XYPLORER HELP

Version: XYplorer 28.00 (64-bit)

Copyright © 2025

XYplorer Help

Manual for the XYplorer File Manager

by Donald Lessau

XYplorer (64-bit)

File manager for 64-bit Windows 11, 10, 8.1, 8, 7, Vista, XP, Server 2025, 2022, 2019, 2016, 2012, 2003

Copyright © 1997-2025 by Cologne Code Company

XYplorer Help

Copyright © 1997-2025 by Cologne Code Company.

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Created: 2025 in Cologne

Table of Contents

Foreword 7

Part I Getting Started 9

1 Installing XYplorer ... 10
2 Removing XYplorer ... 12
3 Glossary ... 13

Part II Configuration 16

1 Configuration Dialog ... 16
2 Multilingual Support ... 122
3 Keyboard Shortcuts and Mouse Tricks ... 125

Part III Main Menu 142

1 File ... 142
2 Edit ... 165
3 View ... 180
4 Go ... 191
5 Favorites ... 195
6 Tags ... 198
7 User ... 201
8 Scripting ... 207
9 Panes ... 209
10 Tabsets ... 213
11 Tools ... 214
12 Window ... 226
13 Help ... 229

Part IV Main Topics 233

1 Rapid Access Folders ... 233
2 Quick Search ... 236
3 Visual Filters ... 250
4 Live Filter Box ... 263
5 Color Filters ... 266
6 Tags ... 284
7 Address Bar ... 292
8 Toolbar ... 304
9 Tabbed Browsing ... 319
10 Tabsets ... 323
11 Breadcrumb Bars ... 325
12 Tree ... 328
13 Mini Tree ... 330

14 Glider ... 332
15 List ... 335
16 Hover Box ... 346
17 Custom Columns ... 351
18 Dual Pane ... 362
19 Sync Folders ... 363
20 Catalog ... 365
21 Click and Search ... 373
22 Status Bar ... 376
23 Status Bar Buttons ... 378
24 Info Panel ... 380
Properties Tab ... 380
Version Tab ... 383
Meta Tab ... 383
Preview Tab ... 383
Raw View Tab ... 391
Tags Tab ... 392
Find Files Tab ... 393
Report Tab ... 427
25 Preview ... 430
26 Floating Preview ... 431
27 Preview Pane ... 437
28 Search Templates ... 439
29 Branch View ... 441
30 Dark Mode ... 444
31 Ghost Filter ... 446
32 Paper Folders ... 448
33 Virtual Folders ... 453
34 Folder View Settings ... 459
35 Custom File Icons ... 463
36 Customize Keyboard Shortcuts ... 469
37 Custom Context Menu Commands ... 472
38 Custom Drag'n'Drop Context Menu ... 476
39 Copying files with XYplorer ... 480
40 Rich Copy ... 482
41 Backup ... 484
42 Undo/Redo ... 486
43 Recycle Bin ... 490
44 Portable Devices ... 492

Part V Advanced Topics 496

1 Admin Settings (Enterprise Edition Only) ... 496
2 Command Line Switches ... 505
3 Custom File Associations ... 512
4 Portable Openwith Menu ... 518

5 User-Defined Commands ... 522
6 Hamburger ... 523
7 Scripting ... 528
8 Scripting Commands Reference ... 572
9 Scripting Command Get ... 827
10 Startup Settings ... 850
11 Variables ... 852
12 Tweaks ... 862

Part VI Registration 864

1 How to purchase a license ... 864

Part VII Support 866

1 Contact ... 866

Part VIII Legal Stuff 868

1 Disclaimer ... 868
2 Copyright ... 869

Index 871

Foreword

This manual describes installation and usage of the XYplorer file manager for Windows.

Part I

Getting Started

1 Getting Started

Introduction

XYplorer is a Windows file manager. It features tabbed browsing, a powerful file search, a versatile preview, a highly customizable interface, an optional dual pane, and many unique ways to efficiently automate repetitive tasks. It's fast, lightweight, and portable.

This help file will guide you through XYplorer's interface, features, usage, and customization.

Yes, it's pretty comprehensive. This manual is really detailed and has lots of examples, so it's a bit overwhelming at first. Just a heads-up, though, that reading this document isn't necessary to use XYplorer. We've always made sure that our user interface is simple and self-explanatory, and we're really proud of that.

