load</b> event, returns an <a href="#">Item</a> object representing the newly loaded image.
viewer	<i>object</i> : <a href="#">Viewer</a>	Returns a <a href="#">Viewer</a> object representing the viewer the event occurred in.
x	<i>int</i>	For the click events, returns the x coordinate within the viewer window that the click occurred.
y	<i>int</i>	For the click events, returns the y coordinate within the viewer window that the click occurred.
w	<i>int</i>	For the click events, returns the width of the viewer window.
h	<i>int</i>	For the click events, returns the height of the viewer window.

## Wild

The **Wild** object allows a script to access the in-built pattern matching functions in Opus. Even though most ActiveX scripting languages have their own pattern matching support (usually via a regular expression system of some sort), you may wish to use the one that Opus provides for consistency with internal Opus functions.

You can create a **Wild** object using the [FSUtil.NewWild](#) method. To use the **Wild** object, you must first give it the pattern to match against (this step is called "parsing the pattern"). You can do this when it is created or later on using the **Parse** method. Once the object has a pattern you can call the **Match** method to test a string against the pattern.

Property Name	Return Type	Description
<default value>	string	Returns the current pattern in the Wild object

Method Name	Arguments	Return Type	Description
Match	<string:test>	bool	Compares the specified string against the previously-parsed pattern, and returns <b>True</b> if it matches.
Parse	<string:pattern> <string:flags>	bool	Parses the supplied pattern. The flags string is optional - if supplied it must be a string consisting of one or more of the following characters:  <b>c</b> - case-sensitive (otherwise pattern matching is not case-sensitive) <b>d</b> - DOS only (only standard MS-DOS wildcards are supported) <b>f</b> - filename mode (special handling for matching filenames) <b>r</b> - regular expression (otherwise <a href="#">standard pattern matching</a> is used)

## WinVer

The **WinVer** object is retrieved from the [Version.winver](#) property. It provides information about the current version of Windows.

Property Name	Return Type	Description
<default value>	string	Full Windows version string.
server	bool	<b>True</b> if running on a Server edition of Windows.
xp	bool	<b>True</b> if running on Windows XP.
xporbetter	bool	<b>True</b> if running on Windows XP or better (this will always be true).
vista	bool	<b>True</b> if running on Windows Vista.
vistaorbetter	bool	<b>True</b> if running on Windows Vista or better (later).
win7	bool	<b>True</b> if running on Windows 7.

win7orbetter	<i>bool</i>	<b>True</b> if running on Windows 7 or better.
win8	<i>bool</i>	<b>True</b> if running on Windows 8.
win8orbetter	<i>bool</i>	<b>True</b> if running on Windows 8 or better.
win81	<i>bool</i>	<b>True</b> if running on Windows 8.1.
win81orbetter	<i>bool</i>	<b>True</b> if running on Windows 8.1 or better.
win10	<i>bool</i>	<b>True</b> if running on Windows 10.
win10orbetter	<i>bool</i>	<b>True</b> if running on Windows 10 or better.

## Scripting Events

Events are functions in a script file that are called by Opus at various times. There are three main classes of events:

- Most events are [script add-in](#) events - they are called whenever various events or activities occur. For example, the [OnBeforeFolderChange](#) event is called before the folder is changed in a tab. Which events a script add-in listens for is completely defined by the script - simply implement the appropriate method and Opus will call it at the appropriate time.
- [Script functions](#) are called via their [OnClick](#) event - although implementing this method is optional, as the entire script is executed when a script button or hotkey is run. The advantage of implementing **OnClick** is the [ClickData](#) object passed to the method which can provide useful information about the command's environment.
- [Rename scripts](#) provide an event that's called for each file being renamed. For historical reasons, there are a number of different rename events that can be implemented - the simplest and recommended is the [OnGetNewName](#) event.

With the exception of the legacy rename events (**Rename\_GetNewName** and **Rename\_GetNewName2**), all event functions have a single argument - an event-specific data object that's passed from Opus to the script each time the event is called. Each object contains properties (and sometimes methods) relevant to the event. The advantage of using a single object is that in the future additional properties can be added without changing the function signature of the event itself.

### ***OnAboutScript***

The **OnAboutScript** event can be implemented by a [script add-in](#) to display an "about" dialog to the user. It is triggered when the user clicks the *About* button for a script on the [Toolbars / Scripts](#) page in Preferences.

<b>Method Name:</b>	OnAboutScript
---------------------	---------------

<b>Argument Type:</b>	<a href="#">AboutData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The usual implementation for this event would use the <b>AboutData.window</b> parameter to display a dialog using the <a href="#">Dialog</a> object.

### ***OnActivateLister***

The **OnActivateLister** event can be implemented by a [script add-in](#) to receive notification whenever a Lister window becomes the active window, or loses activation to another window.

<b>Method Name:</b>	OnActivateLister
<b>Argument Type:</b>	<a href="#">ActivateListerData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>ActivateListerData.lister</b> property identifies the Lister, and the <b>active</b> property indicates whether this Lister has become active or inactive. If the activation moves from one Lister to another this event would be called twice, once for each Lister.

### ***OnActivateTab***

The **OnActivateTab** event can be implemented by a [script add-in](#) to receive a notification every time a tab becomes active (i.e. comes to the front of another tab in the same file display).

<b>Method Name:</b>	OnActivateTab
<b>Argument Type:</b>	<a href="#">ActivateTabData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>ActivateTabData.old</b> property identifies the tab that was previously active, and the <b>new</b> property identifies the new active tab.

## OnAddColumns

The **OnAddColumns** event is called to allow your [script add-in](#) to [add columns](#). Call the [AddColData.AddColumn](#) method once for each column you wish to add.

<b>Method Name:</b>	OnAddColumns
<b>Argument Type:</b>	<a href="#">AddColData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	When Opus starts up, or when a script add-in is added, edited or enabled, its <b>OnAddColumns</b> method is called. This allows a script to <a href="#">add columns</a> to Opus. A script can reinitialize its list of columns at any time by calling the <a href="#">Script.InitColumns</a> method.

## OnAddCommands

The **OnAddCommands** event is called to allow your [script add-in](#) to [add internal commands](#). Call the [AddCmdData.AddCommand](#) method once for each command you wish to add.

<b>Method Name:</b>	OnAddCommands
<b>Argument Type:</b>	<a href="#">AddCmdData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	When Opus starts up, or when a script add-in is added, edited or enabled, its <b>OnAddCommands</b> method is called. This allows a script to <a href="#">add internal commands</a> to the Opus command set. A script can reinitialize its list of commands at any time by calling the <a href="#">Script.InitCommands</a> method.

## OnAfterFolderChange

The **OnAfterFolderChange** event can be implemented by a [script add-in](#) that wants to be notified after a new folder has been read in a tab. Use the [OnBeforeFolderChange](#) event to receive notification *before* the folder is read.

<b>Method Name:</b>	OnAfterFolderChange
<b>Argument Type:</b>	<a href="#">AfterFolderChangeData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	<p>The <b>AfterFolderChangeData.tab</b> property indicates the tab and the <b>path</b> property the path of the folder. The <b>result</b> property indicates the success or failure of the folder read.</p> <p>If <b>result</b> is <b>False</b> (i.e. the folder was not successfully read) then you can return <b>True</b> from this event to stop Opus from going back to the previous folder (which is what normally happens when a folder read fails). If <b>result</b> is <b>True</b> then the return value from this event is ignored.</p>

### ***OnBeforeFolderChange***

The **OnBeforeFolderChange** event can be implemented by a [script add-in](#) to receive notification before a new folder is read in a tab. Use the **OnAfterFolderChange** event if you want notification *after* a folder has been read.

<b>Method Name:</b>	OnBeforeFolderChange
<b>Argument Type:</b>	<a href="#">BeforeFolderChangeData</a>
<b>Return Type:</b>	<i>bool</i> or <i>string</i>
<b>Description:</b>	<p>The <b>BeforeFolderChangeData.tab</b> property identifies the tab, and the <b>path</b> property identifies the folder about to be read. The <b>initial</b> property indicates if this is the first folder read into this tab - if <b>True</b>, it means the tab was previously empty or newly opened.</p> <p>You can return two different types from this event:</p> <ul style="list-style-type: none"> <li>• <i>bool</i>: If you return <b>True</b>, the folder read will be blocked and the tab will be unchanged. If you return <b>False</b> the read will be allowed to continue (this is the default).</li> <li>• <i>string</i>: You can return a <i>string</i> (or a <a href="#">Path</a> object) to change the folder path to be read.</li> </ul>

## OnClick

The **OnClick** event is the main entry point for a [script function](#).

<b>Method Name:</b>	OnClick
<b>Argument Type:</b>	<a href="#">ClickData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	<p>This method is called whenever a button or hotkey containing a <a href="#">script function</a> is run. The <b>ClickData</b> object provides useful information about the command environment, including source and destination tabs and paths, any selected files, command line arguments, any qualifier keys that were held down, etc. This information is all accessible from the <b>ClickData.func</b> property (which returns a <a href="#">Func</a> object).</p> <p>Note that when a script function is run, any instructions not contained inside a function are executed first, before <b>OnClick</b> is invoked. This means that the <b>OnClick</b> method is technically optional (as you could just implement the entire script outside of a function), although it is</p>

recommended that you implement it simply for the benefit of the **ClickData** object it receives.

### ***OnCloseLister***

The **OnCloseLister** event can be implemented by a [script add-in](#) to receive notification whenever a Lister is closed.

<b>Method Name:</b>	OnCloseLister
<b>Argument Type:</b>	<a href="#">CloseListerData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	<p>The <b>CloseListerData.lister</b> property identifies the Lister that is closing. The <b>shutdown</b> property is <b>True</b> if the Lister is closing because Opus (or Windows) is shutting down.</p> <p>If shutdown is <b>False</b>, you can prevent the Lister from closing by returning <b>True</b> from this event (or <b>False</b> to allow the close, which is the default). If Opus or Windows is shutting down then you can't prevent the Lister from closing.</p>

### ***OnCloseTab***

The **OnCloseTab** event can be implemented by a [script add-in](#) to receive notification when a tab is closed in a Lister.

<b>Method Name:</b>	OnCloseTab
<b>Argument Type:</b>	<a href="#">CloseTabData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	<p>The <b>CloseTabData.tab</b> property identifies the tab that is closing. You can return <b>True</b> to prevent the tab from closing, or <b>False</b> (which is the default) to allow it to close.</p>

Note that when a Lister closes this event is **not** triggered for each of its tabs - 1 notification when a Lister closes, and all the tabs in that Lister as discoverable **CloseListerData.lister.tabs** property.

## ***OnDisplayModeChange***

The **OnDisplayModeChange** event can be implemented by a [script add-in](#) to receive notification whenever the user changes the [display mode](#) in a tab.

<b>Method Name:</b>	OnDisplayModeChange
<b>Argument Type:</b>	<a href="#">DisplayModeChangeData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>DisplayModeChangeData.tab</b> property identifies the tab, and the <b>mode</b> property identifies the new display mode.

## ***OnDoubleClick***

The **OnDoubleClick** event can be implemented by a [script add-in](#) to receive notification when the user double-clicks on a file or folder in a tab.

By default your event handler is passed an [Item](#) object corresponding to the item that was double-clicked. Because constructing an Item object may take some time (e.g. on a network drive) you have the option for your handler to be called twice - once with only the path to the item, and a second time (if desired) with the full **Item** object. To do this:

1. Set the [ScriptInitData.early\\_dbclck](#) property to **True** when you initialize your script.
2. Your **OnDoubleClick** event will then be called with a the **early** property set to **True** in the [DoubleClickData](#) object.
3. When **early** is **True**, the **item** property is not present; instead, the **path** property provides the full path of the object, and the **is\_dir** property indicates whether the item is a folder or file.
4. When the **OnDoubleClick** method returns, it will be called a second time, with **early** set to **False** and a full **Item** object available in the **item** property.
5. If you sets the **skipfull** property to **True** in the [DoubleClickData](#) object at the "early" stage, the second call to **OnDoubleClick** doesn't occur.

<b>Method Name:</b>	OnDoubleClick
<b>Argument Type:</b>	<a href="#">DoubleClickData</a>
<b>Return Type:</b>	<i>bool</i> or <i>string</i>
<b>Description:</b>	<p>The <b>DoubleClickData.tab</b> property identifies the tab, and the <b>item</b> property identifies the item that has been double-clicked.</p> <p>You can return two different types from this event:</p> <ul style="list-style-type: none"> <li>• <i>bool</i>: If you return <b>True</b>, the double-click will be cancelled and the file will not be opened. If you return <b>False</b> the double-click will be allowed to continue (this is the default).</li> <li>• <i>string</i>: You can return a <i>string</i> to change the function to be performed on the file. For example, you could return the string "Show" to run the internal <b>Show</b> command on the file.</li> </ul>

### ***OnFlatViewChange***

The **OnFlatViewChange** event can be implemented by a [script add-in](#) to receive notification whenever the user changes the [display mode](#) in a tab.

<b>Method Name:</b>	OnFlatViewChange
<b>Argument Type:</b>	<a href="#">FlatViewChangeData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>FlatViewChangeData.tab</b> property identifies the tab, and the <b>mode</b> property ("off", "grouped", "mixed", "mixednofolders").

### ***OnGetCopyQueueName***

The **OnGetCopyQueueName** event can be implemented by a [script add-in](#) to override the default copy queue behavior when the *Automatically manage file copy queues* option on the [File Operations / Copy Options](#) page in Preferences is turned on. The event is passed the default copy queue name along with information relating to the copy operation. It can accept the default queue name, provide its own or disable queuing and run the operation immediately.

<b>Method Name:</b>	OnGetCopyQueueName
<b>Argument Type:</b>	<a href="#">GetCopyQueueNameData</a>
<b>Return Type:</b>	<p><i>string</i> to provide a new queue name.</p> <p><b>False</b> to accept the default queue name.</p> <p><b>True</b> to bypass the queue and run the copy immediately.</p>
<b>Description:</b>	<p>The <b>GetCopyQueueNameData.sourcetab</b> and <b>desttab</b> properties identify the <a href="#">Tab</a> objects involved in the copy operation, and the <b>source</b> and <b>dest</b> properties provide the source and destination <a href="#">Path</a> objects.</p> <p>The <b>source_drives</b> and <b>dest_drives</b> properties return a string consisting of <b>0</b> and <b>1</b> characters, indicating which physical drives are involved in the operation. For example, if drive A: was involved, the first character would be a <b>1</b>, otherwise it would be a <b>0</b>.</p> <p>The <b>name</b> property indicates the default copy queue name, and <b>move</b> is <b>True</b> if the operation is a <b>move</b> rather than a copy.</p> <p>You can accept the default name by returning <b>False</b>, or return the a new queue name to use. If you return <b>True</b> the queue will be bypassed.</p>

## OnGetCustomFields

The **OnGetCustomFields** event can be implemented by a [rename script](#) to add [custom fields](#) to the *Rename* dialog. This lets you provide one or more controls that users can use to pass parameters to your script.

<b>Method Name:</b>	OnGetCustomFields
<b>Argument Type:</b>	<a href="#">GetCustomFieldData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	<p>The <i>Rename</i> dialog will call this method when your script is loaded from a preset (or when you click the <b>Refresh</b> button in the integrated script editor).</p> <p>The <b>GetCustomFieldData.fields</b> property lets you add one or more custom fields to the <i>Rename</i> dialog. You can also provide labels for your fields using the <b>field_labels</b> property, and cue banners for any edit fields using the <b>field_tips</b> property.</p> <p>Each property is a <a href="#">Map</a> object which you can populate to create your custom fields. See <a href="#">Custom Fields in the Rename Dialog</a> for more information.</p>

## OnGetHelpContent

The

U

event can be implemented by a [script add-in](#) to add its own help content to the Directory Opus local F1 help. You can add a single page of help, or multiple pages (in which case the first page becomes the "topic header" and all subsequent pages appear below it in the index). You can also add images which can be displayed in the help content. Note that script-added help is only possible if the user has local http help enabled (which is the default) - if the user has chosen to use the old *HtmlHelp* interface then your content won't appear.

<b>Method Name:</b>	OnGetHelpContent
<b>Argument Type:</b>	<b>GetHelpContentData</b>
<b>Return Type:</b>	None.
<b>Description:</b>	<p>Use the <b>AddHelpPage</b> and <b>AddHelpImage</b> methods to add your help content.</p> <p>The <b>source_drives</b> and <b>dest_drives</b> properties return a string consisting of <b>0</b> and <b>1</b> characters, indicating which physical drives are involved in the operation. For example, if drive A: was involved, the first character would be a <b>1</b>, otherwise it would be a <b>0</b>.</p> <p>The <b>name</b> property indicates the default copy queue name, and move is <b>True</b> if the operation is a <b>move</b> rather than a copy.</p> <p>You can accept the default name by returning <b>False</b>, or return the a new queue name to use. If you return <b>True</b> the queue will be bypassed.</p>

## OnGetNewName

The **OnGetNewName** event is one of the three alternate entry points for [rename scripts](#). The other two events - **Rename\_GetNewName** and **Rename\_GetNewName2**, are deprecated should only be used for backwards compatibility with Opus 10.

<b>Method Name:</b>	OnGetNewName
<b>Argument Type:</b>	<a href="#">GetNewNameData</a>
<b>Return Type:</b>	<i>bool</i> or <i>string</i>
<b>Description:</b>	<p>When using a rename script, the <b>OnGetNewName</b> event is called for every selected file and folder, giving the script a chance to modify the name.</p> <p>The <b>GetNewNameData.item</b> property identifies the item being renamed. You can access the item's metadata and other information using the various <a href="#">Item</a> properties.</p> <p>The <b>oldname</b> property returns the "old name" pattern as entered by the user in the <a href="#">rename dialog</a>. The <b>newname</b> property returns the proposed new name of the item. This represents the result of all the other non-scripted options in the rename dialog (capitalization, automatic numbering, etc).</p> <p>You can return two different types from this event:</p> <ul style="list-style-type: none"><li>• <i>bool</i>: If you return <b>True</b>, the item will not be renamed. If you return <b>False</b> (this is the default) the item will be renamed to the proposed new name (the <b>newname</b> property).</li><li>• <i>string</i>: You can return a <i>string</i> to specify a new name for the item.</li></ul>

## OnInit

The **OnInit** event is called once for each [script add-in](#) to initialize it. The event will be called on program startup, and also if a script is added or edited while Opus is already running. Implementing the **OnInit** event is optional, but highly recommended as it allows you to provide a name, description and other information to be shown to the user in Preferences. It also provides a way for a script to [add internal commands](#) and [columns](#) (although the [OnAddCommands](#) and [OnAddColumns](#) methods provide a better way to do this).

<b>Method Name:</b>	OnInit
<b>Argument Type:</b>	<a href="#">ScriptInitData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	<p>When Opus starts up, or when a script add-in is added or edited, its <b>OnInit</b> method is called. This gives the script a chance to tell Opus something about itself, by setting the various properties of the <b>ScriptInitData</b> object. The <b>AddCommand</b> method can also be used from this event to <a href="#">add internal commands</a> to the Opus command set, and AddColumn method can be used to add <a href="#">columns</a>.</p> <p>If you return <b>True</b> from this method, the script will be disabled until the next time <b>OnInit</b> is called (which normally would be the next time Opus is run). For instance, you might want to do this if the version of Windows isn't appropriate for your script.</p>

## OnListerResize

The **OnListerResize** event can be implemented by a [script add-in](#) to receive notification whenever a Lister window is resized.

<b>Method Name:</b>	OnListerResize
<b>Argument Type:</b>	<a href="#">ListerResizeData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>ListerResizeData.lister</b> property identifies the Lister, and the <b>action</b> property is a <i>string</i> indicating the element that resize action taken. Use the <b>width</b> and <b>height</b> properties to determine the new window size.

## ***OnListerUIChange***

The **OnListerUIChange** event can be implemented by a [script add-in](#) to receive notification when the state of certain user interface elements in the Lister changes.

<b>Method Name:</b>	OnListerUIChange
<b>Argument Type:</b>	<a href="#">ListerUIChangeData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>ListerUIChangeData.lister</b> property identifies the Lister, and the <b>change</b> property is a <i>string</i> indicating the element that changed. You can discover the actual state of the element using the <a href="#">Command.IsSet</a> method.

## ***OnOpenLister***

The **OnOpenLister** event can be implemented by a [script add-in](#) to receive notification when a new Lister is opened.

<b>Method Name:</b>	OnOpenLister
<b>Argument Type:</b>	<a href="#">OpenListerData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	<p>The <b>OpenListerData.lister</b> property identifies the newly opened Lister.</p> <p>This event is initially called immediately after the Lister has been created, before any folders have been read or any tabs have been opened. If you return <b>True</b> from this event, it will be called again after all tabs have been created. You can use the <code>OpenListerData.after</code> property to distinguish between these calls.</p>

## OnOpenTab

The **On** event can be implemented by a [script add-in](#) to receive notification when a new tab opens.

<b>Method Name:</b>	OnOpenTab
<b>Argument Type:</b>	<a href="#">OpenTabData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>OpenTabData.tab</b> property identifies the newly opened tab. Note that this method is not called when a new Lister is opened, irrespective of how many tabs it contains - instead, you can identify all the tabs in the newly opened Lister by implementing the <b>OnOpenLister</b> event.

## OnScriptColumn

The **OnScriptColumn** event is the entry point for a [custom column](#) added by a [script add-in](#). The actual name of the event is defined by the script itself, when the command is added via the [ScriptInitData.AddColumn](#) method - **OnScriptColumn** is merely a placeholder name.

<b>Method Name:</b>	OnScriptColumn
<b>Argument Type:</b>	<a href="#">ScriptColumnData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	<p>When a script add-in adds a new column using <a href="#">ScriptInitData.AddCommand</a>, it specifies the name of its entry point with the <b>ScriptColumn.method</b> property. When Opus wants to retrieve the value of the column for a particular file or folder, Opus will call that method within your script.</p> <p>The <a href="#">ScriptColumnData.item</a> property provides information about the file or folder in question, and the <b>col</b> property identifies the column you should return data for (in case you use the one method for more than one columns).</p> <p>The return value from this event is ignored - instead, you should return the column data (and optionally, sorting and grouping information) by setting the appropriate values in the <a href="#">ScriptColumnData</a> object.</p>

## OnScriptCommand

The **OnScriptCommand** event is the entry point for an [internal command](#) added by a [script add-in](#). The actual name of the event is defined by the script itself, when the command is added via the [ScriptInitData.AddCommand](#) method - **OnScriptCommand** is merely a placeholder name.

<b>Method Name:</b>	OnScriptCommand
<b>Argument Type:</b>	<a href="#">ScriptCommandData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	<p>When a script add-in adds a new internal command using <b>ScriptInitData.AddCommand</b>, it specifies the name of its entry point with the <b>ScriptCommand.method</b> property. When the internal command is run, Opus will call that method within your script.</p> <p>The <b>ScriptCommandData.func</b> property provides information about the command environment (including any parsed arguments), and the <b>cmdline</b> property provides the raw command line that invoked your command.</p> <p>If this event returns <b>True</b> the function will be aborted - you might do this if an error occurs and the user chooses to abort the operation.</p>

## OnScriptConfigChange

The **OnScriptConfigChange** event can be implemented by a [script add-in](#) to receive notification whenever the user modifies the script's configuration via the Preferences editor.

<b>Method Name:</b>	OnScriptConfigChange
<b>Argument Type:</b>	<a href="#">ConfigChangeData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>ConfigChangeData.changed</b> property identifies the configuration items that were modified.

## ***OnShutdown***

The **OnShutdown** event can be implemented by a [script add-in](#) to receive notification when Opus is shutting down.

<b>Method Name:</b>	OnShutdown
<b>Argument Type:</b>	<a href="#">ShutdownData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	Opus calls this event when it is shutting down. The <b>ShutdownData.endsession</b> property will be <b>True</b> if Opus is quitting because Windows is shutting down. If <b>endsession</b> is <b>False</b> , you can return <b>True</b> from this event to prevent Opus from quitting - the return value is ignored if Windows is shutting down too.

## ***OnSourceDestChange***

The **OnSourceDestChange** event can be implemented by a [script add-in](#) to receive notification whenever a tab's source/destination state changes.

<b>Method Name:</b>	OnSourceDestChange
<b>Argument Type:</b>	<a href="#">SourceDestData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>SourceDestData.tab</b> property identifies the tab, and the <b>source</b> and <b>dest</b> properties identify the new state (if both are <b>False</b> it means the tab is "off" - this can only happen in a single file-display Lister).

## ***OnStartup***

The **OnStartup** event can be implemented by a [script add-in](#) to receive notification when Opus starts up.

<b>Method Name:</b>	OnStartup
<b>Argument Type:</b>	<a href="#">StartupData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	This event is only triggered when Opus starts up, therefore only script add-ins that are already installed will receive it. If a script is installed when Opus is already running it won't receive <b>OnStartup</b> until the next time Opus is started.

### ***OnStyleSelected***

The **OnStyleSelected** event can be implemented by a [script add-in](#) to receive notification when the user selects a new [Lister style](#).

<b>Method Name:</b>	OnStyleSelected
<b>Argument Type:</b>	<a href="#">StyleSelectedData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	The <b>StyleSelectedData.lister</b> property identifies the Lister, and the <b>style</b> property returns the name of the newly selected style.

### ***OnTabClick***

The **OnTabClick** event can be implemented by a [script add-in](#) to receive notification when a tab is clicked with a qualifier key held down. You can use this to override the default behavior (e.g. control-clicking tabs normally links them).

<b>Method Name:</b>	OnTabClick
<b>Argument Type:</b>	<a href="#">TabClickData</a>
<b>Return Type:</b>	<i>bool</i>
<b>Description:</b>	The <b>TabClickData.tab</b> property identifies the tab that was clicked. You can return <b>True</b> from this event to prevent the default action, or <b>False</b> (which is the default) to allow it to proceed.

## OnViewerEvent

The **OnViewerEvent** event can be implemented by a [script add-in](#) to receive notification whenever certain events occur in the [standalone image viewer](#). One possible use would be a script that automatically displays a floating toolbar whenever a standalone viewer is active, and hides it again when the window goes inactive or closes.

<b>Method Name:</b>	OnViewerEvent
<b>Argument Type:</b>	<a href="#">ViewerEventData</a>
<b>Return Type:</b>	<i>none</i>
<b>Description:</b>	<p>The <b>ViewerEventData.viewer</b> property identifies the <a href="#">Viewer</a> that the event occurred in. The <b>event</b> property returns a string identifying the event that occurred, and if applicable the <b>item</b> property identifies the <b>Item</b> involved.</p> <p>The currently defined events are <b>create</b>, <b>setfocus</b>, <b>killfocus</b>, <b>destroy</b> and <b>load</b>.</p>

# DOpusRT Reference

DOpusRT is a separate program (**dopusrt.exe**) that performs various support functions for Directory Opus:

- If [Double-click on the desktop](#) is enabled, it runs in the background on startup to detect double-clicks and launch Opus when it's not already running.
- If [Explorer Replacement](#) mode is enabled, it is used as the default "handler" when opening a folder (and again, it can launch Opus when not already running).
- You can use it to run internal commands from outside of Opus.
- You can use it to display a picture with the standalone viewer from outside of Opus.
- You can use it to manipulate [file collections](#) from outside of Opus.
- You can use it to [retrieve information about currently displayed folders and files](#) from outside of Opus.

The program **dopusrt.exe** is located in the main Opus program folder (normally *C:\Program Files\GPSoftware\Directory Opus*). Note that if you are invoking DOpusRT from an Opus button or hotkey, you don't need to use the full path - you can use the **dopusrt** keyword as a shortcut for the actual location of the .exe. Otherwise, you would normally need to specify the full path of the command when using it (for example, in a shortcut or a batch file). In the examples below, the full path is omitted for clarity.

The DOpusRT command line consists of a *command* followed by *arguments* appropriate to that command. For completeness, all **dopusrt.exe** commands are listed here - but many are designed to only be used by Opus itself. The commands you will most probably be interested in are **/cmd** and **/col**.

Command	Arguments	Description
<b>/acmd</b>	<i>&lt;command and arguments&gt;</i>	Invokes an Opus command from outside of Opus. If one or more Opus listers are open, the command is sent to the one which is, or was most recently, the active window.  <i>Example: <b>dopusrt.exe /acmd Go "C:\Program Files"</b></i>  <i>- navigates to "C:\Program Files" in the last-active window (or in a new window if there isn't one)</i>
<b>/argmsgbox</b>	<i>(no arguments)</i>	Displays any remaining arguments in a message box. This can be useful when you wish to see the exact command-line that an Opus command is generating when calling an external command.

		<p><i>Example: <b>dopusrt.exe /argmsgbox Hello World</b></i></p> <p><i>- display "Hello World" in a message box</i></p>
<b>/argstoclip</b>	<i>(no arguments)</i>	<p>Puts any remaining arguments into the clipboard. As with <b>/argmsgbox</b>, this can be useful for checking what Opus would pass to an external command.</p> <p>Note that Opus has a built-in <a href="#">Clipboard</a> command which is a better choice if you just want to put some text or filenames into the clipboard.</p> <p><i>Example: <b>dopusrt.exe /argstoclip Hello World</b></i></p> <p><i>- put "Hello World" into the clipboard</i></p>
<b>/changelanguage</b>	<i>&lt;language&gt;</i>	<p>Changes the user-interface language. Opus will restart, and toolbars and menus will be translated (if possible).</p> <p><i>Example: <b>dopusrt.exe /changelanguage deutsch</b></i></p> <p><i>- change language to German</i></p>
<b>/cmd</b>	<i>&lt;command and arguments&gt;</i>	<p>Invokes an Opus command from outside of Opus. If one or more Opus listers are open, the command is sent to the Source window. (Compare this with <b>/acmd</b>, above.)</p> <p><i>Example: <b>dopusrt.exe /cmd Go C:\NEW=0,0,640,480</b></i></p> <p><i>- opens a new Lister showing C:\ at the specified coordinates</i></p>
<b>/col</b>	<i>&lt;collection command&gt;</i>	<p>Lets you manipulate file collections from outside of Opus. See the <a href="#">External Manipulation of File Collections</a> page for more information.</p> <p><i>Example: <b>dopusrt.exe /col create "Holiday Photos"</b></i></p> <p><i>- creates a collection called Holiday Photos</i></p>

<b>/dblclk</b>	<i>(no arguments)</i>	<p>Runs in the background to manage the double-click on the desktop feature.</p> <p>You do not need to run this command manually - Opus starts it automatically when the desktop double-click option is enabled.</p>
	<b>=off</b>	<p>Deactivate a running double-click manager. You can use this command if you want to temporarily disable the desktop double-click feature (rather than using Task Manager to kill the <b>dopusrt.exe</b> instance).</p> <p><i>Example: <b>dopusrt.exe /dblclk=off</b></i></p> <p><i>- shuts down the double-click on desktop handler</i></p>
<b>/dde</b>	<i>(no arguments)</i>	<p>Run a DDE server to listen for DDE commands relating to opening folders.</p> <p>This is used in Explorer Replacement mode when you double-click on a folder. You do not need to run this command manually.</p>
<b>/flushplugins</b>	<i>(no arguments)</i>	<p>Flush viewer plugins from memory. If Opus is running, any plugins not currently in use will be unloaded. You can use this if you are developing a plugin and want to replace the DLL file without having to quit Opus.</p> <p><i>Example: <b>dopusrt.exe /flushplugins</b></i></p> <p><i>- flushes unused viewer plugins</i></p>
<b>/help</b>	< <i>help path</i> >	<p>Displays the specified document from the local help in your web browser. This is used to implement the <b>opushelp://</b> URL scheme, which Opus registers when it's installed.</p>
<b>/info</b>	< <i>information command</i> >	<p>Lets you retrieve information about the currently displayed folders and files from outside of Opus. See the <a href="#">Retrieving File and Folder Information</a> page for more information.</p>

		<p><i>Example: <b>dopusrt.exe /info path_list.txt,paths</b></i></p> <p><i>- creates an XML file (path_list.txt) listing the currently displayed paths in all currently open Listers</i></p>
<b>/open</b>	<folder path>	<p>Opens a new Lister showing the specified path.</p> <p><i>Example: <b>dopusrt.exe /open "C:\Program Files"</b></i></p> <p><i>- open a Lister showing the C:\Program Files folder</i></p>
<b>/restart</b>	(no arguments)	<p>Restarts Opus if it is already running. If Opus is not already running it will be started.</p> <p><i>Example: <b>dopusrt.exe /restart</b></i></p> <p><i>- restarts or launches Opus</i></p>
	<b>:norun</b>	<p>Restarts Opus if it is already running, but does not start it if it is not running.</p> <p><i>Example: <b>dopusrt.exe /restart:norun</b></i></p> <p><i>- restarts Opus if it's already running</i></p>
<b>/runopus</b>	(no arguments)	<p>Starts Opus if it is not already running. Under Vista and above, this will always launch Opus as non-elevated, even if the process invoking <b>dopusrt.exe</b> is elevated.</p> <p><i>Example: <b>dopusrt.exe /runopus</b></i></p> <p><i>- runs Opus (non-elevated) if not already running</i></p>
<b>/runstd</b>	<command line>	<p>Lets you launch another program. Under Vista and above, this will run the specified command as non-elevated, even if the process invoking <b>dopusrt.exe</b> is elevated (this is really the only point of this command).</p> <p><i>Example: <b>dopusrt.exe /runstd notepad.exe</b></i></p>

		- runs <i>notepad.exe</i> as non-elevated
<b>/show</b>	<filename>	Shows the specified file using the Opus <a href="#">standalone viewer</a> . You could use this to make the Opus viewer the default system image viewer for a file type.  <i>Example: dopusrt.exe /show %1</i>  - shows the selected image (definition for context menu item)
<b>/vcmd</b>	<command and arguments>	Invokes an Opus command from outside of Opus, in the context of the currently (or most recently) active image viewer. If no image viewers are open the command will be ignored.  <i>Example: dopusrt.exe /vcmd Show VIEWERCMD=selectall</i>  - selects the image in the currently active viewer
<filename>	(no arguments)	DOPUSRT can be called with the name of an external file to launch various Opus files. This behaviour is used by the default file type associations that Opus creates - for example, a <b>.dcf</b> file (an Opus command file, created by dragging a toolbar button to the desktop in Customize mode) is launched by passing its filename to <b>dopusrt.exe</b> .  The file types that can be invoked in this way are <b>.dcf</b> (command files), <b>.dpf</b> (exported Preferences files from older versions of Opus) and <b>.dop</b> (toolbars).  <i>Example: dopusrt.exe C:\Commands\NewLister.dcf</i>  - executes the command in the exported command file

## External Manipulation of File Collections

The [DOPUSRT](#) tool can be used from outside of Opus to manipulate [File Collections](#).

This functionality is accessed using the **dopusrt.exe /col** command - following **/col** you must supply a *collection command*, and the appropriate arguments for that collection command.

As with most command-line tools, if a path or other argument contains spaces, you must "put quotes around it".

Command	Arguments	Description
<b>add</b>	<i>&lt;coll-name&gt;</i> <i>&lt;item&gt;</i> [ <i>&lt;item&gt;</i> ...]	<p><b>Add one or more items to the named collection.</b></p> <p><i>&lt;coll-name&gt;</i> is the name of the collection (if adding to a sub-collection, the full path must be provided).</p> <p><i>&lt;item&gt;</i> is the full path and filename of the item or items to add. If the path or filename contains spaces, it must be enclosed with quotes. The filename can also contain <a href="#">standard wildcards</a> to add multiple items in a folder at once.</p> <p>Example:</p> <pre><b>dopusrt.exe /col add "Photos/Holidays" /mypictures/Hawaii/*.jpg</b></pre>
<b>clear</b>	[ <b>/full</b> ] <i>&lt;coll-name&gt;</i>	<p><b>Clears the contents of the named collection.</b></p> <p><i>&lt;coll-name&gt;</i> is the name of the collection (if clearing a sub-collection, the full path must be provided). By default, sub-collections are not removed when the collection is cleared, but you can specify the <b>/full</b> flag to do this.</p> <p>Example:</p> <pre><b>dopusrt.exe /col clear "Find Results" dopusrt.exe /col clear /full "Marked Pictures"</b></pre>

<b>create</b>	<i>[&lt;flags&gt;] &lt;coll-name&gt;</i>	<p><b>Create a new collection.</b></p> <p><i>&lt;flags&gt;</i> are one or more optional flags, see below for a list of these. If a flag's value contains a space, the entire flag and value must be enclosed in quotes.</p> <p><i>&lt;coll-name&gt;</i> is the name of the collection to create. To create a sub-collection, specify the full path of the collection. The collection name must come after any of the optional flags.</p> <p>Example:</p> <p><b>dopusrt.exe /col create "/desc:My Photos" Photos</b></p>
	<b>/noclear</b>	Does not clear the collection if it already exists.
	<b>/icon:&lt;file&gt;</b>	Specifies a custom icon for the new collection.
	<b>/desc:&lt;desc&gt;</b>	Specifies a description for the new collection.
	<b>/query</b>	Creates the new collection as a <a href="#">stored query</a> .
<b>delete</b>	<i>&lt;coll-name&gt;</i>	<p><b>Delete the named collection.</b></p> <p><i>&lt;coll-name&gt;</i> is the name of the collection to delete. If deleting a sub-collection, the full path must be provided.</p> <p>Example:</p> <p><b>dopusrt.exe /col delete "Photos/Holidays/2010"</b></p>
<b>export</b>	<i>[&lt;flags&gt;] &lt;coll-name&gt; &lt;export-file&gt;</i>	<p><b>Export the contents of a collection to a text file.</b></p> <p><i>&lt;flags&gt;</i> are one or more optional flags, see below for a list of these.</p>

		<p><i>&lt;coll-name&gt;</i> is the name of the collection to export.</p> <p><i>&lt;export-file&gt;</i> is the full path and file to export to - this must be quoted if it contains spaces.</p> <p>If the encoding type is not specified (using the optional flags), the file will be encoded as UTF16-LE if any filenames require Unicode, and otherwise in the current ANSI code-page.</p> <p>Example:</p> <p><b>dopusrt.exe /col export /utf8 "Find Results" D:\Results.txt</b></p>
	<b>/append</b>	If the export file already exists, append to it - otherwise it will be overwritten.
	<b>/utf16be</b>	Force encoding to UTF16-BE.
	<b>/utf16le</b>	Force encoding to UTF16-LE.
	<b>/utf8</b>	Force encoding to UTF8.
	<b>/ansi</b>	Force encoding to the current ANSI code-page.
	<b>/cp:&lt;codepage&gt;</b>	Specify the code-page.
<b>import</b>	<i>[&lt;flags&gt;] &lt;coll-name&gt; &lt;import-file&gt;</i>	<p><b>Import the contents of a text file to a collection.</b></p> <p><i>&lt;flags&gt;</i> are one or more optional flags, see below for a list of these.</p> <p><i>&lt;coll-name&gt;</i> is the name of the collection to import to.</p> <p><i>&lt;import-file&gt;</i> is the file to import from.</p> <p>If the import file has a BOM this will be used to determine the encoding type.</p> <p>Example:</p> <p><b>dopusrt.exe /col import /ansi "New Photos" /desktop/photos.txt</b></p>

	<b>/clear</b>	Clear the collection before importing the new items.
	<b>/create</b>	Create the collection if it doesn't already exist.
	<b>/nocheck</b>	Don't check that the items listed in the import file exist before importing them into the collection.
	<b>/relative:&lt;path&gt;</b>	<p>Specify the path which lines in the list are relative to, if the file does not contain fully qualified paths. If not specified, paths are assumed to be relative to the same folder the list is in.</p> <p>Specify <b>/relative:none</b> if doing something special where you want the lines imported as-is even if they aren't fully qualified, in which case this cannot be combined with <b>/nocheck</b>.</p> <p>If the path contains spaces, the entire argument should be quoted.</p> <p>Example:</p> <pre style="text-align: center;"><b>dopusrt.exe /col import "/relative:C:\My Music" "Jazz" C:\Trane.m3u</b></pre>
	<b>/utf16be</b>	Assume UTF16-BE if no BOM.
	<b>/utf16le</b>	Assume UTF16-LE if no BOM.
	<b>/utf8</b>	Assume UTF8 if no BOM.
	<b>/ansi</b>	Force conversion from the current ANSI code-page.
	<b>/cp:&lt;codepage&gt;</b>	Force conversion from a specific code-page.
<b>remove</b>	<i>&lt;coll-name&gt;</i> <i>&lt;item&gt;</i> [ <i>&lt;item&gt;</i> ...]	<p><b>Remove items from a collection.</b></p> <p><i>&lt;coll-name&gt;</i> is the name of the collection.</p> <p><i>&lt;item&gt;</i> is the name of the item or items to remove. This can be given as either the item's full path on disk, or its name within the collection. If the path or filename contains spaces, it must be enclosed with quotes. The filename can also contain <a href="#">standard wildcards</a> to remove multiple items at once.</p>

		<p>Example:</p> <p><b>dopusrt.exe /col remove "Find Results" *.txt</b></p>
<b>rename</b>	<p><i>&lt;old-coll-name&gt;</i>  <i>&lt;new-coll-name&gt;</i></p>	<p><b>Rename a collection.</b></p> <p><i>&lt;old-coll-name&gt;</i> is the existing name of the collection.</p> <p><i>&lt;new-coll-name&gt;</i> is the new name for the collection.</p> <p>Example:</p> <p><b>dopusrt.exe /col rename "Find Results" "Saved Results 1"</b></p>
<b>runquery</b>	<p><i>&lt;coll-name&gt;</i></p>	<p><b>Run (refresh) a <a href="#">stored query</a>.</b></p> <p><i>&lt;coll-name&gt;</i> is the name of the stored query collection.</p> <p>Example:</p> <p><b>dopusrt.exe /col runquery "Stored Queries\Backup Files"</b></p>
<b>setdesc</b>	<p><i>&lt;coll-name&gt;</i>  <i>&lt;desc&gt;</i></p>	<p><b>Set the description for a collection.</b></p> <p><b>&lt;coll-name&gt;</b> is the name of the collection.</p> <p><b>&lt;desc&gt;</b> is the new description for the collection (or nothing to clear the description).</p> <p>Example:</p>

		<b>dopusrt.exe /col setdesc "Saved Results 1" "Music before 1990"</b>
<b>seticon</b>	<i>&lt;coll-name&gt;</i> <i>&lt;icon-file&gt;</i>	<p><b>Set a custom icon for a collection.</b></p> <p><i>&lt;coll-name&gt;</i> is the name of the collection.</p> <p><i>&lt;icon-file&gt;</i> is the full path and filename of the icon file to use.</p> <p>The icon can come from a .ico or .icl file, or a .exe or .dll file. For files that contain more than one icon, append the desired icon index following the filename.</p> <p>Example:</p> <p><b>dopusrt.exe /col seticon "Pics" C:\Tools\viewer.exe,2</b></p>
<b>setpaths</b>	<p>[<i>&lt;flags&gt;</i>] <i>&lt;coll-name&gt;</i> <i>&lt;path&gt;</i> [<i>&lt;path&gt;</i> ...]</p>	<p><b>Set the search path or paths for a <a href="#">stored query</a>.</b></p> <p><i>&lt;flags&gt;</i> are optional flags, see below for a list of these.</p> <p><i>&lt;coll-name&gt;</i> is the name of the stored query collection.</p> <p><i>&lt;path&gt;</i> is the path or paths to add to the query. If a path contains spaces it must be enclosed in quotes.</p> <p>Example:</p> <p><b>dopusrt.exe /col setpaths "Backup Files" X:\Backup</b></p>
	<b>/add</b>	Add the paths to the query, don't remove any existing ones.

<b>setquery</b>	<i>[&lt;flags&gt;] &lt;coll-name&gt; &lt;query&gt;</i>	<p><b>Set the query string for a <a href="#">stored query</a>.</b></p> <p>&lt;flags&gt; are one or more optional flags, see below for a list of these.</p> <p>&lt;coll-name&gt; is the name of the stored query collection.</p> <p>&lt;query&gt; is the query string, in <a href="#">Advanced Query Syntax</a>.</p> <p>The named collection must have been created as a stored query (e.g. with the <b>dopusrt.exe /col create /query</b> command).</p> <p>Example:</p> <p style="text-align: center;"><b>dopusrt.exe /col setquery "Backup Files" name:*.bak</b></p>
	<b>/auto</b>	Set the stored query to "auto refresh" mode - the query will be run/refreshed whenever it is loaded.
	<b>/noauto</b>	Set the stored query to not automatically refresh.

## Retrieving File and Folder Information

The [DOpusRT](#) tool can be used from outside of Opus to retrieve information about the currently displayed folders and files.

This functionality is accessed using the **dopusrt.exe /info** command - following **/info** you must supply a filename for the output to be saved to, an *information command*, and the appropriate arguments for that information command. The basic template is:

```
dopusrt.exe /info <output file>,<command>[,<arguments>]
```

The data returned by the command will be saved to the specified file in XML format. You can omit the output filename if you do not care about the results. See below for examples of various commands.

Command	Arguments	Description
<b>list</b>	[<tab>]	<p><b>Returns a list of items displayed in the specified tab.</b></p> <p>&lt;tab&gt; is the handle of the tab to retrieve the file list from - this can be obtained from the output of the <b>paths</b> command. Specify <b>0</b> (or omit this argument) to retrieve the file list from the currently active tab.</p> <p>Example:</p> <pre><b>dopusrt.exe /info %temp%\filelist.txt,list</b></pre>
<b>listsel</b>	[<tab>]	<p><b>Returns a list of selected items displayed in the specified tab.</b></p> <p>&lt;tab&gt; is the handle of the tab to retrieve the file list from - this can be obtained from the output of the <b>paths</b> command. Specify <b>0</b> (or omit this argument) to</p>

		<p>retrieve the file list from the currently active tab.</p> <p>Example:</p> <pre><b>dopusrt.exe /info %temp%\filelist.txt,listsel,0x1a508</b></pre>
<b>open</b>	<code>&lt;tab&gt;,&lt;id&gt;[,&lt;id&gt;,...]</code>	<p><b>Opens one or more items in the specified tab.</b></p> <p><code>&lt;tab&gt;</code> is the handle of the tab to open items in - this can be obtained from the output of the <b>paths</b> command. Specify <b>0</b> to retrieve the file list from the currently active tab.</p> <p><code>&lt;id&gt;</code> is the ID of the item to open - this can be obtained from the output of the <b>list</b> command. You can specify multiple IDs by comma-separating them, you can also specify a range of IDs (e.g. <b>3-8</b>).</p> <p>Example:</p> <pre><b>dopusrt.exe /info ,open,0,15</b></pre> <p><i>(note that the output filename has been omitted since we do not care about the results, although the comma separating the filename from the command must still be included).</i></p>
<b>paths</b>	<i>(no arguments)</i>	<p><b>Return a list of paths currently displayed in all tabs in all Listers.</b></p> <p>Example:</p>

		<b>dopusrt.exe /info</b> <b>%temp%\pathlist.txt,paths</b>
<b>select</b>	<i>&lt;tab&gt;</i> , <i>&lt;id&gt;</i> [, <i>&lt;id&gt;</i> ,...]	<p><b>Selects one or more items in the specified tab.</b></p> <p><i>&lt;tab&gt;</i> is the handle of the tab to select files in - this can be obtained from the output of the <b>paths</b> command. Specify <b>0</b> to retrieve the file list from the currently active tab.</p> <p><i>&lt;id&gt;</i> is the ID of the item to select - this can be obtained from the output of the <b>list</b> command. You can specify multiple IDs by comma-separating them, you can also specify a range of IDs (e.g. <b>3-8</b>).</p> <p>Example:</p> <p><b>dopusrt.exe /info ,select,0x1a508,5,8-10</b></p> <p><i>(note that the output filename has been omitted since we do not care about the results, although the comma separating the filename from the command must still be included).</i></p>

# Metadata Keywords

There are several functions that use keywords to represent certain metadata, or file information, elements - for example, file name, size, artist, bit rate, etc. There are two separate sets of metadata keywords, and although the two sets are similar, they are not identical.

- [Keywords for SetAttr META](#) are used to make [programmatic changes to file metadata](#) using the [SetAttr](#) command
- [Keywords for Columns](#) are used with the [Set](#) command to change the information columns that are displayed in the file display, and the [Rename](#) command to rename files using metadata

## Keywords for Columns

The following keywords are used by the [Set](#) command (with the **COLUMNS**, **COLUMNSADD**, **COLUMNSREMOVE** and **COLUMNSTOGGLE** arguments) to change which columns are displayed in the file display (in details and power modes), and with the **GROUPBY** and **SORTBY** arguments to modify the group and sort fields. For example, **Set COLUMNSADD=desc** adds the Description column to the file display.

They are also used by the [Rename](#) function when [renaming using metadata](#), and in the [File Types](#) editor when defining your own [tile](#) and [infotip](#) definitions.

Note that some columns appear in multiple categories. For historic reasons, column keywords sometimes only reflect a narrow usage when the column actually works in a wider range of situations. For example, the *Duration* column (**mp3songlength**) works with various music file types, and some movie types, not just MP3 music files.

Where columns are used to output text (for example, when generating filenames for a rename, or when displaying info tips), the format of dates, times and numeric values can be overridden:

- Date and time fields let you configure the date format, the time format, or both. For example,  
**{datetaken|D#yyyy-MM-dd}** - inserts the date only in **yyyy-MM-dd** format (e.g. 2008-09-22).  
**{modified|T#HH-mm-ss}** - inserts the modified time in **HH-mm-ss** format (e.g. 13:10:55)  
**{datetaken|D#yyyyMMddT#HHmmss}** - inserts both the date and time as **yyyyMMddHHmmss** (e.g. 20051130154410).

As you can see in the examples, **D#** is used to mark the date format, and **T#** is used to mark the time format. See the [Codes for date and time](#) page for information on date and time formats.