If you need further assistance, want to share an experience or make a suggestion, you are always welcome to visit the XYplorer User Forum, where you will find friendly and competent support.

Enjoy XYplorer!

Donald Lessau, Cologne Code Company, Cologne, Germany

1.1 Installing XYplorer

System Requirements

Operating Systems: 64-bit Windows 11, 10, 8.1, 8, 7, Vista, XP, Server 2025, 2022, 2019, 2016, 2012, 2003

Minimal Hardware: 100 MHz Pentium, 32MB RAM.

Monitor: A monitor resolution of at least 900x600 pixels is recommended.

Installation

XYplorer is portable. It doesn't require any "installation" in the proper sense, and using it does not change your system or registry. You can run it directly from your USB drive, and you can have as many parallel "installations" as you wish.

Choose one of the following packages:

(1) Installer Package
(2) No-Install Package

Note: No matter which package you use, Install or No-Install, running XYplorer does not change your system or registry at any time.

Installation using the Installer Package

Use this package to use an installer to install XYplorer, with icons in the start menu and an uninstaller.

Extract xyplorer64_full.zip to some temporary directory and run XYplorer64_xx.xx_Install.exe. The following files will be copied to the directory you selected/created during the install procedure:

XYplorer.exe	Application
XYcopy.exe	Background Copy Handler
XYplorer.chm	Help
Startup.ini	Sets the Application Data Path (see below)
XYplorer Website.url	Link to www.xyplorer.com
License.txt	XYplorer License Agreement
Readme.txt	Readme file
CatalogDefault.dat	Sample Catalog File
Uninstall.exe	Uninstaller

XYicon_*.ico	Some Icon Files

A program group "XYplorer" will be created under "Programs" in your Start menu, containing four icons.

The Application Data Path

The file Startup.ini contains information about the Application Data Path. By default this path is set to %appdata%\XYplorer64 which on a typical system (with English locale) is resolved to C:\Users\<user>\AppData\Roaming\XYplorer64\ (Win 8 and later).

Note that Startup.ini can be manually edited and will not be overwritten on a subsequent installation.

No further files are added anywhere to your system, and nothing will be put into the registry but uninstall information.

Installation using the No-Install Package

Use this package to install XYplorer on a removable drive (e.g. USB), or in any folder on any drive you like. All application data will be stored under the application path.

Simply extract xyplorer64_full_noinstall.zip (or *.rar) anywhere.

Dependencies

None. I.e. none that's not part of Windows anyways.

Digital Signatures

To ensure origin and integrity of the download all executable files (.EXE files) are digitally signed by "Cologne Code Company e.K." (before v18.90.0132 that was "Donald Lessau"). Make sure to check the digital signatures. Do not install or use a copy of XYplorer where the digital signatures are missing or not valid.

1.2 Removing XYplorer

If you find that XYplorer does not meet your needs you can easily remove it from your computer.

Automatic Uninstall

If you installed XYplorer via the Installer Package (e.g. XYplorer64_28.00_Install.exe), you'll find an icon "Uninstall" in the program group created during installation. Click this icon to run the uninstall interface. Alternatively uninstall XYplorer from the Windows Control Panel.

Silent uninstall is supported using the /S switch, for example: C:\Program Files\XYplorer\Uninstall.exe /S

Manual Uninstall

Alternatively you can simply delete XYplorer's application folder and its application data folder (if different).

If you go this way then -- *before* removing the application -- you should take care to clean the registry in case you have enabled any of the settings in Configuration | Shell Integration. In that case simply uncheck these checkboxes:

Configuration | *Shell Integration* | *Default File Manager* | *XYplorer in shell context menu*

Configuration | *Shell Integration* | *Default File Manager* | *XYplorer is default file manager*

1.3 Glossary

Glossary

Abbreviations

AB	Address Bar
AC	Access Control
ADP	Application Data Path
AL	Action Log
BB	Bookmark Buttons
BV	Branch View
CC	Custom Columns
CEA	Custom Event Actions
CF	Color Filters
CFA	Custom File Associations
CFI	Custom File Icons
CKS	Custom Keyboard Shortcuts
CL	Column Layouts
CTB	Custom Toolbar Buttons
DLOC	Dual Locations
DM	Dark Mode
DP	Dual Pane
DUB	Droppable User Buttons
FP	Floating Preview
FSI	Find Similar Images
FVS	Folder View Settings
GF	Ghost Filters
HB	Hover Box
IP	Info Panel
KS	Keyboard Shortcuts
LFB	Live Filter Box

LTCE	Labels, Tags, Comments, and Extra Tags
MBV	Multi Branch View
MDBU	Mouse Down Blow Up
MFS	Multi Field Search
MT	Mini Tree
MUSD	Mouse Up Show Down
MUT	Multi-User Tagging
PF	Paper Folders
PP	Preview Pane
PT	Preview Tab
POM	Portable Openwith Menu
QS	Quick Search
RAF	Rapid Access Folders
RB	Recycle Bin
RFO	Rich File Operation
SB	Status Bar
SC	Scripting Command
SRC	Search Results Caching
TB	Toolbar
UB	User Buttons
UDC	User-Defined Commands
UR	Undo/Redo
VF	Visual Filters

Special Terms

Location term: Every term that is accepted by the Address Bar, e.g. a path, a file, a quick search, a visual filter, a quick script, a dos command, an alias, a Jump and Spot pattern, Quick Select pattern.

Location port: Any interface that accepts a location term, e.g. Address Bar, Catalog, Favorites, Go To dialog, Scripts.

Part II

Configuration

2 Configuration

2.1 Configuration Dialog

Configuration Dialog

Open the Configuration dialog by menu Tools | Configuration (F9). The dialog is organized into various sections:

General: Tree and List - Sort and Rename - Refresh, Icons, History - Menus, Mouse, Usability - Custom Event Actions - Safety Belts, Network - Controls & More - Startup & Exit

Colors and Styles: Colors - Highlights & Dark Mode - Styles - Color Filters - Fonts - Templates

Information: Tags - Custom Columns - File Info Tips & Hover Box - Report & Data

File Operations: File Operations - Undo & Action Log

Find and Filter: Find Files & Branch View - Filters & Type Ahead Find

Preview: Preview - Previewed Formats - Thumbnails - Mouse Down Blow Up

Tabs and Panes: Tabs - Dual Pane

Other: Shell Integration - Features

Tip: Most settings can be Ctrl+Right-clicked to display a menu with variations of their names that can be copied to the clipboard. Can be useful for documentation or communication.

Buttons

At the bottom of the dialog you find the following buttons (and their keyboard shortcuts):

Help (F1)

Opens the Help file at the currently opened section of the dialog.

Jump to Setting... (F3)

Shows an alphabetically ordered list of all settings available in the Configuration dialog, along with a live filter box. From there you can jump directly to a setting to check or modify it. The jump targets are highlighted (where possible -- buttons cannot be highlighted) and focused. If the list shows only one item OK will trigger it even if it is not selected.

Apply (Ctrl+S)

Apply any changed settings without closing the Configuration dialog.

Obviously any settings that are applied using the Apply button cannot be undone anymore by the Cancel button.

Settings that affect the Configuration dialog itself (e.g. Configuration | Colors and Styles | Fonts | Buttons and Labels) are not applied to the dialog. This needs a close and reopen of the dialog.

OK (Enter)

Apply any changed settings and close the Configuration dialog.

Cancel (Esc)

Close the Configuration dialog without applying any changed settings.

Tree and List

Tree

Auto-optimize tree

Automatically collapse non-current nodes in Tree. Auto-optimize sets in when you expand the current tree node, or when you change location via tab switch or pane switch.

Expand tree nodes on browse

Expand a collapsed node in the tree when you browse it in a way other than directly clicking it (which is handled by "Expand tree nodes on single click"). Note that this feature only works in the Maxi Tree, not in the Mini Tree (where it would defeat the purpose of the Mini Tree).

Expand tree nodes on drag-over

Tick it to auto-expand tree nodes when you drag stuff over them. Allows you to drop the stuff into previously invisible subfolders.

Expand tree nodes on single click

When enabled any non-expanded tree node with child nodes will expand on a single click.