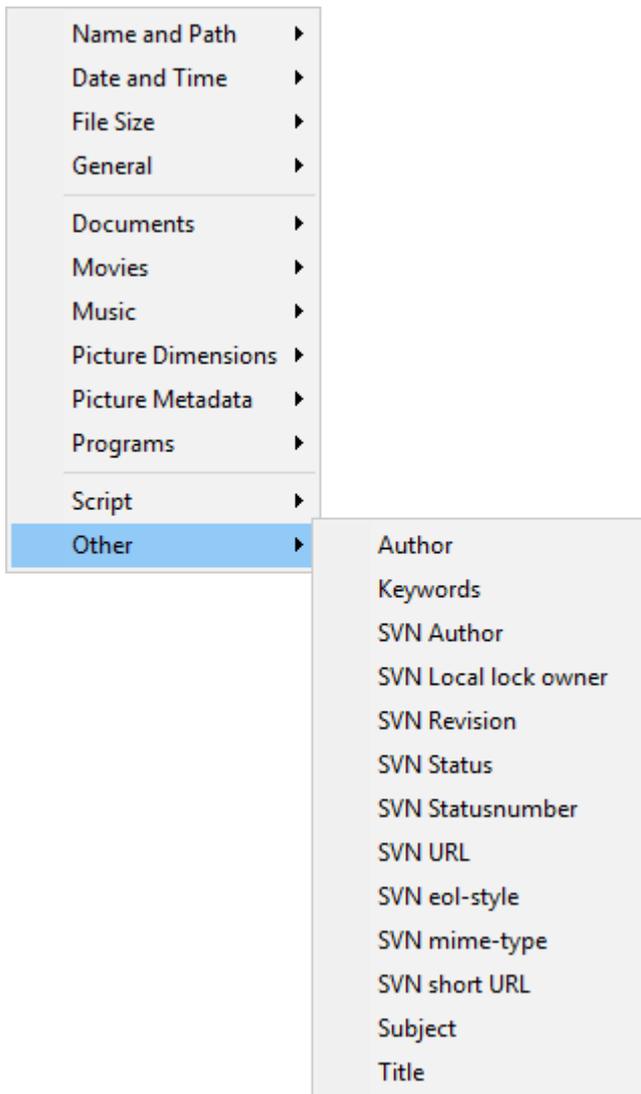
- Numeric fields let you control zero-padding. For example,  
**{size|#8}** - zero-pads the size in bytes to eight places (e.g. 00045412)  
**{mp3track|#2}** - zero-pads the track number to two places (e.g. 08)

<b>Column</b>	<b>Keyword</b>	<b>Column</b>	<b>Keyword</b>
<u><i>Date and Time category</i></u>		<u><i>Name and Path category</i></u>	
Create (relative)	cdaterel	Extension	ext
Date (accessed)	accesseddate	Extension (dirs)	extdir
Date (created)	createddate	Filename	name
Date (modified)	modifieddate	Full path	fullpath
Date and Time (accessed)	accessed	Location	path
Date and Time (created)	created	Location (relative)	pathrel
Date and Time (modified)	modified	Parent folder	parent
Modify (relative)	daterel	Parent folder (full)	parentlocation
Time (accessed)	accessedtime	Parent location	parentpath
Time (created)	createdtime	Path length	pathlen
Time (modified)	modifiedtime	Short name	shortname
<u><i>Documents category</i></u>		<u><i>Picture Dimensions category</i></u>	
Authors	author	Aspect ratio	aspectratio
Category	category	Bit depth	picdepth
Comment	comments	Dimensions	picsize dimensions
Company	companyname	Height	picheight
Copyright	copyright	Physical height	picphysy
Creator	creator	Physical size	picphyssize
Document created date	doccreateddate	Physical width	picphysx
Last edit time	docedittime	Resolution (X)	picresx
Last saved by	doclastsavedby	Resolution (Y)	picresy
Last saved date	doclastsaveddate	Rotation	rotation
Pages	pages	Width	picwidth
Producer	producer		
Subject	subject		
Title	title		
<u><i>File Size category</i></u>		<u><i>Picture Metadata category</i></u>	
Size (KB)	sizekb	Altitude	altitude
Size (auto)	sizeauto	Aperture	apertureval
Size (bytes)	size	Artists	mp3artist
Size (relative)	sizerel	Camera make	cameramake
Size on disk (KB)	disksizekb	Camera model	cameramodel
Size on disk (auto)	disksizeauto	Color space	colorspace
Size on disk (bytes)	disksize	Contrast	contrast
Size on disk (relative)	diskSizerel	Coordinates	coords
Uncompressed Size	uncompressedsize	Copyright	copyright
		Creation software	software
		Date digitized	datedigitized

<u>General category</u>		Date taken	datetaken shootingtime
Attributes	attr	Digital Zoom	digitalzoom
Availability	availability	Exposure bias	exposurebias
Description	desc	Exposure program	exposureprogram
File count	filecount	Exposure time	exposuretime
Font name	fontname	F-number	fnumber
Group	group	Flash	flash
Index	index	Focal length	focallength
Label	label	Focal length (35mm)	35mmfocallength
MD5 checksum	md5sum	ISO speed	isorating
Owner	owner	Image description	imagedesc
Rating	rating	Image quality	imagequality
SHA-1 checksum	shasum	Latitude	latitude
Status Icons	status	Lens type	lenstype
Sub-folder count	dircount	Longitude	longitude
Tags	keywords	Macro mode	macromode
Target	target	Metering mode	meteringmode
Thumbnail	thumbnail	Saturation	saturation
Total File count	filecounttotal	Scene capture type	scenecapturetype
Total Sub-folder count	dircounttotal	Scene mode	scenemode
Type	type	Sharpness	sharpness
User description	userdesc	Shutter speed	shutterspeed
		Subject	subject
<u>Movies category</u>		Subject distance	subjectdistance
Aspect ratio	aspectratio	Special instructions	instructions
Audio codec	mp3type	Title	title
Bit depth	picdepth	White balance	whitebalance
Bit rate	mp3bitrate		
Broadcast date	broadcastdate	<u>Programs category</u>	
Channel number	channel	Copyright	copyright
Credits	credits	Company	companyname
Data rate	datarate	Module Description	moddesc
Dimensions	picsize dimensions	Module Version	modversion
Duration	mp3songlength	Product Name	prodname
Episode name	episodename	Product Version	prodversion
FOURCC code	fourcc		
Frame rate	framerate		
Height	picheight		
High definition?	ishd		
Mode	mp3mode		
Physical size	picphyssize		
Publisher	publisher		
Recording time	recordingtime		

Repeat?	isrepeat
Sample rate	mp3samplerate
Station name	station
Video codec	videocodec
Width	picwidth
<i><u>Music category</u></i>	
Album	mp3album
Album artist	mp3albumartist
Artists	mp3artist
Audio codec	audiocodec mp3type
BPM	mp3bpm
Bit depth	picdepth
Bit rate	mp3bitrate
Composers	composers
Conductors	conductors
Copyright	copyright
Disc number	mp3disc mp3disk
Duration	mp3songlength
Encoded by	mp3encoder
Encoding Software	mp3encodingsoftware
Genre	mp3genre
Initial key	initialkey
Mode	mp3mode
Music comment	mp3comment
Music info	mp3info
Music title	mp3title
Protected	mp3drm
Release date	releasedate
Sample rate	mp3samplerate
Track number	mp3track
Year	mp3year

You can also access some columns that aren't part of the standard set of columns shown above. These can come from various places - third-party shell namespace extensions can provide custom columns, as can Opus plugins. Additionally, Zip and FTP define several columns that are only valid in those types of folders. In Opus, these columns all appear under the **Other** category. To refer to one of these columns from a command you need to know the name of the column, and then use the appropriate prefix to indicate that you want to use a special column.



Columns provided by shell extensions (as in the screenshot above) are prefixed with **sh:** in Opus - the keyword for the column is the name shown in the list without spaces.

For example, to add the **SVN Status** column you might use the command **Set COLUMNSADD sh:svnstatus**.

Columns provided by plugins (e.g. the 7-Zip plugin) must be prefixed by the name of the plugin DLL (e.g. **opus7zip:packed**).

The following column keywords are provided by Opus but only valid in certain folders:

- *FTP* keywords are only valid in an **ftp://** path
- *Zip* keywords are only valid when viewing a Zip archive
- *7-Zip* keywords are only valid when viewing an archive handled by the 7-Zip plugin
- *Computer* keywords are only valid when viewing the [native Computer folder](#)

Column	Keyword	Column	Keyword
<b>FTP keywords</b>			
Transfer time	ftp:xfertime		
Group	ftp:group		
<b>Computer keywords</b>			
Free space on drive	sh:freespace		
File system	sh:filesystem		
Used space (graph)	sh:usedpercent		
Free space (graph)	sh:freepercent		
Network location	sh:netlocation		
Used space on drive	sh:usedspace		
<b>Zip keywords</b>			
Compressed size	zip:compsize		
Compression ratio	zip:compratio		
Compression method	zip:compmethod		
CRC checksum	zip:compcrc		
		<b>7-Zip keywords</b>	
		Packed size	opus7zip:packed
		Ratio	opus7zip:ratio
		CRC	opus7zip:crc
		Block	opus7zip:block
		Index	opus7zip:index
		Is solid?	opus7zip:solid
		Method	opus7zip:method
		Mode	opus7zip:mode
		Link	opus7zip:link
		User	opus7zip:user
		Group	opus7zip:group

Within the **Lister Column Header** [Context Menu](#), you can use **%header%** to refer to the column which was right-clicked. For example, **Set GROUPBY=%header%,toggle** would group by the column which had been clicked to open the menu.

## Keywords for SetAttr META

The following keywords can be used to modify file metadata [programmatically](#) with the **SetAttr META** command. Most fields can only be set to a string (or number), or cleared, however some fields accept more complex instructions, and these are noted below.

Field	Keyword	Notes
<b>Standard Properties</b>		
Attributes	attr	Attributes are specified with one or more of the following letters:  A (archive) R (read-only)

		<p><b>H</b> (hidden)  <b>S</b> (system)  <b>C</b> (compressed)  <b>E</b> (encrypted)</p> <p>You can also use + to turn attributes on and - to turn attributes off.</p> <p><i>Example:</i></p> <p><b>SetAttr META attr:+c-r</b> - sets compression and clears read-only attributes</p>
Date created	createdate	<p>The date and time can be set to an absolute date (in which case the time will be unchanged), an absolute time (in which case the date will be unchanged), or both.</p> <p>The accepted formats for an absolute time and date are:</p> <p><b>YYYY-MM-DD</b> - set just the date  <b>HH:MM:SS</b> - sets just the time  <b>YYYY-MM-DD HH:MM:SS</b> - sets both time and date (needs quotes)</p> <p>The keyword <b>now</b> can also be given to use the current date and time.</p> <p>You can also perform relative adjustments to the current date and time setting, using the following formats:</p> <p><b>&lt;time-adjust&gt;</b> - adjusts only the time  <b>&lt;date-adjust&gt; &lt;time-adjust&gt;</b> - adjusts both date and time (needs quotes)</p> <p>The valid formats of the <b>&lt;time-adjust&gt;</b> string are:</p> <p><b>[+]<b>-</b>]H:M:S</b> - add or subtract hours, minutes and seconds  <b>[+]<b>-</b>]H:M</b> - add or subtract hours and minutes  <b>[+]<b>-</b>]H</b> - add or subtract a number of hours</p> <p>The valid formats of the <b>&lt;date-adjust&gt;</b> string are:</p> <p><b>[+]<b>-</b>]Y:M:D</b> - add or subtract a year, month and day value  <b>[+]<b>-</b>]M:D</b> - add or subtract a month and day</p>

		<p>value [+-]<b>D</b> - add or subtract a number of days</p> <p>You can't adjust the date without also adjusting the time - so specify 0 for <i>&lt;time-adjust&gt;</i> if you only want to adjust the date.</p> <p><i>Example:</i></p> <p><b>SetAttr META createdate:now</b> - sets the creation date to the current date/time  <b>SetAttr META createdate:20100922</b> - sets creation date to Sep 22, 2010  <b>SetAttr META "createdate:20100922 15:30:30"</b> - also sets time to 3:30 pm  <b>SetAttr META createdate:+1:30</b> - adds 1h30m to creation time  <b>SetAttr META "createdate:+1 0"</b> - adds 1 day (and zero hours)</p> <p>You can also copy the values from another date field by specifying the other field's name: <b>createdate</b>, <b>lastmodifieddate</b>, <b>datedigitized</b> or <b>datetaken</b>. (For backward compatibility, <b>modifydate</b> also works as an alias of lastmodifieddate.)</p> <p><i>Example:</i></p> <p><b>SetAttr META createdate:lastmodifieddate</b> - sets the creation date to the file's last modified date/time</p>
Date modified	lastmodifieddate	Accepts the same values as <b>createdate</b> (described above).
<b>Extended Properties</b>		
Comment	comment usercomment	User-defined string.
Rating	rating	Accepts a value from <b>0</b> (to clear the rating) through to <b>5</b> (5 stars).

Tags	tags	<p>Accepts multiple semi-colon separated tags. You can either set the tags absolutely, or add tags to or remove tags from the existing set.</p> <p><i>Example:</i></p> <p><b>SetAttr META tags:one;two</b> - sets the tags to "one" and "two"</p> <p><b>SetAttr META tags:+one</b> - adds the tag "one" to any existing tags</p> <p><b>SetAttr META tags:+one;-two</b> - adds "one" and removes "two"</p>								
<b>Picture Properties</b>										
Aperture	aperture	Accepts either a decimal or fractional value.								
Camera make	cameramake	User-defined string.								
Camera model	cameramodel	User-defined string.								
Contrast	contrast	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1" data-bbox="810 1361 1013 1529"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>normal</td> </tr> <tr> <td>1</td> <td>soft</td> </tr> <tr> <td>2</td> <td>hard</td> </tr> </tbody> </table> <p><i>Example:</i></p> <p><b>SetAttr META contrast:2</b></p>	Value	Keyword	0	normal	1	soft	2	hard
Value	Keyword									
0	normal									
1	soft									
2	hard									
Creation software	software	User-defined string.								
Date digitized	datedigitized	Accepts the same values as <b>createdate</b> (described above in the <b>Standard Properties</b> section).								

Date taken	datetaken	Accepts the same values as <b>createdate</b> (described above in the <b>Standard Properties</b> section).																				
Digital Zoom	digitalzoom	Accepts a decimal or fractional value, as well as the keyword <b>off</b> .																				
Exposure bias	exposurebias	Accepts either a decimal or fractional value.																				
Exposure program	exposureprogram	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>notdefined</td> </tr> <tr> <td>1</td> <td>manual</td> </tr> <tr> <td>2</td> <td>auto</td> </tr> <tr> <td>3</td> <td>aperturepriority</td> </tr> <tr> <td>4</td> <td>shutterpriority</td> </tr> <tr> <td>5</td> <td>creativeprogram</td> </tr> <tr> <td>6</td> <td>actionprogram</td> </tr> <tr> <td>7</td> <td>portraitmode</td> </tr> <tr> <td>8</td> <td>landscapemode</td> </tr> </tbody> </table> <p><i>Example:</i></p> <p><b>SetAttr META exposureprogram:aperturepriority</b></p>	Value	Keyword	0	notdefined	1	manual	2	auto	3	aperturepriority	4	shutterpriority	5	creativeprogram	6	actionprogram	7	portraitmode	8	landscapemode
Value	Keyword																					
0	notdefined																					
1	manual																					
2	auto																					
3	aperturepriority																					
4	shutterpriority																					
5	creativeprogram																					
6	actionprogram																					
7	portraitmode																					
8	landscapemode																					
Exposure time	exposuretime	Specified in seconds or fractions of a second - accepts either a decimal or fractional value.																				
F-number	fnumber	Accepts either a decimal or fractional value.																				
Flash	flash	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> </table>	Value	Keyword																		
Value	Keyword																					

		<table border="1"> <tbody> <tr><td>0x00</td><td>noflash</td></tr> <tr><td>0x01</td><td>fired</td></tr> <tr><td>0x05</td><td>fired,strobereturnlightnotdetected</td></tr> <tr><td>0x07</td><td>fired,strobereturnlightdetected</td></tr> <tr><td>0x08</td><td>yes,didnotfire</td></tr> <tr><td>0x09</td><td>yes,compulsory</td></tr> <tr><td>0x0d</td><td>yes,compulsory,returnlightnotdetected</td></tr> <tr><td>0x0f</td><td>yes,compulsory,returnlightdetected</td></tr> <tr><td>0x10</td><td>no,compulsory</td></tr> <tr><td>0x14</td><td>no,didnotfire,returnnotdetected</td></tr> <tr><td>0x18</td><td>no,auto</td></tr> <tr><td>0x19</td><td>yes,auto</td></tr> <tr><td>0x1d</td><td>yes,auto,returnlightnotdetected</td></tr> <tr><td>0x1f</td><td>yes,auto,returnlightdetected</td></tr> <tr><td>0x20</td><td>noflashfunction</td></tr> <tr><td>0x41</td><td>yes,red-eyereduction</td></tr> <tr><td>0x45</td><td>yes,red-eyereduction,returnlightnotdetected</td></tr> <tr><td>0x47</td><td>yes,red-eyereduction,returnlightdetected</td></tr> <tr><td>0x49</td><td>yes,compulsory,red-eyereduction</td></tr> <tr><td>0x4d</td><td>yes,compulsory,red-eyereduction,returnlightnotdetected</td></tr> <tr><td>0x4f</td><td>yes,compulsory,red-eyereduction,returnlightdetected</td></tr> <tr><td>0x50</td><td>no,red-eyereduction</td></tr> <tr><td>0x58</td><td>no,auto,red-eyereduction</td></tr> <tr><td>0x59</td><td>yes,auto,red-eyereduction</td></tr> <tr><td>0x5d</td><td>yes,auto,red-eyereduction,returnlightnotdetected</td></tr> <tr><td>0x5f</td><td>yes,auto,red-eyereduction,returnlightdetected</td></tr> </tbody> </table> <p><i>Example:</i></p> <p><b>SetAttr META flash:0x50</b>  <b>SetAttr META flash:yes,auto,red-eyereduction</b></p>	0x00	noflash	0x01	fired	0x05	fired,strobereturnlightnotdetected	0x07	fired,strobereturnlightdetected	0x08	yes,didnotfire	0x09	yes,compulsory	0x0d	yes,compulsory,returnlightnotdetected	0x0f	yes,compulsory,returnlightdetected	0x10	no,compulsory	0x14	no,didnotfire,returnnotdetected	0x18	no,auto	0x19	yes,auto	0x1d	yes,auto,returnlightnotdetected	0x1f	yes,auto,returnlightdetected	0x20	noflashfunction	0x41	yes,red-eyereduction	0x45	yes,red-eyereduction,returnlightnotdetected	0x47	yes,red-eyereduction,returnlightdetected	0x49	yes,compulsory,red-eyereduction	0x4d	yes,compulsory,red-eyereduction,returnlightnotdetected	0x4f	yes,compulsory,red-eyereduction,returnlightdetected	0x50	no,red-eyereduction	0x58	no,auto,red-eyereduction	0x59	yes,auto,red-eyereduction	0x5d	yes,auto,red-eyereduction,returnlightnotdetected	0x5f	yes,auto,red-eyereduction,returnlightdetected
0x00	noflash																																																					
0x01	fired																																																					
0x05	fired,strobereturnlightnotdetected																																																					
0x07	fired,strobereturnlightdetected																																																					
0x08	yes,didnotfire																																																					
0x09	yes,compulsory																																																					
0x0d	yes,compulsory,returnlightnotdetected																																																					
0x0f	yes,compulsory,returnlightdetected																																																					
0x10	no,compulsory																																																					
0x14	no,didnotfire,returnnotdetected																																																					
0x18	no,auto																																																					
0x19	yes,auto																																																					
0x1d	yes,auto,returnlightnotdetected																																																					
0x1f	yes,auto,returnlightdetected																																																					
0x20	noflashfunction																																																					
0x41	yes,red-eyereduction																																																					
0x45	yes,red-eyereduction,returnlightnotdetected																																																					
0x47	yes,red-eyereduction,returnlightdetected																																																					
0x49	yes,compulsory,red-eyereduction																																																					
0x4d	yes,compulsory,red-eyereduction,returnlightnotdetected																																																					
0x4f	yes,compulsory,red-eyereduction,returnlightdetected																																																					
0x50	no,red-eyereduction																																																					
0x58	no,auto,red-eyereduction																																																					
0x59	yes,auto,red-eyereduction																																																					
0x5d	yes,auto,red-eyereduction,returnlightnotdetected																																																					
0x5f	yes,auto,red-eyereduction,returnlightdetected																																																					
Focal length	focallength	Specified in millimetres - accepts either a decimal or fractional value.																																																				
Focal length (35mm)	35mmfocallength	Specified in millimetres - accepts either a decimal or fractional value.																																																				
GPS Altitude	gpsaltitude	Specified as metres relative to sea level - accepts either a decimal or fractional value. Specify a negative number to reference below sea level.																																																				

		<p><i>Example:</i></p> <p><b>SetAttr META gpsaltitude:-423</b></p>							
GPS Latitude	gpslatitude	<p>Accepts coordinates in any of the following formats:</p> <table border="1"> <tr><td>45:26:46N</td></tr> <tr><td>45:26:46.302N</td></tr> <tr><td>45N26 21</td></tr> <tr><td>45.446195N</td></tr> <tr><td>45.446195</td></tr> <tr><td>N45° 26.7717'</td></tr> <tr><td>45°26'21"N</td></tr> </table> <p><i>Example:</i></p> <p><b>SetAttr META gpslatitude:45:26:46.302N</b></p>	45:26:46N	45:26:46.302N	45N26 21	45.446195N	45.446195	N45° 26.7717'	45°26'21"N
45:26:46N									
45:26:46.302N									
45N26 21									
45.446195N									
45.446195									
N45° 26.7717'									
45°26'21"N									
GPS Longitude	gpslongitude	<p>Accepts coordinates in any of the following formats:</p> <table border="1"> <tr><td>65:56:55W</td></tr> <tr><td>65:56:55.903W</td></tr> <tr><td>65W58 36</td></tr> <tr><td>65.948862W</td></tr> <tr><td>-65.948862</td></tr> <tr><td>W65° 56.93172'</td></tr> <tr><td>65°58'36"W</td></tr> </table> <p><i>Example:</i></p> <p><b>SetAttr META gpslongitude:65W58.36</b></p>	65:56:55W	65:56:55.903W	65W58 36	65.948862W	-65.948862	W65° 56.93172'	65°58'36"W
65:56:55W									
65:56:55.903W									
65W58 36									
65.948862W									
-65.948862									
W65° 56.93172'									
65°58'36"W									
Image description	imagedesc	User-defined string.							
ISO speed	isospeed	Accepts a numeric value.							

Metering mode	meteringmode	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>unknown</td> </tr> <tr> <td>1</td> <td>average</td> </tr> <tr> <td>2</td> <td>centerweightedaverage</td> </tr> <tr> <td>3</td> <td>spot</td> </tr> <tr> <td>4</td> <td>multi-spot</td> </tr> <tr> <td>5</td> <td>multi-segment</td> </tr> <tr> <td>6</td> <td>partial</td> </tr> <tr> <td>255</td> <td>other</td> </tr> </tbody> </table>	Value	Keyword	0	unknown	1	average	2	centerweightedaverage	3	spot	4	multi-spot	5	multi-segment	6	partial	255	other
Value	Keyword																			
0	unknown																			
1	average																			
2	centerweightedaverage																			
3	spot																			
4	multi-spot																			
5	multi-segment																			
6	partial																			
255	other																			
Rotation	orientation	<p>Accepts the values <b>0, 90, 180</b> and <b>270</b>.</p> <p>Also accepts a delta value to adjust the existing orientation value.</p> <p><i>Example:</i></p> <p><b>SetAttr META orientation:90</b> - set orientation to 90  <b>SetAttr META orientation:-90</b> - rotate orientation 90 degrees counter-clockwise</p>																		
Saturation	saturation	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>normal</td> </tr> <tr> <td>1</td> <td>low</td> </tr> <tr> <td>2</td> <td>high</td> </tr> </tbody> </table>	Value	Keyword	0	normal	1	low	2	high										
Value	Keyword																			
0	normal																			
1	low																			
2	high																			
Scene capture type	scenecapturetype	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>standard</td> </tr> <tr> <td>1</td> <td>landscape</td> </tr> </tbody> </table>	Value	Keyword	0	standard	1	landscape												
Value	Keyword																			
0	standard																			
1	landscape																			

		<table border="1"> <tbody> <tr> <td>2</td> <td>portrait</td> </tr> <tr> <td>3</td> <td>nightscene</td> </tr> </tbody> </table>	2	portrait	3	nightscene				
2	portrait									
3	nightscene									
Sharpness	sharpness	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>normal</td> </tr> <tr> <td>1</td> <td>soft</td> </tr> <tr> <td>2</td> <td>hard</td> </tr> </tbody> </table>	Value	Keyword	0	normal	1	soft	2	hard
Value	Keyword									
0	normal									
1	soft									
2	hard									
Shutter speed	shutterspeed	Specified in seconds or fractions of a second - accepts either a decimal or fractional value.								
Special instructions	instructions	User-defined string.								
Subject distance	subjectdistance	<p>Specified in either metres or millimetres - accepts either a decimal or fractional value.</p> <p><i>Example:</i></p> <p><b>SetAttr META subjectdistance:10.3mm</b> - 10.3 millimetres</p> <p><b>SetAttr META subjectdistance:50m</b> - 50 metres</p>								
White balance	whitebalance	<p>Accepts the following values (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>auto</td> </tr> <tr> <td>1</td> <td>manual</td> </tr> </tbody> </table>	Value	Keyword	0	auto	1	manual		
Value	Keyword									
0	auto									
1	manual									
<b>Music Properties</b>										
Album	album	User-defined string.								

Album artist	albumartist	User-defined string.																																												
Cover art	coverart	<p>This can set, add and remove cover art.</p> <p>To set cover art (removes any existing art), the value must be specified as:</p> <p style="text-align: center;"><i>type:&lt;filename&gt;</i></p> <p>To add cover art (adds to existing art):</p> <p style="text-align: center;"><i>+type:&lt;filename&gt;</i></p> <p>To remove cover art (removes any art of a certain type):</p> <p style="text-align: center;"><i>-type</i></p> <p>The cover art type can be specified as follows (either the <i>value</i> or the <i>keyword</i> can be used):</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th>Value</th> <th>Keyword</th> </tr> </thead> <tbody> <tr><td>3</td><td>front</td></tr> <tr><td>4</td><td>back</td></tr> <tr><td>0</td><td>other</td></tr> <tr><td>1</td><td>icon</td></tr> <tr><td>2</td><td>otherfileicon</td></tr> <tr><td>5</td><td>leaflet</td></tr> <tr><td>6</td><td>media</td></tr> <tr><td>7</td><td>leadartist</td></tr> <tr><td>8</td><td>artist</td></tr> <tr><td>9</td><td>conductor</td></tr> <tr><td>10</td><td>band</td></tr> <tr><td>11</td><td>composer</td></tr> <tr><td>12</td><td>lyricist</td></tr> <tr><td>13</td><td>location</td></tr> <tr><td>14</td><td>recording</td></tr> <tr><td>15</td><td>performance</td></tr> <tr><td>16</td><td>vidcap</td></tr> <tr><td>17</td><td>colorfulfish</td></tr> <tr><td>18</td><td>illustration</td></tr> <tr><td>19</td><td>bandlogo</td></tr> <tr><td>20</td><td>publisherlogo</td></tr> </tbody> </table>	Value	Keyword	3	front	4	back	0	other	1	icon	2	otherfileicon	5	leaflet	6	media	7	leadartist	8	artist	9	conductor	10	band	11	composer	12	lyricist	13	location	14	recording	15	performance	16	vidcap	17	colorfulfish	18	illustration	19	bandlogo	20	publisherlogo
Value	Keyword																																													
3	front																																													
4	back																																													
0	other																																													
1	icon																																													
2	otherfileicon																																													
5	leaflet																																													
6	media																																													
7	leadartist																																													
8	artist																																													
9	conductor																																													
10	band																																													
11	composer																																													
12	lyricist																																													
13	location																																													
14	recording																																													
15	performance																																													
16	vidcap																																													
17	colorfulfish																																													
18	illustration																																													
19	bandlogo																																													
20	publisherlogo																																													

		<p><i>Example:</i></p> <p><b>SetAttr META "coverart:3:/mypictures/Pink Floyd.jpg"</b>  <b>SetAttr META coverart:-back</b> - removes all back cover images  <b>SetAttr META coverart</b> - removes all images</p>
Disc number	discnumber	<p>Accepts either a single digit (the number of the disc), or two digits separated by a forward slash (the number of the disc and the total number of discs in the set).</p> <p><i>Example:</i></p> <p><b>SetAttr META discnumber:3/8</b></p>
Initial key	initialkey	User-defined string.
Original artists	origartist	User-defined string.
Release date	releasedate	<p>Accepts <b>YYYYMMDD</b> or <b>YYYY-MM-DD</b> format for the date.</p> <p><i>Example:</i></p> <p><b>SetAttr META releasedate:1973-03-01</b></p>
Track number	track	<p>Accepts either a single digit (the number of the track), or two digits separated by a forward slash (the number of the track and the total number of tracks on the disk).</p> <p><i>Example:</i></p> <p><b>SetAttr META tracknumber:5/14</b></p>
<b>Video Properties</b>		
Directors	directors	User-defined string. Multiple directors can be separated with semi-colons.

Producers	producers	User-defined string. Multiple producers can be separated with semi-colons.
Writers	writers	User-defined string. Multiple writers can be separated with semi-colons.
<b>Music and Video Properties</b>		
Author URL	authorurl	User-defined string.
Artists	artist	User-defined string. Multiple artists can be separated with semi-colons.
BPM	bpm	Accepts a numeric value.
Composers	composers	User-defined string. Multiple composers can be separated with semi-colons.
Conductors	conductor	User-defined string. Multiple conductors can be separated with semi-colons.
Content group	contentgroup	User-defined string.
Encoded by	encoder	User-defined string.
Encoding software	encodingsoftware	User-defined string.
Genre	genre	User-defined string.
Mood	mood	User-defined string.
Publisher	publisher	User-defined string.

Subtitle	subtitle	User-defined string.
Year	year	Accepts a four digit year (YYYY).
<b>Document Properties</b>		
Authors	author	User-defined string. Multiple authors can be separated with semi-colons.
Category	category	User-defined string.
Company	company	User-defined string.
Content Status	contentstatus	User-defined string.
Content Type	contenttype	User-defined string.
Copyright	copyright	User-defined string.
Creator	creator	User-defined string.
Language	language	User-defined string.
Last saved by	lastsavedby	User-defined string.
Manager	manager	User-defined string.
Producer	producer	User-defined string.
Subject	subject	User-defined string.
Title	title	User-defined string.



# Icon Sets

The Icon Set system allows you to create a custom icon set that can “drop in” and replace, or augment, the internal set of toolbar icons. A list of installed icon sets can be viewed in Preferences on the [Toolbars / Icons](#) page.

An Icon Set is distributed in a **.dis** file, which is a normal ZIP file with a **.dis** extension. Inside the **.dis** file is an XML file that defines the icon set, and one or more image files that contain the actual image data for the icons. Any image format supported by Directory Opus can be used.

## Icon Set XML Definition File

The XML file defines the icons that make up the icon set. Below is an example of an Icon Set .xml file:

```
<?xml version="1.0" encoding="utf-8"?>
<iconset name="sample">
  <display_name>SampleI con Set</display_name>
  <copyright>(c)2007 Fred Bloggs Productions</copyright>
  <artist>Fred Bloggs</artist>
  <set size="small" width="20" height="20"
filename="MyIconsSmall.png">
    <icon name="copy" row="1" col="1" />
    <icon name="move" row="1" col="2" />
  </set>
  <set size="large" width="32" height="32"
filename="MyIconsLarge.png">
    <icon name="copy" row="1" col="1" />
    <icon name="move" row="1" col="2" />
  </set>
</iconset>
```

The root node of the definition file is called **iconset**. The **name** attribute given here specifies an internal name that is used for this set. This is never shown to the user, but is used internally for referencing icons the user has configured in their toolbar. It is important that you pick a unique name for your icon set – if two icon sets with the same internal name are installed, their contents will be merged together as if they were one big set.

Several values below the root node let you specify icon set metadata – **display\_name** is the name of the set that is actually displayed to the user, and **copyright** and **artist** are strings that are shown in the list of Icon Sets in Preferences.

Following this, the **set** nodes are used to specify one or more icons in the set. Each icon set can contain one or two distinct groups of icons – a **small** group and a **large** group. The **size** attribute to the **set** node specifies which icon size is being defined, and the **width** and **height** attributes define the actual pixel sizes of the icons. The **filename** attribute specifies the image file that this group of icons is drawn from.

Below each **set** node are one or more **icon** nodes which define the individual icons that make up the set. Each icon must have a name specified with the **name** attribute, and its row and column within the image file given with the **row** and **col** attributes.

## Icon Sizes

As already mentioned, each icon set can contain one or two distinct groups of icons. For simplicity, these are referred to as the **small** size and the **large** size. However, you are not limited to only two sizes of icon– only two groups of icons. It is possible to have different icons of multiple sizes within the one group. For example,

```
<set size="small" width="20" height="20" filename="MyIconsSmall.png">
  <icon name="copy" row="1" col="1" />
</set>

<set size="small" width="24" height="24" filename="MyIconsMedium.png">
  <icon name="move" row="1" col="1" />
</set>
```

Note that both **set** nodes above have the **size** attribute set to “small”. This example would add two icons to the **small** group – one, called “copy” that is 20x20 in size, and one called “move” that is 24x24 in size. Even though they are different physical sizes, they would both appear together in the small group for this icon set. The icons you provide in the **small** and **large** groups do not have to correspond – you do not **need** to provide both small and large forms of every icon (although it is obviously better for the user if you do.)

## Icon Names

Every icon in an icon set must have a name. The name is used for two things – it lets the user search for icons in the *Select Icon* dialog, and it is stored in the toolbar configuration and used internally by Opus to identify which icon the user has configured for a given button.

Opus supports multiple active icon sets at any given time. When a toolbar button is configured to use an icon, Opus searches the list of installed icon sets for an icon of that name. The first icon found with the specified name is the one that is shown. For example, a toolbar button may be set to use an icon called “copy”. Opus will search its list of icon sets, looking for an icon in the desired size (small or large) called “copy”. The first matching icon that is found is the one that is used.

Opus uses many hard-coded icon names internally to refer to the default functions. For example, “copy”, “move”, “delete”, “rename”, “print”, “play” and “undo” are some of the more than 200 hard-coded icon names. If you want your icon set to be able to replace the default toolbar buttons - *without the user having to specifically configure their buttons to use your icons* – you need to make sure you use the same names as Opus.

However, you are free to use any name you like for your icons. For example, you may wish to create an icon set that provides icons suitable for a source code control program. There are no internal Opus icons that are relevant to this, so your intention would not be to replace the internal icons, but to augment them. Therefore, you would pick names for your icons that don’t match the internal ones.

For a full list of the hard-coded icon names that Opus uses, you can generate a template .xml file using the *Save Icon Set Template* menu command on the *Toolbar Icons* page in Preferences.

## Icon Display Names

By default, the names given to each icon are shown to the user when they hover over each image in the *Select Icon* dialog. If desired, you can specify a “display name” for each icon, which is displayed alongside the icon’s real name.

The internal icons have automatic display names – for example, hovering over the internal Move As icon displays “Move As (moveas)” as the tooltip. “moveas” is the icon’s internal name, but “Move As” is the display name.

If your icons have the same name as the internal icons, they will be given localized display names automatically. Otherwise, you can provide display names for your icons as follows:

```
<set size="small" width="20" height="20" filename="UtilIcons.png">  
  <icon name="burncd" display_name="Burn To CD" row="2" col="8"
```

```
category="Tools" />
</set>
```

## Icon Categories

Directory Opus groups its internal icon set into categories to make it easier for the user to find a particular icon. The *Select Icon* dialog lets the user filter by category as well as by name. The internal icons have all been pre-assigned categories – if your icons use the same names as the internal icons then they will automatically adopt the pre-assigned categories. For example, the “copy” icon is pre-assigned to the “file” category, so if your icon set provides an icon called “copy”, the user will see it in the *Select Icon* dialog when they set the category filter to “File.”

If your icon has a name that is not in the internal icon set, it will by default be listed as Uncategorized. It is possible, but not obligatory, to specify your own categories for your icons. The default categories used by Opus are “config”, “edit”, “file”, “ftp”, “misc”, “go”, “tools”, “view” and “window.” For each icon, you can optionally specify one of the default categories, or define your own custom category.

You can assign categories to your icons on an icon-by-icon basis, using the **icon** node, or to a number of icons at once, using the **set** node. For example,

```
<set size="small" width="20" height="20" filename="UtilIcons.png">
  <icon name="burncd" row="2" col="8" category="Tools"/>
  <icon name="ftpimage" row="3" col="1" category="Image"/>
</set>
```

This would define an icon called “burncd” in the “Tools” default category, and another called “ftpimage” in a custom category called “Image”.

Alternatively, both these icons could be placed in the “File” category as follows:

```
<set size="small" width="20" height="20" filename="UtilIcons.png"
category="File">
  <icon name="burncd" row="2" col="8" />
  <icon name="ftpimage" row="3" col="1" />
</set>
```

Again, it is not essential to categorize your icons. If you use the default icon names they will be categorized automatically (although you can override the automatic categorization if desired), and if not they will still work, but be listed as “Uncategorized” in the *Select Icon* dialog.

## DPI aware Icon Sets

Directory Opus handles different system DPI settings transparently, and will scale icons as needed. However, you may want to provide multiple base icon images in your icon sets to handle different DPI settings. Scaling often results in blurry and unsatisfactory images - providing the same icons in multiple resolutions can result in a better aesthetic result when the user is running on a high-DPI system.

For each `<set>` entry in the XML file you can provide an optional `<dpi>` key which specifies alternate images for different DPI settings, and can also optionally set limits on when the image can be used. For example, if an image doesn't look good scaled up you can set a maximum scale factor it can be used for.

The `<dpi>` key has the following attributes:

<code>set blah...&gt;</code>	
<code>&lt;dpi base="x"&gt;</code>	specifies the base scale factor of the image (e.g. "100")
<code>&lt;scale factor="x"</code>	specifies the scale factor of this alternate image (e.g. "200")
<code>min="x"</code>	minimum scale factor this should be used for (optional, e.g. "200")
<code>max="x"</code>	maximum scale factor this should be used for (optional, e.g. "400")
<code>no_scale_min="x"</code>	minimum scale factor this image should be used without any scaling (optional)
<code>no_scale_max="x"</code>	maximum of above (optional). Use -1 for infinite.
<code>filename="x"</code>	filename of the alternate image
<code>width="x"</code>	width of icons in the alternate image (optional, will be calculated if not provided)

height="x" height of icons in the alternate image (optional, will be calculated if not provided)

```
    />  
</dpi>
```

As many **<scale>** entries can be provided as needed. If the **<dpi>** key is missing altogether a base scaling factor of 100% is assumed. If needed, you can include the base image in the DPI list as well (for example, if you want to specify **no\_scale\_min** and **no\_scale\_max** values for it).

The **no\_scale\_xxx** range can be used to snap to various sizes while avoiding blurring. This is usually only needed at small sizes (since once icons get larger, the scaling works better). Ranges are inclusive of their **min** and **max** values, should not overlap with each other, and the value specified for **factor** should fall within the range.

A real example from the default icon set:

```
<set filename="DEFAULT_ICONS_22.PNG" size="small" width="22" height="22">  
    <dpi base="100">  
        <scale factor="100" filename="DEFAULT_ICONS_22.PNG" width="22"  
height="22" no_scale_min="0" no_scale_max="125" />  
        <scale factor="150" filename="DEFAULT_ICONS_32.PNG" width="32"  
height="32" no_scale_min="126" no_scale_max="175" />  
        <scale factor="200" filename="DEFAULT_ICONS_48.PNG" width="48"  
height="48" />  
        <scale factor="300" filename="DEFAULT_ICONS_64.PNG" width="64"  
height="64" />  
    </dpi>  
    <icon col="1" name="empty" row="1" />  
    <icon col="2" name="spacer" row="1" />  
    ...  
</set>  
  
<set filename="DEFAULT_ICONS_32.PNG" size="large" width="32" height="32">  
    <dpi base="100">  
        <scale factor="100" filename="DEFAULT_ICONS_32.PNG" width="32"  
height="32" no_scale_min="0" no_scale_max="125" />  
        <scale factor="150" filename="DEFAULT_ICONS_48.PNG" width="48"  
height="48" no_scale_min="126" no_scale_max="175" />  
        <scale factor="200" filename="DEFAULT_ICONS_64.PNG" width="64"
```

```

height="64" />
  </dpi>
  <icon col="1" name="empty" row="1" />
  <icon col="2" name="spacer" row="1" />
  ...
</set>

```

## Localization

The Icon Set format optionally supports localization of both the icon display names and custom icon category names. To localize your custom icon names, you need to specify one or more **language\_map** nodes in the definition XML file. For example,

```

<iconset name="sample">
  <language_map language="deutsch">
    <icon name="ftpimage" display_name="Hochladen Sie
dieses Bild" />
    <category name="Image" display_name="Bild" />
  </language_map>
  <language_map language="français">
    <icon name="ftpimage" display_name="Téléchargez cette
image" />
    <category name="Image" display_name="Image" />
  </language_map>
  <set size="small" width="20" height="20"
filename="UtilIcons.png">
    <icon name="ftpimage" display_name="Upload Image"
row="3" col="1" category="Image" />
  </set>
</iconset>

```

This defines a set containing one icon, with the internal name of “ftpimage” and the default display name of “Upload Image”. This icon is placed in a custom category called “Image.”

In English, or any language other than French or German, Opus would display “Upload Image (ftpimage)” as the icon name when hovering over the image in the *Select Icon* dialog. However, thanks to the **language\_map** nodes, German users would see “Hochladen Sie dieses Bild (ftpimage)” and French users “Téléchargez cette image (ftpimage).” Additionally, the custom “Image” category would appear as “Bild” in German.

The name of the language specified in each **language\_map** node must be the same as the name (without extension) of the Language DLL used by Opus, as found in the Languages sub-folder in the program directory.

## Icon Images

As mentioned above, the actual image data for your icons can be supplied in any image format that Directory Opus understands. The most common of these would be JPEG, PNG, GIF and BMP. Any number of separate image files can be used, and each image file can provide one or more icons.

Say you have an image file called “MyIcons.png” that is 100x20 pixels in size, and that contains the imagery for 5 different icons. Each individual icon is therefore 20x20 pixels in size. In the XML definition file, you would have a **set** entry for that image file, and then up to five **icon** entries for each of the icons. The location of the image data for each icon is specified using the **row** and **col** attributes. For example,

```
<set size="small" width="20" height="20" filename="MyIcons.png">
  <icon name="icon1" row="1" col="1"/>
  <icon name="icon2" row="1" col="2" />
  <icon name="icon3" row="1" col="4" />
  <icon name="icon4" row="1" col="5" />
</set>
```

This would add four icons from the image file – the first two and the last two, skipping over the third.

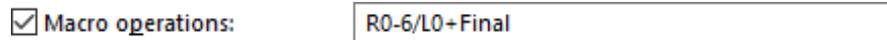
Note that an image file can also contain just a single image. For example, say you have a collection of single image icons that you wish to bundle together as an icon set. You would simply specify multiple **set** nodes, one for each image file, and each **set** node would contain a single **icon** node referring to row 1, column 1.

Images supplied in PNG or GIF format automatically support transparency – GIF supports a single transparent color, and PNG supports full 32 bit alpha. If you wish to provide transparent icons in a different format, you can specify **transparent="yes"** as an attribute of the **set** node. Opus will then make transparent any pixels in the image that have an RGB value of 255,0,255 (#FF00FF).

Filenames which start with “:INTERNAL:” reference internal image resources stored inside of the program. You would not normally use internal images yourself but may see them used in special icon sets, usually created by GP Software.

# Rename Macro Language

Although you don't need to understand the rename macro language (using the [macro builder](#) is much easier!), it's described here for reference.



The macro language describes one or more operations, each with a position relative to either the beginning or end of each filename. Above is a simple macro that removes 6 characters from the end of each name, and inserts the word “Final” at the start of each name.

The components of each macro expression are:

1. **Anchor:** The first character of operation macro is either an **L** or an **R**, to indicate whether the position is relative (anchored) to the **left** or **right** of the filename.
2. **Offset:** Next is the offset in characters from the anchor position.
3. **Operation:** (Optional) One of the following operation characters can appear next.
  - – remove text
  - **C** copy text to clipboard
  - **X** cut text to clipboard
  - **V** paste clipboard contents
4. **Length:** For – (remove), **C** (copy) and **X** (cut), the length in characters to cut, copy or remove is next.
5. **Insert:** (Optional) To insert characters at the current position in the name, use a + followed by the text to insert.
6. **Separator:** If the macro contains additional operations they should be separated by a / character.

Using this information we can now decode the macros shown in the above example:

- **R0-6:** Right-anchor, 0 characters from the end of the filename, **remove** 6 characters.
- **L0+Final:** Left-anchor, 0 characters from the beginning of the filename, **insert** the word “Final”.

And here's an example of this macro in action:

<input checked="" type="checkbox"/> Current Name	New Name	New Ext.
<input checked="" type="checkbox"/> 2007-15-15_draft.txt	Final 2007-15-15	txt
<input checked="" type="checkbox"/> 2012-02-10_draft.txt	Final 2012-02-10	txt
<input checked="" type="checkbox"/> 2015-09-22_draft.txt	Final 2015-09-22	txt

As you can see, the suffix **\_draft** was removed from each filename (by the **R0-6** expression) and the prefix **Final** was inserted (by the **L0+Final** expression).

Let's have a look at a more complicated macro example.

Macro operations: R0-6/R0X2/R0-1/R7V/R7+./R0X2/R0-1/R4V/R4+./L0+Final

Using the guide on the previous page, we can break this down and decode each component as follows:

- **R0-6**: Right-anchor, 0 characters from the end of the filename, **remove** 6 characters.
- **R0X2**: Right-anchor, 0 characters from the end, **cut** 2 characters to the clipboard.
- **R0-1**: Right-anchor, 0 characters from the end, **remove** 1 character.
- **R7V**: Right-anchor, 7 characters from the end, **paste** the clipboard contents.
- **R7+.**: Right-anchor, 7 characters from the end, **insert** a full-stop.
- **R0X2**: Right-anchor, 0 characters from the end, **cut** 2 characters to the clipboard.
- **R0-1**: Right-anchor, 0 characters from the end, **remove** 1 character.
- **R4V**: Right-anchor, 4 characters from the end, **paste** the clipboard contents.
- **R4+.**: Right-anchor, 4 characters from the end, **insert** a full-stop.
- **L0+Final**: Left-anchor, 0 characters from the beginning of the filename, **insert** the word "Final".

As you can see, this macro makes use of the clipboard functionality of the macro language. Clipboard operations are performed on a per-filename basis – when it says “**cut** 2 characters to the clipboard”, that means in each filename two characters – potentially different for each file – are cut to the clipboard. What this means of course is that you can move parts of filenames around using simple clipboard operations.

Here's an example of this macro in use:

<input checked="" type="checkbox"/> Current Name	New Name	New Ext.
<input checked="" type="checkbox"/> 2007-15-15_draft.txt	<input checked="" type="checkbox"/> Final 15.15.2007	txt
<input checked="" type="checkbox"/> 2012-02-10_draft.txt	<input checked="" type="checkbox"/> Final 10.02.2012	txt
<input checked="" type="checkbox"/> 2015-09-22_draft.txt	<input checked="" type="checkbox"/> Final 22.09.2015	txt

Like the first example, the suffix **\_draft** was removed from each filename (by the **R0-6** expression) and the prefix **Final** was inserted (by the **L0+Final** expression). But as well as that, the date in each filename has been rearranged. Using multiple clipboard cut / insert steps, **2007-10-15** has been turned into **15.10.2007**.



# Release History

This is the Directory Opus 12 release history, starting with the changes notes for users upgrading from Opus 11, and followed by a description of all subsequent releases.

- [What's new in Directory Opus 12](#) (full description of changes for users upgrading from Directory Opus 11)
- [Directory Opus 12.2](#) (22nd September, 2016)
- [Directory Opus 12.3](#) (5th December, 2016)
- [Directory Opus 12.4](#) (15th March, 2017)
- [Directory Opus 12.5](#) (27th April, 2017)
- [Directory Opus 12.6](#) (7th June, 2017)
- [Directory Opus 12.7](#) (23rd November, 2017)
- [Directory Opus 12.8](#) (24th April, 2018)
- [Directory Opus 12.9](#) (30th May, 2018)
- [Directory Opus 12.10](#) (3rd October, 2018)

# Welcome to Directory Opus 12!

There are lots of changes and new features in Opus 12, and the length of this document may seem daunting!

Don't be put off scanning through it though as there are lots of goodies lurking within. Here's a summary of just a few of them.