Check existence of subfolders in tree

If turned ON then all freshly displayed tree folders are checked for any contained subfolders (and depending on the result a "+" is display in front of them, or not). If OFF then it is simply *assumed* that subfolders exist, and the "+" icon is set unconditionally. Turning it OFF will speed up browsing the tree considerably.

In network locations as well

Turn it OFF to have the speed booster only where it is really needed: on slow drives (high-latency networks).

Remember state of tree

Tick it to remember which nodes are expanded in the Maxi Tree (the Mini Tree state is remembered

anyway ever since).

The tree state is retained across sessions and also when switching between Maxi and Mini Tree. It also works with a Locked Tree (Tools | Customize Tree | Lock Tree). Also the scroll position is remembered. So now you have a pretty good chance that you find your tree exactly as you left it last time (unless another application changed the folder structure). Of course, recreating the tree does involve additional browsing, so the startup time will be slightly longer depending on the complexity of the tree.

Show localized folder names

Tick it to show special display names (that differ from the real path name) for certain tree folders in non-English Windows installations.

Some examples from German Windows 7:

Real Path Component	Localized Folder Name
Program Files	Programme
Users	Benutzer
Public	Öffentlich

The real name is shown in the tree name hover tooltip.

When you inline rename a folder with a localized name, the real name is used instead.

These localized folder names are currently only provided for the tree.

Select parent of moved folder

If ticked then the focus jumps to the parent of the moved folder when the moved folder was the focused folder. If unticked the focus stays with the moved folder.

Select parent of deleted folder

Check it to always select the parent of a deleted folder. If unchecked the next sibling is selected (if there is any, otherwise the parent is selected).

Scroll selected folder to the top

Tick it to auto-scroll the selected folder to the top of the view.

- Selecting a folder right in the tree does *not* trigger the auto-move. It would just drive you crazy.
- Near the bottom of the tree it might not be possible to bring the folder up to the top. That's tree physics you got to live with.
- The setting of "Configuration | Menus, Mouse, Usability | Usability | Scroll margin" is honored, so you can have the folder auto-moved e.g. to the 3rd row of the view if you like.

Scroll subfolders into view

Check this to have a node automatically scroll up to show as many of its subfolders as possible when you expand it.

This setting also affects the Catalog's behavior when expanding a category.

List

Auto-select first item

Tick it to auto-select the first list item when entering a new folder. Does not overrule Select last used subfolder.

Select last used subfolder

Tick it to have the last used subfolder within the current folder auto-selected. Applies only when going back, forward, up or down.

Select next item after delete and move

Tick it to auto-select the next item in the file list after the current item has been deleted or moved away. If the current item is the last item, then of course not the next but the previous item is auto-selected.

Add new items at the end of the list

Tick it to have new items added at the end of the list. Untick it to have new items added where they belong according to the current sort order.

Always show folder sizes

Tick it to show folder sizes directly in all file lists. Of course, with large and deeply nested folders, this slows down the file browsing considerably, especially on a first browse of that folder. So, use it with care... or press ESC to abort the endless counting of bytes.

Tip 1: There is also a list style Show Folder Sizes, which gives you more control because you can enable it individually per tab, or per folder (via Folder View Settings).

Tip 2: There is also the command Calculate Folder Sizes (Shift+F5) in menu View if you need the folder sizes in the current listing now. You can have this as a toolbar button as well.

In network locations as well

Tick this to also show folder sizes for network locations. This can take a long time on slow drives (high-latency networks).

Cache folder sizes

Calculating Folder Sizes takes time. Tick this option to cache the results of the calculation for each folder, so that the size does not have to be calculated again.

If folder size caching is enabled, it's still not used on all occasions. The main idea of the cache is to speed up browsing, but in other contexts exactness is more important than speed. Here is when/how the cache is actually used:

- The cache is read and written in:
  File List; Properties Tab
- The cache is not read (but written) in:
  Reports; SC foldersize(); Selection Stats; Find Files by Size; menu View | Calculate Folder Sizes (Shift+F5)

Further Notes:

- How to refresh the cache for certain folders: Simply use menu "View | Calculate Folder Sizes" (Shift+F5). This command will refresh the cache for all selected folders, or for all folders in the list if none is selected.
- When freshly calculating a folder size not only this folder's size but also the sizes of its direct child folders are auto-added to the cache. This allows for quickly going down to the next level, and propagates fresh values one level down when refreshing folder sizes. Additionally any subfolders containing more than 256 items are auto-added to the cache.
- Folders cached as "Empty" will always be re-calculated when showing the folder sizes. If they are still empty the costs for this are minimal, and if they are not empty anymore it's really good to know. So when you see "Empty" you can pretty much rely on it since the information is freshly harvested.
- Removable drives are excluded from Folder Size Caching. The variable drive letters don't play well with the implementation of caching.
- The cache stays alive when you turn off showing folder sizes.
- The cache stays alive across sessions (it is stored in the file fsc.dat in the XYplorer application data folder).
- The cache is auto-updated when you rename a cached folder within XYplorer.
- NOTE: When the size or number of items in a folder changes the cache is not auto-updated! So it can and will become stale with time. If that's too risky for you do not enable folder size caching!
- To reduce the risk of falling into the stale cache trap a tilde (~) is prefixed to every folder size when caching is enabled.
- How to clear the cache: The context-menu of the Size column has two commands available:
  Clear Folder Size Cache Here: It will remove the current list path and all its subfolders (i.e. the complete current branch) from the Folder Size Cache.
  Clear Folder Size Cache Everywhere...: It will clear the whole cache (not just the cache for the currently listed folders).
  Note that the clearing is only done in memory, not to disk. So even after clearing it you can still return to your saved cache by choosing "File | Restart without Saving".

Show cached folder sizes only

Tick it to avoid automatic folder size calculation. Only folder sizes already present in the cache are displayed in the list, no automatic (and probably time-consuming) calculation will kick in.

Note that you still can explicitly trigger a calculation by View | Calculate Folder Sizes (Shift+F5) or by "Calculate Folder Sizes" from the Size Column Context Menu. The freshly calculated sizes are then added to the cache.

Show item count with folder sizes

Check it to also show the number of items in each folder in the list's Size column (if folder sizes are shown).

Show folder size on Properties tab

Show Size (total contents bytes sum), Size on disk and Contents for folders on the Properties tab. This feature has to be explicitly activated in configuration since for large deeply nested folders it takes some time to gather this info.

Note: Folder size calculation is not triggered at startup to avoid an undesired slowdown.

Wrap-around list

Check this box to jump to the first/last item on KeyDown/Up when the focus is currently on the last/first item. Note that KeyPageDown/Up behaves the same way in this situation.

Items in Tree and List

Select Items...

Use this button to list the following toggles that allow you to show/hide various items in the tree and list:

Links folder

Show/Hide the Links folder right under the Computer node in the Tree.

The Links folder is functionally quite similar to the "Favorites" folder in Explorer and shows the same items: all and only *.LNK files located in %userprofile%\Links (any other things in this location are ignored). Like in Explorer the Links folder can feature links to folders, to files (documents, photos, etc), and to programs:

- On clicking a folder link you jump to that folder in the Tree.
- On clicking a file link you jump the file's folder in the Tree, and then to the file in the List (Explorer is different here: it will open the file).
- On clicking a program link the program will be opened.
- You can drop items onto folder links (the items are copied or moved to the location).
- You can as well drop items onto program links (the items are opened by the program).

The expansion state of the Links folder is remembered between sessions. The Links inside the Links folder are also accessible via the Breadcrumb Bar.

Desktop folder

Show/Hide the Desktop folder right under the "This PC" node in the Tree.

Documents folder

Show/Hide the Documents folder right under the "This PC" node in the Tree.

Downloads folder

Show/Hide the Downloads folder right under the "This PC" node in the Tree.

OneDrive folder

Show/Hide the OneDrive folder right under the "This PC" node in the Tree.

User folder

Show/Hide the User folder right under the "This PC" node in the Tree.

This special folder points to the current user's profile folder (CSIDL_PROFILE).

Network folder

Show/Hide the Network folder right under the "This PC" node in the Tree.

Recycle Bin folder

Show/Hide the Recycle Bin folder right under the "This PC" node in the Tree.

Notes on the above special folders

If e.g. the Desktop node is hidden then special paths like "Desktop" will not work anymore in the special way but will be auto-resolved to their real file system path.