## Summary of major new features

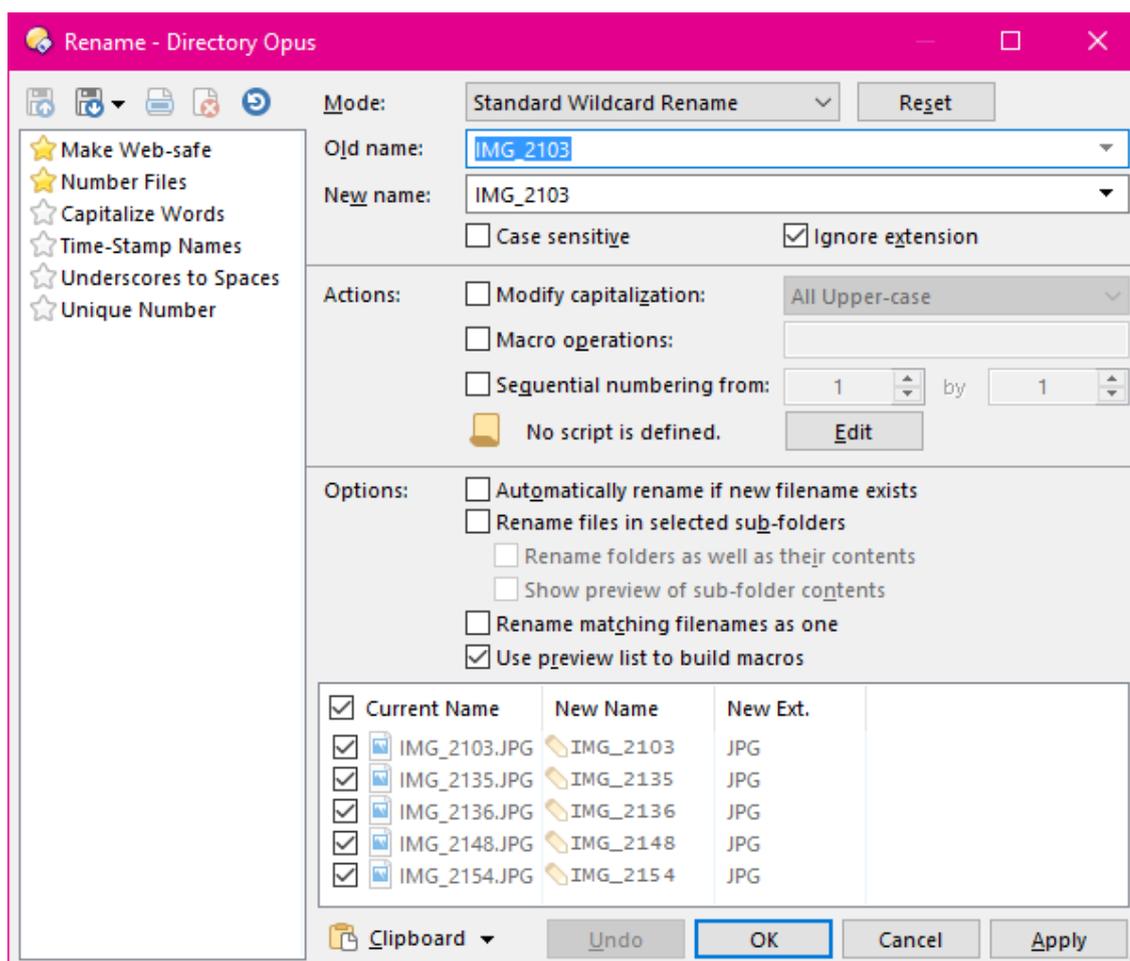
- Full support for **high-DPI** (e.g. 4K and 5K) monitors.
- Redesigned **Rename** dialog, with new features like:
  - A unique macro recorder, which lets you perform complex batch renames without regular expressions.
  - Enhanced scripting capabilities.
  - Better handling of recursive renames and filename clashing.
  - An Apply button which lets you perform multiple renames without closing the dialog.
- Improvements to the **Image Viewer** including:
  - Configurable toolbar and hotkeys, including the ability to run arbitrary commands on the current image file.
  - A new image marking system which makes it much easier to sort through a folder of photos to identify the ones you want to keep, print, share, etc.
  - A read-ahead cache for faster image loading.
  - An integrated metadata panel which lets you edit EXIF and other metadata from within the viewer.
- Enhanced **file and folder labels** including:
  - The ability to assign more than one label at once (label attributes are combined).
  - Label categories (lets you organise your labels into groups).
  - Adjustable label priority (for when more than one wildcard or filter label applies to a file)
  - A new status icon system that lets you assign one or more status icons to each files (e.g. to track which files are 'done', 'watched', 'urgent' or 'to-do').
- A **manual sorting** mode that lets you sort your files and folders exactly how you want.
- An integrated **dialog editor** that lets scripts create their own complex user interfaces.
- **File display** enhancements including:

- Vertical folder tabs (displayed down the left or right side of the file display).
- You can assign your own tab colors for specific folders.
- Optional vertical as well as horizontal gridlines.
- Relative size and age graphs displayed as the background of size and date fields (rather than requiring their own column).
- A new “show everything” mode to quickly disable all filters.
- Improvements to **Folder Options** including:
  - Configure column widths to expand and fill the usable space in the file display.
  - A column filter makes it easier to find and add the columns you want.
  - File and folder name filters can be configured using regular expressions if desired.
- File copy improvements including a **transfer speed graph** in the progress dialog.
- **Toolbar enhancements** including scrollbars and distinct labels in drop-down menus.
- **Lister layouts** can now be arranged into folders and sub-folders.
- ... and as always, much, **much** more!

## Rename

### *An overview of the new Rename dialog*

The **Rename** dialog has been redesigned both to work better and to introduce a number of new features to make batch renaming easier.



The new dialog has five distinct sections:

1. The left panel is the **presets list**, where you can load and manage your rename presets.
2. The top section controls the **mode** (standard / find and replace / regular expression) and lets you edit the **old name** and **new name** (or find and replace) strings.
3. The **actions** section controls the transformations that will be applied to each filename. See below for a description of the powerful new **macro operations** option. The script **edit** button is used to expand the dialog to reveal the rename script editor.

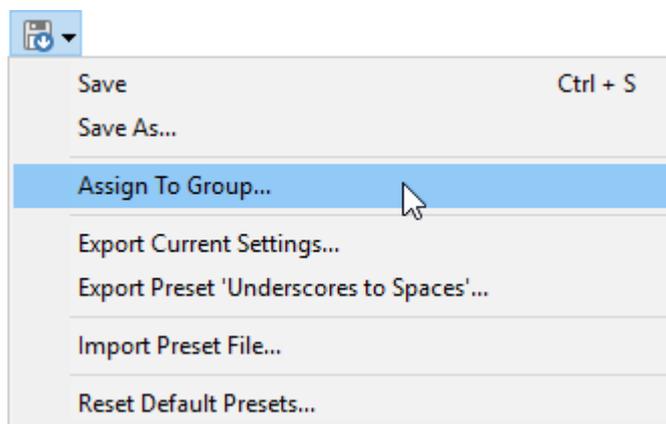
4. The **options** section contains several checkboxes that control the rename operation. New in Opus 12 is the **rename folders as well as their contents** option.
5. Finally, the **preview** section displays a preview of the transformed filenames. In Opus 12 the preview list is always displayed. Each file listed in the preview list (except for those from sub-folders) has a checkbox that you can use to remove the file from the rename operation. The preview list also doubles as the macro builder, which is described below.

At the bottom of the dialog, a new **Apply** button allows changes to be applied to the selected files while leaving the Rename dialog open. Once you use the **Apply** button the **Undo** button next to it becomes available, allowing you to undo the rename you just applied.

### ***The presets list***

The buttons above the list let you manage your saved rename presets. To load the settings from a preset, you can either click the **Load** button or double-click the preset's name in the list. The title bar of the Rename dialog will display the name of the currently loaded preset. Opus 12 will also notice if you've made changes to a loaded preset and ask if you want to save the changes before closing the Rename dialog.

The drop-down **Preset Management** menu provides various commands for managing your presets (you can also right-click a preset to display its context menu).



Each preset in the list has a star icon displayed next to its name, which you can click to designate that preset as a favorite. Favorite presets are displayed at the top of the preset list, and are also displayed separately in dynamically generated toolbar lists (the **Rename PRESET=!list** command has new parameters to control this). You can also assign each preset to a group to visually separate them in the preset list.

### ***The macro builder***

The new **macro operations** feature provides a way to add and remove text to and from filenames without needing to resort to wildcards.

As well as displaying a preview of the rename operation, the preview list at the bottom of the *Rename* dialog also doubles as a macro builder. You can edit file names directly in the preview list, and doing so generates macros which batch-rename all the files in the same way.

To use the macro builder, make sure the **Use preview list to build macros** option is turned on.

Use preview list to build macros

For example, say we have a bunch of files whose names are in the format *YYYY-MM-DD\_draft.txt*, and we want to rename them as *Final DD.MM.YYYY.txt* – removing the “\_draft” suffix, swapping the order of the date fields around and inserting “Final” at the start of each filename.

To do this with wildcards would require a complicated regular expression, but a macro makes it easy. You simply pick one of the filenames and edit it inline just like you would if you were renaming a single file – select areas of text, cut them to the clipboard, paste them in somewhere else, select another area, delete it, type some new characters, and so on.

<input checked="" type="checkbox"/> Current Name	New Name	New Ext.
<input checked="" type="checkbox"/> 2007-15-15_draft.txt	2007-15-15 	txt
<input checked="" type="checkbox"/> 2012-02-10_draft.txt	2012-02-10 	txt
<input checked="" type="checkbox"/> 2015-09-22_draft.txt	2015-09-22 	txt

The macro builder will record your actions as a macro expression and apply the same changes to all selected files.

To build a macro, simply click on a filename in the **New Name** column of the preview list (or press the **F2** key). At this point, any keys you press will be recorded as an expression in the **macro operations** field (although you don’t need to understand the macro language, it’s described in reference section for completeness).

The pencil icon (  ) indicates the current anchor position – that is, which end of the filename subsequent actions will be relative to. In the screenshot above, the anchor point is set to the right. To change the anchor position, you can double-click the pencil icon with the mouse, or position the cursor at the other end of the name and then press the cursor key (**Left** or **Right**) corresponding with that direction. You can also double-press the **Home** or **End** keys.

When you select a range of characters using the mouse or **Shift** plus the cursor keys, the equivalent range is shown as selected in all other files in the preview list – so you can check that your selection is correct before committing it.

In the above example, you can see that the original **\_draft** suffix has been removed – we did this by putting the anchor at the right end of the name, selecting left 6 characters and pressing the **Delete** key.

At the point the screenshot was taken we've selected the last two characters in the name by pushing **Shift + Left** twice, and the next step is to press **Ctrl + X** to cut those characters to the clipboard. This will generate the next macro expression **R0X2**. And so on, until the macro is complete. You can check the preview list at all stages to make sure the macro's having the desired effect on all the selected filenames – if some files are showing incorrect results you can always skip them by turning off their checkboxes.

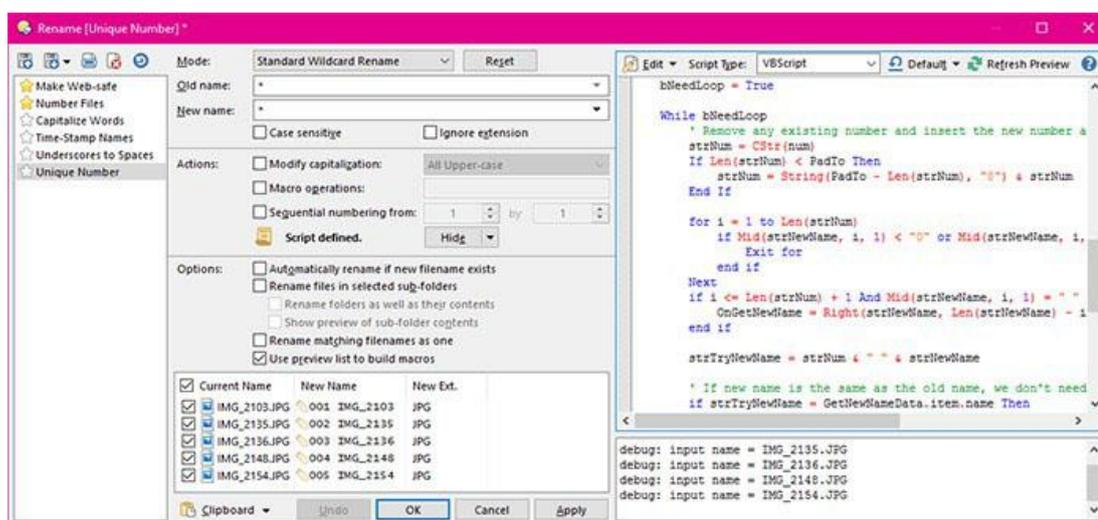
Video example: [Rename Macro Builder \(YouTube\)](#)

If **Use preview list to build macros** is turned off, the preview list lets you edit the new filenames individually. This is great for making small changes to names which the macro (or other batch operations) may not have got exactly right. Once the name for a file has been edited directly it will be displayed in red and the name is locked in until the rename operation occurs (or until you clear the custom name).

## Rename scripting

### Script editor

Click the **Edit** button in the actions section of the dialog to display the in-built script editor.



Compared to Opus 11 the obvious difference is that the script editor appears to the right of the Rename dialog rather than at the bottom. This layout change reflects the prevalence of widescreen displays where there is often more horizontal space than vertical. The width of the script editor can be adjusted using the splitter control between it and the main rename dialog.

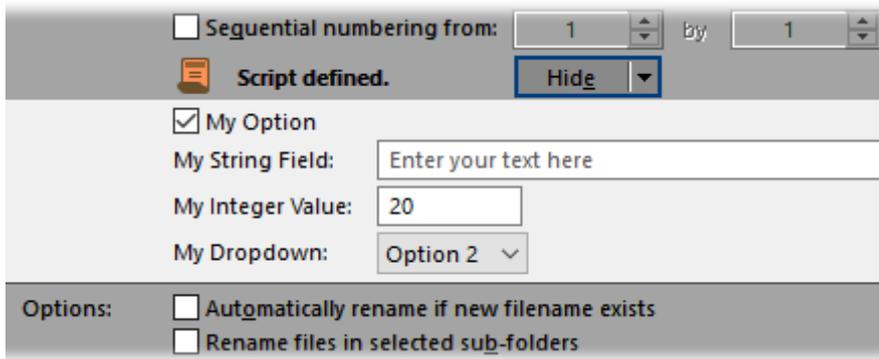
The script editor also displays an output log which will show any error messages generated by the script, as well as any text the script prints to the log using the **Opus.Output** method.

The script language can be changed using the **Script Type** field in the toolbar at the top of the editor. Each language can have a default script saved for it that becomes the template for new rename scripts. Opus ships with default templates for *JScript* and *VBScript* and you can add your own (and overwrite the ones we supply) using the **Save As Default** command in the drop-down attached to the **Default** button.

Click the **Hide** button in the actions section of the rename dialog to hide the script editor – note that if you defined a script it will remain active even though the editor is hidden. When script is defined the **Edit (Hide)** button has a drop-down command that lets you quickly clear the script.

### Custom rename fields

In Opus 12 rename scripts can add their own fields to the rename dialog itself, by implementing the **OnGetCustomFields** method. This lets you provide one or more controls that users can use to pass parameters to your script. Users can also feed parameters to your script using the new **SCRIPTARG** argument for the **Rename** command.



Custom fields can use checkboxes, string fields, number fields and drop-downs.

To add custom fields from your rename script, implement the **OnGetCustomFields** method. The above fields were added using the following code (in VBScript):

```
Function OnGetCustomFields(ByRef getFieldData)

    ' Add the custom fields
    getFieldData.fields.my_option = True
    getFieldData.fields.my_field = ""
    getFieldData.fields.my_value = 20
    getFieldData.fields.my_combo = DOpus.Create.Vector(1, "Option
1", "Option 2",
    "Option 3")

    ' Assign labels to them
    getFieldData.field_labels("my_field") = "My String Field"
    getFieldData.field_labels("my_option") = "My Option"
    getFieldData.field_labels("my_value") = "My Integer Value"
    getFieldData.field_labels("my_combo") = "My Dropdown"

    ' Set cue text for the text field
    getFieldData.field_tips("my_field") = "Enter your text here"

End Function
```

Custom fields are defined in much the same way as Script add-in defines its configuration using the **ScriptConfig** object. The **OnGetCustomFields** method is passed a **GetCustomFieldData** object. Fields are added by assigning properties of the **GetCustomFieldData.fields** object to the variable type you want the field to use (e.g. assign **True** or **False** for a Boolean, a string for a text string, etc.). The value you provide will become the default value for the field.

Each field can also have a label, and text fields can have a “cue banner” which is shown when the text field is empty (as seen above). Two **Map** objects are provided (**GetCustomFieldData.field\_labels** and **GetCustomFieldData.field\_tips**) which allow you to assign these.

The *Rename* dialog will expand automatically to accommodate your custom fields – obviously, screen space isn’t infinite, so you shouldn’t add too many fields or the dialog will grow too big for the screen!

The values that the user enters into your custom fields are provided to your **OnGetNewName** method via the **CustomFieldData** object passed as the **GetNewNameData.custom**. Each field you add in **OnGetCustomFields** will appear as a property of this object. For example, this function will print the provided values to the output log.

```
Function OnGetNewName (ByRef getNewNameData)
    DOpus.Output "Option:    " & getNewNameData.custom.my_option
    DOpus.Output "String:    " & getNewNameData.custom.my_field
    DOpus.Output "Number:    " & getNewNameData.custom.my_value
    DOpus.Output "Dropdown: " & getNewNameData.custom.my_combo

    OnGetNewName = True ' skip rename
End Function
```

When the user automates the **Rename** command to run your rename script directly (using the **PRESET** argument), they can use the new **SCRIPTARG** parameter to pass data for your custom fields through. This argument accepts multiple *name:value* pairs. For example, assume the above script was saved as the rename preset “MyRename”. The user might run the following command:

**Rename PRESET MyRename SCRIPTARG my\_option:True my\_field:moocow**

## ***Other new rename features***

### **Ignore extension**

The **ignore extension** option causes the rename function to ignore file extensions when performing wildcard or batch transformations. With this enabled you can perform regular expression or wildcard renames without having to worry about preserving the file extension in your wildcard pattern. An additional advantage is that the same wildcard patterns can now be used for both files and folders.

### **Macro operations**

The new **macro operations** function is discussed above.

### **Recursive renaming**

When performing a recursive rename using **rename files in selected sub-folders** there are two new options available:

- **Rename folders as well as their contents** lets you rename folders as well as any files within the folders. The ignore extension option is particularly useful in this situation, as it makes it easy to use one wildcard pattern that applies to both files and folders.
- **Show preview of sub-folder contents** adds the contents of any sub-folders to the rename preview list. This lets you see exactly what the rename operation will do.

The new **{parentbase}** code is similar to **{parent}** except it returns the name of the base folder rather than that of the file's parent. This is most useful with a recursive rename, where the name returned by **{parent}** would change for files inside sub-folders.

If you want to use rename to move files into new folders, adding **\$.\** at the start of the new name lets each file be moved relative to the base folder rather than its parent.

### **Clash avoidance**

Rename has new clash-avoidance technology that can prevent problems during batch renames. If the new name of a file would clash with an existing filename Opus will re-order the list of files to remove any dependencies. The **rename matching filenames as one** option has also been improved to work more reliably.

### **Removed options**

The following options have been superseded and removed.

### **Enable file information fields**

In Opus 12, Rename will notice when you use fields like **{picwidth}** in the new name field and automatically generate file information without an option having to be specifically turned on.

### **Include file extension**

This option (which only affected *find and replace* mode) has been replaced by **ignore extension**. Where you might have previously turned **include file extension** on, you should now turn **ignore extension** off (and vice versa).

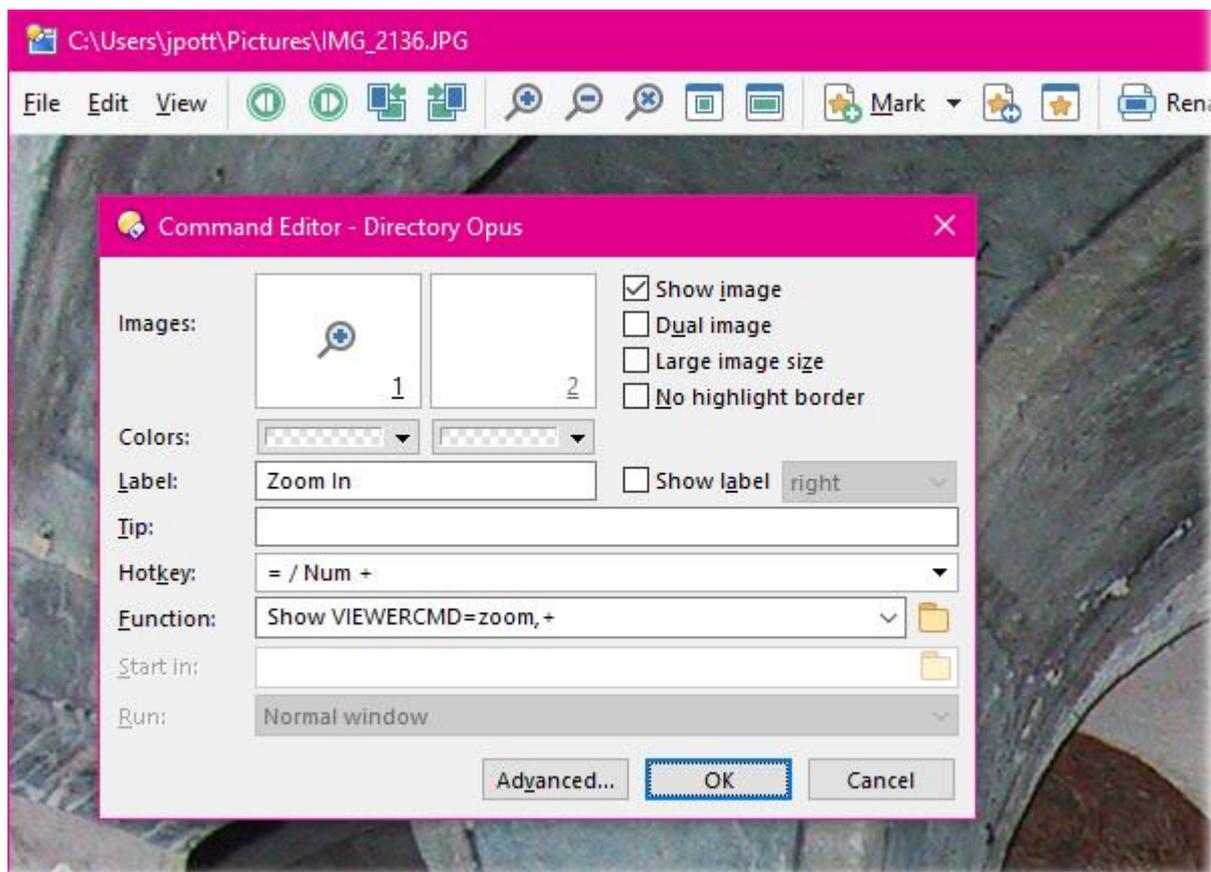
### **Script mode**

In Opus 12, if a rename script has been defined it will *always* run, whether the script editor is currently displayed or not. That is, the Rename dialog is always in “script mode”, but the display of the script editor is optional. This lets you use rename presets based on scripts without needing to have the script editor displayed all the time.

## Image Viewer

### **Configurable toolbar, menus and hotkeys**

The toolbar and context menu in the standalone image viewer are now fully configurable, just like all other toolbars and menus. Additionally, you can create hotkeys that are only active in the viewer.



To edit the toolbar in the viewer, simply select the **Customize Toolbars** command from the Edit menu, just like in a Lister. The viewer context menu can be edited from the *Context Menus* tab in the **Customize** dialog, and you can also create viewer-specific hotkeys on the *Keys* tab.

The default viewer toolbar is called *Image Viewer*, but you can select another toolbar to use from the *Viewer / Appearance* page in Preferences. You might want to do this if, for example, you want to create your own toolbar but leave the default toolbar unchanged.

To support a configurable toolbar a new command, **Show VIEWERCMD**, has been added which is used to invoke all the internal functions of the viewer. These commands only work from a viewer toolbar, menu or hotkey – they will have no effect if you try to run them in a Lister. You

can see from the above screenshot that the command corresponding to the *Zoom In* function is **Show VIEWERCMD=zoom,+**. A full list of **VIEWERCMD** commands is shown in the **Commands Reference** section of this document.

Although the **Show VIEWERCMD** command only works inside the viewer, this doesn't mean it's the only command that does – all other Opus commands and external functions also work inside the viewer. Of course, some commands (for example, **Select**) are not applicable to the viewer, but it's certainly possible to use commands like **Copy** or **Rename**, or have buttons that open the current picture in, say, Photoshop.

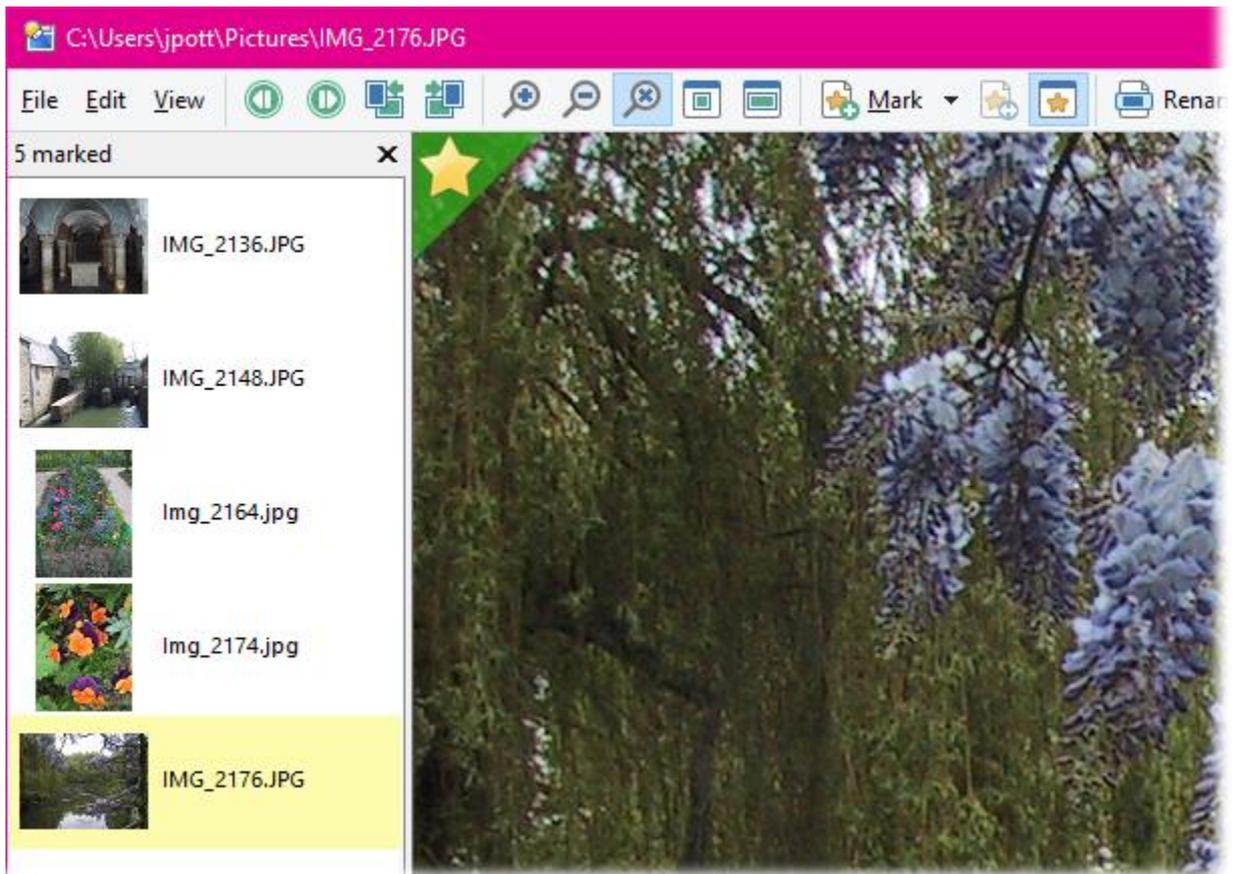
The **@if** directive can test the state of various **Show VIEWERCMD** options when used within the viewer. For example, the following function would toggle between 100% zoom and Grow To Page modes:

```
@if:Show VIEWERCMD=zoom,reset
Show VIEWERCMD=zoom,grow
@if:else
Show VIEWERCMD=zoom,reset
```

## ***Image marking***

A common task is sorting through a bunch of digital photos, working out which ones to keep, which ones to upload, which ones to print, etc. You might recognise this as the old function in Opus 11 called “Tag”, which let you select a picture in the viewer and have its checkbox in the Lister automatically turned on. While this system worked it was a bit clunky – firstly, the Lister had to remain open and in the same folder the whole time, and the checkbox state was easy to lose by, e.g., accidentally pushing F5 or changing the folder. Additionally, the state wasn't persistent – so you really had to finish going through a whole photo collection at once.

In Opus 12 this task has been made much easier by the new “image marking” mode. Images you mark are automatically added to a file collection, which solves all the problems mentioned above. Additionally, thumbnails of the images you've marked are shown in a separate panel in the viewer, and there are commands that make it easy to move around the marked images.



To mark the current image in the viewer, simply push the **M** key (or click the **Mark** button on the toolbar). With the default configuration, a file collection will automatically be created based on the name of the image's parent folder. On the left of the viewer window the marked image pane opens automatically, showing thumbnails of the images you've marked. When this opens it automatically checks the collection for any images you may have already marked in a previous session.

If the current image you're viewing is marked this is indicated with a star icon in the top-left corner, as shown in the above screenshot.

You can jump around the marked images by double-clicking their thumbnail. The marked image pane also lets you rename images by selecting their icon and pressing **F2**. The following keys relating to marking are also defined by default in the viewer:

- **M / Insert**: Mark / unmark the current image
- **Ctrl + M**: Show or hide the marked image pane
- **Ctrl + Left**: Jump to previously marked image
- **Ctrl + Right**: Jump to next marked image
- **Ctrl + Up**: Jump to first marked image

- **Ctrl + Down:** Jump to last marked image
- **Ctrl + Space:** Return from jump to last viewed image
- **Shift + M:** Exchange mark with previously marked image

These keys make it very easy to jump back and forth between images you've marked and your current "position" in the list of images. The **Exchange mark** command comes in handy when you've got multiple photos of the same scene and you're trying to decide which is the best. You might have marked the first photo because you thought it was ok, but then two or three photos later you find one that's slightly better – simply press **Shift + M** to unmark the previous one and mark the new one instead.

If you want to take a break from your session you can simply close the viewer and come back to it later – all images that you've marked will be saved in the file collection. When you've marked one or more images and you close the viewer, Opus will automatically display the file collection for you in a new tab – you can change this behaviour in Preferences.

The name of the file collection can be configured on the *Viewer / Behavior* page in Preferences. When you configure the collection name, you can use the special code **%F** to insert the name of the parent folder, and **%D** to insert the current date. You can also have Opus ask you for a collection name before each marking session. And if you like you can disable the file collection behaviour and return to the old checkbox-based method, although this then has all the drawbacks mentioned above.

## **Scripting interface**

The image viewer is now exposed via a simple scripting interface. All currently open viewers can be enumerated using the **DOpus.Viewers** property – this returns a collection of **Viewer** objects. You can also determine the last active viewer using the **DOpus.Viewers.lastactive** property.

The **Viewer** object implements the following properties:

- **left / top / right / bottom:** window position.
- **foreground:** **True** if the viewer is currently the foreground (active) window in the system.
- **lastactive:** **True** if the viewer is the last active viewer.
- **files:** returns a collection of **Item** objects representing the images in the viewer's list. Each **Item** object has a **current** property that will be **True** if this is the currently displayed image.
- **current:** returns an **Item** object representing the currently displayed image.

The **Viewer** object also implements the following method:

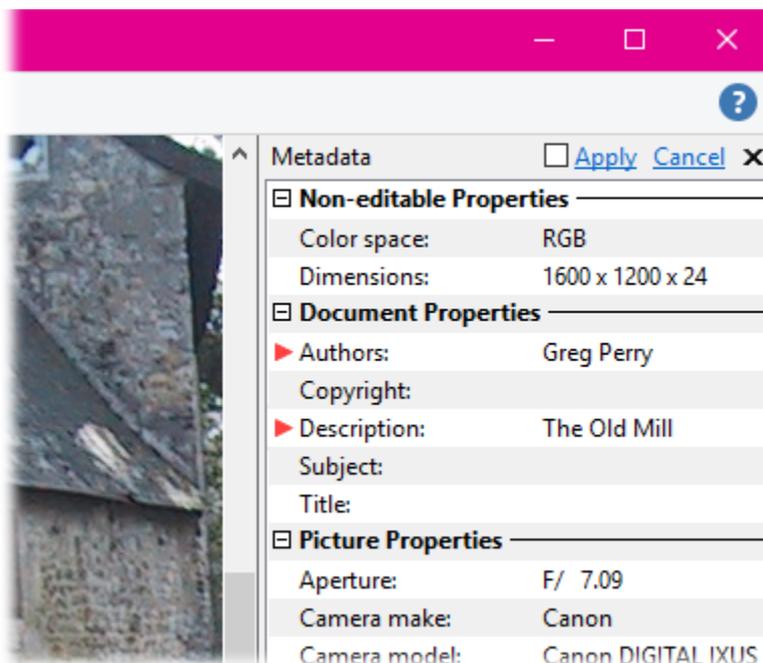
- **command(cmd):** Run a command in the context of this viewer. You can either pass a full **Command** object, or a string. If you pass a string it must be a valid keyword for

the **Show VIEWERCMD** command (e.g. to move to the next picture, you would simply pass the string "**next**" which equates to the command **Show VIEWERCMD=next**).

## Other new viewer features

### Metadata pane

The viewer can now show the Metadata pane to display and edit metadata for the current image. This works very similarly to in the Lister. To display or hide the metadata pane, press the **F9** key.



As in the Lister, click the [Apply](#) link at the top-right of the pane to save any metadata changes you've made. You can also turn on the checkbox next to the [Apply](#) link to have changes saved automatically when you move to the next picture or close the viewer.

### Read-ahead caching

The viewer now has a read-ahead cache, which can make things feel much more fluid when viewing a number of images. While you're looking at one picture, the viewer will be loading the next and previous pictures in the background, so that they're ready to display almost immediately when you move on.

## Reselect command

In the **Edit** menu there's a new **Reselect** command. This sets the selection to the same rectangle that was most recently selected in any viewer window (whether open or closed), which is handy for cropping the same part of multiple images.

## Image conversion of displayed image

When the **File / Convert Image** command is run in the viewer, the conversion tool now gets the image data directly from the displayed image – meaning any rotation or cropping you've done will be reflected in the converted image. This uses the new **@useimagedata** command modifier and can be overridden for an individual command using the **NOUSEIMAGEDATA** argument (which would let you configure a viewer toolbar button to **not** behave like this if needed).

## Minimum width when window auto-sizes

On the *Viewer / Appearance* page in Preferences, it's now possible to configure a minimum width for the viewer window when the *Auto-size* option is turned on. You might want to use this to make sure your toolbar (or a certain amount of it) is always visible when viewing small images. You can set the minimum width automatically from an existing viewer window using the **View / Save Minimum Width** command in the default toolbar.

## Mouse button functions

In Opus 11 you could configure the behaviour of the left and middle mouse buttons from a list of predefined functions. Opus 12 adds a text field letting you specify your own command as well as choosing from the drop-down list. You can now also have a command that runs on left double-click.

## High DPI support

### *An overview of high DPI support*

High DPI monitors are becoming more and more common. 4K or 5K monitors in 24" or 27" form factors have pixels too small to run them at their native resolution (unless you have super-vision) and so these will most commonly be used at higher DPI scaling levels (e.g. instead of the 96 dots-per-inch on a standard monitor, they may be run at 125% (120 dpi), 150% (144 dpi), 200% (192 dpi) or even higher). This lets text and icons be sharper and clearer, but without providing any more "usable" screen space.

In practical terms what this means is that software GUIs have to either be up-scaled by the OS (resulting in a blurry or fuzzy appearance of things like text and icons), or support higher DPI natively with higher resolution icon imagery.

## **Fonts**

Opus 12 configurations store the DPI setting along with your configured font sizes, which means you can move your configuration between different DPIs (say, a high-DPI desktop and a low-DPI laptop) and the correct font size will be used. In Opus 11 this would result in your fonts being either too big or too small, depending on the machine the configuration was saved on.

The **first** time you use an earlier configuration with Opus 12, your existing font sizes will be reset to the defaults.

## **Toolbar icons**

Toolbar icon sets have always supported two icon sizes (small and large); that is, an icon set could contain two completely different sets of images that were used for icons at the two sizes. In Opus 12 icon sets still support two “named” sizes (small and large), but each size can have more than one set of image data, flagged for differing DPI levels. Each set of images can be given a maximum and minimum DPI scaling level that it’s designed for, and Opus will pick the most appropriate size to use automatically.

If an icon set doesn’t have images designed for the current DPI level, Opus will pick the closest size and automatically scale it up (or down) so that all toolbar icons are displayed with the correct dimensions.

## **Status bar and toolbars**

Elements with configurable widths (status bar parts, status bar graphs, and the breadcrumbs path field) are now scaled for the system DPI by default. E.g. a status bar part set to width 200 (with the `{width200}` code) will be 400 pixels wide on a system with 200% DPI scaling.

If you specifically want an absolute width that doesn’t get scaled, simply specify a negative number instead. For example, a status bar part set to `{width-200}` would be 200 pixels wide no matter what the DPI scale factor.

## **Commands and scripts**

The new `{dpi}` command argument lets you use DPI-sensitive values with simple commands. This can be useful if you have buttons which specify column or window sizes and you want consistent results from the same button in different DPIs.

- `{dpi}` on its own will report the current DPI. 96 at standard DPI, 192 at 200% DPI, and so on.
- `{dpi%}` will report the current DPI scale factor. 100 at standard DPI, 200 at 200% DPI, and so on.
- `{dpi<number>}` converts a standard 96 DPI pixel width to the current DPI. For example, if you are at 200% DPI, `{dpi25}` will output 50.

- **{dpi/<number>}** will convert from the current DPI back to standard 96 DPI pixels. For example, if you are at 200% DPI, **{dpi/50}** will output 25.

Example use in a command: **Set LISTERSIZE {dpi|640},{dpi|480}**. For scripts, the new **DPI** object (obtained via the **DOpus.DPI** method) provides similar functionality.

## File and folder labels

### *Stackable labels*

File and folder labels can now “stack” on top of each other. This means that it’s now possible to apply multiple labels to one file, or have multiple wildcard labels that may match a single file. Instead of just the first matching label being used, Opus will continue to search for other matching labels and apply those label settings on top of the previous ones.

For example, you could have one label that colors the names of all JPEG files green, and another that bolds the filenames of all images that are 1920x1080 pixels or larger. In Opus 11 only one of these labels could ever be effective at once, but in Opus 12 any 1920x1080 JPEG files would have their filenames shown as bolded green.

Each label definition now has a “stop on match” flag which lets you prevent this behaviour on a per-label basis.

By default explicitly applied labels override wildcard labels, but the new Preferences setting *Favorites and Recent / Labels / Apply wildcard and label filters to explicitly labeled files and folder* lets wildcard labels stack on top of explicitly-assigned ones.

### *Adjustable priority*

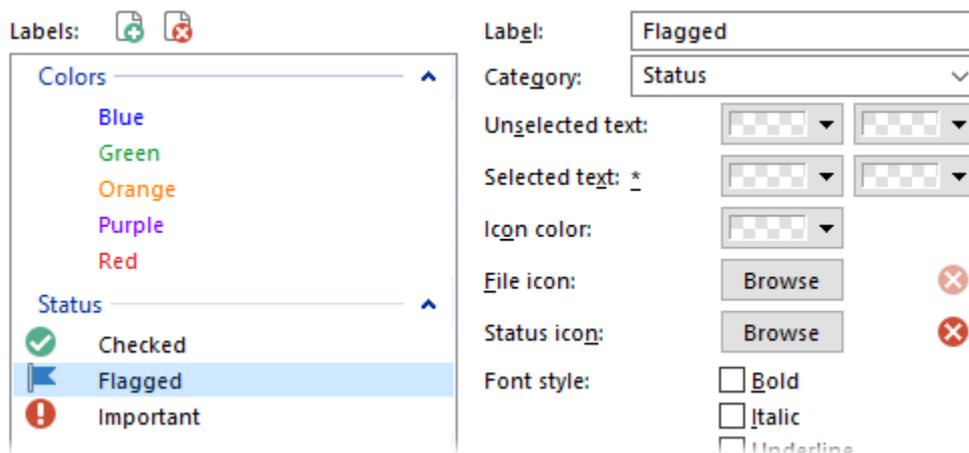
Filter and wildcard label assignments can now be rearranged to control their priority. The new *Label Assignments* page in Preferences has up and down buttons that let you move label assignments above or below each other.

## Categories

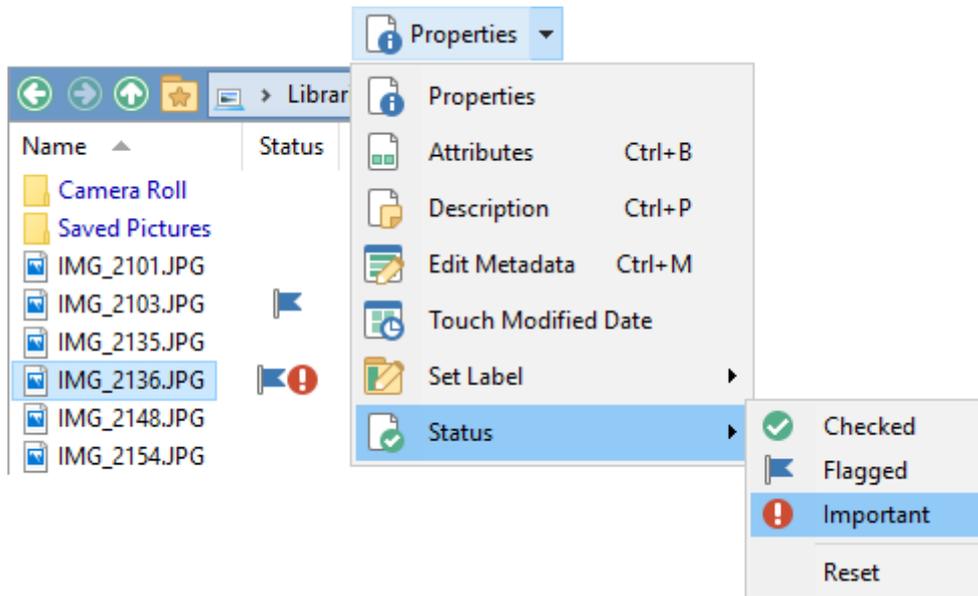
Labels can now optionally be arranged into categories. These are used to group the labels on the *Favorites and Recent / Labels* page in Preferences, and can optionally group them in toolbars when displaying generated lists of labels (e.g. the list generated by the **Properties SETLABEL !menu** command).

## Status icons and the status column

In addition to text color and style, and icon and icon color, labels can now also specify a “status icon”. This is an additional icon that, instead of overriding the regular file icon, is displayed separately (in the new **Status** column). Because labels are now stackable, this means you can apply multiple status icons to the one file.



Three status labels are created by default (shown above). To clarify, these are no different to any other label – they just happen to only define a status icon, and have been placed in the *Status* category. They could just as easily also turn on the bold style or color the file icon.



The default *Properties* drop-down (shown above) uses the new label category functionality to display two separate sub-menus; one (*Status*) for labels in the “Status” category, and another (*Set Label*) for everything else.

You can see in this example that one file has been assigned the “Flagged” label, and another has been assigned both “Flagged” and “Important”. The *Status* column has been added to the list and displays the assigned status icons.

### **TortoiseSVN status**

For developers using the free TortoiseSVN source code control utility, Opus can optionally display a file’s SVN status as an icon in the *Status* column. This is controlled by the *Folders / Folder Display / Show TortoiseSVN status icons in the Status column* option in Preferences. If turned on, Opus will automatically add the appropriate SVN status icon to any other label icons displayed in the *Status* column.

Status	Name	Size	Type
✓	Add to Archive Dialog.xml	2.84 KB	XML Document
✓	Adding a new Site.xml	5.78 KB	XML Document
✓	Adding Cover Art.xml	3.36 KB	XML Document
!	Adding to Archives.xml	3.74 KB	XML Document
✓	Adding, Removing and Editing Clauses.xml	2.91 KB	XML Document
✓	Additional Functionality.xml	2.45 KB	XML Document

Other than the status icon appearing larger and more distinctive than the usual icon overlay that Tortoise uses, this can help with the problem of limited icon overlays. Windows only allows a maximum of 15 icon overlays in the system, and Tortoise normally uses 8 or more of these for

itself, crowding out other shell extensions. Using this option in Opus gets around the limit as the status is taken directly from Tortoise rather than via the icon overlay.

In Preferences there's also an option to disable the Tortoise icon overlay handler in Opus completely (as shown above – no point showing the same status icon twice), and an option to choose the SVN status icon set.

Note that this feature requires TortoiseSVN version 1.9.3 or greater.

## Commands

The **Properties SETLABEL** command can now apply more than one label to a file at once, by comma-separating the label names - for example, **Properties SETLABEL red,green**. The keyword **stoponmatch** can be added to override the *Apply wildcard and label filters to explicitly labeled files and folders* Preferences option, to stop a file inheriting from wildcard and filter labels.

The **Properties SETLABELTOGGLE** argument now accepts the optional keyword **shift** - if specified, the labels will only be toggled if the **Shift** key is held down (e.g. **Properties SETLABEL red SETLABELTOGGLE shift**).

The new **Properties ADDLABEL** argument lets you add a label to a file without replacing any existing labels it might have. If the optional keyword **ctrl** is specified, the label will only be added if the **Control** key is held down (e.g. **Properties SETLABEL green ADDLABEL ctrl**). **ADDLABEL** also works in conjunction with **SETLABELTOGGLE**, to toggle a label on and off without affecting any others the file may have.

The **Properties SETLABEL !submenu** command can be used to generate a list or menu of labels grouped into categories – this can also be combined with the **!menu** keyword. You can also combine the **ADDLABEL** and **SETLABETOGGLE** to dynamically generate a list of “toggle label” commands (e.g. **Properties SETLABEL !menu !submenu ADDLABEL SETLABELTOGGLE**).

The new **Properties LABELCATEGORY** argument can be used to filter the generated list of labels by category. It accepts one or more wildcard strings which let you match the name of categories to include. E.g. to display a list of labels in the *Status* category, the default drop-down uses the command:

## Properties SETLABEL !menu LABELCATEGORY raw:Status ADDLABEL SETLABELTOGGLE

The specified categories will also be used when resetting labels using the **Properties SETLABEL !reset** command - if **LABELCATEGORY** is used as well, only labels in the specified categories will be cleared. This lets you have a command that, for example, clears all *Status* category labels (removes all status icons) without affecting any other labels the file may have.

## Manual sorting

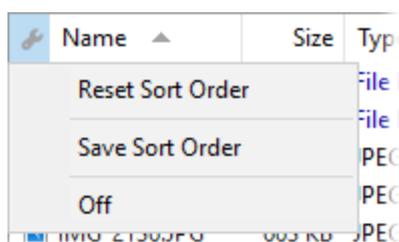
### *An overview of manual sorting*

Manual sorting refers to being able to control the display order of files and folders arbitrarily. Although for most purposes the automatic sorting methods (e.g. sorting files alphabetically by name) are all you need, many people have use cases where being able to control the sort order exactly would be useful.

Manual sort mode needs to be activated in a Lister before files can be arbitrarily arranged. To activate manual sorting, use the **Set MANUALSORT** command, or turn on the *Sorting Options / Manual sorting* option on the *Display* tab in the Folder Options dialog. Obviously as a folder options setting, this mode can also be made permanent by saving the folder format.

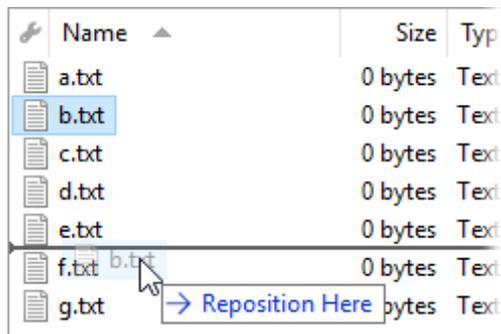
If a manual sort order had previously been defined (and saved) for the current folder, the file list will resort automatically when manual sorting is turned on. Otherwise, turning on manual sorting will have no immediate effect on the file list.

In details or power mode, or when the column header is enabled in the other modes, a new button (  ) appears on the left of the column header when in manual sort mode. Clicking this button displays a control menu with a few commands in it relating to manual sort mode:



## How to manually sort your folders

Once manual sort mode is active, you can reposition a file using drag-and-drop, or from the keyboard using **Shift + Alt** in conjunction with the cursor keys.



By default, your manual sort order will be persistent (saved automatically), whenever possible (see below for current limitations on manual sorting). If you want the freedom to change the sort order temporarily without making it permanent, you can turn off the *Folders / Folder Behaviour / Automatically save manual sort order where possible* option in Preferences. With that option turned off, you need to use the **Save Sort Order** command in the control menu, or the **Set MANUALSORTSAVE** command, to save the current sort order permanently.

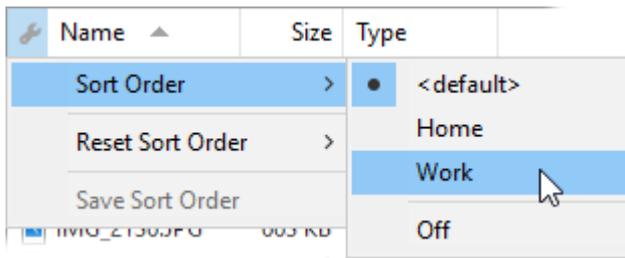
At any time, you can reset the sort order to the current automatic order (e.g. alphabetical by name) using the **Reset Sort Order** command in the control menu (shown above), or with the **Set MANUALSORTRESET** command.

## Multiple custom sorts

By default you're limited to just a single manual sort order per folder (one manual sort order ought to be enough for anyone!) but if you need to be able to define more than one, you can add “named orders” through the *Miscellaneous / Advanced / manual\_sort\_names* option in Preferences.

layout_string	
manual_sort_names	Home; Work
max folder thumb time	5000

When named orders are defined they appear in the column header's control menu, letting you switch between them at will.



The Folder Options dialog also displays a drop-down (on the *Display* tab) showing all the named sort orders, which lets you select the default for a folder if desired. The **Set MANUALSORT** command also lets you specify the named order to switch to a particular sort from a command.

## Scripting

The **Format** script object provides the following properties relating to manual sorting:

- **manual\_sort**: Returns **True** if manual sorting is active.
- **manual\_sort\_name**: The name of the current manual sort (or empty if the sort order is not named).
- **manual\_sort\_order**: Returns a **SortOrder** object that allows the current sort order to be queried and modified.

The **SortOrder** object returned by the **manual\_sort\_order** property has the following methods:

- **GetOrder**: Returns a **Vector** of strings representing the current sort order of the folder. An optional parameter lets you specify a particular named sort.
- **SetOrder**: Accepts a **Vector** of strings representing the desired sort order of the folder.
- **ResetOrder**: Resets the current sort order of the folder to the default. An optional parameter lets you specify a particular named sort.

## Limitations

Currently manual sort orders can't be saved for some types of folders:

- Non-NTFS disk folders (e.g. FAT/FAT32)
- Archives
- FTP

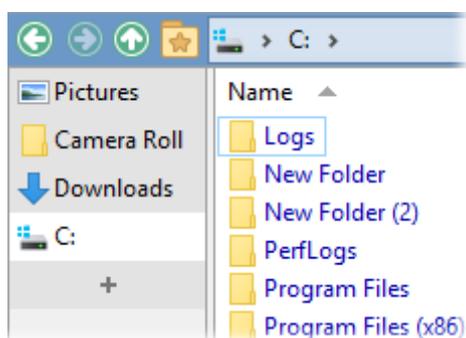
By default, manual sorting is disabled completely in folders that the order can't be saved for. If you turn on the *Folders / Folder Behaviour / Allow manual sorting in all folders* Preferences option then manual sorting will be available everywhere, but changes to the sort order in those types of folders will only be temporary.

Manual sorting is currently not supported in Flat View or when the file display is grouped. It will also not work correctly in a folder when compatibility files are displayed and there are two files with the same name.

## File displays

### **Vertical folder tabs**

Folder tabs can now be displayed vertically (to the left or right of the file display) as well as horizontally (above or below).



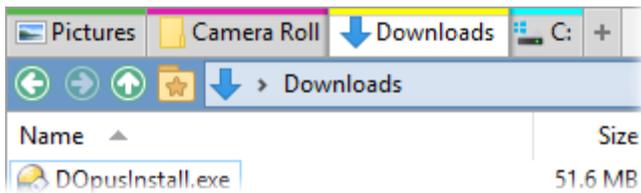
The *Folder Tabs / Options / Tab Position* option in Preferences lets you choose where your tabs are to be positioned. Select *left* or *right* for vertical tabs. The initial width of the tab “well” can be configured in *Preferences / Folder Tabs / Appearance*, or you can choose automatic sizing.

The old option to display tabs at the top and bottom in **Dual Horizontal** display mode has been replaced with a *Dual display position* option (on the *Folder Tabs / Options* page). This lets you choose “together” (the equivalent of the old option) as well as “apart” (the opposite effect).

Note that when folder tabs are displayed vertically the overflow menu/scroll button is replaced with a scrollbar.

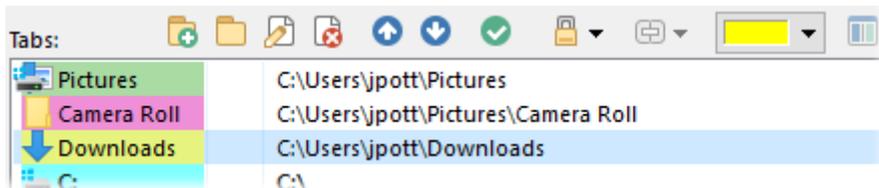
### **Folder tab colors**

Folder Tabs can now be colored individually.



There are a number of ways you can do this:

- Using the **Go TABCOLOR** command you can change the color of the current folder tab, for example **Go TABCOLOR #ff8000** would make the current tab orange. Use **Go TABCOLOR reset** to reset the tab's color.
- If you right-click on a folder tab you can select the **Set Tab Color** command from the context menu, which lets you change the color interactively.
- When defining a tab group, you can assign colors to individual tabs within the group.

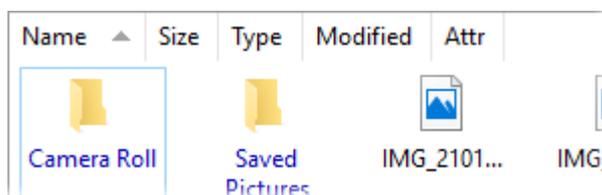


- Folder formats can specify a tab color on the *Options* page of the *Folder Options* dialog; this color will be used for a tab when that format is active unless the color has been overridden by something else.

The **Tab** script object has a new **color** property which lets you query the color of a tab (run the **Go TABCOLOR** command from the script to change it).

### **Column headers in icon modes**

The column header can now be shown in the icon modes, providing an quick way to resort the file list (plus access to the column header context menu).



This is controlled by the new *File Displays / Options / Show sort header in icon modes* Preferences option. Also the **Set ICONMODESORTHEADER** command lets you toggle the header on and off from a button or hotkey.

## Relative graphs behind size and date fields

The old *Relative Age* column (which shows a graph indicating a file's age) has been renamed to *Modify (relative)*, and a new *Create (relative)* column has been added. This lets you see relative ages based on the creation time as well as last modification time of a file. A new option in *Preferences / Display / Fields* lets you invert the meaning of these graphs (so, e.g. the widest bar indicates the newest file rather than the oldest one). Graphs are now calculated separately for files and folders, although you can revert to the old behavior using the `graphs_separate_files_dirs` option in *Preferences / Miscellaneous / Advanced*.

As well as the relative age columns and the separate *Relative Size* column, these graphs can now be shown as the “background” to normal size and date fields. This gives the benefits of the graph without having an additional column take up space.

	DeviceSetupManagerAPI.dll	115 KB	30/10/2015	6:19 PM		
	dhcpcsvc6.dll	65.5 KB	28/05/2016	2:24 PM		
	dhcpcsap.dll	210 KB	30/10/2015	6:17 PM		
	DiagCpl.dll	1 MB	30/10/2015	6:17 PM		
	DiagnosticLogCSP.dll	281 KB	30/10/2015	6:17 PM		
	diagperf.dll	1.37 MB	1/07/2016	1:38 PM		
	diagtrack.dll	1.53 MB	1/07/2016	2:45 PM		
	diagtrack_win.dll	360 KB	1/07/2016	1:46 PM		
	discan.dll	247 KB	30/10/2015	6:17 PM		

	DeviceSetupManagerAPI.dll	115 KB	30/10/2015	6:19 PM		
	dhcpcsvc6.dll	65.5 KB	28/05/2016	2:24 PM		
	dhcpcsap.dll	210 KB	30/10/2015	6:17 PM		
	DiagCpl.dll	1 MB	30/10/2015	6:17 PM		
	DiagnosticLogCSP.dll	281 KB	30/10/2015	6:17 PM		
	diagperf.dll	1.37 MB	1/07/2016	1:38 PM		
	diagtrack.dll	1.53 MB	1/07/2016	2:45 PM		
	diagtrack_win.dll	360 KB	1/07/2016	1:46 PM		
	discan.dll	247 KB	30/10/2015	6:17 PM		

To enable these graphs, use the *Show relative graphs behind...* options in *Preferences / Folders / Folder Display*. When enabled, the opacity of the background fill is controlled the *Background graph opacity* setting in *Preferences / Display / Options*.

## Vertical gridlines in Details and Power modes

The *File Display Modes / Details* and *File Display Modes / Power Mode* pages in Preferences now both have options to enable vertical gridlines as well as horizontal.

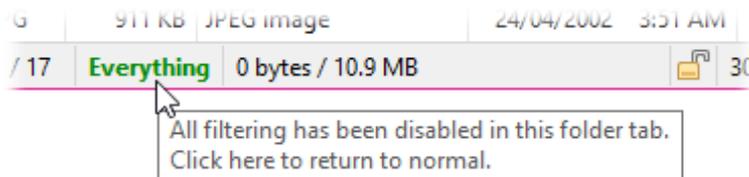
	IMG_2101.JPG	678 KB	JPEG image	22/04/2002	5:15 AM	-a----	
	IMG_2103.JPG	633 KB	JPEG image	22/04/2002	5:16 AM	-a----	
	IMG_2135.JPG	406 KB	JPEG image	23/04/2002	2:56 AM	-a----	
	IMG_2136.JPG	663 KB	JPEG image	23/04/2002	2:57 AM	-a----	
	IMG_2148.JPG	769 KB	JPEG image	23/04/2002	10:02 AM	-a----	
	IMG_2154.JPG	765 KB	JPEG image	23/04/2002	11:29 PM	-a----	
	Img_2164.jpg	676 KB	JPEG image	8/06/2002	5:52 PM	-a----	

You can configure line style, color and opacity separately for the vertical and horizontal lines.

The **Set GRIDLINES** command (which still works for backwards compatibility) has been split into two – the command **Set GRIDLINESH** controls the horizontal lines, and **Set GRIDLINESV** controls the vertical lines.

### Show Everything

*Show Everything* is a one-click toggle to show everything below the current folder, ignoring all local and global filters. It's local to the folder tab it is used in.



Clicking the count of hidden items on the status bar toggles *Show Everything* mode. The new default status bar displays **Everything** if *Show Everything* is active (you won't see this unless you reset your status bar to the defaults).

- On the default toolbars, **Folder > Show Everything** replaces the old **Folder > Clear Local Name Filters** command.
- The Filter Bar has a new **Show everything** checkbox which also lets you toggle the mode on and off.
- You can use the command **Set SHOWEVERYTHING** to toggle the mode from a button or toolbar.
- The new **{hse}** status bar code lets you display the **Everything** indicator on your status bar. The panel that contains **{hse}** can also be clicked on to toggle Show Everything mode.

## **Other file display-related changes**

- If the *Rating* column is displayed in the file display, you can now edit a file's rating directly by clicking the stars (rather than going through the Metadata panel).
- When the file display is grouped, the collapse state of each group is now remembered in the back / forward history in the file display.
- The Filter Bar drop-down now lets you select File Type Groups, for example if you want to quickly filter on images.
- The file display will now ensure the focus item is visible after clearing the quick filter.
- The file display now makes the focus item visible when changing display modes.
- The file display now keeps the focus item visible, if it already is, when resizing the window or zooming the font/thumbnail size.
- Thumbnails, Tiles and Icon modes now automatically increase horizontal spacing to spread across the available space, rather than leave the unused space after the last column. The spacing setting still specifies the minimum spacing.
- The Thumbnail column can now be used in This PC (aka Computer and My Computer).
- Drag and drop of text (from applications that allow you to do that) to a Lister is now supported (it behaves the same as using **Ctrl V** to paste clipboard text to a Lister, creating a file called *Clipboard Text.txt*).
- Selected files hidden by **Select HIDESEL** and similar commands will no longer be selected when revealed again, since it was usually unexpected.
- The new *Uncompressed* column displays the uncompressed size of supported archive files.
- The new *Encoding Software* column, displays the TSSE frame for MP3 music files.

## **Preferences options relating to the file display**

- *Display / Fields*: Now allows the alignment (left/right/center) for individual columns to be modified from the defaults.
- *File Displays / Mouse / Double-click on file display background*: The old “go up” option has been removed, now there is just a field that allows any command to be entered. If you had “go up” selected in your configuration this will turn into the **Go UP BACK** command automatically.
- *File Displays / Mouse / Middle double-click on file display background*: New option to run a command when the file display background is double-clicked with the middle mouse button.
- *File Displays / Options / Hover to switch source/destination state*: Hovering the mouse cursor over a file display for the specified time (in milliseconds) will set it as the source.
- *File Displays / Options / Specify initial folder when switching to dual file display*: The new *Tab group* option lets you configure a tab group to be opened automatically when switching to dual display mode (instead of just a folder).
- *Folders / Auto-Loading*: The *Prevent automatic loading of certain types of folders* options now include one for **All other drive types**, which applies to everything not listed

separately. For each folder type, you can also now select **Load on tab activation** as an option which prevents the loading initially if the tab isn't the active one, but will load the folder automatically the first time the tab is activated.

- *Folders / Global Filters*: The two wildcard filters now have options to use regular expressions.
- *Folders / Virtual Folders*: New options to control the appearance of drive labels in the native My Computer display. Also a new option to *Sort drives by name instead of letter*.
- *Miscellaneous / Advanced / scroll\_lock*: If the **Scroll Lock** key is turned on the cursor keys will scroll the file display without changing the focus.

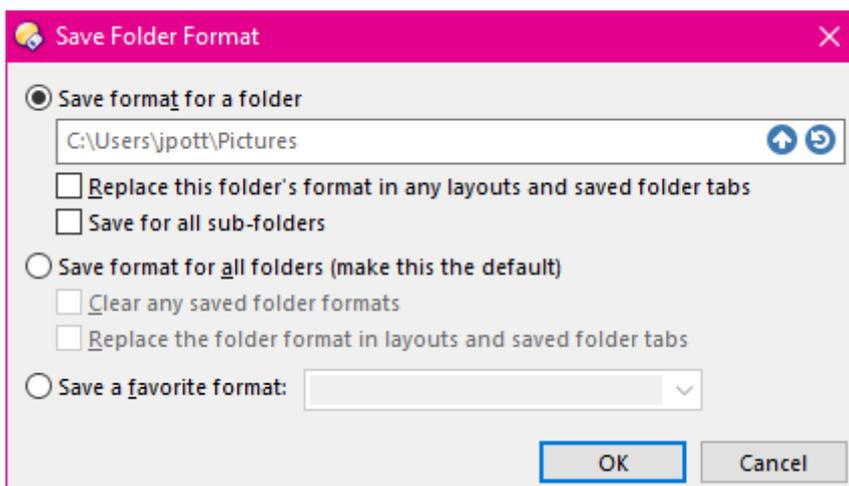
### Commands relating to the file display

- **Set GLOBALHIDEFILENAME** and **GLOBALHIDEFOLDERS**: You can now use a regular expression with these commands by specifying **regex:** as a prefix (e.g. **Set GLOBALHIDEFILENAME regex:^tmp\.**)
- **Go TABCLOSEALL**: Now has a **dest** argument that lets it be run against the destination file display in a dual-display Lister.
- **Go TABGROUPLIST**: Now works in conjunction with the **USEQUALKEYS** and **KEYARGS** arguments.

## Folder Formats and Folder Options

### Save format

The function in the *Folder Options* dialog to *Save the current folder format* now uses a dialog rather than a drop-down menu, making it easier to choose what you want to save the format for.



There are three main options when saving a format:

- **Save format for this folder**: The format will be saved for this specific folder (the one currently displayed in the file display). If you turn on the **Replace this folder's format...**

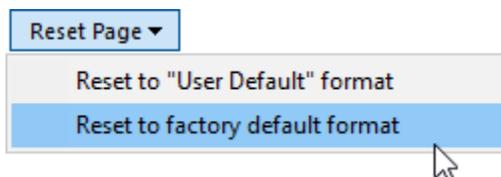
option , Opus will search through all your saved Lister layouts, saved folder tabs (and styles), and any currently open folder tabs for this folder, and update their format as well. The **Save for all sub-folders** option causes the **Use as the default format for all sub-folders** option to be turned on in the saved format.

- **Save format for all folders:** The format will be saved as the new *User Default* format, which is used for a folder that doesn't have a specific folder format saved for it. If you turn on the **Clear any saved folder formats** option, and folders for which you've previously saved a specific format will be reset. And the **Replace the folder format...** option causes any saved layouts, tabs and styles to also have their formats reset.
- **Save a favorite format:** This lets you save the format as a favorite format (either a new one, or over the top of an existing favorite).

### **Default format**

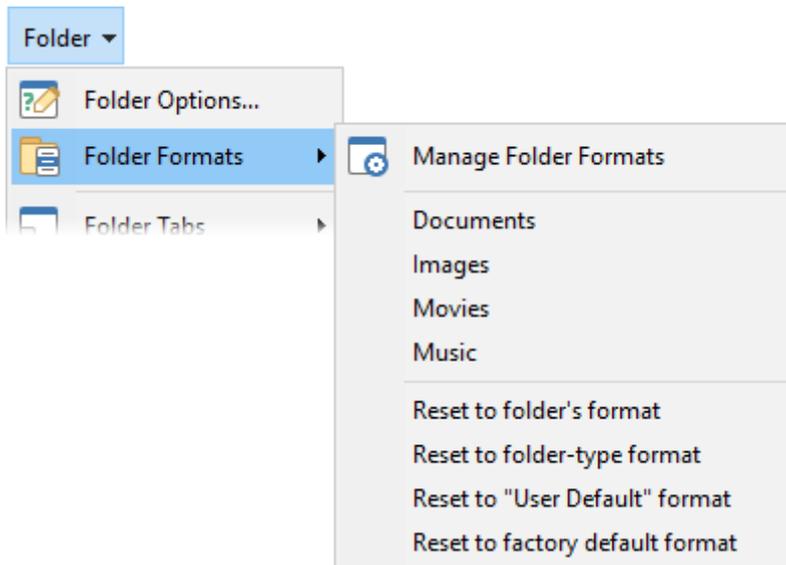
The old *Custom* format has been renamed to *User Default* to try to make its purpose clearer (that is, a default format defined by the user).

The “hard-coded” default format (basic *Name/Size/Type/Date/Attributes*) is now referred to as *Factory default* (this really only crops up in a few places, like the **Reset Page** drop-down at the bottom of the Folder Options dialog).



### **Folder menu**

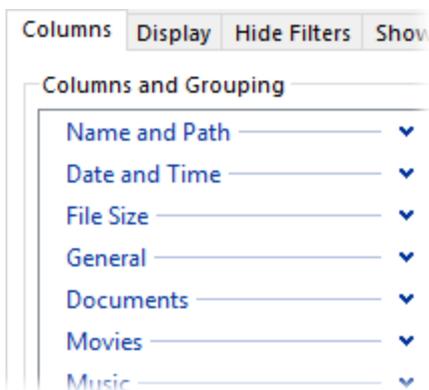
Under the *Folder > Folder Formats* menu in the default *Menu* toolbar there are new reset options, including **Reset to folder's format**, which undoes any changes you've made and resets things as if you had opened a fresh window and gone to the folder.



## Columns tab

### Available column categories

The old drop-down category list for available columns is gone, and the list of columns is now split into groups in the list itself.



The categories themselves have been expanded and reorganised to make it easier to locate the columns you're looking for. (These new categories are also reflected in any column lists shown in menus throughout Opus).

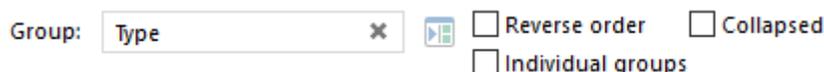
### Column filter

Under the list of available columns there's now a filter control that lets you filter the list of columns by name.



## Grouping

In Opus 11 grouping was enabled (and the column to group by chosen) from the list of displayed fields; however, it's possible to group by a column that's not actually displayed and the old user interface didn't allow this.



Grouping is now controlled using a separate setting at the bottom of the *Columns* tab. The current group column is displayed, if any, and to clear it click the **x** symbol. To group by a new column, select the desired column in either of the two column lists above, and click the arrow button. You'll notice that the arrow button changes between  and  depending on which list was most recently active – this is a visual cue to indicate which list the group column will be set from when the button is clicked.

The **Reverse order** option lets you set the initial group order (how the individual groups are arranged) to the reverse of the default, and **Collapsed** is a new option that lets you have the groups start out as collapsed.

## Column width

The list of displayed fields has new options for controlling the widths of columns.

Column widths have always been slightly complicated by the fact that the *Auto-size all columns* option is hidden away on another tab (the *Display* tab). In Opus 12 we've tried to improve this by making it more obvious when column widths are being controlled by the global option, and also making that option turn itself off automatically as soon as you edit a column width manually.

If the *Auto-size all columns* option on the *Display* tab is turned on, all columns will be shown with their widths as *Auto* (and grayed out to indicate that option is in effect):

Displayed fields	Width	Max	Sort
Filename	Auto		✓
Size (auto)	Auto		
Type	Auto		
Date and Time (modified)	Auto		
Attributes	Auto		

As soon as you edit the width of a field, the *Auto-size all columns* option is automatically turned off. All the other columns will still show their widths as *Auto*, but it won't be grayed out any more – this indicates that the column's width is specifically set to *Auto* rather than the global option overriding.

If you click the *Width* field of one of the displayed columns, you now see a drop-down menu with a number of width options available:

Displayed fields	Width	Max	Sort
Filename	Auto		✓
Size (auto)	Auto		
Type	Auto		
Date and Time (modified)	Auto		
Attributes	Auto		

The options are:

- **Auto:** Automatically size this column. This has the same effect as the *Auto-size all columns* option on the Display tab, but lets you apply it on a per-column basis.
- **Expand:** Automatically sizes the column the same as **Auto**. The difference is the widths of **Expand** columns are ignored in the calculation of **Fill** columns(see below), and columns set to **Expand** will go off the right hand side of the file display rather than making **Collapse** columns start to collapse.
- **Collapse:** Automatically sizes the column, but its width is able to collapse (down to zero width if necessary) to allow other fields to fit without horizontal scrollbars appearing. For example, you might want the *Description* column displayed, but not have it force other fields off the edge of the file display. Setting it to **Collapse** means it will only appear if there's space for it.
- **Fill:** Columns set to fill will be automatically sized to fill any available horizontal space in the file display. If there is more than one **Fill** column they divide the available space between them. Columns set to **Fill** can potentially end up wider than they need to be (contrast with **Auto** + **Fill** described below).

You can also enter a desired pixel width into the *Width* field.

If you're using the new **Fill**, **Expand** and **Collapse** column modes, you may want a quick way to switch everything to auto-size. The following script auto-sizes all columns if any have their widths restricted, and otherwise will reset the folder format (including columns and column widths) to what a new window would show for the folder.

*Script Type: JScript*

```
function OnClick(clickData)
{
    var anyColumns = false;
    var cmdLine = "Set COLUMNSADD=";
    for (var e = new Enumerator(clickData.func.sourcetab.format.columns);
    !e.atEnd(); e.moveNext())
    {
        var col = e.item();
        if (!col.Autosize || col.Max != 0)
        {
            if (anyColumns) cmdLine += ",";
            cmdLine += col.Name;
            cmdLine += "(!,a,0)"; // Keep position. Auto-size. No maximum.
            anyColumns = true;
        }
    }
    if (!anyColumns) cmdLine = "Set FORMAT=!folder";
    clickData.func.command.RunCommand(cmdLine);
}
```

## Column maximum width

Columns that are set to automatically size (**Auto**, **Expand** or **Collapse**) can also have a maximum width set using the *Max* field.

Displayed fields	Width	Max	Sort
Filename	Auto		✓
Size (auto)	Auto	Fill	
Type	Expand	85	
Date and Time (modified)	Auto		

You can enter a maximum size in pixels. For a column set to **Auto** width you can also choose **Fill** for the maximum, which makes it auto-size up to but not beyond the width of the file display (to avoid horizontal scrolling), but unlike setting **Fill** for the width it won't auto-size larger than it needs to be.

## Display tab

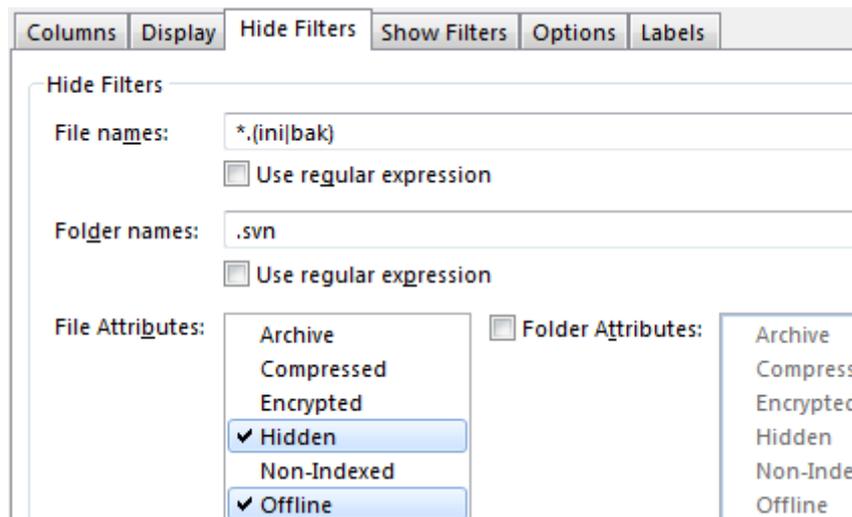
There are two new options on the *Display* tab:

- **Manual sorting:** Lets you enable manual sorting in a folder.

- **When grouped, combine groups with only one member into the “Other” group:** When enabled, and the file display is grouped, any items in a group by themselves will instead be shown in a group called *Other* (prevents cluttering up the folder with lots of groups containing only one file).

## Filters tabs

The old *Filters* tab has been split into two – *Hide Filters* and *Show Filters*. Other than the outcome of the filter these tabs are identical.



Options have been added to the *File names* and *Folder names* options to use regular expressions as well as standard wildcards.

The attributes options are now presented as a list showing the full attribute names rather than checkboxes with the single-letter abbreviation of the previous version.

## Preferences options relating to folder formats

- *Folders / Folder Formats*: The old *Default Formats* section has been renamed to *Folder Type Formats* (as these are formats that are used for certain types of folders). The *User Default* format (which used to be called *Custom*) is now in a section by itself at the bottom of the list.
- *Folders / Folder Formats*: Wildcard Path formats now have an **Expand aliases** option. If this is turned on then Opus will attempt to expand folder aliases and environment variables in the entered string before performing the pattern matching. For example, */SDATA* (which is a folder alias for a drive called *Data*) would let you create a folder format that applies to a drive labelled *Data* no matter what drive letter it has.

## Commands relating to folder formats

- **Set SAVEFORMAT:** Lets you save the current folder format without going through the Folder Options dialog.
- **Set COLUMNSADD** and **COLUMNSTOGGLE:** When specifying the size for fields added with these commands, you can now use **a** for *Auto*, **f** for *Fill*, **e** for *Expand* and **c** for *Collapse*. For example, to add the *picture width* field with its width set to auto, you might use **Set COLUMNSADD picwidth(\*,a)**.

You can also specify the maximum width with an additional parameter; e.g. to add the *picture width* field with its width set to auto and maximum width set to fill, you might use **Set COLUMNSADD picwidth(\*,a,f)**.

- **Set SHOWFILTERFILENAME** and similar commands: You can now use a regular expression with these commands by specifying **regex:** as a prefix (e.g. **Set SHOWFILTERFILENAME regex:^tmp\.**)
- **Set GROUPELLAPSE:** When the file display is grouped, this command can be used to expand or collapse groups by name.
- **Set COMBINESINGLEGROUPS:** Lets you control the state of the *When grouped, combine groups with only one member into the "Other" group* option.
- **Set FORMAT** and **Print FOLDER FORMAT:** As well as accepting the name of a favorite format, the **FORMAT** argument for both these commands accepts the following special keywords:
  - **!factory:** Reset to factory defaults.
  - **!user:** Reset to the user default (in Opus 11 this was **!custom**, which still works for compatibility).
  - **!default:** Resets to Folder Type format applicable to current folder.
  - **!folder:** Resets to the format for the folder that a brand new window would use.
  - **!current:** Only useful for **Print FOLDER FORMAT=!current** – uses the current format shown in the Lister.

## Folder formats scripting

- The **Column** object has new properties to expose the new column width options (**autosize**, **fill**, **expand**, **collapse**, **max**).
- The **Format** object has four new properties which indicate whether the file and folder filters are set to regular expression mode (**hide\_files\_regex**, **hide\_dirs\_regex**, **show\_files\_regex**, **show\_dirs\_regex**).

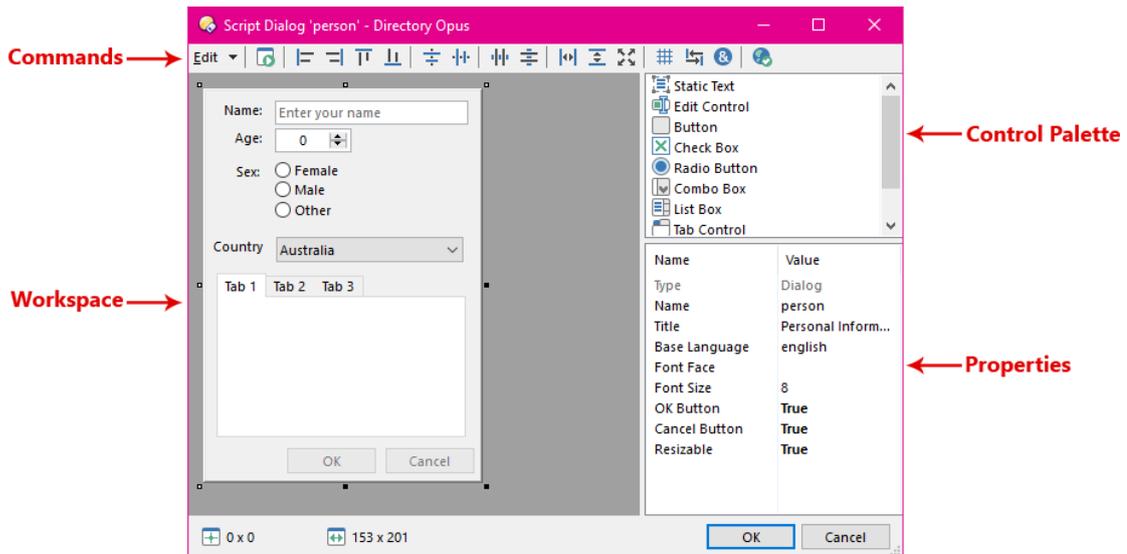
## Script dialogs

Scripts have always been able to display simple dialogs (e.g. *Yes/No* style requesters, or a dialog letting you enter a simple string or choose an item from a list). In Opus 12 scripts are now able to define free-form dialogs in much the same way that “proper” Windows software can, using many of the standard Windows controls.



Above is an example of the type of dialog that scripts can now create. Dialogs are defined as “resources” – XML formatted data that defines the dialog and control layout. Any script can have resources attached – either a script in a button, or one in the Script Add-Ins folder.

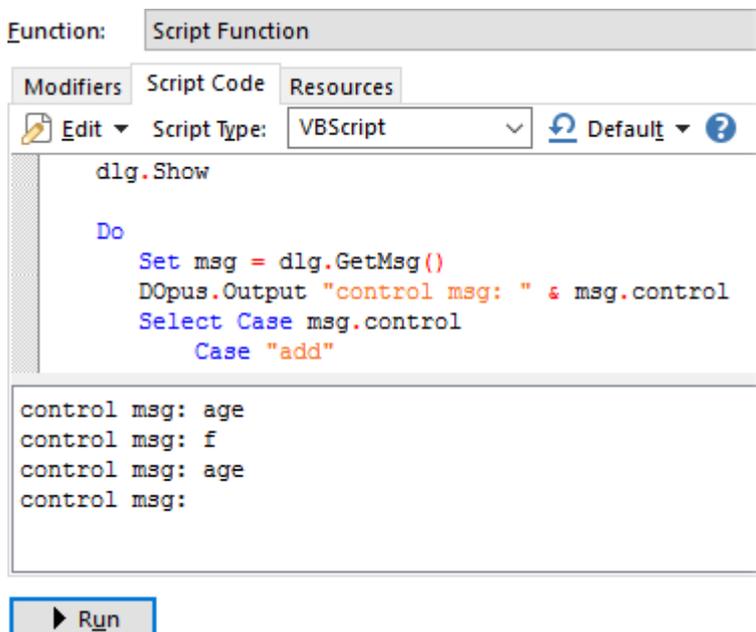
A full GUI-based dialog editor is provided inside the function editor, which makes it very easy to design script dialogs.



## Function editor

When the function editor has been set to run a *Script function*, it now has three separate tabs which split the function into:

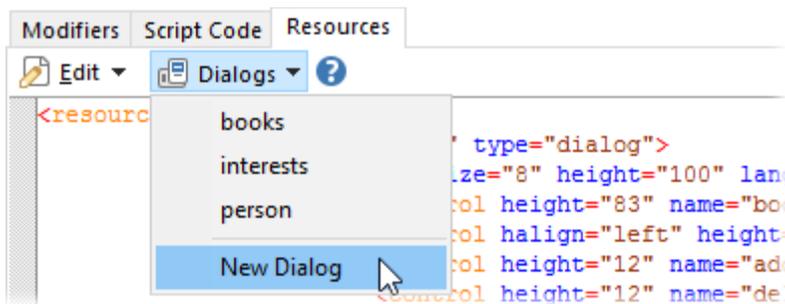
- **Modifiers:** Any command modifiers that apply to the script (e.g. **@filesfromdroponly**).
- **Script Code:** The actual code that defines the script.
- **Resources:** Script resources.



The script type (language) is now set using the **Script Type** drop-down in the toolbar – you don't need to type the **@script** line explicitly any more. The **Default** button lets you save a script “template” as the default for a particular language, and revert to the default at any time.

At the bottom of the function editor a new **Run** button lets you test the current script immediately, without having to exit *Customize* mode. When you use the **Run** button an output panel will appear (shown above) which displays any errors or script text output.

The **Resources** tab defines any resources available for the script to use. Dialogs are the main type of resource, but also supported are string resources which let you define strings in multiple languages.



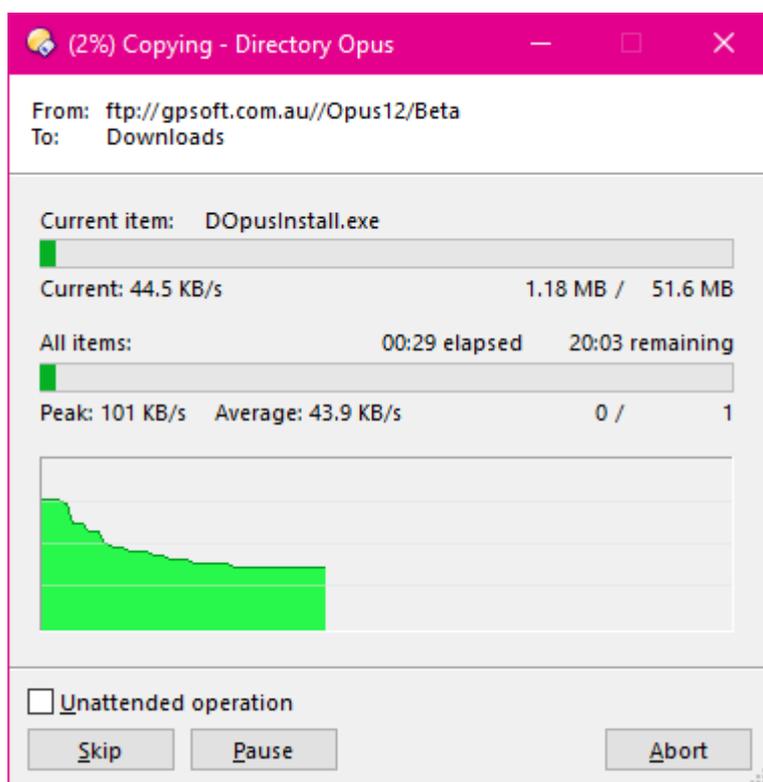
While you can hand-code dialog resources in XML if you wish, it's much easier to design them using the in-built dialog editor. To create a new dialog using the editor, click the **Dialogs** drop-down and choose the **New Dialog** command. You can also edit existing dialogs by selecting them from the drop-down list.

The dialog editor and how to use dialogs from scripts is explained in detail in the reference section.

## File copying

### *Transfer speed graph*

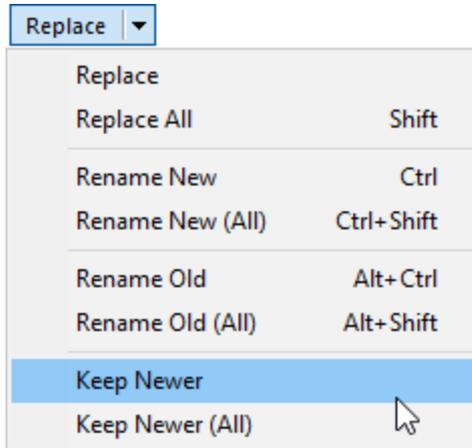
The Copy progress display now has a graph that indicates overall transfer speed for the duration of the operation.



The graph is “unitless”; each point on the graph represents the instantaneous average speed compared with the overall peak speed. If the peak speed increases dramatically the graph will dynamically rescale itself. It also scrolls when it reaches the end of the display – it’s not a progress bar, so the width of the graph isn’t related in any way to the duration of the operation.

## Replace file – Keep Newer

The *Replace File* dialog has new **Keep Newer** and **Keep Newer (All)** options.



This option will copy the incoming file over the top of the existing file, but only if the incoming file was modified more recently than the existing one. Otherwise, it leaves the existing file alone and doesn't copy the incoming file at all. This is done by comparing the modified timestamps of the two files. In the case of a tie, where both timestamps are identical, the incoming file is skipped.

## Commands relating to file copying

- **Copy WHENEXISTS=keepnewer:** This option used to be called **replacenewer**, but it has been renamed for clarity. The old name still works to maintain compatibility. It also now works to and from Zip files.
- **Copy AUTOSELECT:** This argument lets you override the *File Operations / Copy Options / Automatically select newly copied files* Preferences setting.
- **Copy IGNOREEXT:** This argument makes the copy command ignore file extensions when using a wildcard rename (e.g. **Copy PATTERN \*\_old AS \*\_new IGNOREEXT**). This lets you have the same command work on both files and folders using the same pattern.

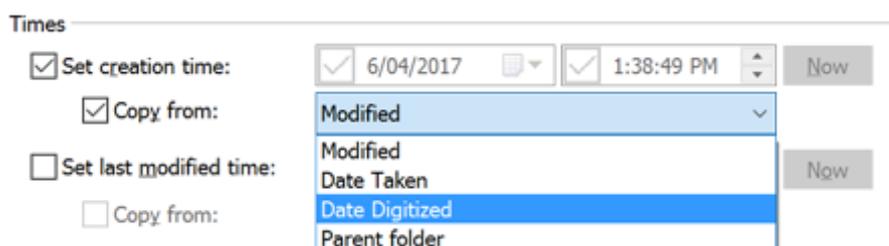
## Copy Queue scripting

The new **OnGetCopyQueueName** script event lets a script override the copy queue name for automatically-managed copy queues. This lets you implement your own copy queue logic if desired. The event function is passed a **GetCopyQueueNameData** object containing information about the copy operation as well as the default queue name. Your function can return a new queue name, or return **False** to accept the default name. If your function returns **True** the queue will be bypassed and the operation will run immediately.

## File operations

### Change Attributes & Times

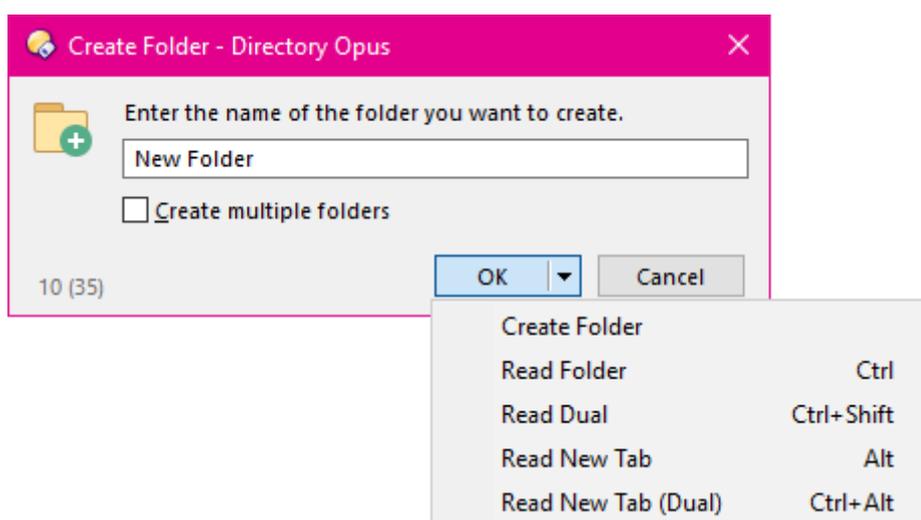
The *Change Attributes & Times* dialog can now copy timestamps from the created and modified timestamps, in the case of pictures from the date taken and date digitized EXIF fields, and also the timestamp from the parent folder.



The **SetAttr** command can also do this, using the **created**, **modified**, **taken**, **digitized** and **parent** keywords (e.g. **SetAttr MODIFIED=taken**).

### Create Folder

The *Create Folder* dialog now uses a drop-down on the **OK** button to select the various "read auto" options, rather than a series of checkboxes.



You can also press the **Return** key in conjunction with one of the listed hotkeys to activate that option automatically (e.g. **Ctrl + Shift + Return** to read the new folder in the dual display).

The *Create Folder* dialog now allows multiple folders to be created at the same level (with the **Create multiple folders** option on), using the | character to separate the folder names. For example, **Blah\one|two|three** would create **Blah\one**, **Blah\two** and **Blah\three**.

## Miscellaneous

- The new **notin** keyword for the **Select SOURCETODEST** and **Select DESTTOSOURCE** commands let you select all files on one side of a Lister that *aren't* on the other side.
- The new *File Operations / Options / Append " - Shortcut" when creating shortcuts* option in Preferences lets you control whether a suffix is automatically added to the names of shortcuts that Opus creates using the **Copy MAKELINK** command.

## Find

### Improvements to Simple Find mode

The *Simple* find mode has been redesigned to be less cluttered and easier to use.

The screenshot shows the 'Simple' find mode dialog. It features a 'Name matching' section with a text input containing 'horse donkey', a 'Date' dropdown set to 'Within', and three checkboxes: 'Wildcards' (unchecked), 'Any word' (checked), and 'Partial match' (checked). The 'Containing text' section has an empty text input, a 'Time' dropdown set to 'Ignore', and two checkboxes: 'Wildcards' (unchecked) and 'Case sensitive' (unchecked). The 'Filetype' section has a 'File Type Group' dropdown set to 'File Type Group' and a 'Size' dropdown set to 'Ignore'. A 'Reset' button is located at the bottom left.

The labels of each search criteria appear in **bold** when something is defined for them, making it a bit easier to see exactly what you’re searching for.

Under **Name matching** there are three options:

- **Wildcards**: Lets you enable or disable the use of wildcards (simple pattern matching only).
- **Any word**: Treats every word you enter as a separate search term. For example, in the above example we’re searching for a filename containing either “horse” or “donkey”. This saves you having to construct complicated *OR* wildcard patterns.
- **Partial match**: Enables partial matching; as shown above, “very big horse.jpg” would match because partial matching is enabled.

The **Filetype** option now has two drop-downs; the first lets you select the “type” of filetype (if that makes sense):

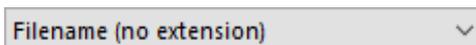
- **All files and folders:** Search for all files or folders matching the other conditions.
- **Files:** Search for only files.
- **Folders:** Search for only folders.
- **Junctions and Links:** Only search for junctions or soft-links.
- **File Type Group:** Search files belonging to a specified file type group. You can use the second drop-down control to choose the group to search for. In the above example, only files belonging to the *Images* group will be considered.
- **File Type:** Search files of a particular file type. Use the second drop-down to choose the file type to search for.

The *Type* clause in the *Advanced* find mode also has the same two drop-downs.

At the bottom of the simple find dialog there’s a new **Reset** button which makes it easy to quickly reset the find criteria to the defaults.

## Improvements to Duplicate File Finder

The Duplicate File Finder has a new **Filename (no extension)** search mode.



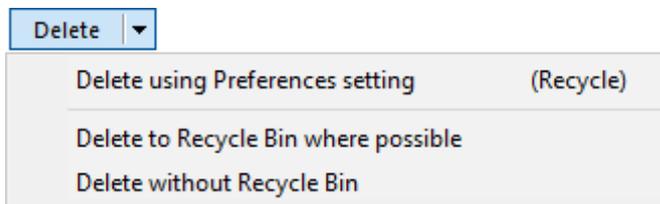
This will search for files with the same filename stem, ignoring their file extension. For example, "grandma.mp3" and "grandma.jpg" would match.

When searching by checksum, there's a new option to only calculate a hash for a percentage of the file. This lets you speed up the operation for large files at the expense of accuracy. Use the slider control to adjust the percentage from 1% to 100%.



Directory Opus 12 has added an optional MD5 checksums cache for large files located on NTFS partitions, and the Duplicate File Finder can make use of this. The checksum cache is only used if specifically requested; use the new **Use MD5 cache** option to turn it on. Note that cached checksums will only be updated if a file's last modified timestamp or size has changed.

When using the **Delete mode** option, you can now override the main Preferences Recycle Bin setting when deleting any selected duplicate files.

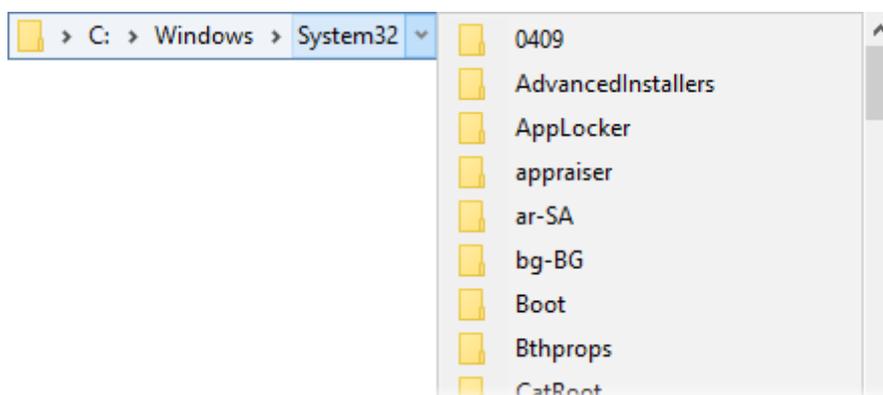


The **Delete** button in the bottom-right of the panel has a drop-down menu attached which lets you choose exactly how you want to delete the duplicates.

## Toolbars

### *Drop-down menu scrollbars*

When drop-down menus are too big for the screen they now use a scrollbar (and support the mouse wheel) rather than the up / down arrows that previous versions of Opus used.



### ***Drive button drive letters***

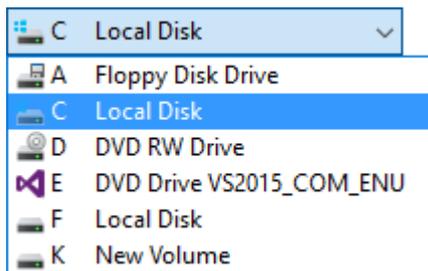
Drive Buttons on toolbars can now draw small drive letters over the icons themselves. This is done by default if the buttons are configured to show only icons and no labels, since otherwise you often end up with a line of identical icons with no way to distinguish them except by hovering the mouse over them to display their tooltip.



You can override this to force it on or off if you want, using the **iconletterson** and **iconlettersoff** keywords for the **Go DRIVEBUTTONS** command.

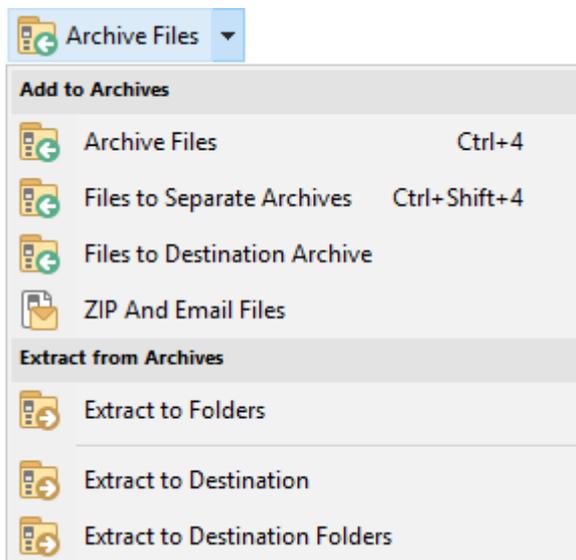
### ***Drives drop-down labels***

The Drives toolbar drop-down field can now display the drive's label (as well as its letter), if you add the keyword **labels** to the button's **Args** field.



### ***Drop-down menu labels***

Label fields in drop-down menus are now drawn in a bold font and left-aligned to stand out visually from the other items in the menu.



## ***Taskbar-style floating toolbars in Windows 10***

In Windows 10, *Taskbar-style* floating toolbars now mimic the look of the Windows 10 taskbar. The new style also looks better on Windows 8 and 8.1.



## ***Buttons can test recycle bin state***

The `@ifset` and `@icon` directives can use the new **RECYCLEBINEMPTY** argument to test if the recycle bin is empty or not. For example,

```
@ifset:RECYCLEBINEMPTY
@confirm The recycle bin is empty!
@ifset:else
Delete EMPTYRECYCLE
```

Toolbar buttons that use `@icon` with **RECYCLEBINEMPTY** will refresh themselves automatically when the recycle bin state changes (this lets you have a button whose icon reflects the state of the recycle bin).

## ***Dynamic buttons for thumbnail sizes***

The new command `Set THUMBNAILSIZE=list` generates dynamic buttons that allow you to switch thumbnail sizes. The sizes it generates buttons for start with 32px and then double until the maximum size is reached (and the maximum size is determined by your system DPI).