The Mini Tree does not really care about these settings but shows and hides what you tell him to show and hide.

Floppy drives

Uncheck this box if you want to reduce hardware access to rarely used Floppy drives (A: and B:).

CD-ROM drives

Tick it to show CD-ROM drives in Tree and List.

Hidden drives

Check to show drives that Explorer and other file managers do not show because they are marked as hidden.

Hidden files and folders

Uncheck if you don't want to see hidden system folders like "System Volume Information" and "Recycler" or files like "C:\boot.ini" or "desktop.ini". Operations on these items are often disallowed by the system or dangerous.

If hidden, those items won't also be found by a search. They will, however, be contained in folder reports and backups.

System files and folders

Clearing this check box will hide all items with the SYSTEM attribute set (regardless of the HIDDEN attribute and the "Hidden files and folders" check box setting).

If hidden, those items won't also be found by a search. They will, however, be contained in folder reports and backups.

Protected operating system files

Tick this box to show all items (file and folders) that have the HIDDEN and the SYSTEM attribute both set. If unchecked, this setting overrides the "Hidden files and folders" and "System files and folders" settings if both are checked.

Junctions

Uncheck to hide junctions (items with the REPARSE_POINT attribute set). The factory default is unchecked. This command has mainly been added to get rid of the plethora of junctions in Win7 and later.

Note that in the folder tree, on hovering a junction the tooltip displays the Junction Target.

Portable devices

Tick it to show Portable Devices and their contents in Tree and List.

Sort and Rename

Sort

Sort method

Here you can choose one of three sort methods:

- Binary: Results in a sort order based on comparisons of the internal binary representations of the characters as determined by the active code page.
  `A!=a, A<B<a<b, 1<12<2`
- Textual: Results in a sort order based on case-insensitive comparisons of the characters as determined by the system's locale.
  `A==a, 1<12<2`
- Natural: Results in a textual sort order, except that multi-digit numbers are ordered according to their numeric value. Also known as "intuitive filename sorting".
  `A==a, 1<2<12`

```
Examples:    Binary        Textual        Natural
             ----------------------------------
             Ba.txt        a1.txt         a1.txt
             a-a.txt       a12.txt        a2.txt
```

```
a-b.txt       a2.txt        a12.txt
a1.txt        aa.txt        aa.txt
a12.txt       a-a.txt       a-a.txt
a2.txt        ab.txt        ab.txt
aa.txt        a-b.txt       a-b.txt
ab.txt        Ba.txt        Ba.txt
```

Sort folders apart

Check this to make XYplorer behave like Explorer in sorting folders apart from files. Uncheck it if you want everything in one alphabet.

Keep folders on top

Check it to keep the folders at the top of the list always (independently of the sort order and column).

Sort folders always ascending

Tick it to have the folders always sorted by name in ascending order. Especially useful if also "Keep folders on top" is ticked.

Note that "Mixed sort on date columns" will overwrite this setting for the Date columns.

Mixed sort on date columns

Check to use a mixed sort (mix folders and files when sorting) always on the Date columns.

Mixed sort on tag columns

Check to use a mixed sort (mix folders and files when sorting) always on the columns Label, Tags, and Comment.

Mixed sort on path columns

Check to use a mixed sort (mix folders and files when sorting) always on the columns Path (Search Results) and Original Location (Recycle Bin).

Sort filenames by base

Check it to sort filenames primarily by base, secondarily by extension. Note that Windows Explorer always sorts by full name (base + extension). For example:

```
Natural numeric sort order ON
-------------------------------------
Sort by base ON     Sort by base OFF
-----------------   -----------------
a.txt               a (copy).txt
a (copy).txt        a.txt
a2.txt              a2.txt
a10.txt             a10.txt
```

```
Natural numeric sort order OFF
--------------------------------------
Sort by base ON     Sort by base OFF
-----------------   -----------------
a.txt               a (copy).txt
a (copy).txt        a.txt
a10.txt             a10.txt
a2.txt              a2.txt
```

Treat hyphens and apostrophes like normal characters

Unless it's a binary comparison, Windows treats hyphens and apostrophes in a special way during string comparison that might not produce the desired sort order. This option allows you to control this behavior. For example:

```
Not ticked:         Ticked:
  File.txt            File.txt
  File'A.txt          File'A.txt
  File-A.txt          File'C.txt
  FileB.txt           File-A.txt
  File'C.txt          File-C.txt
  File-C.txt          FileB.txt
```

- This setting only affects the Text and Natural sort methods.
- And it only affects sorting in the main file list. Not in the folder tree and not anywhere else.
- Ticking this setting will slow down sorting a tiny bit, so if you don't need it, don't tick it.