## ***DPI scaling for breadcrumbs path fields***

The various components of the breadcrumbs path field that can have their sizes configured are now scaled for high DPI displays. For example, `size=1+1+1+3+3` would equate to `size=2+2+2+6+6` on a 200% DPI system. You can prevent scaling if you want by specifying negative numbers for the size; e.g. `size=-1+-1+-1+-3+-3`.

## **Button context sensitivity**

It's always been possible to use the **@disablenosel** modifier to make your buttons disabled when no files are selected. The new modifier, **@hidenosel**, works just the same – except a button with this modifier will be hidden instead of disabled when no files are selected.

Both **@disablenosel** and **@hidenosel** can now check for files or folders matching a certain wildcard pattern being selected, rather than just “any files”. For example, **@hidenosel:type=\*.jpg** would hide the button unless at least one .jpg file was selected. Normal pattern matching is supported.

You can also test for files or folders specifically (otherwise the pattern can match both). For example, **@disablenosel:files,type=old\*** would disable the button unless files whose names begin with “old” are selected. **@hidenosel:dirs** would hide the button if no folders were selected.

## **Condition testing**

The **@ifset:** directive allows you to test if a **Set** command condition is active (e.g. **@ifset:TREE**). While the **Set** command is the main command that uses context sensitive commands, a few other commands do too in some cases (e.g. the **Show THUMBNAILSIZE** command will “highlight” a button when the thumbnail size is set to the size specified in the command).

You can now use the more generalised **@if:** directive to test for those commands (it also works with **Set** as well). For example, **@if:Set TREE** is equivalent to **@ifset:TREE**. This will be mostly useful in the viewer now that the viewer toolbar is configurable. For example, a button in the viewer toolbar can test if the zoom mode is set to “fit” with the directive **@if:Show VIEWERCMD=zoom,fit**.

The script form of this test, **Command.IsSet()**, now takes an optional second argument which specifies the command to test against (and this defaults to “Set” if it's not provided).

## **FAYT / Filter Bar**

The Preferences pages relating to the *Find-As-You-Type field* (FAYT) and *Filter Bar* have been reorganised.

## ***FAYT and Filter Bar Keys***

This page contains all settings relating to activation of the *FAYT* and *Filter Bar* via the keyboard. The new option **Default mode** lets you choose whether the *Filter Bar* or a specific *FAYT* mode is activated by default; that is, when you press a key that hasn't been assigned to a specific mode.

## ***FAYT and Filter Bar Options***

This page contains options relating to the behaviour of the *FAYT* and *Filter Bar*. The new options on this page are:

## **Find-As-You-Type Filter Mode**

- **Allow return key to open selected item:** In the *FAYTFilter* mode, pressing the **Return** key will open the item with focus rather than simply closing the *FAYT* field.
- **Select first matching item:** In the *FAYTFilter* mode, the first item that matches the filter will be automatically selected.

## **Filter Bar**

- **Match any word:** In the *Filter Bar*, the **Match Any word** option treats all words you enter as separate patterns. For example, you can type “moo cow” and it would automatically match a file called “moo” or a file called “cow”. This saves you having to build up complex *OR* wildcards (the equivalent wildcard would be “(moo|cow)”).
- **Use regular expression:** When turned on, the pattern you enter into the *Filter Bar* will be treated as a regular expression rather than a simple wildcard.

## **Status Bar**

### ***Individual file information***

The status bar can now display information about the most recently selected file, using the new **{sel:x}** code.

The **x** in the code specifies the information to display; valid keywords are:

- **name:** Name of the file or folder.
- **size:** File size. Follow this keyword with **b** or **k** to specify the units as *bytes* or *KB* (otherwise the units are automatically chosen).
- **create:** Creation date stamp. Follow this keyword with **d** or **t** to specify *date* or *time* (otherwise both are shown).

- **write**: Last write (modification) date stamp. Follow this keyword with **d** or **t** to specify *date* or *time* (otherwise both are shown).
- **access**: Last access date stamp.
- **attr**: File or folder attributes.
- **desc**: Description string (the same as is displayed in the *Description* column).
- **path**: Full path of the file or folder.
- **index**: Index in the file display.

For example, `{sel:sizek} {sel:desc}`.

### **Show everything mode**

The new `{hse}` code is used with the new *Show Everything* mode. When *Show Everything* is enabled, this code will display the text **Everything**. When *Show Everything* is disabled, this code will display the text **Filtering** (to indicate that filters may be in effect).

On the default status bar, `{hse}` is wrapped inside a `{h!}...{h!}` clause, which will cause it to be hidden unless *Show Everything* is turned on. That is, you may see the text **Everything** but not the text **Filtering**.

### **DPI scaling for status bar graphs and parts**

Status bar graph width numbers are now scaled to DPI by default. You can use negative numbers to prevent scaling from happening. (i.e. Just put a - before the number you want.)

Status bar `{widthXX}` is similar: It now scales the pixels to the current DPI, and can be made to use an absolute value using negative numbers. e.g. `{width-10}` will be 10 pixels in all DPIs, while `{width10}` will be 10 pixels at 100% DPI and 20 pixels at 200% DPI.

### **Miscellaneous things**

Well there had to be a miscellaneous section didn't there! Here are all the minor changes (in no particular order) that don't fit into any of the other categories in this document.

- In *Preferences / Layouts and Styles / Layouts*, Lister layouts can now be organised into folders.
- Windows 8 and above add a significant delay before launching applications at startup. The new option *Preferences / Launching Opus / Startup* now indicates if this delay is in effect, and has buttons to remove or restore it.

Changing this option affects all programs launched via the Start Menu startup folder, not

just Opus, and the setting is in the registry rather than the Directory Opus configuration (so you'll need to change it on individual machines if desired). It's a Windows setting, not an Opus one - we just provide a convenient way to change it in Preferences.

For those who are interested, the option has the effect of setting `KEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Serialize:Startupdelayinmsec (DWORD) = 0` to remove the delay, or deleting the value to restore it.

- The **Set LISTERTITLE** command (and the *Display / Options / Custom title* option in Preferences) can now use the code **%G** to display the target of the current folder if it's a junction or soft-link.
- There is a new optional checksum cache for large files located on NTFS partitions. The checksum cache is only used if specifically requested - the **GetSizes** command has a new **USEHASHCACHE** argument, and **Clipboard COPYNAMES** has a new **hashcache** parameter to enable it. The Duplicate File Finder can also use it, either by setting the option in the user interface, or by specifying the **MD5=cache** argument for the **Find** command. Note that cached checksums will only be updated if a file's last modified timestamp or size has changed.
- The *FTP Address Book* now allows a private key file to be specified explicitly for SSH connections (previously you had to use the external Pageant tool to use a key file for authentication instead of a password).
- The Lister now automatically moves out of the current drive if it detects the drive has been removed. Files on the drive which are open in the viewer pane will be closed as well.
- The *Folder Tree / Contents* page in Preferences has an option to hide the Homegroup from the folder tree, and separate options for *Hidden* and *Protected operating system folders*.
- Added option to the *Folders / Virtual Folders* page in Preferences to change what happens when you do a Windows Search from the Desktop virtual folder. By default, Opus now only searches your personal Desktop folder (**/desktopdir**) and, if configured to appear, the shared desktop folder (**/commondesktopdir**). If the setting is changed (or if native display of the Desktop folder is off), you will return to the old behavior: The search will include the profile and shared Desktop folders and also the Documents folder, and potentially others. (It is up to Windows exactly which folders are searched, and you'll see similar from Explorer.)
- "Opus" audio format (no relation :) tags can be read by the AudioTags plugin. (Read-only for file display columns, rename and scripting. Not supported in the metadata editor at time of writing.)
- Added support for MP4 and basic MKV metadata under Windows 7 and up.
- A version of the *FileTypeDiag* tool has been built into Opus. (**File > Diagnostic**, from the *File Types* dialog.)
- Updated to 7z.dll 15.12. RAR unpacking is done using 7z by default again, since it now supports RAR 5 and has more accurate timestamps, but you can switch back to UnRAR.dll if preferred.

- Various Preferences settings which used to allow a user command to be chosen from a drop-down list (e.g. the desktop double-click settings) now allow any single-line command to be typed. This means you no longer need to set up a user command just to run a single-line internal or script command.
- Opus now has better support for dark color schemes:
  - On the *Display / Colors and Fonts* page in Preferences, the *Pane borders* section has new options **Use for lister column headers** and **Use for lister scrollbars**. (The scrollbar option is only available if Windows visual styles are active, and is also disabled if WindowBlinds is detected).
  - Labels for disabled toolbar buttons are now faded instead of being drawn using the system “gray” color, so they'll look better with non-standard colors, especially inverted or very dark colors.
- The new folder alias **/scripts** takes you to the script add-ins folder (equivalent of **/dopusdata/Script AddIns**).
- Zip AES encryption now uses the WinZip standard instead of the PKWare one. This seems to work with more applications, including popular ones like WinRar and 7-Zip which did not work with Opus encrypted archives in the past.
- The animated GIF plugin supports image flipping (mirroring).
- The *Miscellaneous / Advanced* section in Preferences has a new *clipboard\_image\_paste\_dpi* option. When turned on, pasting clipboard image data to a file (by pressing **Ctrl+V** in a Lister) will automatically scale it to compensate for the system DPI. A similar option added to the dialog displayed by the **Clipboard PASTE AS=ask command**.
- The *Miscellaneous / Advanced: gloss\_and\_gradients* Preferences setting now has an *Automatic* option which is selected by default. When set to *Automatic*, a flat and simple look is chosen if on Windows 8 or above, and a shiny look with gradients is chosen on Windows 7 and below.
- You can now disable internal image viewers via *Preferences / Miscellaneous / Advanced: viewer\_disable\_internal*. For example, this allows you to divert the TIFF viewer to a third-party ActiveX control which handles multi-page TIFFs.
- The font used in the metadata control (in the Lister and the Viewer) is now configurable via **Preferences / Display / Colors & Fonts**.
- Added **Preferences / Folder Tabs / Options / Click selected tab to go to previous one** option.
- In Windows 10, the new Quick Access folder can be shown in the folder tree by turning on the *Preferences / Folder Tree / Contents/ Quick Access* option. The new **/quickaccess** path alias can be used to navigate to the Quick Access folder from a button (e.g. **Go /quickaccess**).
- Added **mycomputerfull** argument for breadcrumbs pathfields, to enable the display of folders in the drop-down for the Computer folder.
- You can now block the Movie plugin from handling certain extensions (even if their registry information specifies they are video types) by adding them to the list of extensions with a ~ before them. For example **.avi,.mpg,~.wmv** would tell the plugin to handle **.avi** and **.mpg** but ignore **.wmv** files.

- The Navigation Lock algorithm has been redesigned to hopefully address various complaints about the old algorithm (in terms of how and when it goes out of sync). It should be much better now at recovering after having gone out of sync. The "slave tabs" system now uses the same algorithm as Navigation Lock in terms of staying in and going out of sync (i.e. when it goes out of sync it will show a warning message rather than changing the tab to show the same folder as the other side). The term "Slave Tab" has been replaced with "Navigation Lock" in the user interface to avoid confusion (since there's really only one system now instead of two completely separate ones).
- The Scripts Preferences page has been redesigned:
  - It now uses a multicolumn list to display the list of scripts. Columns can be added/removed by right-clicking the column header.
  - Information about the selected script is displayed at the bottom of the list if not shown in a column.
  - If a script supports configuration clicking its name now opens the configuration dialog.
  - The list now remembers the previously selected script and reselects it next time it opens.
  - Scripts that fail to parse are no longer removed from the list; instead they will display the failure reason in the *Status* column.
  - Scripts that you disable are now **completely** disabled - no code from them is run at all. (previously, the **OnInit** function would run and commands/columns would still be queried).
  - The new **Import** command can be used to import a script (as well as drag-and-drop like before).
  - The new **Delete** command can be used to remove a script. It deletes the script file and removes the record of any configuration for it (so that if later on you add a new script with the same name, it won't inherit the old script's configuration).
  - The new **Disable All Scripts** command lets you disable all script add-ins at once - individual enable/disable states will be preserved. The new **Set SCRIPTDISABLE** command lets you toggle this from a button or hotkey.
  - Scripts can now specify a group in their **OnInit** function (using the **ScriptInitData.group** property). If any scripts specify a group this will be reflected in the Preferences list.
  - Scripts can also specify a group for their configuration values using the new **ScriptInitData.config\_groups** property (this accepts a **Map** in the same way as the **config\_desc** property).
- Added script support for Windows shell properties, making it easy for a script to enumerate properties in the system and retrieve the properties for a file.

## Script Miscellaneous

### *Shell Properties*

Opus 12 has new script support for Windows shell properties, making it easy for a script to enumerate properties in the system and retrieve the properties for a file.

- Use **FSUtil.GetShellPropertyList** to retrieve a list of properties (optionally matching a wildcard pattern).
- Use **FSUtil.GetShellProperty** to get the value of one or more properties for a file.
- Use **Item.ShellProp** to get the value of a single property for the item.
- Additionally, the new **ScriptColumn.userdata** property has been added which lets a column specify an item of data that's passed back to column handlers. This is used below.

Below is an example script that adds columns to Opus that show the value of shell properties for DWG (AutoCAD) files added by a third party tool.

```
Function OnInit(initData)
    initData.name = "DWG Columns"
    initData.desc = "Adds DWG Columns from the JTB World
extension"
    initData.copyright = "(c) 2016 jpotter"
    initData.version = "1.0"
    initData.default_enable = true
    initData.min_version = "12.0.8"

    Dim props, prop, col
    Set props = DOpus.FSUtil.GetShellPropertyList("dwg.*", "r")

    for each prop in props

        Set col = initData.AddColumn
        col.name = prop.raw_name
        col.method = "OnDWGColumn"
        col.label = prop.display_name
        col.justify = "left"
        col.autogroup = true
        col.userdata = prop.pkey
    next

End Function

Function OnDWGColumn(scriptColData)
    scriptColData.value =
scriptColData.item.shellprop(scriptColData.userdata)
End Function
```

## **Viewer Events**

There is a new **OnViewerEvent** script event, which is called when certain events occur in a standalone image viewer.

- The event is passed a **ViewerEventData** object, with properties **viewer**, **event** and **item** (if applicable).
- The events currently defined are **create**, **setfocus**, **killfocus**, **destroy** and **load**.

One possible use would be a script that automatically displays a floating toolbar whenever a standalone viewer is active, and hides it again when the window goes inactive or closes.

## ***Copy Queue scripting***

The new **OnGetCopyQueueName** script event lets a script override the copy queue name for automatically-managed copy queues. This lets you implement your own copy queue logic if desired.

The event function is passed a **GetCopyQueueNameData** object containing information about the copy operation as well as the default queue name. Your function can return a new queue name, or return **False** to accept the default name. If your function returns **True** the queue will be bypassed and the operation will run immediately.

## Reference

### Commands

The following is a summary of the changes to the internal command set.

### CLI

- **COMMANDAPPEND**: This argument has been removed (the old toolbar *Command* field has been removed).
- **COMMANDSET**: This argument has been removed (the old toolbar *Command* field has been removed).
- The **CLI** command lets you use **noselect**: to set the text in the field and begin typing at the end of it, instead of typing over the top of it. For example, you could bind a hotkey to **CLI QUICKGO=noselect:C:\** and then push it and start typing the name of a folder below C:\.

### Clipboard

- **COPYNAMES**: This argument has a new **hashcache** parameter to enable use of the checksum cache when copying checksums to the clipboard.
- **PASTE**: An option to scale pasted images to compensate for the system DPI has been added to the dialog displayed by the **Clipboard PASTE AS=ask** command.

### Copy

- **AUTOSELECT**: Lets you override the *Preferences / File Operations / Copy Options / Automatically select newly copied files* option.
- **IGNOREEXT**: Makes the function ignore file extensions when copying with a wildcard rename (e.g. so a button can work on both files and folders using the same wildcard pattern).
- **WHENEXISTS**: The **replacenewer** keyword has been renamed to **keepnewer** to make it more obvious what it does.

### Delete

- **FAILNOTEMPTY**: Fail when attempting to delete a non-empty folder (must be combined with **NORECYCLE**).
- **SKIPNOTEMPTY**: Skip over without error when attempting to delete a non-empty folder (must be combined with **NORECYCLE**).

## Find

- **GOOGLE:** This argument has been removed (Google Desktop Search is no longer supported).
- **MD5:** This argument now lets you specify a percentage of the file's checksum to calculate, e.g. **MD5=50** for 50% accuracy. You can also enable the checksum cache, e.g. **MD5=cache,25**.

## GetSizes

- **USEHASHCACHE:** Enables the use of the checksum cache when calculating checksums for display in the file display.

## Go

- **DRIVEBUTTONS:** Use the keywords **iconletterson** and **iconlettersoff** to control whether drive letters are displayed in the icons for each drive.
- **FOLDERCONTENT:** Now supports non-filesystem folders. For example, in Windows 10 you can use **Go /quickaccess FOLDERCONTENT** to show the Quick Access folder in a pop-out menu
- **TABCLOSEALL:** Added the **dest** keyword, which lets you close tabs in the destination file display in a dual-display Lister.
- **TABCOLOR:** Lets you change the color of the current tab (e.g. **Go TABCOLOR #ff8000** or **Go TABCOLOR 255,127,0**). Use **Go TABCOLOR reset** to reset the tab color.
- **TABGROUPLIST:** This argument now works in conjunction with the **USEQUALKEYS** and **KEYARGS** arguments.
- **TABLINK:** The **slave** keyword has been renamed to **navlock**. The new **reset** keyword lets you reset the sync position when the current tab is linked in **navlock** mode with another.

## Help

- **REF:** You can now use the **Help** command to directly open the page for any internal command. For example: **Help REF=cmd\_CreateFolder**

## Image

- **NOUSEIMAGEDATA:** When used with the **CONVERT** argument (in the standalone image viewer), overrides the **@useimagedata** command modifier and makes the image converter load the image from disk rather than obtaining it from the viewer.

## Prefs

- **PAGE:** The keywords **viewer** and **slideshow** have been renamed **viewer1** and **viewer2** (the old ones still work for compatibility). Additional page keywords are **viewer3** and **assignedlabels**.

## Print

- **FORMAT:** When used with the **Print FOLDER** command, this argument accepts the following special keywords, as well as the name of a favorite format:
  - **!factory:** Reset to factory defaults.
  - **!user:** Reset to the user default (in Opus 11 this was **!custom**, which still works for compatibility).
  - **!default:** Resets to Folder Type format applicable to current folder.
  - **!folder:** Resets to the format for the folder that a brand new window would use.
  - **!current:** Uses the current format shown in the Lister.

## Properties

- **ADDLABEL:** In conjunction with the **SETLABEL** argument, this lets you add one or more labels without clearing any existing ones. If the optional keyword **ctrl** is specified, the labels will only be added if the **Control** key is held down (otherwise they will replace existing labels as normal).
- **LABELCATEGORY:** When used on a command that generates a list of labels (e.g. **Properties SETLABEL** or **Properties SETLABEL !menu**) this argument lets you filter the generated list by category. It accepts one or more comma-separated wildcard strings which let you match the name of categories to include. The specified categories will also be used when resetting labels using the **Properties SETLABEL !reset** command - if **LABELCATEGORY** is used as well, only labels in the specified categories will be cleared. You can match uncategorized labels using the pattern **~\*** (which means "not anything").
- **SETLABEL:** This argument now accepts multiple values, to assign more than one label to a file or folder. Label names must be comma-separated. Commas and back-slashes in label names must be escaped with a back-slash. There's also a new **Properties SETLABEL=!submenu2** mode which puts uncategorized labels in an *Uncategorized* category.

When used without an argument, to generate a dynamic list of labels, it now groups labels by their categories, while still producing a flat list. Similarly, **SETLABEL=!menu** does the same inside a sub-menu. You can use **SETLABEL=!nogroup** or **SETLABEL=!menu,!nogroup** to intermix the categories as before, if you wish.

- **SETLABELTOGGLE:** This argument now accepts the optional keyword **shift**, which makes it only take effect if the **Shift** key is held down.
- **SETWALLPAPER:** This argument now supports multi-monitor "Span" mode on Windows 8 and above (e.g. **Properties SETWALLPAPER=span**).

The **Properties SETLABEL** command supports embedded commands when it's used to generate dynamic buttons.

## Rename

- **FILEINFO**: This argument has been removed (file information is now generated automatically when needed).
- **IGNOREEXT**: Lets you control the *Rename* dialog's **Ignore extension** option when automating the **Rename** command.
- **MACRO**: Lets you specify a rename macro operation string when automating the **Rename** command (e.g. **Rename MACRO R0-6/L0+Final**).
- **NOFILEINFO**: Disable automatic file information inserts.
- **PRESET**: With the **!list** keyword (which generates a list of your rename presets), the additional keywords **favesonly** (only displays presets marked as favorites), **nofaves** (only displays presets not marked as favorites) and **nogroup** (does not group favorite and non-favorite presets separately) can also be used (e.g. **Rename PRESET=!list,favesonly**).
- **SCRIPTARG**: Used to pass the value of custom fields to a rename script (see the section on rename scripting for more information).
- **SHOWPREVIEW**: This argument has been removed (the rename preview is now always displayed).

## Select

- **DETTOSOURCE**: This argument now accepts the **notin** keyword which lets you select all files in the source file display that aren't in the destination. This argument now accepts the **in** keyword which lets you select all files in the source that are in the destination.
- **DESELECTOTHERTYPE**: When used with the **TYPE** argument to restrict a selection to either files or folders (or with the **ALLFILES** and **ALLDIRS** arguments), **DESELECTOTHERTYPE** causes all items of the other type to be deselected.
- **DESELECTNOMATCH**: This argument now works when selecting with a filter (i.e. when using the **FILTER** argument)
- **SOURCETODEST**: This argument now accepts the **notin** keyword which lets you select all files in the destination file display that aren't in the source. This argument now accepts the **in** keyword which lets you select all files in the destination that are in the source.

## Set

- **COLUMNSADD**: When specifying the size for fields added with this argument, you can now use **a** for *Auto*, **f** for *Fill*, **e** for *Expand* and **c** for *Collapse*. For example, to add the *picture width* field with its width set to auto, you might use **Set COLUMNSADD picwidth(\*,a)**.

You can also specify the maximum width with an additional parameter; e.g. to add the *picture width* field with its width set to auto and maximum width set to fill, you might use **Set COLUMNSADD picwidth(\*,a,f)**.

**Set COLUMNSADD** is also capable of changing the auto-size and max-size properties

of existing columns without changing their positions. For example, **Set COLUMNSADD=name(!,a,0)** will set the Name column to auto-size with no maximum width.

**Set COLUMNSADD** and similar now allow column positions to be specified relative to existing columns. For example, to add the Status Icons column after the Name column: **Set COLUMNSADD=Status(1+Name)**. "0+Name" is the position of the Name column. "1+Name" the next position. "1-Name" the previous position.

- **COLUMNSTOGGLE**: See **COLUMNSADD** above.
- **COMBINESINGLEGROUPS**: Lets you control the state of the *When grouped, combine groups with only one member into the "Other" group* option for the current file display.
- **FORMAT**: This argument accepts the following special keywords, as well as the name of a favorite format:
  - **!factory**: Reset to factory defaults.
  - **!user**: Reset to the user default (in Opus 11 this was **!custom**, which still works for compatibility).
  - **!default**: Resets to Folder Type format applicable to current folder.
  - **!folder**: Resets to the format for the folder that a brand new window would use.
- **GLOBALHIDEFILENAME**: The supplied pattern can be prefixed with **regex:** to specify the pattern is a regular expression.
- **GLOBALHIDEFOLDERS**: The supplied pattern can be prefixed with **regex:** to specify the pattern is a regular expression.
- **GRIDLINESH**: Control the display of horizontal grid lines in the current file display (this argument used to be called **GRIDLINES**).
- **GRIDLINESV**: Control the display of vertical grid lines in the current file display.
- **GROUPCOLLAPSE**: Controls the *Collapsed* option for grouping in the current file display.
- **HIDEFILTERFILENAME**: The supplied pattern can be prefixed with **regex:** to specify the pattern is a regular expression.
- **HIDEFILTERFOLDERS**: The supplied pattern can be prefixed with **regex:** to specify the pattern is a regular expression.
- **ICONMODESORTHEADER**: Control the visibility of column headers in the icon modes (large icons, thumbnails, list, etc).
- **LISTERTITLE**: The title string can now use **%G** to display the target of the current folder if it's a junction or soft-link.
- **MANUALSORTRESET**: Reset the current manual sort arrangement in the file display and revert to automatic sorting.
- **MANUALSORT**: Enable or disable manual sorting in the current file display.
- **MANUALSORTSAVE**: Save the current manual sort arrangement (if the folder type supports saving the manual sort order).

- **RECYCLEBINEMPTY**: Only used with the **@ifset** and **@icon** modifiers, tests if the recycle bin is empty.
- **SAVEFORMAT**: Save the folder format for the current folder without having to go through the *Folder Options* dialog.
- **SCRIPTDISABLE**: Toggle the global "disable all script add-ins" option on or off.
- **SHOWEVERYTHING**: Toggle the *Show Everything* mode on or off in the current file display.
- **SHOWFILTERFILENAME**: The supplied pattern can be prefixed with **regex:** to specify the pattern is a regular expression.
- **SHOWFILTERFOLDERS**: The supplied pattern can be prefixed with **regex:** to specify the pattern is a regular expression.
- **UTILITY**: Accepts the new keywords **expand** and **noexpand**. Functions which toggle the utility panel will now, by default, expand it when turning the panel on, if it was saved in a collapsed state. You can add the **noexpand** keyword to prevent this, e.g. **Set UTILITY=find,toggle,noexpand** would open a collapsed Find panel if that was how it was last saved. You can also explicitly specify **expand** if you would like expansion to take priority over closing a panel if the panel is currently in a shrunken state, e.g. **Set UTILITY=find,toggle,expand** (you'd probably want to add **focus** as well). When neither **expand** nor **noexpand** is explicitly specified, the default is to expand when showing the panel (including when changing which panel is displayed) and to not expand when hiding the panel.

## SetAttr

- **CREATED**: Now accepts the keywords **modified**, **taken**, **digitized** and **parent** to copy the date from another field or the parent folder.
- **MODIFIED**: Now accepts the keywords **created**, **taken**, **digitized** and **parent** to copy the date from another field or the parent folder.

## Show

- **THUMBNAILSIZE**: This argument accepts the special keyword **list** which makes it generate a list of thumbnail sizes automatically (based on your system DPI setting).
- **VIEWERCMD**: This argument is only used within the standalone viewer. Keywords are:
  - **alpha**: Toggle the *Hide Alpha Channel* option on and off.
  - **close**: Close the viewer.
  - **copy**: Copy the current selection to the clipboard.
  - **copyto**: Copy the currently viewed file to another folder.
  - **crop**: Crop the image to the current selection.
  - **cut**: Cut the currently viewed file to the clipboard.

- **delete**: Delete the currently viewed file.
- **first**: Move to the first file in the list.
- **flip**: Flip the current image (use either **flip,horiz** or **flip,vert**).
- **fullscreen**: Toggle full-screen mode on and off.
- **gamma**: Adjust the gamma of the image display (use **gamma,+<val>** or **gamma,-<val>** for relative adjustments, **gamma,<val>** for an absolute positive value, **gamma,0-<val>** for an absolute negative value, and **gamma,reset** to reset the gamma to the default).
- **goto**: Go to a specified image in the list (use **goto,<index>** where *<index>* is the 0-based offset of the file from the beginning of the list).
- **help**: Display help on the viewer.
- **hex**: Toggle the display in and out of hex mode.
- **info**: Toggle the *Show Information* overlay on and off.
- **last**: Move to the last file in the list.
- **mark**: Control image marking. This keyword has many different uses:
  - **mark**: (no sub-keyword) Toggle mark state of current image.
  - **mark,toggle**: Toggle mark state of current image.
  - **mark,on**: Mark the current image.
  - **mark,off**: Unmark the current image.
  - **mark,view**: Toggle the marked panel on and off.
  - **mark,browse**: Browse marked pictures.
  - **mark,clear**: Clear all marks.
  - **mark,exchange**: Exchange the current image for the previously marked image.
  - **mark,first**: Jump to the first marked image.
  - **mark,last**: Jump to the last marked image.
  - **mark,prev**: Jump to the previously marked image.
  - **mark,next**: Jump to the next marked image.
  - **mark,return**: Return from a jump to the image you were previously viewing.
  - The **nohighlight** keyword can also be added, which stops a toolbar button from displaying as highlighted when the condition in it is true (e.g. **Set VIEWERCMD=mark,toggle,nohighlight**).
- **meta**: Toggle the meta pane on and off, and control its width.
  - **on**: Turn the meta pane on.
  - **off**: Turn the meta pane off.
  - **toggle**: Toggle the meta pane state.

- **grow**: Grow the viewer window to accommodate the meta pane if possible.
  - `<width>`: Specify the width of the meta pane (in pixels).
  - **nofocus**: Does not give focus to the meta pane when it opens.
- **minwidth**: Save the width of the current viewer window as the new minimum width.
- **moveto**: Move the currently viewed file to a new folder.
- **next**: Move to the next file in the list.
- **nextlist**: Generate a list of the next files in the list.
- **notfullscreen**: Button will only be visible when the viewer is not in full-screen mode (combine with other keywords, e.g. **Show VIEWERCMD=delete,notfullscreen**).
- **onlyfullscreen**: Button will only be visible when the viewer is in full-screen mode (combine with other keywords, e.g. **Show VIEWERCMD=close,onlyfullscreen**).
- **open**: Open a new file. Can be optionally given a filename to open (e.g. **Show VIEWERCMD="open,c:\my pictures\image.jpg"**)
- **pluginabout**: Display the about dialog for the current viewer plugin.
- **plugincfg**: Display the configuration dialog for the current viewer plugin.
- **plugincmd**: Trigger a command provided by the current viewer plugin (given as an 0-based index).
- **plugincmds**: Generate a list of commands provided by the current viewer plugin.
- **prev**: Move to the previous file in the list.
- **prevlist**: Generate a list of the previous files in the list.
- **print**: Print the currently viewed file.
- **refresh**: Refresh (reload) the currently displayed file.
- **reselect**: Reselect the previous selection.
- **restore**: Undo the previous crop operation.
- **rotate**: Rotate the current image (use **rotate,+<val>** or **rotate,-<val>** for relative rotations, **rotate,<val>** for an absolute rotation and **rotate,reset** to reset the rotation).
- **save**: Save changes you have made to the current image. Add the **quiet** option to replace the existing file silently (e.g. **Show VIEWERCMD save,quiet**)
- **saveas**: Save the current image as a new file. Can be optionally given a filename to save (e.g. **Show VIEWERCMD="saveas,c:\my pictures\image.jpg"**)
- **selectall**: Select the entire image.
- **selectfile**: Selects the viewer's current file in the folder tab it came from.
- **scroll**: Scroll the currently displayed image. You must specify either **horiz** or **vert** to indicate the dimension you want to scroll, and then another keyword (comma-separated) to indicate how far to scroll.

- **horiz**: Scroll horizontally.
- **vert**: Scroll vertically.
- **up**: Scroll up or left.
- **down**: Scroll down or right.
- **pageup**: Scroll up (or left) a page.
- **pagedown**: Scroll down (or right) a page.
- **top**: Scroll to the top (or far left).
- **bottom**: Scroll to the bottom (or far right).
- **shortcutbar**: Toggle the display of the shortcut bar on and off.
- **slideshow**: Toggle slideshow mode on and off.
- **statusbar**: Toggle the display of the status bar on and off.
- **toolbar**: Toggle display of the toolbar on and off.
- **wallpaper**: Set the current image as your desktop wallpaper (use the keywords **center**, **tile**, **stretch**, **fit** and **fill** to specify the wallpaper mode, e.g. **Show VIEWERCMD=wallpaper,fit**).
- **zoom**: Adjust the zoom level of the current image:
  - **zoom,+**: Zoom in.
  - **zoom,-**: Zoom out.
  - **zoom,+<val>**: Zoom in a specified amount.
  - **zoom,-<val>**: Zoom out a specified amount.
  - **zoom,<val>**: Set zoom to the specified level.
  - **zoom,fit**: Set zoom mode to “fit to page”.
  - **zoom,grow**: Set zoom mode to “grow to page”.
  - **zoom,tile**: Set zoom mode to “tile”.
  - **zoom,reset**: Reset the zoom level.

The **Show** command supports embedded commands, which let you open the viewer and run a command in the context of the viewer from the one button. For example, to show the selected file and automatically turn on the metadata pane in the viewer, you might use a command like:

Show

[Show VIEWERCMD=meta]

The **@if** directive can test the state of various **Show VIEWERCMD** options when used within the viewer. For example, the following function would toggle between 100% zoom and Grow To Page modes:

```
@if:Show VIEWERCMD=zoom,reset
Show VIEWERCMD=zoom,grow
@if:else
Show VIEWERCMD=zoom,reset
```

### Command control codes and modifiers

- Added **{dpi}** control code so you can use DPI-sensitive values with simple commands. This can be useful if you have buttons which specify column or window sizes and you want consistent results from the same button in different DPIs.
  - **{dpi}** on its own will report the current DPI. **96** at standard DPI, **192** at 200% DPI, and so on.
  - **{dpi|%}** will report the current DPI scale factor. **100** at standard DPI, **200** at 200% DPI, and so on.
  - **{dpi|<number>}** will convert a standard 96 DPI pixel width to the current DPI. For example, if you are at 200% DPI, **{dpi|25}** will output **50**.
  - **{dpi|/<number>}** will convert from the current DPI back to standard 96 DPI pixels. For example, if you are at 200% DPI, **{dpi|/50}** will output **25**.

Example use in a command: **Set LISTERSIZE {dpi|640},{dpi|480}**

- The new **@if** directive allows commands other than **Set** to be tested, e.g. **@if:Show VIEWERCMD=zoom,fit** tests whether the zoom mode is set to “fit to page” in the current viewer.
- **@if**, **@ifset** and **@icon** directives can now test if the recycle bin is empty, with the **RECYCLEBINEMPTY** argument (e.g. **@ifset:RECYCLEBINEMPTY**). Toolbar buttons that use **@icon** with **RECYCLEBINEMPTY** refresh automatically when the recycle bin state changes.
- The new **@hidenosel** modifier works like **@disablenosel**, except the button is hidden instead of disabled if no files are selected.
- Both **@hidenosel** and **@disablenosel** can now check for specific file types being selected; for example, **@disablenosel:type=\*.jpg** would disable the button unless at least one .jpg file was selected. Standard pattern matching is supported. You can also test for files (e.g. **@hidenosel:files**) or folders (e.g. **@hidenosel:dirs.,type=\*\_tmp**).
- Added **sep=** as a modifier for the **{allfilepath}** etc. codes, allowing you to change the separator character from the default space. E.g. **{allfilepath\$|sep=,}** would product a comma-separated list of filepaths.
- Date and time codes, when renaming files or running commands, can now specify the "system invariant locale" instead of local language settings. This locale is similar to North American and gives the same results on all machines and in all languages. For example,

running `{date|Idd-MMM} {time|Ihh:mm tt}` on June 23rd at 10:52 PM will always output "23-Jun 10:52 PM", always using English month names and AM/PM strings.

## Script objects

The following is a summary of the changes to scripting objects.

### CloseListerData

The **CloseListerData** object has the following new properties:

- **prevent\_save**: Set this to **True** to prevent the closing Lister from being saved as the new default Lister.

You can use this to revert to the old behavior of not auto-saving Listers that close when the Synchronize or Duplicate File Finder panels are showing:

*Script Type: VBScript*

```
Function OnCloseLister(closeListerData)
    If closeListerData.lister.utilpane = 1 Then
        If closeListerData.lister.utilpage = "sync" Or
closeListerData.lister.utilpage = "dupe" Then
            closeListerData.prevent_save = True
        End If
    End If
End Function
```

### Column

The **Column** object has the following new properties:

- **autosize**: **True** if the column width is set to *auto*.
- **collapse**: **True** if the column width is set to *collapse*.
- **expand**: **True** if the column width is set to *expand*.
- **fill**: **True** if the column width is set to *fill*.
- **header**: provides the name of the column as displayed in the Lister column header.
- **label**: provides the name of the column as displayed in the Columns tab in the Folder Options dialog.
- **max**: Maximum width of the column, or “fill” if the maximum is set to *fill*.

## Command

- **IsSet:** The **IsSet()** method now takes an optional second argument specifying the command to test against (defaults to "Set" if not provided).

## Control

The **Control** object represents a control on a script dialog; it's returned by the **Dialog.Control** method. It has the following methods and properties:

- **AddItem:** Add an item to the control (list box or combo box). The first parameter is the item's name, and the second optional parameter is a data value to associate with the string. The item is added to the end of the list. You can also pass a **DialogListItem** object obtained from another control.
- **InsertItemAt:** Inserts an item in the control (list box or combo box). The first parameter is the index to insert the item at (0-based), the second parameter is the item's name and the third (optional) parameter is a data value to associate with the item. Instead of a name/data value you can also pass a **DialogListItem** object obtained from another control.
- **GetItemAt:** Returns a **DialogListItem** object representing the specified item in the list box or combo box control (specify the item's 0-based index).
- **GetItemByName:** Returns a **DialogListItem** object representing the specified item in the list box or combo box control (specify the item's name).
- **RemoveItem:** Removes an item from a list box or combo box. You can pass either the index of the item to remove (0-based), or a **DialogListItem** obtained from the **GetItemAt** or **GetItemByName** methods.
- **SelectItem:** Selects an item in a list box or combo box. You can pass either the index of the item to select (0-based), or a **DialogListItem** obtained from the **GetItemAt** or **GetItemByName** methods.
- **SelectRange:** Selects text within an edit control (or the edit field in a combo box). The two parameters represent the start and end position of the desired selection. To select the entire contents, use **SelectRange(0, -1)**. The return value is a **Vector** with two members that provide the current start and end of the selection. To query the range without changing it, simply call **SelectRange()** with no arguments.
- **count:** Returns the number of items contained in the control.
- **enabled:** Set or query the enabled state of the control.
- **visible:** Set or query the visible state of the control.
- **focus:** Set or query the input focus state of the control.
- **label:** Set or query the control's label.
- **value:** Set or query the control's value. This property depends on the type of the control:
  - **Edit control:** returns or accepts a *string* representing the current contents of the edit control.

- **Checkbox:** for a simple on/off checkbox, returns or accepts a *bool* - **True** for checked, **False** for unchecked. For a tri-state checkbox, returns or accepts a *long* - **0** (unchecked), **1**(checked) or **2** (indeterminate).
- **Radio button:** returns or accepts a *bool* - **True** for checked, **False** for unchecked.
- **Tab:** returns or accepts a *long* indicating the current page.
- **List box / combo box:** returns or accepts a **DialogListItem** representing the selected item. When setting the value, also accepts a *long* representing the 0-based index of the item.

## CustomFieldData

The **CustomFieldData** object is passed to a rename script's **OnGetNewName** method via the **GetNewNameData.custom** property. If a rename script adds custom fields to the *Rename* dialog using **OnGetCustomFields**, this object lets you access the values the user provided for each field.

## Dialog

The **Dialog** object has the following new properties and methods:

- **detach:** Set to **True** if you want a script dialog to run in “detached” mode, where your script provides its message loop.
- **language:** Set this property to create a script dialog in a particular language (if one or more language overlays have been provided), rather than the currently selected language.
- **template:** Set this to the name of the script dialog to display.
- **Create:** Instead of setting **detach** to **True** and then calling **Show**, you can instead call **Dialog.Create**. This creates the dialog but does not show it immediately, letting you initialize controls first before the dialog goes visible. Call the **Show** method once you're ready to show the dialog.
- **Control:** Returns a **Control** object corresponding to one of the controls on the dialog. The first parameter is the name of the control; the optional second parameter is the name of the dialog (if you have any child dialogs hosted in tabs), and the optional third parameter is the name of the tab control (if you have multiple tab controls hosting the same dialog).
- **EndDlg:** Ends a dialog running in detached mode. Normally dialogs end automatically when the user clicks the close button or another button that has its **Close Dialog** property set to **True**. This method lets you end a dialog under script control. The optional parameter specifies the result code that the **Dialog.result** property will return.
- **GetMsg:** Returns a **Msg** object representing the most recent input event in the dialog (only used in detached mode).
- **RunDlg:** Turns a previously detached dialog into a non-detached one, by taking over and running the default message loop. The **RunDlg** method won't return until the dialog has closed.

## DialogListItem

The **DialogListItem** object is returned by the **Control.GetItemAt** and **Control.GetItemByName** methods. It represents an item in a combo or listbox control in a script dialog. It has the following properties:

- **data**: Returns or sets the optional data value associated with this item.
- **index**: Returns the 0-based index of this item within the control.
- **name**: Returns or sets the item's name.

## DOpus

The **DOpus** object has the following changes:

- **DPI**: Returns a **DPI** object which provides information and utility methods relating to the system DPI setting.
- **strings**: Returns a **ScriptStrings** object which lets your script access any strings defined as string resources.
- **viewers**: Returns a **Viewers** object which represents any currently open standalone image viewers (each one is represented by a **Viewer** object).
- **Output**: Scripts can now request timestamps for log messages via an optional 3rd argument to **DOpus.Output**. e.g. **DOpus.Output("Hello", false, true)**. Timestamps only appear in the utility panel, not in places like the Button Editor's output panel. Error messages always get timestamps so if the second argument is true then the third is ignored.

## DPI

The **DPI** object is returned via the **DOpus.DPI** property. It contains the following properties and methods:

- **factor**: Returns the DPI settings as a “scale factor” (e.g. 100, 125, 200).
- **dpi**: Returns the system DPI setting as a “dpi value” (e.g. 96, 192).
- **Scale**: Scales the provided size by the system DPI; e.g. if the system DPI was set to 200%, **DPI.Scale(75)** would return **150**.
- **Divide**: Divides the provided size by the system DPI; e.g. if the system DPI was set to 150%, **DPI.Divide(60)** would return **40**.

## Format

The **Format** object has the following new properties:

- **hide\_dirs\_regex**: **True** if the current **hide\_dirs** pattern is using regular expressions.
- **hide\_files\_regex**: **True** if the current **hide\_files** pattern is using regular expressions.
- **manual\_sort**: Returns **True** if the manual sort option is active.

- **manual\_sort\_name**: If manual sort is active, returns the name of the current sort order (if it has one).
- **manual\_sort\_order**: If manual sort is active, returns a **SortOrder** object which lets you query and change the sort order.
- **show\_dirs\_regex**: **True** if the current **show\_dirs** pattern is using regular expressions.
- **show\_files\_regex**: **True** if the current **show\_files** pattern is using regular expressions.

## FSUtil

- **OpenFile**: The **OpenFile** method now accepts "**NoElevate**" or "**ElevateNoAsk**" to prevent triggering UAC prompts when opening files. "NoElevate" avoids elevation entirely while "ElevateNoAsk" gains elevation only if something else has already triggered it within the script's context. Script columns which open files should use these to avoid triggering annoying UAC prompts in the background.
- **ReadDir**: The **ReadDir** method can now optionally enumerate using the shell, which means non-filesystem folders like `\\server\` or `/mycomputer` can be enumerated. Set the optional third parameter for this method to **True** to use this.
- **GetShellPropertyList**: Returns a list of shell properties, optionally matching the supplied wildcard pattern.
- **GetShellProperty**: Returns the value of one or more shell properties for the specified file. You can either provide the name of the property or a **Map** object to retrieve multiple properties at once.

## Item

The **Item** object has the following changes:

- **current**: This is only present for **Item** objects obtained from a **Viewer**. It will be **True** if the item represents the currently displayed image.
- **name\_stem\_m**: Returns the name stem, taking multi-part file extensions into account (e.g. "cat.and.dog" instead of "cat.and.dog.part1")
- **ext\_m**: Returns the file extension, taking multi-part file extensions into account (e.g. ".part1.rar" instead of ".rar")
- **ShellProp**: Returns the value of a shell property for the item.
- **Open**: The **Open** method now accepts "**NoElevate**" or "**ElevateNoAsk**" to prevent triggering UAC prompts when opening files. "NoElevate" avoids elevation entirely while "ElevateNoAsk" gains elevation only if something else has already triggered it within the script's context. Script columns which open files should use these to avoid triggering annoying UAC prompts in the background.

## Lister

The **Lister** object has the following new property:

- **utilpage**: If the utility panel is open, returns a string representing the currently selected utility page.

## GetCopyQueueNameData

This object is passed to the new **OnGetCopyQueueName** event. It lets a script override the name of the automatically-generated copy queue. The properties are:

- **dest**: Returns a **Path** object indicating the destination path of the copy operation.
- **desttab**: Returns the destination **Tab** object.
- **dest\_drives**: Returns a binary string indicating the physical drive indices that the destination path is located on (if any).
- **move**: **True** if this is a move operation.
- **name**: The default name of the copy queue. To override this, return the new queue name from the **OnGetCopyQueueName** event. To accept the default name, return **False**.
- **source**: Returns a **Path** object indicating the source path of the copy operation.
- **sourcetab**: Returns the source **Tab** object.
- **source\_drives**: Returns a binary string indicating the physical drive indices that the source path is located on (if any).

## GetCustomFieldData

If a rename script implements the **OnGetCustomFields** method it will be passed a **GetCustomField** data object that it can use to add custom fields to the Rename dialog. The object contains the following properties:

- **fields**: Assign values to sub-properties of the fields property to add a field. For example, **fields.my\_option = True** would create a Boolean field called “my\_option”.
- **field\_labels**: A **Map** that lets you provide display names for your fields. For example, **fields.field\_labels("my\_option") = "Option name"**.
- **field\_tips**: A **Map** that lets you provide cue banner text for text fields. For example, **fields.field\_tips("my\_text") = "Enter your text here"**.

## GetNewNameData

The **GetNewNameData** object has the following new properties:

- **custom**: Returns a **CustomFieldData** object which lets your script access the values of any custom fields you added via the **OnGetCustomFields** method.
- **oldname\_field**: content of the *Old Name* field in the rename dialog (synonym of existing **oldname** property, which still works).
- **newname\_field**: content of the *New Name* field in the rename dialog; was previously not available to scripts.
- **newname\_ext**: the proposed new name, extension only, just the last part e.g. ".rar".
- **newname\_stem**: the proposed new name, everything before **newname\_ext**.
- **newname\_ext\_m**: the proposed new name, extension only, handles multi-part extensions e.g. ".part1.rar".
- **newname\_stem\_m**: the proposed new name, everything before **newname\_ext\_m**.

## Msg

The **Msg** object represents a script dialog input event message. It's returned by the **Dialog.GetMsg** method which you call when running the message loop for a detached dialog. It has the following properties:

- **default value**: Returns **True** if the message is valid, or **False** if the dialog has been closed (which means you should exit your message loop).
- **control**: The name of the control involved.
- **data**: The data associated with the current selection (for a combo box or list box). For a check box or radio button, indicates the check state of the control.
- **dialog**: The name of the dialog involved.
- **event**: The event that occurred (**invalid**, **click**, **dblclk**, **selchange**, **editchange**).
- **focus**: **True** if the control had input focus when the message was generated.
- **index**: The current selection index (for a combo box or list box).
- **result**: Returns **True** if the message is valid or **False** if the dialog has been closed.
- **tab**: The name of the tab control hosting the dialog (if any).
- **value**: The current text value of the control (e.g. for an edit control).

## Path

The **Path** object has the following changes:

- **stem**: Returns the name stem, not taking multi-part file extensions into account (e.g. "cat.and.dog.part1" instead of "cat.and.dog")
- **stem\_m**: Returns the name stem, taking multi-part file extensions into account (e.g. "cat.and.dog" instead of "cat.and.dog.part1")
- **ext\_m**: Returns the file extension, taking multi-part file extensions into account (e.g. ".part1.rar" instead of ".rar").

## Script

The **Script** object has the following new method:

- **LoadResources**: Load script resources from an external file (or a raw XML string). If the script is included in a package the file must have *.odxml* as a file extension.

## ScriptColumn

- **autorefresh**: This property can now be set to **2** to force Opus to update the value for the column when the file's attributes change (normally it would only update if the file modification time or size changed).
- **userdata**: Allows you to associate a data value with a column. The value will be passed to your column handler in the **ScriptColumnData.userdata** property.

## ScriptColumnData

- **userdata:** Returns the user data value associated with this column.

## ScriptInitData

The **ScriptInitData** object has the following new properties:

- **group:** Lets you specify an arbitrary group for this script. If scripts specify a group they will be displayed in that group in the list in Preferences.
- **url:** Lets you provide a URL where the user can go to find out more information about the script.
- **config\_groups:** Lets you place configuration values in groups (which can make their display in Preferences easier for the user if you have a lot of options). Accepts a **Map** object in the same way as the **config\_desc** property.

## ScriptStrings

The **ScriptStrings** object is returned by the **DOpus.strings** property. It lets you access any strings defined via string resources. It contains the following methods and properties:

- **langs:** Returns a **Vector** of strings representing the languages that strings are defined for.
- **Get:** Returns the text of a specified string (by name). Optionally accepts a second parameter, specifying a language (otherwise the string is returned in current language if it's defined).

## ShellProperty

The **ShellProperty** object represents a shell property - an item of metadata for a file or folder that comes from Windows or third-party extensions (as opposed to metadata from Opus's native metadata system).

The **FSUtil.GetShellPropertyList** method lets you retrieve a list of available shell properties. You can then use **FSUtil.GetShellProperty** or **Item.ShellProp** to retrieve the value of a property for a particular file.

- **defwidth:** The default width a column displaying this property should use.
- **display\_name:** The display name of this property (the name that should be shown to users).
- **justify:** The default column justification for this property (**left**, **right**, **center**).
- **pkey:** The PKEY (property key) for this property. This is a property's unique ID and the canonical way to refer to a property. You can use the **raw\_name** and **display\_name** values to access properties as well, but they are potentially inaccurate (since it's possible to have two properties with the same name) and also slower as the property has to be looked up by name each time.
- **raw\_name:** An internal name used by the property provider.
- **type:** The type of data this property returns; **string**, **number**, **datetime** are the only supported types currently.

## SortOrder

The **SortOrder** object is returned by the **Format.manual\_sort\_order** property if manual sort mode is active. It lets you query and modify the sort order. The object supports the following properties:

- **GetOrder**: Returns a **Vector** of strings representing the current sort order of files in the folder. You can optionally provide the name of a sort order as a parameter if you have defined more than one.
- **SetOrder**: Pass this method a **Vector** of strings to change the sort order. You can optionally provide the name of a sort order as the second parameter if you've got more than one sort order defined.
- **ResetOrder**: Resets the current sort order to the default. You can optionally provide the name of a sort order as a parameter if you have defined more than one.

## Tab

The **Tab** object has the following new property:

- **color**: Returns the current color of the tab (in "R,G,B" format) if one has been assigned.

## Viewer

The **Viewer** object represents a standalone image viewer. A collection of **Viewer** objects is returned by the **Viewers** object, which is obtainable via the **DOpus.viewers** property. As well as querying its properties, you can also use a **Viewer** object as the parameter for the **Command.SetSourceTab** method, which lets you run commands against a viewer window.

The **Viewer** object contains the following properties and methods:

- **current**: Returns an Item object representing the currently displayed image.
- **bottom**: Returns the bottom coordinate of the viewer window.
- **files**: Returns a collection of Item objects representing the images in the viewer's list.
- **foreground**: True if the viewer is currently the foreground (active) window in the system.
- **lastactive**: True if the viewer is the last active viewer.
- **left**: Returns the left coordinate of the viewer window.
- **right**: Returns the right coordinate of the viewer window.
- **top**: Returns the top coordinate of the viewer window.
- **Command**: Runs a command in the context of this viewer. You can either pass a command string (e.g. **Command("Show VIEWERCMD=next")**) or a **Command** object).

## Viewers

This object is obtained from the **DOpus.viewers** property. It represents all currently open standalone image viewers. It is a collection of **Viewer** objects and can be enumerated as such. It also has the following property:

- **lastactive**: Returns a **Viewer** object representing the last active viewer.

## ViewerEventData

This object is passed to the **OnViewerEvent** method, whenever certain events occur in a standalone image viewer.

- **event**: A string indicating the event that occurred. The events currently defined are **create**, **setfocus**, **killfocus**, **destroy** and **load**.
- **item**: An **Item** object representing the image file involved (if applicable).
- **viewer**: A **Viewer** object representing the viewer the event occurred in.

## Script resources

You may have noticed in the section on Script dialogs that scripts can now have “resources” associated with them. This is XML-formatted data that provides resources to the script but doesn’t actually form part of the script code.

When you use the command editor to design your script, the resources are split out onto a separate tab to make it easier to work with. But in a *Script Add-in*, or when using the *CLI* in script mode, script resources are included at the end of the script code itself. A separator line marks the boundary between script code and resources, like this:

```
If BlahBlah Then
    BlahBlah Blah
End If
```

```
==SCRIPT RESOURCES
```

```
<resources>
  <resource name="blah1" type="dialog">
    <dialog blah blah>
    </dialog>
  </resource>
  <resource name="blah2" type="dialog">
    <dialog blah blah>
    </dialog>
  </resource>
```

```

    <resource type="strings">
      <strings lang="blah">
        <string id="blah" text="Blah!" />
      </strings>
    </resource>
  </resources>

```

Everything before the line `==SCRIPT RESOURCES` is considered part of the script code, and everything after it is the XML-formatted resources.

### ***String resources***

As well as *dialog resources*, scripts can also have *string resources* which provides an easy way for a script to support languages for ad-hoc strings (that is, for strings not part of a static dialog but used programmatically by the script). Strings defined in dialogs (e.g. *Button* control labels) can be translated using the *Language overlays* feature.

As a very simple example, consider the following VBScript fragment.

```

If DOpus.language = "français" Then
  DOpus.Output "Bonjour!"
ElseIf DOpus.language = "deutsch" Then
  DOpus.Output "Guten Tag!"
Else
  DOpus.Output "Hello!"
End If

```

This tests the current language Opus is running in and prints an appropriate “hello” string in that language, or in English if the language isn’t known. That’s fine for a single string, but having to do that for every string in the script could be a lot of typing. Using string resources, the above script fragment reduces to:

```

DOpus.Output DOpus.Strings.Get("hello")

```

The actual strings themselves are provided as an XML resource, much the same way as dialog definitions are. Here’s the string resource that provides the above strings:

```

<resources>
  <resource type="strings">
    <strings lang="français">
      <string id="hello" text="Bonjour!" />
    </strings>
    <strings lang="deutsch">
      <string id="hello" text="Guten Tag!" />
    </strings>
    <strings lang="english">
      <string id="hello" text="Hello!" />
    </strings>
  </resource>
</resources>

```

As you can see, there are three **strings** tags, each one with a different **lang** attribute (this specifies the language). Inside each **strings** tag are one or more **string** tags which provide the actual strings. The **id** attribute is a name of your choosing – it’s how your script refers to the string. The **text** attribute provides the translation of the string in the specified language.

The values for the **lang** attribute correspond to the names of the Directory Opus language libraries (which can be found in the `/home/Language` folder) – **english**, **deutsch**, **français**, **cat**, **czech**, **español**, etc.

Note that there isn’t currently a GUI for editing string resources like there is for dialogs, so you need to code the XML resources yourself if you want to use them.

### **Icon sets**

The Icon Set XML format has been extended to allow multiple resolutions of icon images to be provided to cater for high DPI displays. Opus will use the image closest to the system DPI (and then scale if necessary).

For each `<set>` entry in the XML file you can provide an optional `<dpi>` key which specifies alternate images for different DPI settings, and can also optionally set limits on when the image can be used. For example, if an image doesn’t look good scaled up you can set a maximum scale factor it can be used for.

The `<dpi>` key has the following attributes:

```

set blah...>
  <dpi base="x">                specifies the base scale factor of the image (e.g.
                                "100")
    <scale factor="x"           specifies the scale factor of this alternate image (e.g.
                                "200")
      min="x"                  minimum scale factor this should be used for
                                (optional, e.g. "200")

      max="x"                  maximum scale factor this should be used for
                                (optional, e.g. "400")
      no_scale_min="x"        minimum scale factor this image should be used
                                without any scaling (optional)

      no_scale_max="x"        maximum of above (optional). Use -1 for infinite.

      filename="x"            filename of the alternate image
      width="x"               width of icons in the alternate image (optional, will
                                be calculated if not provided)

      height="x"              height of icons in the alternate image (optional, will
                                be calculated if not provided)

  />
</dpi>

```

As many **<scale>** entries can be provided as needed. If the **<dpi>** key is missing altogether a base scaling factor of 100% is assumed. If needed, you can include the base image in the DPI list as well (for example, if you want to specify **no\_scale\_min** and **no\_scale\_max** values for it).

The **no\_scale\_xxx** range can be used to snap to various sizes while avoiding blurring. This is usually only needed at small sizes (since once icons get larger, the scaling works better). Ranges are inclusive of their **min** and **max** values, should not overlap with each other, and the value specified for **factor** should fall within the range.

A real example from the default icon set:

```

<set filename="#DEFAULT_ICONS_22.PNG" size="small" width="22"
  height="22">
  <dpi base="100">
    <scale factor="100" filename="#DEFAULT_ICONS_22.PNG"
width="22" height="22" no_scale_min="0" no_scale_max="125" />
    <scale factor="150" filename="#DEFAULT_ICONS_32.PNG"
width="32" height="32" no_scale_min="126" no_scale_max="175" />
    <scale factor="200" filename="#DEFAULT_ICONS_48.PNG"
width="48" height="48" />
    <scale factor="300" filename="#DEFAULT_ICONS_64.PNG"
width="64" height="64" />
  </dpi>
  <icon col="1" name="empty" row="1" />
  <icon col="2" name="spacer" row="1" />
  ...
</set>
<set filename="#DEFAULT_ICONS_32.PNG" size="large" width="32"
  height="32">
  <dpi base="100">
    <scale factor="100" filename="#DEFAULT_ICONS_32.PNG"
width="32" height="32" no_scale_min="0" no_scale_max="125" />
    <scale factor="150" filename="#DEFAULT_ICONS_48.PNG"
width="48" height="48" no_scale_min="126" no_scale_max="175" />
    <scale factor="200" filename="#DEFAULT_ICONS_64.PNG"
width="64" height="64" />
  </dpi>
  <icon col="1" name="empty" row="1" />
  <icon col="2" name="spacer" row="1" />
  ...
</set>

```

- Added **FileType NEWCOUNT** argument which lets you create more than one new file at once (e.g. **FileType NEW=.txt NEWCOUNT=10**).
- Added **CreateFolder ASK** argument to cause the dialog to be displayed even if a name is provided (e.g. **CreateFolder "New Folder Name" ASK**).
- Added **drivelabel** argument for breadcrumbs path fields, to display drive label along with the drive letter.
- The advanced filter control now has a *Case sensitive* option for *Name* and several other clauses.
- The *Favorites* item in the folder tree now has a context menu which lets you open the favorites editor (Preferences page).
- The *Save Tab Group* dialog (that's displayed by the **Go TABGROUPSAVE** command) now has a checkbox letting you control the *Close existing tabs* flag for the new group.
- The **Find DUPES** command now lets you use the **NAME** argument to provide a wildcard name filter when searching for duplicates.
- The *Replace File* dialog's rename field now supports more of the features of the regular inline rename field (including capitalization, F2 to cycle selection, and left/right cursor handling).
- Photoshop CMYK images with alpha are now supported.
- The *Music* category in the filter control (advanced find) now lets you search on the *Year* field.
- External icon sets can now refer to internal image resources (such as the default toolbar icons) without having to include redundant copies of the image data. The main goal is to allow alternative size (and DPI-scaling) variants of the internal sets. Internal icon sets contain four image sizes (22, 32, 48 and 64 pixels) but until now you only had access to two of the sizes at any given DPI. This also makes it possible to use the flat and glass icon styles at the same time.
- Scripting:
  - Added **Viewer.parenttab** property.
  - The **OnGetCustomFields** rename script method now lets you assign default focus to one of the custom fields, by setting the **GetCustomFieldData.focus** property to the name of the field. You can also use **!oldname** and **!newname** to assign focus to the standard old and new name fields.
  - Added an interface for multiple-selection listboxes in script dialogs.
    - **Control.SelectItem** now works with multi-selection listboxes. -1 can be given as the index to select all items in the listbox.
    - New method **Control.DeselectItem** lets items be deselected.
    - **Control.value** now returns and accepts a Vector of **DialogListItem** objects for multiple-selection listboxes.
    - New property **DialogListItem.selected** provides another way to query or set the selection state of items in a multiple-selection listbox.
  - Script dialog status controls now correctly handle setting their initial text when the supplied string contains `\n` (to insert a line-break).
  - Script dialogs now support the **icon** property to set the dialog's titlebar icon.
  - Added **x**, **y** and **position** properties to **Dialog** object, to allow the position of script dialogs to be controlled. **position** can be "center" (the default), "parent" (relative to parent window), "monitor" (relative to monitor) or "absolute". If **x** and **y** are specified and **position** is not "center" then the coordinates are treated as relative to the specified position.
- Fixed problem which could cause sort header to not appear in the default tab of a new Lister when in one of the icon modes.

- Fixed problem with displaying correct filetype description for matlab .m files.
- The **Go FOLDERCONTENT=move** (or **=copy**) command now works from the toolbar in the standalone image viewer.
- Fixed problem which could cause slow startup with network favorites displayed in the tree.
- Fixed problem which could cause the vertical scroll offset to be reset when going forwards/back in the history (instead of preserving the saved offset).
- Scrolling horizontally in list mode by clicking the scrollbar gutter will no longer skip partially visible columns.
- The relative date graphs are now based on the timestamps as displayed in the file display. So if milliseconds are not displayed, they also won't be considered when calculating the graphs (same goes for seconds).
- **CreateFolder** now works properly when pipe-separated names are specified on the command line (e.g. **CreateFolder blah\1|2|3**).
- Opus no longer inspects the contents of *.tmp* files in an attempt to generate thumbnails for them.
- Updated to latest libpng (fixes problem loading image in the forum thread [PNG images cannot be loaded as images](#))
- Fixed crash in *Rename* dialog that could be caused by having an empty new name string in *Find & Replace* mode and turning on the *Rename matching filenames as one* option.
- The move up/down buttons in the *Folder Options / Columns* tab did not work correctly.
- **Select FROMSCRIPT** (when used from a script, obviously) now works reliably with libraries.
- The **Copy WHENEXISTS** argument is now respected when extracting from zip archives.
- When copying files, the replace dialog now re-opens relative to its previous top/right position, so that the mouse doesn't need to move to find the same button from one file to the next.
- Fixed problem with folder sizes in infotips which could leave a partially calculated size behind if the initial calculation was aborted.
- The *Locate toolbar* button in the *Customize / Keys* dialog will now flash the exact button the hotkey comes from rather than just the toolbar (and will expand sub-menus to make it visible, if needed).
- If tree path highlighting is on and set to use the configured tab color, it now updates in real time when the tab color is changed.
- **Ctrl-S** (to save a preset) now works in the rename dialog even when the script editor has focus.
- The **Go OPENCONTAINER** command now works correctly in conjunction with the **EXISTINGLISTER** argument.
- Drop-down menu buttons (e.g. those generated by **Go FOLDERCONTENT=button**) can now be accessed from the keyboard.
- When dragging items around on the File Type editor's *Context Menu* tab, the line indicating where the item would end up could appear one item above or below the actual location.
- The image viewer no longer resets the scroll position when moving from one image to another (unless the zoom level is also reset due to Preferences).
- The *Preferences / File Operations / Double-click on Files / Use internal picture viewer for...* option didn't work if the *Left double-click* event for the *Recognized images* filetype was undefined. It now defaults to **Show** if not set but can still be overridden via the filetype editor if needed.
- On Windows 8 and 10, the *Set Wallpaper* command had to be run twice to be effective if the desktop had been set to a solid color. A workaround has been added so this is no longer the case.
- Improved Lister resizing performance when resizing from top/left of Windows when composition is enabled.
- Setting labels on files in library paths always saved the label to the config, ignoring the "save to NTFS" flag.

- Fixed problem where labeled folders would lose their labels in the folder tree when moved/copied (until a tree refresh).

## 12.3 5th December, 2016

---

- Added **@disableif** and **@hideif** command modifiers, that let buttons be hidden or disabled based on the result of a function (e.g. **@hideif:!Set DUAL=Toggle** to hide a button when not in dual-display mode).
- Added **@disableifpath** and **@hideifpath** command modifiers, to allow buttons to be hidden or disabled based on the current source path.
- **@if**, **@disableif** and **@hideif** can now test if a command would be enabled rather than toggled. E.g. **@if:enabled Go GROUPEXPAND=\***.
- The **@hidenosel** and **@disablenosel** modifiers can now be negated, e.g. **@hidenosel:!** would hide a button if anything at all was selected.
- **Set COLUMNSADD** and similar commands can now specify a position for the column and flag that it is only to be used if the column isn't already present. Prefix the position with **!** to do this. For example **Set COLUMNSADD=Size(!2)** would leave the *Size* column as-is if it already exists, and add it in position 2 otherwise. This also works with relative positions: **Set COLUMNSADD=Status(!1+Name)**.
- Adding an **!** before various conditional modifier tests now work to negate the tests. This has been added for **@if:!**<blah>****, **@ifset:!**<blah>****, **@ifpath:!**, **@ifpathr:!**, **@ifexists:!** and **@keydown:!** (plus the new **@disableif**, **@hideif**, **@disableifpath** and **@hideifpath** modifiers).
- Added **Preferences / Viewer / Appearance / Show status icons** option to display current image's status icons in the viewer.
- Added **Custom title** option to **Preferences / Viewer / Appearance**.
- The font used for the rename macro builder is now configurable via **Preferences / Display / Colors and Fonts**.
- Added **Preferences / Folder Tree / Options / Expand selected branch when changing tabs** option.
- A new setting, **Preferences / Miscellaneous / Advanced: context\_menu\_icon\_set**, allows you to change the icon set used when generating the *Add to Archive* and similar context menu items within Opus. For example, if you want the menu items to use the standard 16x16 size (32x32 at 200% DPI, etc.) then you can now do that, assuming a suitable icon set is installed.
- The *Target* clause in the advanced filter control now lets you limit matches to shortcuts only to files or only to folders.
- The breadcrumbs path field has a new **dragsafetyoff** argument which allows you to copy or move from the breadcrumbs field using drag and drop, without having to explicitly hold **Ctrl** or **Shift**. Without the argument, drag and drop from breadcrumbs defaults to creating shortcuts, to avoid accidents.
- The **Customize / Keys** list now displays a column showing which Toolbar a key comes from (if applicable). The toolbar name is shown in italics if its *Always enable hotkeys* option is turned on. The **Locate Toolbar** button will now open any sub-menus necessary to locate the button that contains the selected hotkey.
- The **Join** dialog now adds an output filename by default.
- Minor improvements to the **Split** dialog. It now remembers the last size and UUEncode settings you used. "Automatic" size on non-removable devices is now 100MB instead of 1.44MB. Handles splitting to the current folder (vs a destination folder) better.
- Moved the **Start Folder Tree at** Preferences option from **Folder Tree / Options** to **Folder Tree / Contents**.
- RAR decompression uses UnRAR.dll again unless overridden. It is faster for some operations and a recent update added support full timestamp accuracy and all three timestamps (Created, Modified, Accessed).
- The Preferences option **Launching Opus / Explorer Replacement / Open external folders in a new tab** now routes new tabs to the most recently used Lister, rather than the source Lister.

- In the advanced button editor, the help button (or F1) will open the page about command modifiers if clicked while on a line for one.
- Reduced overhead of the Copy progress dialog when copying thousands of tiny files.
- Fixed Preferences ignoring the "show built-in aliases" checkbox when first opening the *Aliases* page if no user-defined aliases exist.
- Fixed problem with script dialogs that could lead to a script still processing messages in its message loop after a dialog had closed (which depending on the script and the exact timing, could lead to a script error).
- A button-menu with "always enable drop-down" set would change the spacing between its label and drop-down arrow when the main button part became disabled. The spacing now stays the same in both states.
- Added an option to **Preferences / File Operations / Progress Indicators** to disable the progress indicator speed graph.
- When copying two or more files the progress dialog now shows the total bytes copied / remaining as well as the values for the current file.
- It's now possible to assign the "stop on match" label to a file by itself, to prevent wildcard/label filters from applying to it without actually specifying a label explicitly.
- Added workaround for Google Drive opening its folder using the \\?\ prefix even when it is unnecessary. (Things worked, but it was unsightly.) The prefix is now removed if it isn't needed.
- The folder tree and file display should now update immediately in response to assigning labels to drive roots under the This PC (My Computer) folder.
- The **Prefs STYLESAVE** dialog now asks if you want to replace an existing style instead of failing.
- The **Go DRIVEBUTTONS** command would incorrectly omit MTP devices if any arguments were used (unless "mtp" was specified as well).
- If the viewer custom title contained %L (for file label), the viewer would not correctly update file labels / status icons when advancing to the next image.
- The TGA plugin's minimum size limit has been reduced from 45 bytes to 19 bytes, to allow extremely small TGA files to be viewed.
- The **Preferences / Favorites and Recent / Jump List** page now supports Layouts that have been organised into folders. Note that on the jumplist itself they will appear as a flattened list.
- Fixed problem that could cause problems with FTP in certain specific situations (e.g. with two single Listers, clicking the **Up** button in the FTP Lister while copying a file to it would cause the transfer to be aborted).
- Music Comment in the file display and the scripting **mp3comment** field will now return comments up to 1000 characters long before truncation, instead of the old limit of 100.
- If **Rename** is used in simple mode (**Rename SIMPLE**) and a wildcard pattern isn't entered, the rename dialog will now re-open automatically allowing the next selected file to be renamed (compatibility with Opus 11).
- The rename script editor now works correctly in the Light edition.
- Moving the mouse over the text viewer no longer causes the cursor to flicker between pointer and caret.
- Fixed some problems with image marking in MTP folders
- Tab groups now obey the **Preferences / Folders / Auto-Loading** settings. The new **TABGROUPFORCE** switch can be used with the **Go TABGROUPLOAD** command to load a tab group and override the auto-load settings.
- Fixed/improved a problem with native MTP support where creating folders (or sometimes even reading a directory) could cause Opus to freeze until the MTP device was unplugged.
- Fixed problem where drag & drop of folders from a library to another Lister would not correctly count the files for the progress indicator.

- Fixed line spacing not changing if you switched between Details and Power modes (without another mode in between) and the two had different line spacing set.
- Fixed cosmetic issue when scrolling up in Tiles mode with certain combinations of visual settings.
- Right-click on the viewer's title bar now shows the normal window menu rather than the viewer's context menu.
- Image conversion no longer fails when updating EXIF data in JPG and PNG images if the destination requires UAC elevation to modify. Also changed what happens when the destination is an FTP site or Zip archive or similar, where the EXIF data will be dropped on conversion now (as it was in Opus 11) instead of always causing the operation to fail.
- The viewer pane now only tries to display .URL files using a web browser if they point to HTTP, HTTPS, FTP or FILE URLs. In particular, shortcuts to Steam apps/games will no longer be triggered by the viewer; you'll only see the text content of the shortcuts instead.
- Fixed details/power mode rendering issue with graphs (date/size) behind another column being truncated in some cases (e.g. if the column was set to left-justify and the width of the graph was greater than the width of the text).
- If, while a folder is being read, you press a key that triggers the FAYT field, the FAYT will no longer automatically close when the folder read finishes.
- Fixed not being able to change the file display toolbar back to the default in **Preferences / File Displays / Border** on a clean install of Opus Light.
- The **Customize / Keys** list now includes all default toolbars in the Light version so that you can easily see their hotkeys and check for clashes (previously they would only be shown if the toolbar files actually existed on disk, which would only happen if you had run the Pro version before switching to Light).
- Made a change which stops the **Target** column showing an out-of-date value for .url shortcuts if the .url shortcut file was edited manually (e.g. in a text editor) rather than through the Properties dialog.
- Added **Show VIEWERCMD=cmdbar** command which shows a FAYT-style command field in the viewer.
- Added **Preferences / Folder Tabs / Options / Lister closes when last tab closes** option.
- When using the metadata panel to set an image's last modification timestamp, Opus now sets the EXIF *DateTime* tag to the same value automatically.
- **dopusrt.exe** has a new */vcmd* argument which lets you send commands to the last active viewer window.
- Default toolbars now include a *PowerShell Here* menu item which runs the new **CLI DOSPROMPT=powershell** command. **CLI DOSPROMPT=powershellISE** is now also supported, for opening the PowerShell ISE instead of a basic PowerShell prompt.
- In the viewer, you can now press **Ctrl+TAB** to switch the focus from a metadata field back to the main viewer window, and **Ctrl+TAB** to switch focus back to the same metadata field again.
- In the button editor, increased the length of strings which the **Label** and **Tip** fields will accept.
- Navigating to locked BitLocker devices now automatically prompts to unlock them on Windows 8 and Windows 10. (This already worked on Windows 7.)
- When selecting icons for Labels and Collections, the icon path is automatically made relative to folder aliases if possible. (e.g. */home* for where Opus is installed, */dopusdata* for your config folder, */programfiles*, etc.) This helps make your configuration more portable.
- Paths to background images are now stored relative to standard aliases if possible, for better portability. (This was already the case for the *Images* folder in the configuration, but now works with more folders.) Paths to external toolbar and context menu icons are now stored relative to aliases as well.
- The various options (color, image, label state, etc) on the **Customize / Toolbars** page for the standard toolbars were editable in Opus Light but not saved. These settings are now preserved correctly.
- **Preferences / Folder Tree / Options / Expand selected branch when changing tabs** now also applies in a dual file display, with a single shared folder tree, when you change from one side to the other.

- If a Folder Format specifies a background image that no longer exists, the effect is the same as if *No Image* was chosen in the Folder Format editor, but the editor instead displayed *Default Image*. The editor now displays *No Image* to match the actual behavior.
- The Preferences search function now understands some alternate English spellings (e.g. color/colour, behaviour/behavior).
- The **Rename PRESET=last** command was incorrectly documented as running immediately rather than opening the rename dialog. The docs have been corrected to reflect the actual behaviour. Also added **Rename PRESET=!last** which actually does let you run the last rename command without opening the rename dialog.
- The rename preview no longer shows items as ghosted if their location has changed without the name changing too.
- The rename dialog now updates any script-added custom fields correctly when loading a preset that has values for them saved, even if the control had previously been added by a different script.
- The Replace dialog now respects the preferences flag to auto-rotate thumbnails using EXIF data.
- Folder tree highlight option **Use color from tab** now works for tab colors which come from folder formats.
- The control used for the list of Preferences pages, and in a few other places, now works better with dark Windows themes.
- The command editor menus for the **Prefs LAYOUT** command no longer hide hidden layouts, since that's one place you would normally still want to see them.
- Scrolling an image in the viewer by click+drag now "tracks" the image with the mouse when zoomed unless the control key is held down.
- Fixed the Preferences dialog opening and then immediately closing if no Listers were open and the program was set to automatically close when no Listers were open. Preferences (and several other windows) will now keep the program alive similar to Lister windows.
- Fixed an issue with miscalculation of ISO week numbers in some cases.
- Fixed the breadcrumbs path field getting a bit confused if you were in a folder below Quick Access and then navigated directly to the real parent of that folder.
- The **Favorites** command has a new **BRANCH** argument which allows you to specify the branch of the favorites tree to add new favorites to. If used with the *Add To Favorites* dialog, you can make it select or create a particular branch by default but still be able to change it when the dialog appears.
- In the *Add To Favorites* dialog, you can now create nested branches with fewer clicks. e.g. Click **New Branch** and type **Cat\Dog** to create **Cat** below the selected branch and **Dog** below that.
- Added **Select MAKEVISIBLE=immediate** option, which activates the **MAKEVISIBLE** action immediately instead of after a small delay (and also disables smooth scrolling).
- Added **Select NEXT=nodeselect** and **PREV=nodeselect** keywords, which prevent the current file from being deselected before selecting the next/previous file.
- When adding a submenu to a filetype's context menu, you can now make the submenu also act as a 'menu button' that runs the first command within it when it's clicked. To do this, right-click on the first child menu item and turn on the **Button** option. Note this only works if the first command in the submenu is an 'Opus-only' command.
- The right-click menus for Favorites items now hide the **Rename**, **Cut** and **Delete** commands in the folder tree, toolbars and breadcrumbs menu. (**Delete** was already hidden in the case of toolbars).
- **@disablenosel** and similar command modifiers now work if they are preceded by another modifier line in the function that's been commented out.
- The *Save Toolbar Set* dialog now displays a drop-down of existing toolbar set names to choose from (as well as entering a new one).
- The standalone viewer now respects configured Pane Border colors for the Marked and Metadata panel headers.

- File infotips will no longer appear while inline rename is active, since they were easy to trigger by accident and sometimes appeared on top of the rename field.
- Fixed bug where a queued **Copy As** function would forget the new name you had entered for the file if the copy was initiated via drag & drop.
- Fixed problem with the **Copy HERE** argument picking the wrong folder when used in flatview.
- Fixed problem with Rename which meant the **New Location** column in the preview could flicker on and off as you scrolled through the preview list.
- Fixed incorrect drag & drop tooltip in some situations when dragging over the Favorites root or branches in the folder tree.
- The viewer now correctly detects when the image it's viewing has been renamed if the file is located in a junction or linked folder.
- (Experimental) Fix for missing separators in some situations, due to menu items which get hidden. If you notice extra or missing separators, or anything wrong with toolbars in menus, please report it so we can investigate.
- Fixed problem with image metadata writing not working on Windows XP.
- Fixed problem when using the **Show** or **Slideshow** commands on a folder. The viewer correctly enumerated the folder's contents, but also tried to display the folder itself as the first image, resulting in an error message.
- Fixed toggling checkbox mode not adjusting *Auto-Fill* column sizes appropriately.
- Fixed not being able to drag folders from the breadcrumbs field and drop them into file displays.
- Fixed file displays not always using the drag & drop source's preferred action (link, copy, move) if one was specified and no key was held down to override it.
- Fixed **Click selected tab to go to previous one** option interfering with ctrl-clicking folder tabs to link them.
- The Advanced Filter *Type* clause used raw names instead of user names for file type groups - didn't affect the default groups but user-created groups would appear as {GUID} strings in the drop-down.
- If the Copy command was launched with the default Ctrl+I hotkey it wouldn't prompt for a filter when filter mode was on.
- Fixed Lister not always coming to the front if you activated it by clicking on an inactive folder tab, where one tab had a warning banner and the other did not.
- Fixed folder tabs opening next beside the active tab instead of the end when dragging folders or tabs to the end of the tab bar, if certain Preferences settings were combined.
- Dragging drive roots (e.g. "C:\") from breadcrumbs to the tab bar is no longer blocked and can be used to open them in new tabs.
- The **Properties SETLABEL** command would ignore the **ADDLABEL** switch if a) a category filter was specified and b) **SETLABELTOGGLE** wasn't also used.
- Fixed problem with layout separators appearing in the wrong positions.
- Change in 12.2 would cause replace dialog to move to the right on the second and subsequent invocations if the generated file description was over a certain length.
- Fixed **Inherit columns from other matching formats** folder formats option causing the sort order to be inherited/replaced as well.
- Fixed crash if a script passed **Command.AddFiles** a **Vector** containing strings (instead of file **Item** objects).
- Explorer Replacement is now stricter about finding existing windows/tabs when another program requests the user profile desktop directory and the Desktop virtual folder is showing in a lister (or vice versa). In specific situations, this will improve the response of "show in folder" options in other programs.
- Fixed problem with {sel:xxx} status bar date codes, which displayed UTC instead of local time.

- Improved the way the Update Checker shows you the folder with the downloaded installer, and similarly how the File Type Diagnostic utility shows you the zip file it makes.
- Fixed crash if your toolbar had a **Set ENABLELABELFILTER** button added to it but **Preferences / Favorites and Recent / Label Assignments** was completely empty.
- Added **rename\_default\_focus** option to *Preferences / Miscellaneous / Advanced*, to control which field in the *Rename* dialog gets input focus by default (scripts can override this).
- Added **Assume UTF-8 without BOM** option to text viewer configuration.
- Inline rename in the Folder Tree now supports the same case-changing hotkeys as most other rename fields in the program. (**Ctrl+L** for lowercase, **Ctrl+U** for uppercase, **Ctrl+W** to capitalize words and **Ctrl+P** to capitalize just the first letter).
- **CLI DOSPROMPT** command now works from a context menu in the tree (previously it would always be disabled).
- Fix for item in the **Preferences / Favorites and Recent / Labels** list sometimes becoming unselectable if you renamed one item and then clicked another in the list before clicking anywhere else.
- If you run the **Settings / File Types** command with the *File Types* dialog already open but minimized, it will now be restored (previously nothing would happen).
- Scripting:
  - In script buttons launched via the viewer, **clickData.Func.Dialog** now has the viewer as its parent window and not the active file display.
  - Fixed problem where the **Metadata** object could return outdated data in some cases.
  - Group boxes in script dialogs did not refresh properly when their label was changed.
  - The **Dialog** object's **x** and **y** properties can now be set after the dialog has been shown, to move the dialog around the screen.
  - The **Dialog** object now has **cx** and **cy** properties to get and set its size (when the dialog is resizable).
  - Added **Dialog.SavePosition** and **Dialog.LoadPosition** methods, to save and load dialog window position.
  - Added **Dialog.Vars** method which returns a **Vars** object associated with the dialog.
  - The **Control.RemoveItem** method now accepts **-1** to clear the entire listbox/combo contents.
  - The **Viewer** object has new **title** property which allows the title in a viewer to be changed from a script.
  - Added **fg**, **bg** and **style** properties to the **Control** object. Currently only supported for static controls.
  - The **Command.AddFiles** method can now be given a **Vector** containing **Path** or strings (full paths), in addition to the **Item** objects which it previously allowed.
  - Added **opacity** property to the **Dialog** object (and also the dialog editor), allowing a dialog's opacity level to be controlled.
  - Added **Dialog.SetTimer** and **KillTimer** methods, which allow scripts to create recurring timer events.
  - Added **Dialog.AddHotkey** and **DelHotkey** methods so script dialogs can implement hotkeys (keyboard accelerators).
  - Added *List View* control type to script dialogs - this is like a *List Box* except it supports multiple columns and different display modes. To support this, the **Control** object has new **columns** and **mode** properties and the **DialogListItem** has new **subitems** and **icon** properties.
  - Added **Lister.state** property to report the source/destination/off state of a non-dual display Lister.

- *Static Text* controls in script dialogs can now display images, by setting the **Image** property to true in the dialog template (the control's label then becomes the image filename).
- Script dialogs can now accept files via drag & drop if the **Accept Drops** property is set in the dialog template. Dropped files will generate a **drop** event and the new **Msg.object** property provides a **Vector** of **Item** objects representing the files that were dropped. If the drop was over a control the **Msg.control** property will identify the target control.
- Added **Control.MoveItem** method to move an existing item in a *List Box*, *Combo Box* or *List View* control.
- Added the **FSUtil.Drives** method which returns a **Vector** of **Drive** objects, providing information about the drives present in the system.
- The scripting **File.Read** method did not correctly read the entire file if not passed a size.
- The scripting **DOpus.Language** property now always returns a lowercase version of the language name, to simplify testing against it.
- Added **OnFlatViewChange** script event, to allow a script add-in to detect changes to the flat view mode in a tab.
- Script dialogs now generate resize events when the window is resized. Must be enabled by setting **Dialog.want\_resize** to **True**. **Msg.cx** and **Msg.cy** provide size information.
- Added **DialogListColumns.GetColumnAt** method.
- Added **DialogListColumn.resize** property - set to **True** to specify a column that auto-resizes with the listview.
- Added **NOSCRIP**T argument to the **Close** command to stop the command triggering the **OnCloseLister** event.
- The **StringTools.Decode** script method now skips over the UTF-8 BOM when decoding a **Blob** containing UTF-8 encoded data.
- Fixed problem with Rename script custom fields where default field values would not be valid the first time the rename preview was refreshed.
- The script **Command** object now correctly runs functions asynchronously if **SetModifier("async")** has been called.
- Fixed bug which could cause the program to crash if a script dialog encountered a script error while processing the dialog message loop.
- Fixed problem with **OnCloseLister** script event causing infinite loop/stack overflow if the event handler invoked the **Close** command to close the same Lister.
- Added **OnListerResize** script event.
- The **Metadata** object now returns a **Date** object for date values (e.g. **Metadata.image.datetaken**) rather than **VT\_DATE** variants.
- Added **Item.Labels** method which returns a **Vector<string>** of all labels applying to an item.
- The **Date.Format** method can now be passed flags to override Preferences settings like day names in dates, when formatting dates and times according to the user's locale.
- Added script **Item.Update** method to update an existing Item object's size/date/attributes if the file has been modified on disk
- The scripting **Item.Labels** method's category filter now works properly with user-defined categories.
- Fixed a (rather esoteric) problem which meant the **GetSizes MD5** command, if run from a script, did not work correctly with files selected.
- Added **Prefs NOSCRIP**T argument; when used with the **LAYOUT** argument it prevents **OnOpusLister/OnOpenTab** script events from firing for the new Listers opened by the layout.
- Dialog events triggered by control initialization are no longer passed through to the script.
- The **Date.Sub** method now works properly for years and months.



- Added functions to the *Rename* dialog to copy the list of old names to the clipboard, and to paste a list of new names in. The **Use preview list to build macros** option must be turned off to use these functions.
- Added two new options to the **File Operations / Progress Indicators** page in Preferences:
  - **Always display the jobs bar:** Jobs bars will always be displayed at the bottom of Listers, even when no operations are running.
  - **Show in all Listers:** When the jobs bar is shown automatically, it will be shown in all existing Listers, not just the active one.
- Added **File Displays / FAYT and Filter Bar Options / Prioritize shorter filenames** option to Preferences. When enabled, the FAYT will favor shorter matching filenames over longer ones when searching in Find mode, which may be desirable when the list isn't sorted alphabetically.
- On the **File Display Modes / Thumbnails** page in Preferences, the settings for folder thumbnails have been moved to a sub-dialog that's displayed when you click the **Adjust folder thumbnail settings** link. This dialog contains some new options:
  - **Single image:** When NOT using the shell to generate folder thumbnails, this causes Opus to only use a single image from within the folder rather than up to four images when building the thumbnail. If the single image option is turned on, a filename or wildcard pattern can be specified to control the files that are looked for.
  - **Display folder frame:** When NOT using the shell to generate folder thumbnails, this lets you control whether the big "folder" image is used for thumbnail folders or not. If turned off folder thumbnails will be displayed with a normal thumbnail border. The options underneath let you control the color of the folder frame if it's enabled.
- Added **Viewer / Appearance / Reset scroll position for each picture** option to Preferences. When enabled, the scroll position of will be reset to the top/left when moving from one image to the next.
- Added two new options to the various mouse button settings on the **Viewer / Mouse Buttons** page in Preferences:
  - **Expand/Scroll Image:** When this option is enabled, and the displayed image is reduced from the original size, clicking and holding the appropriate mouse button displays the image in its original size for as long as the mouse button is held down.
  - **Script event:** When enabled, any script add-ins that implement the **OnViewerEvent** script event will be triggered when the button is clicked. The event type will be "click", "dblclk" or "mclick" as appropriate.
- Added **Viewer / Mouse Buttons / Click left/right edges to go to previous/next picture** option to Preferences. When this is turned on, the left mouse button will move to the next or previous picture when the left or right edges of the window are clicked irrespective of the actual setting for this button. You can configure the percentage of the window that is considered to be the "edge" (defaults to 20% of the window width).
- Added **Viewer / Viewer Pane / Expand and scroll** option to Preferences. When this option is enabled, and the displayed image is reduced from the original size, clicking and holding the appropriate mouse button displays the image in its original size for as long as the mouse button is held down.
- Auto-hide toolbars set to the *Frame* and *NoFrame* appearances are now invisible when hidden, instead of causing a solid line to appear down the edge of the screen. (This was already true for the other appearances).
- Using the **Select** command to show or hide files now also triggers column widths to be re-evaluated if auto-sizing is on.

- Added a **Use Simple Rename** option to the **Rename** drop-down on the default *Operations* toolbar. When turned on, the default **Rename** button will show the simple rename dialog instead of the advanced one. Note you'll need to reset your *Operations* toolbar to the defaults to see the new option.
- Floating toolbars set to auto-hide now have configurable delays before they start to slide on or off the screen, in addition to the old settings for the speeds of the slides themselves. The new settings, in **Preferences / Toolbars / Options**, can help avoid accidentally triggering a hidden toolbar when moving the mouse nearby.
- Added an option to save the key file passphrase for SSH connections.
- The *Edit File Type* dialog has a new tab (*Replace Menu*) which lets you configure menu items that will be shown in the context menu displayed when right-clicking on the file icons in the *Replace* dialog. This lets you add commands to compare the two files using an external tool.
- The *Save Folder Format* dialog now has an "up" button which lets you quickly save the format for a parent folder of the current one.
- When adding files to a collection using the **Copy** command, you can now use **WHENEXISTS=replace** or **WHENEXISTS=skip** to suppress error messages about any files which are already in the collection.
- The **{allfilepath|filem}** and similar command sequences now have **CROnly** and **LFOnly** flags to specify that the output files should only have CR or LF characters between lines, rather than the default of CR,LF pairs. For example, **{allfilepath|filem|lfonly}**.
- Wildcards now allow expressions such as **~grp:Images** and **~(grp:Images)** for negating file type group wildcards.
- The **Properties SETLABEL** dynamic button command to generate menus or buttons for setting/toggling labels and status icons has new **!noreset** and **!nostoponmatch** parameters to prevent the *Reset* and *Stop On Match* options being added to the end of what it generates.
- The **Home** and **End** keys in the *Copy As* dialog now stop at the file extension (unless already on it and pushed a second time), as already done in places like the *Rename* dialog.
- Added **Show VIEWERCMD=scroll,center** command (scrolls image to center).
- The image viewer now preserves the current relative scroll position when zooming in and out.
- Added **Set RELATIVESIZEGRAPHS** and **Set RELATIVEDATEGRAPHS** commands to toggle the **Preferences / Folders / Folder Display / Show relative graphs behind size columns** and **Show relative graphs behind modified date columns** options.
- The viewer and preview pane have new options for accelerated scrolling while dragging images with the mouse. If off, the image moves 1:1 and the pixel you grab stays under the mouse pointer. If on, the image scrolls faster such that you can move across the entire image with a small mouse movement.
- The **Show** command now has an **AUTOFILELIST** argument which is similar to **LISTSIBLINGS**, but uses the folder tab's file list rather than the directory. (The two are different if the file display is filtered or using Flat View, for example.) While **AUTOFILELIST** has existed as a hidden argument for some time, it is now documented and supported and has some new behaviors. See the updated manual for full details of what it does and how it interacts with the **LISTSIBLINGS** argument and related Preferences option.
- Added **{grp}** status bar code, which returns the number of file groups displayed in the file display (when it's set to group).
- Added the **Show VIEWERCMD selaspect** command, which allows the selection aspect ratio to be fixed in the standalone viewer.
- Added option to folder thumbnail preferences to prevent folder thumbnail images from using the most recent images in the folder (instead images will be used alphabetically).
- Moved the **Cycle through pictures with mouse wheel** option from **Preferences / Viewer / Behaviour to Viewer / Mouse Buttons** page, and added an additional option to zoom with the mouse wheel (without the **Ctrl** key needing to be held down).
- Moved the **Reset scroll position** and **Reset zoom level** options from **Preferences / Viewer / Appearance to Viewer / Behaviour**.

- Added an **Edit** menu to the **Customize / Keys** page, allowing copy/cut/paste of hotkeys. You can also right-click on individual hotkeys in the list to copy them to the clipboard.
  - In FlatView mode you can now drag files from sub-folders to the "root" folder by dropping on the file display background, rather than having to drag to the last node of the breadcrumbs bar or folder tab.
  - Dragging files from nested sub-folders in FlatView mode to the parent folder now bypasses the prompt and defaults to copying all files to the same folder (since "recreate" in this case makes no sense anyway).
  - Some improvements relating to disconnected network drives:
    - Drive buttons no longer read the volume name for network drives when displaying a tooltip, which could cause a lock up if the network drive was offline (the tooltip will show the UNC path instead of the volume name).
    - Drop-down drive lists are now updated on a background thread which should prevent the Lister locking up when mapped network drives are unavailable.
    - Made some changes to fix Lister lockups when switching between tabs that point to disconnected network drives
  - In **Rename**, you can now use **{parent|noext}** and **{parentbase|noext}** to get the parent folder name without file extension (useful if the parent is an archive file).  
The **Copy** command's **CLEARREADONLY** argument is now used for all copy sources, not just CDs/DVDs. Note that the Preferences option still only applies when copying from CDs.
  - Added **Set DISABLEGLOBALHOTKEYS** command to temporarily disable all system-wide hotkeys.
  - Each Lister now gets its own dedicated background file information thread (for e.g. extracting metadata from images, calculating labels, etc). Previously there was just one file info thread for the whole program. This should help improve performance with multiple Listers and also mitigate against the problem of the thread getting stuck (e.g. the recent problem with certain PDF files) which meant all file information and labels etc. stop working.
  - Renaming a file or folder into a sub-folder with the same name as itself is now supported. For example, you can rename "moo" to "moo\cow" or even "moo\moo\cow".
  - Added an option to **Preferences / Folder Tree / Contents** to display all Folder Aliases in the tree instead of just user-defined ones.
  - Added support for basic audio metadata from .mka files (requires Windows 10 or codec/splitter to be installed).
  - In inline rename, **Ctrl-'** now works the same as **Ctrl+Shift+Up** (to copy the filename from the previous file).
  - Added **Preferences / Viewer / Viewer Pane / Expand and scroll (Left double-click)** option.
- 
- The (simple) *Select Files* dialog now has a drop-down history of previously used selection patterns.
  - The Advanced Find function can now correctly match .wav files by sample rate, bit rate, duration and codec.
  - Fixed an issue when copying files to FTP that could occasionally result in the new file not appearing in the destination folder until after a refresh.
  - Fixed a problem where progress dialogs could come to the front when other progress dialogs were created or activated.
  - If the Abort button was clicked in a progress dialog that had additional queued items attached, the abort confirmation message would block any other functions that use progress dialogs from running until the confirm dialog was closed.
  - Fixed initial enabled/disabled state of **Always Highlight Full Row** checkbox when the *Preferences / File Display Modes / Details* page first opens.

- Fixed problem with embedded command in a newly opened Lister that opens multiple new tabs; in some cases the new tabs could open in the original Lister.
- Fixed problem with **Go TABGROUPSAVE** command failing when selecting an existing folder from the dropdown and then appending a new name to it.
- Fixed problem with sequential numbering rename - if rename of a file failed and was retried multiple times before succeeding, the number assigned to the file was being incremented for every retry.
- The *Rename* preview now correctly indicates that files in subfolders aren't going to change if their parent folder is deselected.
- Fixed rendering issue in the *Rename* dialog's macro editor when visual styles (themes) are disabled.
- The Archives plugin now allows you to add and remove file extensions associated with most of its archive types.
- Fixed Advanced Filter Control *Compare* clause layout issue in non-English (when "Size" string is longer than "Date" string).
- The folder thumbnail "background" that Opus renders now scales for DPI.
- Inline rename on MTP devices (when using native MTP support) was broken.
- Made a change which means the Copy Handler context menu extension now works in Opus.
- Fixed "view as hex" option in the viewer not always being disabled if no hex viewer plugin was enabled.
- Image marking now works even if the configured tag collection contains invalid filename characters.
- Fixed problem with SSH connections where server certificate warnings (unknown key / mismatched key) were converted to ANSI before being displayed in a dialog (not an issue in English, but in other languages caused corruption of the message text).
- The paths passed to the **OnGetCopyQueueName** script event were incorrectly stripped to the root paths for non-disk drive letters (e.g. mapped network paths).
- The **Properties** command is no longer forcibly disabled in context menus when files from two or more different folders are selected.
- Actions shown in the *Undo Log* are now time-stamped.
- Fixed problem on FTP sites with cut-and-paste; pasting files to a sub-folder and then going up to the parent folder could end up in the wrong location.
- Breadcrumbs path no longer shows "Undef (FTP)" as the top level branch for FTP sites in some cases.
- Added **@noprogess** modifier which lets you disable the automatic progress dialog for a function.
- The **Set COLUMNSTOGGLE** and similar commands now recognise **audiocodec** as a synonym for the **mp3type** column.
- The properties dialog for files on FTP sites is now resizable.
- Fixed a problem when copying files using the *Select Destination Folder* dialog to pick the destination, which meant selecting a folder from the **Favorites** drop-down and then clicking **OK** didn't work properly.
- Fixed sub-branches of the Favorites tree not expanding after forcing the Folder Tree to refresh.
- Processing of **@disableifpath** and **@hideifpath** modifiers now works even if they are preceded by a `//comment` line.
- The breadcrumbs field will now show ghost paths for a folder if any depth of child folder hierarchy exists, instead of requiring the full hierarchy of the previous location.
- Fixed issue with breadcrumbs path field showing all "ghost path" and no "real path" when going back from subfolders underneath QuickAccess.
- Improved folder tree's handling of the Windows 10 Quick Access folder.
- Items below Favorites, Quick Access and similar folder tree branches now respond to icon and label/color changes without a forced refresh.

- Fixed issue dragging to folder tabs on the left or right if they had been scrolled but no longer needed a scrollbar.
- Layouts and Folder Tab Groups saved with tabs in Quick Access shortcuts will now be restored the same way, rather than navigating to the real paths the Quick Access shortcuts resolve to.
- The **{alias}** command sequence now resolves library paths to their real disk paths if possible, e.g. **{alias|libraries}/Tools/calc.exe** would resolve to the full path of *calc.exe* within the *Tools* library. Previously **{alias|libraries}** would turn into **lib://** which is no good for passing to external programs.
- Fixed a problem where another window could be brought to the front when the "confirm file replace" dialog closes
- On a slow machine, it was possible to right-click a file, choose rename, then select another file before the rename began and end up renaming the second file. This has been fixed.
- The **Select FROMSCRIPT HIDESEL** command now works.
- The **Preferences / Launching Opus / Startup / Run a command** option only worked if a user command was entered, but now works for any command.
- Fixed Rename *Find and Replace* mode presets saved from Opus 12 always having the **Ignore Extension** checkbox turned on when re-loaded.
- A **@hideif** test for a variable now works when the condition is negated (e.g. **@hideif:!\$glob:abc**).
- When files are filtered out of a recursive copy operation the progress dialog now adjusts the "total" counts to compensate.
- Lister Themes now save and restore the **Use for lister column headers** and **Use for lister scrollbars** options in **Preferences / Display / Colors and Fonts / Pane borders**.
- Fixed a bug where a toolbar button click could be ignored if at the same time a script triggered a particular change to the Lister (e.g. changed its state from source to destination).
- Fixed a problem where the *Delete* confirmation dialog would fail to activate if the command was preceded by a **@confirm** line.
- The regular expression option in **Folder Formats / Show filters** and **Hide filters** wasn't being respected when a folder format was loaded in a file display.
- Fixed DPI bug in Flickr sync dialog.
- Fixed incompatibilities reading some PDF metadata.
- A picture displayed in the viewer using the **Show VIEWERCMD=open** command is now properly rotated for its EXIF value if the option is turned on in Preferences.
- Fixed a Flat View / Grouped sorting error.
- Fixed appearance of non-functional "Share with" context menu item on the File menu.
- Fixed problem importing some filetype settings from filetype export files.
- Fixed toolbar layout error where a horizontal toolbar could overwrite a vertical one in some situations.
- Fixed crash which could occur if you went to a directory within an archive, then went up to the archive's parent folder.
- Improved metadata handling of .ai files.
- Fix for image viewer crash if you deleted the 2nd last image in a folder and the last image was invalid or could not be loaded.
- The context menu from very old versions of NotePad++ (which have been known to cause crashes) is now blocked by default.
- Opus Light users can now set the **HKEY\_CURRENT\_USER\SOFTWARE\GPSSoftware\Directory Opus\ContextMenuDebug** DWORD value to 1 to enable context menu debugging.
- Blocked erroneous "click" event being sent to script dialogs when a radio button gains the focus.

- Fixed expand/scroll viewer mode with certain images not making all pixels of the original image visible.
- Made a change to hopefully fix problem with Photoshop not recognising keywords set through Opus in some images.
- Setting the rating for a FLAC file is now supported.
- Fixed two problems with vertical folder tabs:
  - Tab tooltips were not updated when the tabs were scrolled.
  - When resized the tabs would automatically scroll to make the current selection visible.
- It wasn't possible to move the focus away from a drive dropdown toolbar control with the tab key.
- Activating a toolbar button by pressing its **&** key now returns focus to the file display after running the function.
- Scripting:
  - The script **Tab** object now has **backlist** and **forwardlist** properties, which provide a collection of **Path** objects in the back and forward history lists for that tab.
  - Fixed an issue with script columns that set the **multicol** property but then fail to return the data for all columns at once.
  - A new **DOpus.GetQualifiers** method allows you to get the state of keys like **Shift**, **Ctrl** and **Alt** in situations where you aren't already given them. See the updated F1 help for a discussion of when it makes sense to use this vs other methods.
  - A new **DOpus.TypeOf** method lets you query the type of an Opus script object (returns a string identifying the object type corresponding to the type names documented in the *Scripting Reference* section of the manual).
  - The **metadata.image.picdepth** property incorrectly returned **picheight** and has been fixed.
  - Fix for **Item.ShellProp** failing on a file if some of its properties could be obtained but not all of them.
  - The **OnViewerEvent** is now called synchronously. This may break some existing viewer scripts, but was needed because the events could come in a random order before and events like "create" were inconsistent as to whether viewer.current had the first file or was empty. (It will always be empty now; use "load" to get the first file.)
  - **Item** objects returned by the **FSUtil.ReadDir** method now have their **realpath** property set correctly.
  - Added **DOpus.favorites** script property which returns a **Favorites** object, allowing the user-defined favorite folder list to be queried and modified.
  - Added **DOpus.smartfavorites** property to give access to the smart favorites data.
  - Added **def\_value** property to all script objects, which allows the default value to be accessed like a normal property to avoid weirdness with some script languages.
  - Added **Show VIEWERCMD=dragsel** command which lets a script trigger drag scroll/select/expand actions when responding to an **OnViewerEvent** mouse event.
  - Added **Set NOSCRIPT** argument; currently only used with the **Set LISTERCMD=tofront** command, to prevent **OnActivateLister** scripts from firing.
  - Added **DOpus.favoriteformats** script property which returns a collection of user-defined favorite formats (**Format** objects).
  - Added **DOpus.Filters** script property to provide access to global filter settings.
  - Added **BusyIndicator** script object.

- File and folder labels can now be set as "pin to top". Any file with a label assigned to it that specifies pin to top will sort at the top of the file list, irrespective of the overall sort order. There's a new default Status category label (Pinned), although you note you'll need to reset the **Preferences / Favorites and Recent / Labels** page to defaults to see this in an existing configuration.
- The GIF plugin now fully supports the Expand/Scroll mouse button option in the viewer, including quick rendering and animation support.
- The **Change Attributes & Times** dialog now lets the time and date fields for **Creation** and **Last Modified** time be individually turned off (so that you can, for example, change the dates of selected files without affecting their times).
- Added support for the "non-content indexed" file attribute (the **SetAttr** command and metadata panel now let it be set or cleared, advanced **Find** lets you search for files with it set, and the file display displays it in the *Attr* column as *i*).
- The **{thumbnail}** infotip code can now specify a different size to the regular thumbnail size, by adding an additional **:** parameter (following the parameter for the border type, which must also be included). E.g. **{thumbnail:0:512}** for 512 pixel thumbnails with no border.
- In the **Rename** dialog, the options to copy and paste the list of names to/from the clipboard have been moved into a menu-button above the preview list. The menu also contains three new options: Prefix, Append and Reset. Prefix and Append let you add the clipboard content to the start or end of the existing names instead of replacing the names. If one line is in the clipboard, it will be added to every name. If multiple lines are in the clipboard, one will be added to each name, and blank lines can be used to skip names. (If the number of files is larger than the number of lines, the clipboard content will loop around). Finally, the Reset option clears any new names set via the same menu or via manually typing over individual names. As before, this functionality is only available when macro-building mode is turned off, via the adjacent checkbox.
- Added **@ifrunning** command modifier (and **SysInfo.FindProcess** script method).
- Added **Show in Label column** option for labels, which lets you prevent their name being shown in the Label column in a Lister.
- Added script dialog **click** event for listbox and listview items. The **Msg** object also has new **mousex** and **mousey** properties that give the mouse location (in screen coordinates) when the message was generated.
- It's now possible to define a hotkey that overrides the **Escape** key in the standalone viewer.
- The **Label Assignments** list in Preferences now uses the configured file display background color.
- Context menu items in the registry as "static verbs" now work correctly from the file display background context menu (e.g. Git for Windows).
- Fixed problem in List mode where the filename of a newly created folder could be displayed in the wrong location until the file display was repainted.
- Fixed Text viewer's "assume UTF-8" checkbox becoming stuck if turned on.
- Fix for drag & drop from TortoiseSVN repo browser creating zero byte files.
- The Info Tip font is now configurable through Preferences.
- The **{dlgsave}** code now allows the "save as type" field to be populated, e.g. **{dlgsave|Title|Default Name.txt|type=Text Files!\*.txt!Doc Files!\*.doc}**. Add **#** following the **type=** string to automatically include the "All files" item.
- The rename dialog new name field now allows **{sep:...}** to be typed, to allow script column values to be used in the rename operation (previously the colon would be blocked).

- The viewer was failing to indicate if the first image it opened with was marked unless the "Display Marked Pictures pane when a picture is marked" option was turned on.
- Fixed a fairly obscure problem which could cause inline rename to be cancelled automatically when middle-clicking on a file in power mode with an **OnActivateLister** script installed that automatically sets the active lister to be source.
- File tooltips triggered by holding down the control key are now clipped to the file display to prevent them appearing off-screen if the display is scrolled too far horizontally.
- Fixed a problem with inline rename where the file extension could incorrectly be displayed (if it was hidden) when moving from one filename to the next using the cursor keys.
- Fixed drag image/tooltip going behind submenus when dragging to toolbar in customize mode.
- When clearing a collection using **dopusrt.exe /col clear**, sub-collections are no longer removed (previously they would be removed from the list, but not from disk, which meant they would come back after a restart). You can now use **/col clear /full <name>** to remove all contents including sub-collections.
- The "Pin current folder to Quick Access" context menu command (when right-clicking on the root of the Quick Access folder in the tree) now works
- Rename macro builder now uses Consolas 9pt as its default font on Vista and above.
- Fixed filenames not lining up in the Rename macro editor with some fonts.
- After using "paste new names" in the Rename dialog, the preview's New Name column now auto-sizes.
- Fixed incorrect preview, and sometimes crash, in Rename dialog when using "Show preview of sub-folder contents" and a large number of parent folders.
- Fixed problem with the rename macro builder that could lead to an erase/insert/erase macro being incorrectly collapsed to erase/insert
- The image viewer's optional picture frame now scales with DPI.
- The Labels list in Preferences is now displayed using the configured file display background color instead of the system "window" color.
- Added a confirmation prompt when the dialog editor is cancelled if the dialog has been modified.
- The @noprogess modifier works from scripts now.
- If all folder aliases were displayed in the folder tree (instead of just user-defined ones), the user-defined ones didn't work.
- Fixed problems with archives not appearing in the tree underneath libraries in some cases.
- Using the **Set COLUMNS** command to set only one column (e.g. **Set COLUMNS=35mmfocallength**) would cause a crash in some cases.
- The **Set COLUMNS** and similar commands misidentified the **35mmfocallength** column as the **companyname** column.
- The Quick Access branch of the tree now displays the properly localized names of its member folders.
- Fixed a problem which could cause a crash if a function modified image files in a Lister and then changed folder.
- Fix for toolbar layout issues on high DPI systems after multiple soft restarts have occurred (e.g. after changing Preferences settings that require the program restarts itself).
- Label lists in **Preferences / Folder Options** now use the configured file text color as well as background color.
- On the **Label Assignments** page in Preferences, it was possible to open multiple label filter dialogs simultaneously, which could cause a crash when the dialogs were closed.
- The **Change Attributes** dialog now resizes to make sure the list of attributes is completely visible.
- Fixed problem where a label filter that used *Type Match Folders* could misidentify some archives as folders when in a library.

- Fixed a problem with labels not working properly on folders in the tree if they have localized filenames.
- Fixed problem with PDF metadata not decoding some strings correctly.
- If a toolbar button was highlighted with the mouse and then the mouse moved over an adjacent spacer, the button would remain highlighted.
- Fixed problem with progress dialog ending up the wrong size if it was minimized with the copy queue visible and all but the last item finished (removing the queue) while still minimized.
- Info tips will now display strings enclosed in angle brackets (e.g. <blah>) unless they're a supported html-style tag.
- Fix for double-clicking images if the Windows Photos app was set as the default image viewer, after the Windows 10 Creators Update.
- Fixed problem with metadata pane not setting exif tags in some circumstances.
- **Set JOBSBAR** command was missing from the manual.
- It's now possible to make a link to a drive root, e.g. **Copy MAKELINK=softlink E:\**
- Fixed problem introduced in 12.4 where you could not use AltGr+' -- actually AltGr + (the key bottom-left of Return, depending on keymap) -- to type special characters in some locales, as it was interpreted as the Ctrl+' which now copies the name from the file above.
- Scripting:
  - In the **OnOpenLister** script event, it's now possible for the event to fire after a Lister has finished opening (including opening all its tabs) as well as before. If the new **OpenListerData.after** property is **False**, you should return **True** from the event handler. The event will then be called again with **OpenListerData.after** set to **True** once all the tabs are open.
  - The **ImageMeta.latitude** and **longitude** properties now return decimal coordinates. The **coords** property now returns a string containing *latitude,longitude* as decimal coordinates.
  - Added **Command.filecount** and **linecount** properties
  - For combo edit controls in script dialogs, **Control.value.name** now returns the string entered by the user if they typed something in rather than picking from the dropdown list. The editable text can be set using **Control.label**.
  - In scripts, a **Dialog** object can now be used as the parent window for things that need one. E.g. a dialog can have another dialog as its parent.
  - The new **Dialog.disable\_window** property lets a dialog automatically disable a window while it's visible, and re-enable it again once the dialog closes.
  - Save dialogs shown via the **Dialog.Save** method now support a fourth argument to populate the "save as type" dropdown.
  - Fixed a problem with "multicol" script columns incorrectly being called for non-multicol columns.
  - The script method **Control.GetItemByLabel** was incorrectly documented as **GetItemByName**. Both versions work now.
  - The script **Blob.ToArray** and **ToVbArray** methods now take optional "from" and "size" parameters, similar to **Blob.CopyFrom**.

## 12.6 7th June, 2017

---

- Fix for crash which could occur when refreshing the folder display, opening new tabs, or renaming files in some cases.
- Fixed crash which could sometimes occur with certain command sequences in a button. (e.g. CreateFolder, then open the folder in the dual display.)
- The Filter Bar now automatically disables partial matching when a file type group is being matched. For example, **grp:Archives** will no longer match **example.rar.jpg**, even if partial matching is enabled.
- Fixed minor problem displaying some licence counts in the licence manager.

- Added support for the **Windows 10 Fall Creators Update** version of Microsoft **OneDrive**. OneDrive for Business and SharePoint folders synced by the new OneDrive client are also supported. Sync status for files is shown in the **Availability** column, and an icon representing the sync status is shown (by default) in the **Status** column (a new option in **Preferences / Folders / Folder Display** lets you turn this off if you want). The sync icons are also shown in the icon display modes.
- The new **P** attribute (for "pinned") is shown for files that are marked as *Always available on this device*.
- A new Folder Type format has been added to **Preferences / Folders / Folder Formats** which controls the default format for OneDrive folders (turning on the **Status** and **Availability** columns).
- A new [Launch Options](#) dialog is displayed when you drag an exe file to a toolbar in Customize mode. The dialog lets you choose whether the program will run on its own or be passed selected files, and similar options. You can set it to always use those options when dragging other exes to toolbars in the future (hold **Ctrl** to make it appear again afterwards).
- The tooltip for the **Clipboard PASTE** and **PASTELINK** commands (e.g. in the **Edit** menu) can now display a preview of the clipboard contents. To enable this in existing toolbars, edit the function for the command and add **%1** to the tooltip definition. For example, the new default tooltip for **Clipboard PASTE** is *Paste files and folders on the clipboard to the destination folder\n\n%1*.
- Added **Go REBUILDTREE** command. Equivalent to toggling tree off and on again.
- Added metadata support for the IPTC/XMP **Special Instructions** field.
- Added **Favorites ALIAS** command which lets folder aliases be added, modified and deleted.
- Added **Favorites** command **COPYTO** and **MOVETO** arguments, which cause the generated list of favorite folders to contain commands for copying or moving selected files to your favorite folders.
- **Copy TO=ask** and **TO=ask\$** now allow a default path to be specified, e.g. **Copy TO=ask:c:\data**.
- The new advanced options **custom\_time\_format** and **custom\_date\_format** allow you to override the standard system date and time formats.
- Pressing **Ctrl-T** in the Find-As-You-Type field now toggles the state of the **fyat\_firstchar\_repeat** advanced Preferences setting.
- **Preferences / File Operations / Progress Indicators** has a new option to turn off the slide animation when a queued operation begins. The animation is also automatically suppressed when using Remote Desktop or if client-area animations are turned off system-wide.
- The dialog for configuring folder thumbnails now has a **Defaults** button to reset just its settings without having to reset the whole **Thumbnails** preferences page as well.
- Added compatibility with some incorrectly formatted BMP images.
- The folder tab strip's empty space can now be clicked to activate that side of the Lister (unless the **tab\_click\_nofocus** Advanced Preferences option is on).
- Clicking a file display's scrollbars will now make it the source in all cases. (Previously, this only happened for Details and Power modes).
- The **Go FOLDERCONTENT** command now accepts the **useshell** keyword to force the folder to be enumerated using the shell (to get shell ordering and display names).
- Added **Creator** and **Producer** fields to the list of file display columns (currently these columns are only supported by PDF files - the metadata panel already supported them). You can also search on them using the **Advanced Find** function.
- **Preferences / Folder Tree / Contents** now has an option to hide **Creative Cloud Files** from the Tree (only present if Adobe Creative Cloud is installed on the machine).

- The **Select ADVANCED** command now allows the name of a saved filter to be specified to open the selection dialog with that filter already loaded. e.g. **Select ADVANCED=MyFilter**.
- The **Set Attributes** dialog can now copy the various "document date" fields (created, edited, last saved) to the modification and creation time fields.
- The "Save Tab group" dialog now displays an indicator in the drop-down list to show which tab group was most recently loaded.
- The **SetAttr META** command now accepts **usercomment** as a synonym for **comment**.
- Opus now displays a confirmation message before generating a context menu for a large number of files. The limit defaults to 1000 but this can be changed with the **context\_menu\_max\_files** advanced option. Set it to **0** for no limit like before.
- The **Browse** button in the advanced function editor now displays a drop-down menu letting you browse for files or folders, and also insert a folder alias or FTP site.
- Updated 7z and UnRAR components to current stable versions.
- Improved the behavior of the **Preferences / Folder Tabs / Options / Preserve folder tree expansion when switching tabs** option.
- Improved the performance of the **Browse for folder** dialog when a network path is pasted into the path field.
- Added **Composers** and **Conductors** columns to the **Music** category.
- Added setting for "off" file display background color to **Preferences / Display / Colors & Fonts**.
- The **Replace File** dialog now shows the locations (not just names) of the files, and you can now hover over extremely wide, truncated strings to see tooltips with their full details.
- In path fields, when the path completion drop-down is visible, you can now use **Tab** and **Shift-Tab** for the next and previous matches, as synonyms for the **Down** and **Up** cursor keys.
- In path fields with path completion enabled, when neither the drop-down nor the path-completion pop-up list is visible, the up/down cursor keys now do nothing. Previously, they would trigger immediate navigation to an entry from the history list, or whichever list the path field's drop-down was set to. Once the completion pop-up or drop-down list is open, the cursor keys will move through the list. (Note that **F4** is the hotkey to open the drop-down, as is standard in Windows.)
- Breadcrumbs path fields have a new **DragIgnoreSelf** option which blocks drag & drop from the field to itself, to avoid accidents.
- Breadcrumbs path fields have a new **EditEnd** option which positions the cursor at the end of the path string when you start editing the path, making it easier to type a sub-directory or modify the last path component. By default, the whole path will be selected, making it easier to type a completely new path, as before.
- Added **Ctrl-L** as a default hotkey for the path field, and changed the default **Calculate Folder Sizes** hotkey from **Ctrl-L** to **Ctrl-K**. (This will only affect new configurations, unless you revert your toolbars to the factory defaults or make similar changes yourself.)
- JPEG2000 decoding now respects EXIF rotation, if enabled.
- The colors used by the text viewer plugin are now configurable via **Preferences / Viewer / Plugins / Text / Configure**.
- Added an option to suppress the warning about slow searches due to unindexed folders when using Windows Search. **Preferences / Miscellaneous / Advanced: search\_warn\_nonindexed**.
- Added a Preferences option to disable the underline when single click mode is active and the mouse hovers over a filename. **Preferences / File Displays / Mouse / Underline items on hover**.
- Added new **unique** parameter to **Close ALLLISTERS=collapse,unique** which will collapse all tabs in all windows into the current Lister without opening any duplicate tabs.
- Added new advanced settings, **notify\_max\_time** and **notify\_min\_items**, which can diagnose rare situations where the file display cannot keep up with the number of filesystem change events being produced. See the help file for more detail.

- The **Prefs BACKUPRESTORE** command now respects the **TO**, **PASSWORD** and **DESC** arguments to override the default backup filename and pre-supply a password or description when using the interactive backup UI. (Previously, these only worked when doing fully automated backups.)
- Configuration backup filenames now use **yyyy-MM-dd** as the default date format, so the backups are easier to sort.
- In path fields, pressing the **End** key twice now turns into a path slash. This is to make path completion easier with keymaps where pressing the path separator keys is difficult.
- Diacritics are now ignored when grouping by name (and other text fields). For example, *Ábc* will now group under **A-H** rather than in the **Unspecified** group. (Vista and above).
- Context menu items that showed up as a long "internal" string (e.g. *@{Microsoft.Windows.Photos.blahblahblah}*) are now displayed correctly.
- Made the New Text Document menu and command work better when there is incorrect data in the registry.
- Copying music file cover art in the metadata editor via drag and drop now works correctly (previously the copied cover art would not be saved to the target file).
- DPI scaling is now applied to the positions of toolbars on shared rows and columns.
- Improved alignment of drive letters when added to the corners of drive buttons.
- **Status Icons** changes:
  - The **Status Icons** column now sorts and groups by the names of the status labels, so you can change the order by editing the names in Preferences. (Previously, the sorting/grouping of the icons was somewhat arbitrary.)
  - Added a small space between status icons.
  - Status Icon overlay in the full-screen viewer now moves down and out of the way when the toolbar is made visible by clicking the top of the screen.
  - The **Status Icon** column now shows "..." to indicate when there are more icons than will fit in the column.
  - Group names should now be correct when grouping by Status Icons which have their **Show in Label column** option turned off.
  - Fixed **Preferences / Viewer / Appearance / Show status icons** not working if the image viewer started as a normal window and then went full-screen.
- A tab that has its loading deferred until tab activation will now correctly add the path to the history list when a sub-directory is entered.
- The file log no longer truncates paths to 260 characters when saving it as a text file.
- Added option to save the file log as a CSV file, and a "copy to clipboard" context menu.
- Opus will now attempt to preserve zip file metadata (rating, tags, etc) when the archive is modified.
- Fix for crash caused by the Windows 10 Creators Update which could happen at a random time after the credentials dialog was displayed for a network drive.
- Advanced Find wasn't able to find labels that had their **Show in label column** option turned off.
- The **Go TABUNDOCLOSE** command now preserves the tab color setting of the closed tab.
- Fixed FTP login problems with SwiFTP android server.
- A separator immediately following a list of buttons generated by **Properties SETLABEL** was not displayed.
- Keyboard accelerators (e.g. in Preferences) where the key in some non-English languages (e.g. Japanese) appears at the end of the translated label (e.g. *xxxxxxx (Y)*) now work correctly.
- Fixed issue with certain corrupt RAR files which could cause Opus to keep trying to open them instead of giving up on failure.

- Fixed folder with a name like `::{018D5C66-4533-4307-9B53-224DE2ED1FE6}` appearing in Desktop if Opus was configured to show the OneDrive folder there but OneDrive was broken or disabled.
- Fixed Synchronize tool not properly matching files between sides if special folders with localized names were below the starting point of the sync, and **Preferences / Folders / Folder Display / Display localized folder names** was on. (e.g. If you synced the parent of the special *Documents* or *Music* folders, either having moved them to folders with different names or when using a non-English version of Windows).
- Fixed problems with labels in the tree disappearing when the label Preferences are edited.
- The two default context menu items for files in a collection were not translated when changing languages.
- If the file display format is set to group by Labels, the "collapsed" option now works correctly.
- When using the **Copy As** function from a zip file, the "enter a new name" dialog continued to offer the first file's name as a default for second and subsequent files.
- Fixed problem when copying out of zip files in flat view mode - only files in the root folder were copied, files in sub-folders were ignored.
- Fixed shared rename presets from older versions sometimes getting numeric names when imported into Opus 12.
- Copying files out of zip files now obeys the "unattended" settings correctly.
- The **Copy MAKELINK** command now works from a file collection.
- The **SetAttr META** command now reports when errors occur, and allows you to skip over errors and continue setting metadata on subsequent files.
- The **Preferences / Viewer / Appearance / Display full path** option works again.
- Flickr photo syncing works again (Opus now supports the *OAuth* authentication system which Flickr has recently switched to).
- Saving an image from the viewer (e.g. after cropping it) now preserves the original EXIF data from the source image if possible.
- Fix for new **Go REBUILDTREE** command not working properly in a dual display Lister.
- Fixed internal "IDL:" type strings appearing in tooltips in generated **Go FOLDERCONTENT** menus.
- Fixed crash if a button or script rapidly changed the status icons of the file open in the image viewer.
- Fixed some arguments to **Properties SETLABEL** being ignored if a single button used it multiple times.
- Fixed script and FTP logs not line-wrapping if the windows they were on were never resized.
- If the **Rename** dialog's script panel was open, refreshing the rename preview cleared both the dialog's script output and the global script log. It now only clears the dialog's output.
- Fixed Opus not recognizing the date taken field in some image files (specifically, files that use the Xmp "CreateDate" field rather than the Xmp or Exif "DateTimeOriginal" fields).
- Fixed problem with **{parent}** code in **Rename** dialog revealing an internal path when in the root of a library.
- Clicking the **Edit Labels** link from a folder format dialog opened via the Preferences **Folder Formats** page will now save any changes made in that dialog before it closes.
- `.dcf` files (created by drag & drop from the toolbar) are now written as UTF-8 if they contain any characters above the ASCII set (> 0x7f). Previously they would be saved as UTF-16, and only if they contained characters above 0xff.
- On Windows 10, if the file display is showing a tooltip and the mouse is over it, using the mousewheel now scrolls the file display. (By default, Windows 10 delivers mousewheel events to the window under the mouse pointer, not the window with focus. The tooltip now forwards them to the file display.)
- Any file display tooltip is now hidden when you scroll with the mouse wheel.
- When using **Alt** + mouse wheel to scroll both sides of a dual-display Lister at once, fixed the sides not always scrolling the same distance if wheel acceleration was on.

- Fixed a problem with file change notifications getting lost when a large number of changes are generated at once (particularly on slow devices like network shares).
- Windows key system hotkey overriding works again after the Windows 10 Fall Creators Update.
- Fix for context menu icons from the "File Menu Tools" shell extension (and possibly others) having black fringes.
- Fixed crash running **SetAttr META \*** command on some MP3 files.
- Tree label filters:
  - Fix for drive roots and certain folders under Desktop being matched by label filters that specified only matching files.
  - Fix for drive roots in the folder tree not working properly with label filters which included path or name clauses.
  - Fix for library roots in the folder tree not being colored by label filters until a refresh.
  - Fix for the Desktop branch's user profile folder not updating for label filter changes after the tree was first built.
- Fix for rare situation where the folder tree opened with branches expanded to two levels instead of one.
- Fixed rename via the tree being canceled if you clicked a folder in the tree and pushed F2 to rename it too quickly, when **Position selected item in the middle of the tree** was on.
- The Size On Disk column for folders now takes into account full-volume compression. (This was already the case for files, as well as for folders where individual files were compressed but not the whole volume.)
- The standalone viewer no longer blocks things like the Calculator key on certain keyboards and mice.
- Standalone viewer hotkeys now allow you to re-bind special keys such as the Calculator one on some keyboards. (This was already possible for Lister hotkeys.)
- Fixed rare situation where deleting a standalone viewer hotkey did not work or crashed.
- Fixed mouse over viewer drifting while the shift key was held down in some DPI-scaling situations which Windows handles incorrectly (RDP from standard DPI client to high DPI server).
- Fixed crash if you ran **Copy TO=ask**, selected a library, and then edited the path.
- Fixed problem with status bar incorrectly showing a double bottom border in some situations.
- Fixed some issues with sub-collections. For example, after deleting a sub-collection, you would not be able to create a new one with the same name and path until Opus was restarted.
- Fixed **Remove from Collection** not working via right-click context menu.
- Fix/workaround for bug in Windows 10 where dragging a large number of files could result in the drag cursor and description being drawn incorrectly, or even a crash in some cases.
- Fixed problem in some zip files where the description for a folder within the zip could be generated from one of the files within the folder.
- Fixed duplicate entries when the Quick Access folder is shown in the Recent list in the folder tree.
- Fix for **{allfilepath}** and similar forcing a space before the first path, if there wasn't one already, when automatic quoting was off.
- Opus now handles ERROR\_NO\_SUCH\_LOGON\_SESSION (1312) errors correctly when connecting to a network share.
- Fixed crash which could occur sometimes if you ran (a large number of) Find operations in parallel from the same script.
- Tooltips for buttons at the bottom of the screen (e.g. docked toolbars) will no longer appear overlapping the mouse pointer (which caused them to vanish as soon as they appeared).
- Tooltips on the status bar are moved up a bit when appearing above the mouse, to help avoid accidentally popping them by moving the mouse into them.

- Fixed the Rename dialog's "clipboard" button being pushed off-screen when the presets list was resized.
- Fix/workaround for Windows bug where double-clicking a .URL shortcut in a folder path with non-ANSI/OEM characters would result in an error message, at least with some web browsers.
- Fixed **Close ALLLISTERS=collapse** so it no longer turns on the dual file display with an empty folder tab if none of the collapsed windows had dual displays.
- Fixed issue with rename preview showing the wrong file numbering when two folders with the same name (e.g. via Find or Flat View) were selected for recursive renaming.
- Fixed Rename Preset Save-As ignoring edits of the preset name if you selected an existing preset from the tree in the prompt.
- Fixed incomplete wav thumbnail being cached if thumbnail generation was canceled part-way through.
- Fixed crash when viewing thumbnails of certain WAV files (or loading their thumbnail into the viewer pane)
- Fixed error 32 (file in use) when dragging files from WinRAR to the folder tree.
- Scripting / Plugin changes:
  - Fixed **DVP\_LoadText** plugin API.
  - Added **FSUtil.GetErrorMsg** script method to get (localized) plain text error message from an error code.
  - Added **QuickFilter** object (accessed via **Tab.quickfilter** property) which provides information on the state of the quick filter in the tab.
  - The **DOpus.Strings.Langs** property can now be dereferenced directly (e.g. **DOpus.Strings.Langs(0)** now works).
  - Added **DOpus.Strings.HasLanguage** method to test if a particular language is included in the string resources.
  - Added **Item.shortpath**, **Path.shortpath** and **Path.longpath** properties.
  - Added **Tab.displayed\_label** property which returns the currently displayed label of the tab (whether a custom label has been set or not).
  - The **FSUtil.Hash** script method can now calculate sha256 and sha512 hashes.
  - Added a warning message if your rename script return an object which cannot be converted to a string. In particular, this helps if you mistakenly do something like 'return new String("Hello World")' from JScript. (You can make that work by removing 'new' or calling '.toString()' on the object.)
  - Scripts can now pass a **Tab** object to the **Go TABPOS** command to reposition tabs other than the currently active ones.
  - For the **FSUtil.GetShellProperty** and similar methods, properties that are returned as SAFEARRAYs (e.g. the shell's "Composers" column) are now converted to Opus **Vector** objects automatically (since JScript can't easily handle SAFEARRAYs).
  - Clicking the **Abort** button in a progress dialog obtained via the **Command.progress** property no longer aborts the script (instead, the abort event can be polled for via **GetAbortState** as documented).
  - The **Aliases.Add** script method now correctly updates the **Aliases** object when replacing/modifying an existing alias.
  - Reading the label property of a **Control** object referring to an editable combo box now works correctly after the dialog has been closed.
  - The minimum value setting for a numeric edit control was being ignored unless the maximum value was also set.
  - Fixed problem with the **Script.RefreshColumn** method which could cause script columns to stop working until the folder was refreshed (e.g. by pressing **F5**).

- Fixed problem with script columns not being generated in *Find Results* collections until **F5** pressed.

## 12.8 24th April, 2018

---

- Added support for *Dropbox Smart Sync* (Opus now recognises when files are "available online only", can display thumbnails for offline files, etc).
- Added the ability to blur filenames and paths in the Lister, in order to take screenshots without revealing potentially sensitive information. The default **Help** menu has a new **Secure Screenshot** command in it which lets you take a secure screenshot of the current Lister or the whole desktop (with all Listers blurred).

Note that at the moment there is no blurring in secondary windows (progress indicators, error dialogs) or in dropdown menus (favorites, breadcrumbs path, etc).

The **Clipboard SCREENSHOT** command can take a screenshot of either the current Lister or the whole desktop, with or without blurring of filenames. By default the screenshot is placed in the clipboard, but it can optionally be saved to the desktop automatically. The command can also display an optional countdown timer for taking more complicated screenshots.

- The **Set BLURFILENAMES** command can be used to manually turn blurring on and off in the current Lister, if you want to use an external screenshot tool.
- Added basic support for WSL (Windows Subsystem for Linux):
  - Added **CLI DOSPROMPT=wsl** command to open a linux shell (assuming WSL is installed in Windows 10).
  - The **Clipboard COPYNAMES** command has a new **wsl** argument that copies filenames in WSL (Linux) format, e.g. instead of **C:\Test** it would copy **/mnt/c/test**.
  - The **/mnt/** format is also understood by path fields, etc (anywhere aliases work).
  - Added a new WSL Command mode to the FAYT; this lets you run linux commands (providing WSL is installed under Windows 10) from the FAYT like you can already run DOS commands. By default the activation key is | (vertical bar).
- Added **Clipboard COPYNAMES quote** keyword, to force copied pathnames to be quoted whether they contain spaces or not.
- The **Rename** tool has a new mode, **Regular Expressions + Find And Replace**, which combines both modes (lets you search and replace within filenames using regular expressions, without having to construct a pattern to match the whole name).
- The **Duplicate Finder** tool now has a **Size** mode which only finds duplicates based on their size (ignoring name, date and contents). The **Find** command has a new **SIZEONLY** argument to activate this mode from a command.
- Buttons can now be hidden or disabled based on the number of selected files or folders. For example, **@hidenosel:minfiles=2,maxfiles=4** will hide a button unless 2, 3 or 4 files are selected.
- The **{foldercontents}** and **{foldersize}** infotip fields can now be configured to control what they show:
  - **noprefix** can be used to remove the normal prefix added to the field (e.g. **{foldersize:noprefix}**)
  - **files** and **dirs** can be used for **{foldercontents}** to only show one type of content (e.g. **{foldercontents:files,noprefix}**)

- The **Preferences / Miscellaneous / Advanced / use\_color\_management** option now allows an external ICC file to be selected (as well as the built-in sRGB profile). Additionally, PNG files with embedded ICC profiles are now supported as well as JPEGs.
- The **Show** command now has **POS** and **SIZE** arguments that let you specify a position and/or size for the standalone viewer window.
- Added **Set VIEWPANELOCK** command. When the viewer pane lock is turned on, it will continue to display its current image even if the file selection is changed in the Lister.
- Folder tab group changes:
  - Menus for selecting folder tab groups now highlight the "active" group, i.e. the one which was last loaded. You can use **Go TABGROUPLIST=nohighlight** to prevent this on user-editable toolbars and menus.
  - When saving over a tab group, the suggested group name now comes from the display you are saving from (if applicable) and not always the source display.
  - When saving over a tab group, the dialog now opens with the existing tab group name pre-selected.
  - You can now use **Go TABGROUPSAVE=!forget** to tell the file display to de-couple itself from any loaded tab group (e.g. so the group's name is no longer selected in the list of tab groups).
  - The **Go TABGROUPSAVE** command also now has **!closeall** and **!nocloseall** keywords which override the state of the **Close all other tabs** flag. These existed previously but were undocumented, and the way the flag behaves when saving over existing groups has been improved slightly.
  - You can now use **Go TABGROUPSAVE=!current** to save over the last folder tab group which was loaded. If no group is loaded, you will be prompted to name a new one, unless you add **!quiet** as well. You can also use **!unless** to skip saving if a particular group is currently loaded. Details in the manual.
  - You can now use **Go TABGROUPLIST=savecurrent** on a toolbar to generate a list of buttons for each folder tab group where any changes to the current group (if any) will be saved automatically before switching to another group. Note that this does not cover saving the current group in other situations like closing the Lister, and consideration is needed if you use this with multiple Listers at once. More detail in the manual.
- The metadata panel can now be used to set the *Lens Type* EXIF field.
- Added .mkv to the default list of extensions viewed via Generic ActiveX. This should mean that if the Opus Movie plugin cannot view an MKV then Opus will fall back on the Windows Media Player ActiveX control, which can view them with newer versions of Windows.
- The metadata panel now notices and updates external changes to the metadata of files it's currently showing in more cases (e.g. if the file is modified externally using a "safe save" technique the panel should now notice).
- Made changes to the file notification system to improve performance when copying files across a network.
- The dialog confirming that a copy operation has been queued now has a checkbox on it letting you disable all further confirmations (as well as the existing checkbox which disables them for the current queue only).
- Opus's Select Folder dialog now has a drop-down providing quick access to any paths currently open in a Lister or tab.

- Added a workaround to try to prevent TeamViewer from locking up Opus when using the Image Convert function. Opus now also tries to detect TeamViewer and add itself to the exclusion list for TeamViewer's QuickConnect feature, since it seems to cause nothing but problems (for lots of software, not just Opus).
- *WebP* images can now be displayed in the viewer, preview pane, thumbnails. The image converter can convert from WebP to other formats (but not the other way around). Alpha channel transparency is supported. Metadata, tiled images (over 16kx16k) and animated images are not currently supported.
- The advanced command editor now allows commands to be set as **WSL Script** functions; similar to **DOS Batch** mode, this runs the command as a WSL (Bash) script. Note that WSL needs to be installed from the Windows Store.
- Enabled several columns for movie files that previously only worked for music files (even though the metadata panel could show them for movies). The columns are **Year**, **Genre**, **Title**, **Artist**, **Encoded By**, **Initial Key**, **Composers**, **Conductor**, and **Producers**. Also added **Directors** column since that was missing altogether.
- The **Preferences / File Displays / Border** page now lets more than one toolbar to be selected for the file display border.
- Under **Preferences / Toolbars / Appearance**, the maximum allowed button spacing values now scale with DPI. (The values themselves already scaled when you moved between machines or DPIs, but the maximum the UI let you enter was always 10 pixels until now.)
- The vertical button spacing option in **Preferences / Toolbars / Appearance** can now be applied to individual buttons on vertical toolbars or items in pop-up menus via two new options on the same page.
- Added **Preferences / Folder Tabs / Options/ Click selected tab** option which lets you control what happens when you click the already active folder tab.
- A new **Extra Line Padding** option in **Preferences / File Display Modes / Details** and **Power** which complements their older **Extra Line Spacing** option. Padding makes each line taller, adding extra space inside the clickable area, similar to using a taller font. Padding is scaled with DPI. (On the other hand, spacing increases the gap between items and is outside the clickable area. With large spacing you get gaps between items which act like the lister background when clicked. Spacing is not DPI scaled as you will probably only want exactly 0 or 1 pixel of it: You can set spacing to 1 pixel to prevent the overlap of selection boxes on newer versions of Windows which causes a thin line between selected items. For everything else, you probably want to use the new padding option.)
- Improved the interaction between grid lines and extra line spacing in Details and Power modes. If you notice any glitches in how the grids and backgrounds behind/between files are painted, please report them.
- Added Preferences option **Folders / Folder Behaviour / Ignore junctions and softlinks** when calculating folder sizes. Also added the **GetSizes IGNOREJUNCTIONS** argument for to override the option setting.
- Added new Preferences option **Folder Tabs / Options / Process file changes in background tabs**. When this is turned off (the default is on), tabs that aren't visible will no longer process file change notifications. Instead they'll be flagged as dirty, and automatically refreshed when you switch to that tab.
- Added **Recent** command **COPY** argument which lets generated recent list buttons copy (or move) selected files to recent folders automatically. The **KEYARGS** argument also supports the use of the **Copy** command in this context.
- Updated **7z.dll** to version 18.01 to address two potential security issues ( <https://landave.io/2018/01/7-zip-multiple-memory-corruptions-via-rar-and-zip/> ). Note that we don't think either would affect Opus in its default configuration, but they could be a problem if you switch Opus from Unrar.dll to 7z.dll for RAR

files, or if you turn on the ZipX archive type via Preferences.

- Added the **Set TABPOSITION** command that lets tab position be modified on a per-Lister basis.
- Added the **Set LISTERSIZE=auto** command, which automatically resizes the Lister to fit the current columns (in details/power mode only).
- Opus can now show product version, product name, company name and digital signature status for MSI installer packages.
- AudioTags plugin updated to populate Composers and Conductors file display columns for FLAC, iTunes/M4A, Ogg Vorbis and APE audio files.
- The Jobs Bar text and background colors can now be configured in Preferences.
- When pasting a clipboard image to a file in a Lister the busy indicator is now shown to indicate activity (as otherwise it can sometimes not be obvious that anything is happening).
- The viewer now shows a wait cursor while saving over the current image, since it can take a while (e.g. huge PNG images).
- The **CLI DOSPROMPT** command will now use the current folder set by the CD instruction (if supplied), e.g:

```
cd "c:\program files"  
CLI DOSPROMPT
```

- Added **Select GROUPNAME** argument to allow selection based on file group (when file display is grouped).
  - Combine with **PATTERN** (or **NOPATTERN**) argument to select files in a specific group or groups.
  - You can also use it to give focus to a specific group header (**Select NOPATTERN GROUPNAME <name> SETFOCUS**).
- Added built-in folder alias **/dropbox** (when Dropbox is installed).
- Added support for checkboxes to listviews in script dialogs.
  - The listview control can have checkboxes set to **Automatic** or **Manual** (as well as **Off**).
  - In Automatic mode, the state is changed automatically when user clicks the checkbox. In Manual mode, you're notified of the click but must change the state yourself.
  - The **Msg** object has a new event **checked** that indicates that the checkbox was clicked.
  - The **Msg** object has a new **checked** property that indicates the old check state (for Manual mode) or new check state (for Automatic mode).
  - The **DialogListItem** object has a new **checked** property that lets you get or set the check state for an item.
  - Check states are: **0** (unchecked), **1** (checked), **2** (indeterminate), **3** (unchecked/disabled), **4** (checked/disabled), **5** (indeterminate/disabled)
- Added **Blob.Find** and **Blob.Reverse** script methods.

- Added **Viewer.AddFile** and **Viewer.RemoveFile** script methods to modify the list of viewed pictures in an existing viewer.
- Added a new **FileGroup** script object which exposes information about a file group (when a **Tab** is set to group by a particular column).
  - The **Tab** object has a new **filegroups** property which returns a collection of **FileGroup** objects representing all file groups in the **Tab**.
  - The **Item** object has a new **filegroup** property which returns a **FileGroup** object representing the group the item is a member of.
- Added scripting **FileAttr** object, to better represent file attributes.
  - A new **FileAttr** object can be created by the **FSUtil.NewFileAttr()** method.
  - The **Format** object's **hide\_attr**, **show\_attr**, **hide\_folder\_attr** and **show\_folder\_attr** properties now return **FileAttr** objects instead of strings.
  - The **Item** object has a new **fileattr** property that returns a **FileAttr** object (the existing **attr** and **attr\_text** properties are unchanged).
  - The **File.SetAttr()** method now accepts a **FileAttr** object as well as a string.
- If a string is passed to **File.SetAttr()** (rather than a **FileAttr** object), + and - can be used to set and clear attributes rather than replacing the attributes entirely. For example, **File.SetAttr("-r")** to clear the read-only attribute.
- Script columns can now use the additional types **graphrel**, **igraphrel** and **percentrel**. These are similar to the existing types **graph**, **igraph** and **percent** except that the values your script provides do not need to be limited to the range 0 to 100. Instead, Opus will keep track of the smallest and largest values returned and automatically calculate each file's relative percentage. You can also use the types **graphrel0**, **igraphrel0** and **percentrel0** which always use 0 as the lower-bound value (and Opus will simply keep track of the largest value you return).
- The **ScriptColumn** script object has new properties (**graph\_colors**, **graph\_colors2** and **graph\_threshold**) that let you control the color of graphs drawn in a graph column.
- Script dialog controls now have properties and methods to get and set their position within the client area of the dialog. The properties **x**, **y**, **cx** and **cy** let you get and set the left, top, width and height of the control. The **SetPos**, **SetSize** and **SetPosAndSize** methods let the position and size be set in a single call.
- Scripts that open files for writing using the **FSUtil.OpenFile** method can now specify "f" as part of the mode string to automatically remove the read-only attribute from existing files.
- Scripts that ask to overwrite read-only (**R** attribute) files will no longer trigger unnecessary UAC prompts.
- **FileSize** objects can now be initialised from a **Blob** as well as the existing value types. The **Blob** must contain exactly 1, 2, 4 or 8 bytes. The new **ToBlob** method lets you convert a **FileSize** to a **Blob**.
- String dialogs shown by the **Dialog.GetString** script method now support the **select** property.
- **StringSet** objects can now be initialised from automation arrays (vbscript) and **Vector** objects.
- Added the **Func.argmap** property which presents the arguments for a script command as a **Map** object.

- Added a way for a script that implements the **OnDoubleClick** event to indicate that it wants to be invoked with only the path to the double-clicked item rather than a full **Item** object (which may be slow to construct, e.g. on network paths):
- Set the new **ScriptInitData.early\_dbclck** property to **True**.
- **OnDoubleClick** will then be called with a new property **early** set to **True** in the **DoubleClickData** object.
- When **early** is **True**, the **item** property is not present; instead, the new **path** property provides the full path of the object, and the new **is\_dir** property indicates whether the item is a folder or file.
- When the **OnDoubleClick** method returns, it will be called a second time, with **early** set to **False** and a full **Item** object available in the **item** property.
- If the script sets the new **skipfull** property to **True** in the **DoubleClickData** at the "early" stage, the second call to **OnDoubleClick** doesn't occur.
- Fixed incompatibility with TumaSoft's PresetViewer Argus in the Opus viewer pane (for viewing Photoshop brushes, shapes, and so on).
- Fixed a crash in the script dialog editor with a particular combination of control modifications.
- Fixed a problem where modifying a 7zip archive that was open in the file display could cause the Lister to jump to the parent folder.
- Fix for submitted crash dump involving Flickr Sync.
- The tooltips for Go commands which use explicit paths can now use %1 to insert the path into the tooltip string.
- Fixed the folder tree showing the same library twice when the library was the Lister's default path and libraries were configured to appear under the Desktop branch.
- Fixed grouping by Label or Status Icon putting each file into its own group in the previous version.
- The maximum length of the string that can be entered into a **{dlgstring}** dialog has been increased to 2048 characters.
- Fixed email address field from appending a backslash after using completion or pushing End twice.
- Path-completion controls in various dialogs will no longer submit the dialog if you push return while the completion drop-down is open. They now select the chosen path, close the drop-down, and give focus to the edit control. You can then use tab to move to other controls, or push return a second time to submit the dialog.
- Added a workaround to prevent corrupted files when dragging and dropping from UltraISO to an Opus window.
- Fixed an error which could leave large (> 2 GB) image files locked after trying to extract metadata from them.
- Fixed incorrect size/positioning of folder thumbnails using a single image (folder.jpg) and a non-square thumbnail size set via **Show THUMBNAILSIZE**.
- In Preferences, DPI scaling is now applied to the maximum size you can set marked image thumbnails within the viewer.

- Fixed height of marked image thumbnails in the viewer if the thumbnail size was not configured to be square.
- Fixed file display scrollbars not updating in existing windows when line spacing was changed in Preferences.
- Fixed a bug which meant a file's ADS streams may not have been preserved when converting an image in place with the **Image CONVERT** command.
- The **Image CONVERT** command now preserves any XMP metadata embedded in the image file (previously only EXIF data was preserved). For example, in PNG files tags are stored in an XMP tag and previously weren't preserved.
- Fixed some very minor DPI issues with the search field (top right of default toolbars, and bottom left of Preferences dialog).
- Fix for separators in the Layouts list appearing in the wrong place within the Desktop context menu, if you had more than one separator.
- Fixed a situation where the viewer failed to display Office documents if they were currently open in Microsoft Office.
- Workaround for Microsoft Word documents sometimes appearing clipped in the viewer with newer versions of Microsoft Office.
- Removed OpenOffice and Adobe Flash extensions from the list which Opus will attempt to view via Internet Explorer by default, since these days they are more likely to cause a Save As dialog to appear than to result in a working viewer via IE. If you were still relying on IE plugins for any of those formats, you can add their extensions back via Preferences.
- Microsoft Office and other preview handlers are now cached by the viewer pane so that viewing one document and then another will be faster.
- Fixed problem where the rename **Automatically number files when names clash** option could get confused if the original name ended in a number followed by a ) character (e.g. **ABC (2018-02-06)** ).
- Tidied up the way accelerator keys are assigned in the **Rename** dialog. (e.g. Previously, if you opened the dialog in *Find and Replace* mode, the **Alt+F** key for refreshing the script preview would not work.) Some of the keys have changed as a result.
- When copying files out of non-zip archives (ones handled by the Archives plugin), you can now retry after a disk full error.
- In scripting, if a file had no tags, the **Item.metadata.tags.count** property was undefined instead of reporting **0**.
- Fixed issue which could cause an infinite loop when performing certain rename operations recursively.
- Fixed an issue when running the **Clipboard PASTE** command from a script that could cause it to incorrectly paste to the destination file display instead of the source.
- Fixed problem removing all tags from a PDF file that had no other metadata.
- MD5 accuracy slider in the **Duplicate Finder** tool has up/down cursor key direction swapped.

- Status bar codes for showing the name and other details about the last-selected file now work in all types of folder.
- Fixed an empty archive being left around if archive creation failed or was cancelled, when using the **Add To Archive** dialog, or when the destination archive required elevation to create.
- Fixed another issue where modifying a non-zip archive could cause the Lister to react as if the archive had been deleted (e.g. go up to the parent path).
- Fixed the **Favorites ALIAS=set** and **ALIAS=delete** commands when used on toolbars. They previously only worked via hotkeys and scripts.
- **@hidenosel** and **@disablenosel** patterns and file/dir limits now work in the context menus for the folder tree and file display.
- The **New** context menu in Opus no longer filters out types that have identical *content type* specifications. This check (and questionable filetype registrations by other programs) may have resulted in the **New Text Document** menu being missing on some machines. Note that you may now see two almost identical items in the New menu for **Microsoft Access**, but this behavior is correct, and Microsoft's fault, and the same is true in Explorer.
- Fixed incompatibility with drag & drop from ISO Buster.
- The standalone viewer slideshow speed setting in Preferences now allows fractions of a second to be specified (using a decimal point).
- The three tabs in the Advanced Function editor when it's in script mode now remember their cursor position and selection range when you switch between them.
- The **Create Folder** dialog no longer allows strings containing colons (and other invalid filename characters) to be pasted in from the clipboard.
- Fixed not being able to set **Preferences / Miscellaneous / Advanced: clipboard\_image\_paste** to JPG for some configs which date back to older versions of Opus.
- The path field's ghost path breadcrumbs are now cleared when you right-click the path field and choose **Clear History**, or when you run the **Recent CLEAR** command.
- Fixed archive extract-to-subfolder failure when the archive name had a space before the extension.
- Experimental fix for file change notification not working in collections when files are added via a junction or mount point.
- The **@toggle** and **@icon** modifiers can now test conditions based on any internal command, not just the **Set** command.
- Opus will now show an error when you try to copy a file > 4GB in size to a FAT or FAT32 drive.
- The confirmation dialog when deleting a single file now truncates long filenames in the middle (e.g. aaa...ccc) rather than at the end (aaabbb...).

- The current per-tab font scaling setting is now saved as part of layouts and styles.
- The **Rename** dialog's preset list now shows a tooltip for any presets whose names are too wide for the list to show completely.
- Removing files from a collection can now be undone.
- The metadata pane should now recognise .m4a files properly (previously it was only checking for audio/x-m4a content type, it now checks for audio/mp4 as well).
- Dragging a "modern" application (e.g. **Calculator**) from the start menu to an Opus Lister now correctly creates a shortcut to the app.
- Combining the **Select SOURCETODEST** command with the **HIDEUNSEL** argument now correctly hides unselected items in the destination file display.
- The **Align other Lister element headers with the file display border** Preferences option is now disabled if more than one toolbar is selected for the FDB.
- The **COPYDIRTIMES** and **COPYFILETIMES** (and several other arguments) for the **Copy** command did not work in conjunction with the **EXTRACT** argument.
- Fixed a problem where renaming a folder by only changing the case of its name (e.g. "test" to "TEST") would cause it to be removed from the folder tree.
- Fixed problem with misreported duration of some AVI video files.
- When the file display is grouped and in details mode, pushing shift+up/down would sometimes not scroll the list far enough to show the next selected file.
- In the viewer pane, Opus would show the wrong cursor if the **Expand and scroll** was turned on but the parent option **Scroll with left mouse button** was turned off.
- If you used the Grid Lines toggle in the default toolbars (**Set GRIDLINESH=toggle**) to override the grid line settings in Preferences, then used it again to go back to normal, that lister would then ignore changes to grid lines Preferences. The second toggle will now reset the lister back to the current Preferences settings, and it will then track them as normal.
- Fixed the metadata panel's date fields showing the wrong day names when not being edited, if you had added day names to the system-wide short date format.
- Fixed "inherit columns from other formats" preventing a format from overriding the Default format's column sizes. Also fixed an issue where certain combinations of columns in the inherited and main format resulted in column sizes being shifted (e.g. Column 3's size applied to column 2, and so on).
- Fixed grid lines being invisible over columns with background colors in some situations, and improved centering of grid lines when "use visual styles to draw items" is off.
- Fixed tooltips displaying Opus-generated folder thumbs without proper borders.
- When using the **Go FOLDERCONTENT** command with the **useshell** option, sub-folders that only contain files and not folders will now be expandable.

- Fixed problem where the tooltip for a folder with **folder.jpg** inside would not render any folder thumbnail at all.
- Fixed issues with folder thumbnails in Tiles and Details+Thumbnails modes (images could draw outside the folder border).
- Fixed problem where a newly created library would not update in the tree to show added member folders until the tree was refreshed.
- Reworked the way Opus enumerates printers to prevent delays during startup (and possibly other times) caused by Windows taking a long time to process network printers.
- Fixed problem where using **Image CONVERT** to convert from a picture in a zip file to the same zip file could cause a deadlock.
- The simple Find panel now shows help bubbles again when wildcard characters like ( and ) are typed.
- The simple Select dialog no longer shows the wildcard help bubble as soon as it appears if the edit control already has wildcards in it from the last time.
- Workaround for Windows bug that caused parts of the **Preferences / File Operations / Filters** page to be drawn incorrectly after displaying the wildcard help bubble.
- Fixed **Expand & Scroll** not working in the preview panel for raw camera files, and others loaded via simple bitmap plugins.
- Improved **Open With** menu compatibility with Explorer when multiple versions of a program are installed side-by-side (e.g. Adobe Premier).
- It's now possible to drag a file collection to a toolbar in Customize mode to add a button to go to that collection.
- Fixed problem with the **Filetype CONTEXTMENU** command when a folder was selected - if it was present in the **File** menu following **Filetype CONTEXTMENU=Directory\Background**, it would generate commands (e.g. "Open in Explorer") for the wrong folder.
- Also fixed longstanding issue where context menus in the **File** menu could end up in the wrong spot (e.g. below the **Exit Directory Opus** command).
- The **GetSizes** command no longer ignores offline files when the **Ignore junctions and softlinks** option is on (so calculating the size of an offline OneDrive or Dropbox folder now works properly).
- The *Add folder to search* dialog in **Find** did not let you add a path by typing it in manually in some cases.
- In the icon modes, when the file display is grouped and the last group is collapsed, pressing the **End** key now puts the focus on the last group rather than the last (hidden) file within it.
- The Metadata dialog now correctly displays the focus rectangle if you push the cursor down key rather than Tab when the cursor opens.
- Using the **Image CONVERT** command with **REPLACE=always** now correctly replaces existing files when the destination is a zip file.

## 12.9 30th May, 2018

---

- Added **COPYSPARSE** argument for the **Copy** command. This option preserves regions in copied files that are marked as sparse (as long as sparse files are supported on the destination drive).
- The **Regular expression + Find and Replace** rename now allows you to anchor the search pattern at the start or end using **^** and **\$** like in regular regex mode.
- When multiple file copy jobs are queued and one job triggers a UAC prompt, the UAC permission is now passed through automatically to subsequent jobs in the queue
- **@hidenosel** and similar modifiers now work in the viewer pane's right-click > File menu, and in context menus you get when right-clicking file lists generated by **Go FOLDERCONTENT**.
- **@disableifpath** and **@hideifpath** now work with aliases, environment variables, **{apppath}**, and so on.
- Workaround for Windows bug where a case-only rename was treated as if the file was removed and a new version created. The difference was usually subtle but could be seen easily if the **Sort newly created and copied files** option was turned off, and could affect whether files remained selected after renaming.
- Fix for viewer crashing if you ran a custom command to jump forward or back multiple files when the target file had been deleted. (Normal single-step jumps were not affected.)
- Fixed a crash introduced in 12.7.1 when processing some file change notifications involving short path names.
- Commands like **Show VIEWERCMD=goto,-10** now move to the first file if there are fewer than 10 before the current file. (Previously, nothing happened.)
- Fixed issue where if you had chosen the *Filter Bar* as the default FAYT mode in 12.7, it would have changed to *WSL Command* or blank in 12.8. (As a side-effect of fixing this, if you chose *WSL Command* as the default in 12.8, you'll need to choose it again, as a one-off after installing this update.)
- Fix for the "tray icon" menu sometimes not closing if you click elsewhere, until you click on it first, if you had left-clicked the icon and then clicked somewhere else before the menu was triggered (after the system double-click time).
- Re-worked the way Opus prevents system sleep while copying files (etc.), and sleep and display power-down while playing slideshows (viewer) and videos (Movie plugin), as the old method was not working in Windows 10.
- Archive auto-extract now only prompts or happens when there's more than one file inside the archive.
- Updated 7z.dll to 18.5 which contains a security fix for RAR handling.
- Updated unrar(64).dll to 5.60.3 which contains a security fix for RAR handling.
- Fixed problem with Dropbox support when the Dropbox folder path contained unicode characters.

- The latest Windows 10 update added a non-functional "Open in new tab" item to folder context menus. We now filter it out. (The similar "Open in new Folder Tab" item which Opus itself adds to the same menu is still there and still works).
- Fixed "Launch" in the Visual Studio 2017 installer opening the folder VS was installed in rather than launching the program, if Explorer Replacement was turned on.

- HEIC/HEIF images are now supported, as long as the relevant WIC codecs are installed. (If you have the Windows Photos app working with HEIC files, they'll work in Opus as well.) Both the CopyTrans HEIC decoder (works with Windows 7 and up) and the Microsoft decoder (works with Windows 10) are supported.
- 3D objects (3mf, fbx, glb, glbt, obj, ply, stl) can now be displayed in the preview pane on Windows 10 (Creators Update and above). (3mf seems limited to files created by Microsoft's own tools and won't load most 3mf files from the web.) See here for an example and additional details: <https://resource.dopus.com/t/3d-object-preview-3mf-fbx-glb-glbt-obj-ply-stl/29541>
- Added **Individual groups** option to the **Folder Options / Columns** dialog. If this is turned on and the file display is grouped, one group will be created for each distinct value rather than a range of values falling into a single group (e.g. instead of A-H you would have A, B, C, D, E, F, G, H). This is only really useful for text fields like "User description". Also added the **Set GROUPINDIVIDUAL** command to toggle the setting on and off programmatically.
- Improved support for virtual desktops in Windows 10.
  - A new **virtual\_desktop\_isolation** setting, on by default, makes changes like these:
    - The "Save All Listers" command only saves windows from the current desktop.
    - Layouts set to close existing Listers when they open will only close Listers on the active virtual desktop.
    - Explorer Replacement (and things like the Go EXISTINGLISTER command), if set to reuse existing windows or open new tabs in them, will tend to open a new window if it can't find an existing one on the current desktop. (There are some exceptions, where Windows itself decides which Lister to activate for a folder.)
    - "Reuse existing viewer window" only considers viewer windows on the current desktop.
    - "Close All Viewers" from the viewer's system menu only closes viewers on the current desktop.
    - **Set LISTERCMD=showall, minimizeall, and toggleminimizeall** only act on the current desktop's windows.
  - **Close ALLVIEWERS, ALLLISTERS** and **ALLOTHERLISTERS** commands now work with a new **CURRENTDESKTOP** argument to restrict them to acting on the current desktop's windows.
  - **Set LISTERCMD=tileh, tilev, and cascade** always only act on the current desktop's windows, regardless of Preferences.
  - Fixed being able to dock a Lister into one that is on another virtual desktop and not currently visible.
- For Windows 10 only: **Preferences / Launching Opus / From the Win + E hotkey** now exists for you to change what happens when the **Win-E** hotkey is pushed. (While configured independently, this is tied to the Explorer Replacement mechanism. It thus requires Opus Pro and will only work with portable versions if they have been configured to enable Explorer Replacement.) **WHEN UPDATING OPUS, THIS NEW FUNCTIONALITY WILL NOT WORK CORRECTLY UNTIL YOU HAVE REBOOTED.**
- Local F1 help (as opposed to the help on the GPSoftware website) is now shown in your web browser by default. A new option, **Preferences / Miscellaneous / Advanced [Behavior]: help\_interface**, lets you set

it back to CHM (*HTML Help*) if you prefer this interface. You can also use that setting to specify a custom port for the local http help server if needed. When help is shown in your web browser, script add-ins and plugins are now able to add their own help pages.

- For users who wish to use dark themes:
  - Under **Preferences / Display / Colors and Fonts / File group header**, you can now configure the color of the expand/collapse glyph.
  - Under **Preferences / Display / Colors and Fonts / Folder Tree**, and **Folder Tree (Destination)**, you can now configure the color of the expand/collapse glyph.
  - If the status bar is set to use a dark background color, it no longer draws its frames using visual styles.
  - Relative size and date graphs now let you configure their frame and background colors.
  - Colors used in free space graphs in This PC (My Computer) can now be configured.
  - The free space graph in the status bar now uses the colors you define in Preferences rather than system colors, when applicable.
  - Fixed the Filter Bar's checkbox labels not being readable if you used a dark panel background.
  - The marker when dragging toolbar buttons around is now easier to see.
  - Fixed some glyphs within toolbars and menus which were difficult to see when using light-on-dark colors.
  - The system menu visual style is no longer used for menus which do not use dark text colors.
  - The frames around pop-up menus now use configurable colors, and menus will be drawn using your toolbar colors by default, if they have been customized. (Does not affect all menus in the program, at least not yet, but will affect most toolbar and context menus.)
  - Default menu background color can now be configured separately.
  - Toolbars and menus now have configurable "toggled" and "selected" colors for buttons and menu items (e.g. when pushed in or as the mouse moves over them).
- The codes available to display information on the status bar have been expanded:
  - You can now use variables on the status bar. e.g. `{var:tab:MyVariable}` will display the contents of the file display variable *MyVariable*. You can also hide and show sections depending on whether or not a variable exists, and then control which information is shown on the status bar from buttons and scripts. e.g. `{h!{var:glob:ShowExtraInfo}} ... {h!}`
  - Status bar data can now be conditional on the current path or tests for whether paths exist. For example, `{ifpath:"C:*"}` will return **1** if you are in a path below **C:\** and nothing otherwise. `{ifexists:".*.dll"}` will return **1** if there are any DLLs below the current folder. These can be used with the `{h!}` code to show or hide things. Further examples and details can be found in the manual.
  - Status bar data can now be conditional on the same `@if` and `@ifset` tests you can use in buttons and scripts. For example, if you want the status bar to show the selected file's size and date, but only when not in Details mode: `{h!{if:!Set VIEW=Details}} {sel:size} {sel:write} {h!}`

- Added new **Preferences / Miscellaneous / Advanced** setting, **status\_metadata\_trigger**, which lets you choose when/if things such as duration codes on the status bar should cause file metadata to be calculated.
  - The **@toggle:update** command directive can now be used after modifying variables to force status bars to update.
- Added **Publisher** column.
  - The **Select HIDESEL**, **HIDEUNSEL** and **SHOWHIDDEN** arguments can now be made to operate only on files or only on directories (or both, by default).
  - The services supported by the **Image LOCATE** command are now configurable via **Preferences / Miscellaneous / Advanced: image\_locate\_services**. Also added *Open Street Map* as a default service (**Image LOCATE=osm**).
  - The **CreateFolder** command now has a **MULTI** argument to force multi-line mode on and off, and now lets you pass multiple default folder names to the multi-line dialog.
  - The **Rename** command's **FROM** argument now understands aliases mixed with wildcards, e.g. **Rename FROM /downloads\\*.zip** to rename all zip files in the system downloads folder.
  - The **Rename** command now lets you specify the *Rename* dialog's default sequential numbering parameters without having to turn sequential numbering on. This is done by adding **!** to the **NUMBER** parameter. For example, if you want the *Rename* dialog to open with numbering off but padding set to two digits when/if you turn it on: **Rename ADVANCED NUMBER=!01**
  - **Select FILTER** now works in conjunction with the **TYPE** argument to limit the filter to either files or directories.
  - Added **Select NEXT=row** and **Select PREV=row** arguments, to allow commands that move the selection up or down a row in icon modes.
  - The **Rename** command now has a **WHENEXISTS** argument, letting you specify the action to take when the new filename already exists.
  - The *Simple Find* function (and the **Find NAME** parameter when automating a find from the command line) now supports **regex:** at the start of the pattern to specify a regular expression.
  - Added some new **Find** command arguments to control things the *Simple Find* dialog has check-boxes for, when automating the command: **NOWILD**, **ANYWORD**, **CONTWILD**, **CONTCASE**, **NOPARTIAL**.
  - The **Go** command (and other commands that support things like /aliases) now supports **file://** URIs (so e.g. you can now do **Go file:///C:**).
  - Commands for setting show/hide attribute filters now support the **I** (non-indexed) and **P** (pinned) attributes. Also, previously undocumented: You can specify two sets of attributes to alternate between the two each time the command is run. e.g. **Set HideFilterAttr=HS,R** would toggle between hiding files with the **Hidden** or **System** attributes and hiding files with the **Read-Only** attribute. When only one set is specified, the command acts as a toggle.
  - File display variables can now be prefixed with **left:**, **right:** and **tab:** to designate the left, right and active file displays, in addition to the older **src:** and **dst:** prefixes. (**tab:** and **src:** will usually point to the same thing, but can be different, e.g. when dealing with status bar codes for a particular side of the Lister.)

- You can now configure the font used in code editors such as the advanced button editor, rename scripts, and status bar Preferences.
- Added **Preferences / File Operations / Copy Attributes / Copy sparse files as sparse** option. This option preserves regions in copied files that are marked as sparse (as long as sparse files are supported on the destination drive). (Note this option was originally included in 12.8.1 but was removed in 12.9)
- Added **Preferences / Miscellaneous / Advanced: grid\_lines\_first\_last**. For solid grid lines, you can now make the alternating pattern start on the first line instead of the second. For thin grid lines, you can now have a line before the very first file or after the very last file. (Caveats in the manual.)
- Opus now recognises command codes like **{allfile|sep=|}** where you want a pipe character as the separator between files.
- **@if:set focus** now allows more flexible testing of which window currently has input focus (e.g. **@if:set focus=filedisplay** to test if any file display has focus).
- The action for normal left-click on items in the tree is now configurable through **Preferences / Folder Tree / Selection Events**. The default action is **Go**; to get the same result as the old **Switch to existing tab if already open** option, change it to **Go TABFINDEXISTING**. You can also change it to e.g. **Go NEWTAB=findexisting** to make left-clicks on the tree open new tabs by default.
- The new setting **clipboard\_change\_delay**, under **Preferences / Miscellaneous / Advanced [Troubleshooting]**, allows you to tell Opus to delay before checking the clipboard for changes. This may help avoid problems using the clipboard in certain software (e.g. Microsoft Office, although a \*lot\* of other software triggers the same problem in Office, including parts of Windows itself, so YMMV). The default delay is 100ms; it was effectively 0 in the past.
- You can now override the default icons used by new File Collections via **Preferences / Miscellaneous / Advanced [Cosmetic]: collection\_icon\_default, collection\_icon\_find, collection\_icon\_flickr, and collection\_icon\_marked**. Also fixed custom collection icons not being used in certain places, and some cosmetic issues with the File Collection properties dialog.
- Thumbnail Threads (**Preferences / File Display Modes / Thumbnails**) maximum limit increased from 8 to 32. (When Opus launches it also limits the value to the number of CPU threads.)
- Added **Preferences / Miscellaneous / Advanced [Behavior]: slow\_dbclk\_rename** option, which allows you to disable inline rename via slow double-click (F2 not affected). (Does not apply to Power mode, which has its own mouse button settings.)
- The colors for relative size and date graphs can now be configured separately for files and folders.
- The background and glyph colors of the breadcrumbs path field, search field, filter field and filter bar controls can now be configured under **Preferences / Display / Colors and Fonts / Toolbar and menu defaults**.
- If you hide the media duration parts of the status bar using a variable, the duration will no longer be calculated (unless something else still triggers it). If you then toggle the variable to make the duration parts show again, the duration will be calculated immediately (if it hasn't been already).
- Status bar **{var:...}** codes can now negate the variable using **{var:!...}**.

- The distance the mouse has to move before a drag & drop begins is now DPI scaled. (The base value still comes from the registry, via the `GetSystemMetrics` API, and can be changed under `HKCU\Control Panel\Desktop\DragWidth` and `DragHeight`. Note that most other software does not DPI-scale it, but we believe this is incorrect as the default has remained at 4 pixels since long before Windows supported high DPI.) Similarly, the bounding box the mouse must stay in between clicks to register a double-click is scaled as well.
- Scripting changes:
  - Added **Tab.hidden**, **hidden\_files** and **hidden\_dirs** properties, which provide information about files and folders hidden from the current view in a tab.
  - Added **Format.group\_individual** property to query the state of the new **Individual groups** option.
  - The **Blob.CopyFrom** method now lets you initialise a **Blob** from a string. By default the **Blob** will be set to the Unicode form of the string; if you pass "utf8" as the second parameter it will initialise the **Blob** with the utf8-encoded form of the string.
  - The **FSUtil.Hash** method can now hash the contents of a **Blob** as well as a disk file.
  - Fixed **item.name\_stem\_m** and **item.ext\_m** ignoring multi-part extensions when obtained via **tab.selected\_files**.
  - Script columns now respect the *Folder Options* **Numeric order filename sorting** setting instead of always applying it. You can also now specify **numERICsort**, **nonumERICsort**, **WORDsort** and/or **noWORDsort** for a script column's type to enable or disable the numeric and word sorting modes, overriding the setting in *Folder Options*.
  - Scripts can now use the *Progress* dialog's **Skip** button via the new **Progress.EnableSkip** method.
  - The **Progress.GetAbortState** method can now handle pausing automatically, can be told to only check for certain states, and will now only indicate the most important state by default.
  - The new **StringTools.IsASCII** method can be used to test if a string uses only 7-bit ASCII characters, or if it would be better written in a Unicode format when saving it to a file.
  - The **FolderEnum.Next** method can now return more than one directory entry at once, in a **Vector**. Specify the number of items you want returned at once, or **-1** to return the whole directory in a single call. You can pass a **Vector** object to use as the second argument if desired, or one will be created for you. If no arguments are specified then only a single **Item** object is returned as before.
  - Added **OnTabClick** script event, which lets scripts intercept mouse clicks (with a qualifier key) on tabs and override the default behaviour.
  - The **StringTools.Encode** method understands a new format string **utf-8 bom** to encode into UTF-8 with a byte-order-mark on the front.
  - The **Progress.SetFromTo** methods now clears the "To:" line entirely if the argument for it is omitted. (It's still best to use **Progress.SetStatus** if you only want one line all of the time.)
  - The new **Item.InGroup** method lets you test if a file belongs to a particular file type group without having to loop through all the groups it belongs to.
  - The **StringTools.Encode** method no longer adds a null at the end of the binary data.

- Fixed **Blob.Compare** method incorrectly returning **0** when two different sized blobs were compared, without the comparison size being specified, if one was a prefix of the other.
- In script dialogs, the **readonly** property of edit controls can now be modified by the script rather than being fixed at design time.
- The **QuickFilter** object has a new **showeverything** property which indicates if **Show Everything** mode is on for the folder tab. When **Show Everything** mode is on, it also now implicitly makes the **QuickFilter** object report **disabled=true**, **showallfiles=true**, and similar.
- A new script event, **OnGetHelpContent**, lets script add-ins add their own help content to the **F1** help (only when help is shown in the user's web browser). The new **Script.LoadImageFile** and **Script.LoadHelpImage** methods let you easily load your script's help content when bundled as a script package.
- Added **Script.HttpHelpEnabled** and **Script.ShowHelp** to let scripts trigger their own help pages. There are also matching functions for viewer/VFS plugins.
- Added workaround for FTP connections to Pure-FTPd server when using TLS, where the directory listing would not complete or require a timeout before completing.
- It is now possible to remove items from collections which were imported without any paths (just names) from a file list using **dopusrt.exe**.
- When importing file lists to collections via **dopusrt.exe**, lines which are not fully qualified paths are now assumed to be relative to the same folder the list file is in. You can override this using **/relative:<path>** or suppress it using **/relative:none**. If using **/relative:none**, you cannot also use **/nocheck**.
- Fixed cosmetic issue where, if a column was less than 4 pixels wide, background colors applied by file labels would have a tiny gap where the narrow column was.
- Previously, if the **Automatically select next file after deleting** option was on and you removed the drive you were on, with the folder tree open, Opus would navigate you to the next drive. It will now always take you to the This PC (My Computer) level instead.
- The **Pin to Start Menu** context menu item now works in Opus.
- The **Go TABGROUPFORCE** argument can now be used with **Go TABGROUPLIST**.
- Fixed the filter bar's file extensions drop-down not indicating which extension has keyboard focus on systems with visual styles disabled.
- Fixed crash which could occur after changing the "when image is shared" mode for an image under **Preferences / Display / Images**.
- **Preferences / Display / Images** now accepts new images via drag & drop. Drop on the list to add a new image, or select an image and drop on the thumbnail to change it to a new image.
- Fixed Opus Light incorrectly allowing you disable hotkeys which come from the default toolbars, since changes to the default toolbars are never saved to disk in Opus Light. Standalone hotkeys can still be edited and are saved to disk.

- Drag & Drop with the right mouse button now works correctly within the same tab when manual sort is enabled.
- Auto-extract now works with 7z, RAR, Tar and other plugin archive types.
- Fix for double-clicking programs inside non-zip archives causing further double-clicks in the file display to be buffered until the launched program exited.
- When merging two folders via a move, any sub-folders in the source that had labels assigned to them would lose their labels when they were moved across.
- When manual sort mode is on, and "Sort newly created and copied files" is turned off, copying a file by dragging and dropping to a specific place now maintains the insertion position.
- When the viewer is set to use an automatic background color, it now ignores any transparent pixels when selecting a color to use.
- The **Secure Delete** function now moves files to a temporary folder (with random name) before deleting them, to try to stop their original location being visible in recovery tools.
- A button that runs **Set BLURFILENAMES=Toggle** now correctly highlights when filenames are blurred.
- When renaming a folder in the folder tree you can now use the down/up cursor keys to accept the edit and move to the next/previous sibling folder.
- Added **.mkv** and **.flac** extensions to the Windows Media Player preview handler (if no other preview handler is assigned to them), since the Windows 10 version can handle both formats but doesn't designate itself as handling them in the registry. Note that the handler is still disabled by default, so this only affects it if you turn it on via the ActiveX + Preview + Office + Web plugin. (Also note that Microsoft have fixed the preview handler so it works again in Windows 10 build 1803, after it was broken for about a year.)
- Added **.flac** extension to the Generic ActiveX list, since the Windows 10 WMP ActiveX control can play FLAC files in the viewer.
- The **{smp3}** and **{tmp3}** status bar codes now work with WAV files.
- Fix for drag & drop from Adobe Bridge not working.
- Archives plugin (7z, RAR, etc.) and internal zip code now has better handling for (somewhat dubious) archives that use the same paths for multiple files. Each item will now have a unique name generated for it. In the case of RAR archives, such archives will also be made read-only.
- Fixed problem with zip files which contained sub-folders with spaces at the end of their names.
- Detection of WSL only worked in 12.9 if you were still on the Windows 10 Creators Update. It now works with the newer 1803 build again.
- Changing **Preferences / Folders / Folder Behaviour / Sort shortcuts to folders like folders** now triggers a refresh of all open folders, so the setting's effect is seen immediately.

- Secure screenshot wasn't blurring some paths in the breadcrumbs field.
- Secure screenshot now blurs filenames in Explorer-provided folders (e.g. *Quick Access*).
- Fixed problem with navigation lock where navigating out of a zip file on one side would not cause the other side to follow along.
- Dragging a tab from file display to another now makes the new tab active.
- When an external icon set is cached (for performance) it now preserves the display names of icons within the set.
- Fixed problem with Duplicate Finder's **Select** button - if the duplicate search was initially run with **Delete mode** turned off, and then you put the folder into **Checkbox Mode** manually and clicked the **Select** button, weird things would happen.
- Fixed the file display font-size pop-up going off the side of the monitor if it opened too near the edge.
- **Preferences / Miscellaneous / Advanced** is now grouped into categories. The list is also now filtered when using the search function in Preferences, rather than drawing matching items in red.
- The **Combine groups with only one member into the 'Other' group** option is now ignored when grouping by duplicates.
- Improved support for creating a toolbar button to go to a web page by dragging from a web browser to the toolbar in Customize mode. (Tested with Chrome and IE. Should work with Firefox. Does not work with Microsoft Edge, since you can't drag from Edge to *\*anything\**, even to create a .URL file in File Explorer).
- Added support for dropping .URL files (including Steam game shortcuts) on a toolbar in Customize mode to create buttons.
- When Opus receives a "media removed" notification from the OS it now reads the Computer folder automatically in any tabs that were showing folders on the removed media.
- When test-running commands in the button editor, the command will now always run against the Lister that launched the editor. (Previously, the command would look for a source window and potentially run against a different Lister, or even fail to find a source if you had two or more dual-display windows and no single-display ones).
- Copying files out of a zip file in flat view now gives more consistent results (both in "recreate" and "flatten" mode).
- It's now possible to prevent Opus from saving FTP passwords, by setting the **PreventSaveFTPCredentials** value under **HKEY\_LOCAL\_MACHINE\Software\GPSSoftware\Directory Opus** to **DWORD:1**.
- Pasting a file from a virtual data source (e.g. Remote Desktop) now behaves the same as pasting a real file when the destination already exists - you'll be prompted to overwrite/rename etc, rather than the new file just being renamed automatically.
- Workaround for issue with SumatraPDF Open With menu item.

- Fixed Open With menu showing just an icon and no name for Notepad.
- Fixed **SetAttr META coverart** only allowing numeric specification of the type of cover-art when adding, not removing.
- Fixed metadata editor's **Add cover-art** button not working correctly when in the Desktop folder.
- Fixed labels applied via folder formats not doing anything if the global labels-in-NTFS option was off and the global list of assigned labels was completely empty.
- Added missing music cover-art image types "other file icon" and "colorful fish" to the Metadata editor, **SetAttr** command and AudioCoverArt script object. (The fish thing is, unfortunately, part of the ID3 specification. It is hidden in the UI except when editing a file which already uses it, because it is ridiculous and some tagging software won't recognise it.)
- **Prefs LAYOUTLIST SHOWICONS** now assigns folder icons to branches of the layout list, instead of only assigning icons to the layouts themselves.
- Fixed the default "Lister Styles" menu heading being mistranslated to "Rotate" (in the target language) if you changed language after installation.
- Fixed crash if you started a Find Files operation from a folder with a localised name. (You would normally have had to type the localised name into the Find In field manually to make this happen).
- Fixed issue where clicking "skip" to the error shown when attempting to copy a file > 4GB in size to a FAT32 drive would abort the rest of the function rather than just skipping the file.
- The **Print FOLDER** command no longer flashes a progress dialog on screen if delayed progress dialogs are turned on in Preferences and it is outputting to a file or the clipboard without showing the Print Folder dialog.
- Fixed a Favorite which pointed to an Alias called **/user** not doing anything when clicked.
- Fixed Properties dialogs showing zero size when opened for multiple items in different folders (e.g. via Flat View or collections like Find Results).
- Fixed right-click > Properties on an FTP Site in the folder tree opening the FTP Address Book but not selecting the site.
- Fixed right-click > Create Shortcut on the File Collections root in the folder tree crashing when used.
- Possible fix for folder tabs sometimes getting stuck "Reading Folder" with TortoiseSVN installed. (The problem stopped happening for us by itself so it is difficult to verify the fix does what it is supposed to. Please let us know if you still encounter the problem with this or later versions.)
- Made change to solve problem of Opus not allowing files > 4GB to be copied to Google Team Drive (which misrepresents itself as a FAT32 filesystem).
- Fixed problem that meant when an FTP folder was added to favorites and selected from the Favorites branch in the tree the folder would open in a new Lister (or even an external program, depending on your

settings).

- The *Browse* dialog shown for the **Preferences / Miscellaneous / Advanced [Image Formats]: use\_color\_management** setting now has a filter set to **\*.icc;\*.icm** to make it clear what type of file is required.
- Fixed alignment of Opus-generated folder thumbnail when shown in a tooltip.
- Config files, script add-in code, and *Containing Text* searches in the *Find Panel* now support characters outside the BMP range (UTF-16 surrogate pairs and 4-byte UTF-8 sequences). Previously, such characters - e.g. in button labels -- were dropped when saving and turned into '?' when loading.
- Fixed crash in the button editor which could happen after using **Shift+Tab** to de-indent lines (usually while editing scripts).
- Addressed Explorer Replacement not working in some cases where applications initialized COM in a non-standard way for calling shell APIs (e.g. Google Chrome).
- When Opus is installed it now registers a URL scheme, **opushelp://**, which allows for clickable links to local help in your browser. **dopusrt.exe** has a new **/help** argument that this scheme uses to trigger display of local http help. The HTTP help server will be enabled and started automatically if it was turned off in Preferences.
- The API for non-Unicode viewer and VFS DLLs has been disabled. We believe there are only two 64-bit plugins affected by this, both of which have alternatives and appear not to have been downloaded recently. If this is wrong, or there are non-Unicode plugins we are unaware of (e.g. privately developed for corporate environments), please let us know. If needed, we can reinstate the legacy API, but our current plans are to remove the code behind it if there are no complaints.
- On the *FTP Logs* panel, clearing the *[All Activity]* log now does the same as the **Clear All Logs** option, removing site-specific logs at the same time.
- Opening virtual folders via Explorer Replacement now opens them in your normal default Lister, instead of forcing the dual file display, tree, viewer and other panels to be off.
- Commands like **Go LASTACTIVELISTER** are now better at finding the Lister which was most recently active. Previously, they were a bit too literal: If the last active Lister had since been closed, without another becoming active, then they did not find anything.
- If a file copy fails mid-way with *ERROR\_NOT\_READY* (which can happen if a USB drive is pulled out) Opus will restart the copy when you click **Retry** rather than just attempting the last write again.
- Opus now tries to stop its DLLs being injected into chrome.exe to avoid Chrome incorrectly labeling (libeling!) us as an incompatible application. Unfortunately this will stop the "Open Downloads Folder" command in Chrome from opening in Opus when Explorer Replacement is enabled.
- The *Rename* dialog's **Clipboard** menu/button is now always enabled, even if the macro builder is turned on. Using one of the paste options will turn off the macro builder automatically.
- Label filters in the folder tree are now re-calculated when you push **F5** (or run similar refresh commands which affect the tree).

- Tweaked syntax highlighting for the script and button editors.
- Status bar configuration can now include comments by starting lines with //.
- Folder tooltips using **{foldercontents}**, and the status bar hidden-count tooltip, now show each file/folder on a separate line, and allow names to be longer before truncation, and more names to be displayed in total.
- Infotip **{foldercontents}** has new parameters: **SingleLine** makes it show files and folders on a single line (each) instead of one line per item. **MaxItems** limits the number of items within each category (up to a hard maximum of 20). **MaxItemLength** limits the length of each item (up to a hard maximum of 260). **Indent** lets you change the indentation before each item in multi-line mode (defaults to four spaces), or the separator between each item in single-line mode (defaults to ", "). To get single-line infotips like they were prior to Opus 12.9.3, use: **{foldercontents:singleline:maxitems=4,maxitemlength=20}**
- If the **Add File Collections list to the Send To menu** option was turned on, then later turned off, the collections in the **Send To** menu at the time would be left there (except on Windows XP). They are now cleaned up.
- With newer versions of WinRAR installed, the Archives plugin always created RAR 4 archives even if configured to make RAR 5 ones. This now works again.
- Fixed **Preferences / Launching Opus / Default Lister / In a fixed position relative to the monitor the mouse is on** behaving like **Always over the mouse pointer**.
- Fixed incorrect behavior of the viewer pane's context menu items for spanned wallpaper and resetting the viewer plugin override.
- Fixed dialog editor crash if you pushed the **Del** key on an item with multi-line contents to reset the contents to empty.
- Fixed problem with FTP SSH not returning directory listings if the list output omitted file permissions.
- The **Replace Explorer for all file system folders** option now includes the Quick Access folder on Windows 10. Also improved this option so it excludes more control panels (e.g. the "classic" desktop wallpaper control panel opened by running "shell::{ED834ED6-4B5A-4bfe-8F11-A626DCB6A921} - Microsoft.Personalization\pageWallpaper"). If updating Opus, this change will only take effect after you reboot.
- Fixed crash if the filter bar was in use and set to clear when changing folders, and you then changed folders while inline-renaming a file.
- Double-clicking shortcuts to FTP sites added under This PC will now add the FTP site under the folder tree's FTP branch, not under the This PC branch (which had some issues with "... on SiteAddress" being added to the end of folder names).
- Fixed a quirk of the Filter Bar which meant that, with filtering disabled, you could activate the disabled edit control via the keyboard and type one character into it.
- Fixed crash if the Filter Bar was open at the time you toggled the Preferences setting to make it use regular expressions.

- Fixed status bar code parsing so using {{ to get a single literal { works even when it is before something that looks like a valid status bar code.
- Fixed crash if you pushed **Alt** to put the Lister toolbars into keyboard mode, then used the **Left** cursor key to move past the leftmost menu item.
- **Rename FROM** now works correctly in conjunction with the **PRESET** argument.
- The context menu item for *TreeSizeFree* is now shown in Opus.
- Fixed problem where the **New Folder** button would be incorrectly disabled in the **Copy TO=ask\$ dialog** if the last destination was a network drive.



# Index

## 1

12.10 1299

12.2 1258

12.3 1261

12.4 1269

12.5 1275

12.6 1278

12.7 1279

12.8 1286

12.9 1297

## 7

7z Options 211

## A

AboutData 960

Acknowledgements 6

Actions 533

ActivateListerData 960

ActivateTabData 961

Add to Archive Dialog 208

AddCmdData 961

AddColData 961

Adding a new Column 599

Adding a new Column from Shell Properties 601

Adding a new Internal Command 597

Adding a new Site 261

Adding Cover Art 235

Adding Dialog Controls 564

Adding to Archives 207

Adding, Removing and Editing Clauses 168

Additional Functionality 269

Advanced Command Editor 497

Advanced Find 102

Advanced Options 396

Advanced Rename 181

Advanced Selection 89

AfterFolderChangeData 962

Alias 963

Aliases 34, 963

Archive and VFS Plugins 433

Archive Context Menu 434

Archive Options 436

Archives 138

Args 964

Argument Types 648

AudioCoverArt 965

AudioMeta 966

Auto-Loading 361

Automated image conversion tasks 281

## B

Backing up and Restoring Preferences 315

Bar graphs and Percentages 639

Basic Concepts 17

BeforeFolderChangeData 966

Blob 967

Breadcrumbs Configuration 486

Breadcrumbs Location Field 27

BusyIndicator 970

Button Properties 573

**C**

Calculating Folder Sizes 76

Changing Attributes 219

Check Box Properties 574

Checkbox Mode 86

CLI 310, 650

ClickData 971

Clipboard 655

Close 669

CloseListerData 972

CloseTabData 972

Codes for clipboard and variables 928

Codes for date and time 927

Codes for disk space 626

Codes for file and folder counts 622

Codes for graphical elements 628

Codes for music and video duration 627

Codes for passing filenames 914

Codes for passing paths 921

Codes to display dialogs 924

Colors and Fonts 319

Column 972

Columns 112

Combo Box Properties 574

Command 973

Command Editor 494

Command modifier reference 930

Command modifiers 515

Command Reference 648

Commands 441, 1234

Common Dialog Properties 570

Compatibility Files 140

ConfigChangeData 980

Content Types 122

Context Menu 538

Context Menus 455

ContextMenu 674

Control 981

Controlling Floating Toolbars 300

Copy 678

Copy and Paste 142

Copy Attributes 352

Copy Options 354

Copy Queues 148

Copying Updated Files 156

Copying using the toolbar buttons 145

Copying, Moving and Deleting Files 142

Copyrights 1

CreateFolder 698

Creating Archives 206

Creating Folders 203

Creating Script Dialogs 561

Creating your own buttons 458

Creating your own Themes 74

Custom Fields in the Rename Dialog 555

CustomFieldData 986

Customize 439

**D**

Date 988

DDE Functions 520

Default Lister Settings 387

Default Settings 250

Defining a Filter 166

Delete 701

Deleting Files 176, 355

Detached Dialogs 579

Details Mode 343

Dialog 991

Dialog Control Mnemonics 568

Dialog Control Properties 570

Dialog Control Tab Order 568

Dialog Editor 562

Dialog Editor Commands 563

Dialog Properties 571

DialogListColumn 1004

DialogListColumns 1004

DialogListItem 1005

DialogOption 1006

Directory Opus 12 1177

Directory Opus File Types 526

Display 115, 319

Display Options 326

Display Page 254

DisplayModeChangeData 1006

Dock 1007

DocMeta 1007

Document Properties 230

DOPus 1008

DOPusFactory 1012

DOPusRT Reference 1129

Double-click Files 356

DoubleClickData 1013

DPI 1015

DPI aware Icon Sets 1167

Drag and drop 143

Drive 1015

Drive Buttons and Lists 30

Drive Buttons Configuration 477

Drive List Configuration 490

Drop Menu 542

Drop-down Buttons and Menus 475

Dual Display 46

Duplicate File Finder 286

Dynamic Buttons 477

Dynamic Toolbars 56

**E**

Edit Category (Pre-defined commands) 443

Edit Control Properties 573

Editing Metadata 227

Editing the Toolbar 458

Email 383

Embedded functions 521

Embedding Rename Scripts 519

Events 535

Example Rename Script 594

Example Scripts 594

ExeMeta 1016

Explorer Replacement 79, 388

Exporting to USB 304

Extended Properties 237

External control codes 914

External Manipulation of File Collections 1134

**F**

Favorite 1016

Favorites 32, 330, 705, 1017

Favorites and Recent 330

FAYT and Filter Bar 1227

FAYT and Filter Bar Keys 339

FAYT and Filter Bar Options 340

Field Buttons 482

Fields 322

File 1019

File and Folder Labels 333, 1195

File Category (Pre-defined commands) 443

File Collections 132

File Commands (Pre-defined commands) 444

File copying 1218

File Descriptions 241

File Display border 25, 338

File Display Modes 343

File Display Options 342

File Display Toolbar 55

File Displays 337, 1202

File Operations 141, 352, 1220

File Operations Options 358

File Type Groups 527

File Types 523

FileAttr 1023

FileGroup 1024

FileSize 1025

FileType 710

Filetype Editor 532

FiletypeGroup 1026

Filter Bar 91

Filter Clause Types 168

Filter Field Configuration 489

Filtered Operations 163

Filters 117, 356, 1027

Find 716, 1221

Find and Replace 189

Find Files 97

Find-as-you-type Field 58

Flat View 126

FlatViewChangeData 1028

Flickr 384

Flickr Synchronization 289

Floating Toolbars 299

Folder Aliases 331

Folder Behaviour 361

Folder Display 363

Folder Formats 121, 365

Folder Formats and Folder Options 1207

Folder Options 109

Folder Options Dialog 110

Folder Tab Appearance 372

Folder Tab Options 372

Folder Tabs 371

Folder Tree 23, 377

Folder Tree Appearance 378

Folder Tree Contents 378

Folder Tree Options 380

FolderEnum 1028

Folders 360

FontMeta 1029

Format 1030

FSUtil 1032

FTP 139, 247

FTP Address Book 248

FTP Connect 263

FTP Log 266

FTP Paths 267

Func 1039

**G**

GetCopyQueueNameData 1041

GetCustomFieldData 1041

GetHelpContentData 1042

GetNewNameData 1043

GetSizes 722

Global Filters 368

Go 724

Go Category (Pre-defined commands) 445

Group Box Properties 577

**H**

Help 761

Help Category (Pre-defined commands) 446

Hiding sections on the status bar 644

High DPI support 1193

**I**

Icon Categories 1166

Icon Display Names 1165

Icon Images 1170  
Icon Names 1165  
Icon Set XML Definition File 1163  
Icon Sets 1163, 1255  
Icon Sizes 1164  
Identifying the current format 124  
Image 762  
Image Conversion 279  
Image Marking 276  
Image Viewer 1188  
ImageMeta 1044  
Images 323  
Images Toolbar 56  
Index Page 255  
Info Tip 546  
Inline Rename 179, 356  
Installing and Registering 11  
Interacting with Dialog Controls 583  
Internal Command Arguments 510  
Internal Commands 649  
Internet 383  
Introduction 9  
Item 1044  
**J**  
Join 769  
Joining Files 295  
Jumplist 332

## **K**

Keys 453  
Keywords for Columns 1144  
Keywords for SetAttr META 1149

## **L**

Label Assignments 334  
Labels 119, 223  
Language 325  
Language Overlays 569  
Launch Options 463  
Launching Opus 387  
Launching Opus from the Desktop 389  
Launching Opus from the Taskbar Icon 390  
Launching Opus from the Win + E hotkey 390  
Launching Opus On Startup 391  
Layouts 68, 393  
Layouts and Styles 393  
Libraries 135  
Licence 4  
List Box Properties 575  
List View Properties 575  
Lister 1050  
ListerResizeData 1052  
Listers 1052  
ListerUIChangeData 1053  
Localization 1169  
Locking the Format 123

Logging 357

## **M**

Making Links and Junctions 296

Manual Sorting 106, 1199

Map 1053

Marker 770

Menu Toolbar 50

Metadata 357, 1055

Metadata Keywords 1144

Metadata Pane 66

Misc Page 256

Miscellaneous 396, 1229

Miscellaneous Category (Pre-defined commands) 447

Mouse 342

Mouse Buttons 429

MS-DOS Batch commands 516

Msg 1057

MTP 139

Multiple Function Buttons 473

Music Properties 234

## **N**

Navigation 21

Navigation Lock 46

Network Page 252

New Category (Pre-defined commands) 447

Numbering Files 193

## **O**

OnAboutScript 1110

OnActivateLister 1110

OnActivateTab 1110

OnAddColumns 1111

OnAddCommands 1111

OnAfterFolderChange 1112

OnBeforeFolderChange 1112

OnClick 1113

OnCloseLister 1114

OnCloseTab 1114

OnDisplayModeChange 1115

OnDoubleClick 1115

OnFlatViewChange 1116

OnGetCopyQueueName 1117

OnGetCustomFields 1118

OnGetHelpContent 1119

OnGetNewName 1120

OnInit 1121

OnListerResize 1121

OnListerUIChange 1122

OnOpenLister 1122

OnOpenTab 1123

OnScriptColumn 1123

OnScriptCommand 1124

OnScriptConfigChange 1125

OnShutdown 1125

OnSourceDestChange 1125

OnStartup 1126

OnStyleSelected 1126

OnTabClick 1127

OnViewerEvent 1127

Opening a Lister 20

OpenListerData 1059

OpenTabData 1060

Operations Toolbar 52

Options 118

Other Codes 629

OtherMeta 1060

## **P**

Padding sections on the status bar 646

Passing files to external programs 514

Path 1061

Path Field Configuration 485

Pattern Matching Syntax 614

Picture Properties 231

Play 771

Playing Sounds 278

Power Mode 345

Power Mode Buttons 347

Preferences 313

Preferences Categories 318

Prefs 772

Print 782

Print Folder 283

Programmatic setting of Metadata 238

Progress 1062

Progress Indicators 359

Properties 787

Proxy 386

Proxy Page 259

## **Q**

QuickFilter 1067

## **R**

Radio Button Properties 574

RAR Options 211

Reading Dialog Control Values 581

Read-Only mode 214

Recent 793

Recent and History Lists 33

Recent List 336

Rect 1068

Reference 613, 1234

Regular Expression Syntax 617

Regular Expressions 190

Regular Expressions + Find and Replace 191

Release History 1176

Rename 797, 1179

Rename Actions 192

Rename Macro Language 1173

Rename Macros 194

Rename Modes 186

Rename Options 197

Rename Presets 183

Rename Scripts 201, 553

Renaming Files 179

Renaming with Metadata 198

Replace Menu 545

Resources 590

Responding to Events 609

Results 1068

Retrieving File and Folder Information 1141

**S**

Script 1069

Script Add-ins 587

Script Dialog Example 606

Script Dialogs 560, 1215

Script Functions 558

Script Miscellaneous 1232

Script objects 1244

Script Package 588

Script resources 1253

ScriptColumn 1071

ScriptColumnData 1076

ScriptCommand 1080

ScriptCommandData 1081

ScriptConfig 1082

Scripting 551

Scripting Events 1109

Scripting Objects 953

Scripting Reference 953

ScriptInitData 1083

Scripts 425

ScriptStrings 1085

Searching and Filtering 91

Secure Delete 177

Select 806

Selecting Files 82

Selecting with the mouse and keyboard 83

Selection Events 382

Self-Extracting Zip Files 214

Set 818

SetAttr 881

Settings Category (Pre-defined commands) 448

Sharing functions with others 492

ShellProperty 1086

Show 888

Show Everything 93

ShutdownData 1086

Simple Command Editor 496

Simple Dialogs 577

Simple Dialogs and Popup Menus 602

Simple Find 99

Simple Script Function 595

Simple Wildcard Rename 180

Simple Wildcard Selection 88  
 Single-click mode 85  
 Site Page 251  
 Site Properties 265  
 Sizing and Positioning Dialog Controls 565  
 SmartFavorite 1087  
 SmartFavorites 33, 337, 1087  
 Sorting and Grouping 103  
 SortOrder 1088  
 Sounds 418  
 Sounds Page 256  
 Source and Destination 80  
 SourceDestData 1089  
 Special Page 258  
 Speed Page 258  
 Split 903  
 Splitting Files 294  
 Standard Rename 187  
 Starting Opus 16  
 StartupData 1089  
 Static Text Properties 572  
 Status Bar 77, 327, 1228  
 Status Bar Codes 622  
 Stored Queries 134  
 String Resources 591, 1254  
 StringSet 1089  
 StringTools 1090  
 Styles 70, 394  
 StyleSelectedData 1092  
 Synchronize 158  
 Synchronous and Asynchronous functions 509  
 SysInfo 1092  
 System virtual folders 130  
 System-wide Hotkeys 302  
**T**  
 Tab 1093  
 Tab Control Properties 576  
 Tab Groups 41, 375  
 TabClickData 1096  
 Tabs 35  
 TabStats 1096  
 TAR BZip2 Options 212  
 TAR GZip Options 213  
 The Confirm File Replace Dialog 153  
 The Customize Dialog 441  
 The Default Lister 69  
 The Default Toolbars 49  
 The Dialog Message Loop 577  
 The Jobs Bar 152  
 The Lister 18  
 The Open With editor 529  
 Themes 72  
 Thumbnails Mode 348  
 Tiles Mode 350, 549

Time Shifting 232

Toolbar 904, 1097

Toolbar Appearance 420

Toolbar Context Menus 470

Toolbar Filter Fields 94

Toolbar Icons 421

Toolbar Options 422

Toolbar Sets 57, 424

Toolbars 48, 351, 420, 451, 1099, 1223

Tools Category (Pre-defined commands) 449

Tracking and Undoing File Operations 217

Transition Animations 329

**U**

UAC and Administrator Mode 244

Unattended operation 150

Undo 913

Up, Forwards, Back 21

Update Checker 308

Updates 386

User-defined Commands 449, 504

Using the Hotkey Control 500

Using the Layout Commands 566

Using Wildcards when Copying 146

Utility Panel 67

**V**

Var 1099

Vars 1100

Vector 1102

Version 1103

Video Properties 237

VideoMeta 1104

View Category (Pre-defined commands) 449

View Modes 42

Viewer 426, 1104

Viewer Appearance 427

Viewer Behavior 428

Viewer Keys and Toolbar 272

Viewer Pane 63, 430

Viewer Plugins 432

ViewerEventData 1107

Viewers 1106

Viewing Images 270

Virtual File System 130

Virtual Folders 369

**W**

Wild 1107

Wildcard Reference 614

Windows Integration 419

Windows Search 96

WinVer 1108

**Z**

Zip and Other Archives 433

Zip Comment 213

Zip File Options 436

Zip Files 213

Zip Options 210