Sort size (date) columns descending by default

Allows you to control the default sort order for certain column types in List.

Keep current item in view after sorting

If checked then the current item (the item that's focused and selected) stays in view (if possible) after the list is resorted. And not only in view, but -- again: if possible -- exactly in the same place in the viewport. So your eyes don't have to go on a trip looking for it.

This setting is ignored if there are no selections in the list. In that case what happens after sorting is controlled by Scroll to top after resorting following here below.

Tip: If the setting is unticked you still can enforce it by holding CTRL while clicking the column header. This will even keep the current item in view when it is not selected, so CTRL gives you even a little more power than the config setting.

Scroll to top after resorting

If checked the list is auto-scrolled to top when you change the sort order. Additionally, if there are no selections in the list, the focus is auto-shifted to the top item. Note that the option is honored only

when no current item is kept in view due to an enabled Keep current item in view after sorting (in other words, the latter setting takes precedence).

Show sort headers in all views

Tick it to have sort headers (column headers) also in non-Details views, e.g. in Thumbnails. You can click the column headers to sort the list by this column.

Note that in views "Small Icons", "List", and "Tiles" you can modify the column width by resizing the Name column header.

Show implicit secondary sort order arrow

Tick it to show the implicit secondary sort order (if any) by a very light small sort arrow.

Rename

Preselect name

Tick it to preselect the name in the edit box.

Exclude file extension from initial selection

Check to exclude extension from the rename pre-selection.

Hide extensions from rename edit box

Tick it hide extensions from the inline rename box. Can be quite useful to have them out of the way. Very rarely you want to change the extension of a file anyway.

While renaming the extension is shown at the right end of the edit box.

There is still a way to modify the extension: While the rename box is open, and nothing is selected, and the cursor is at the right end, you can press the Right Arrow key (alternatively the End key) to "move" the extension into the rename box. So, to quickly rename the extension while "Hide extensions from rename edit box" is enabled you can press the sequence F2 - Right - Right or F2 - End - End.

Serial rename with Up and Down keys

Check it to use the cursor keys UP and DOWN to go to the next and previous file while renaming them. Additionally, the current caret position (e.g., your caret is at the 8th letter) is kept if possible (if the filename is long enough), so renaming files in a list now is almost like moving and typing around in a multi-line text block. Very intuitive and convenient!

Notes:

- Serial Rename is only invoked if nothing is selected in the current rename box.
- Hold down the CTRL key if you want to measure the cursor position from the right end of the file name. This is useful when editing serial numbers and other parts that are usually appended to the name.
- Serial Rename with (Shift+)Tab is always enabled by default, regardless of this setting. However, no caret positioning is performed.
- Serial Rename is also supported in various lists under Tools | List Management and all other small

lists where items can be renamed.

- In thumbnails views with more than one caption line you can use PageUp/PageDown for serial rename.

Show name length while renaming

Tick it to have a little info box at the right end of the inline rename box showing the current length of the filename and also (in parentheses) the length of the full path.

- Applies to Tree and List.
- The full path length is that of the real path without trailing backslash, for example:
  Tree path: Desktop\sub\
  Measured path: C:\Users\Donald\Desktop\sub

The info box is color coded:

Red with yellow text	Invalid filename! (e.g. "Test>.txt", "con", "lpt1")
Red with white text	Name is too long! (longer than 259 characters without any trailing backslash)
Orange with white text	Move on rename. (List only)
Dark grey with white text	Same as source name.
Purple with white text	Name exists.
Green with white text	Name is available.
Blue with white text	Change in capitalization.

Use dialog to rename single items

Tick it to use a dialog instead of the usual inline rename in Tree and List.

Note that the setting only affects the F2 rename; the slow double-click will always open the inline rename.

Allow move on rename

A cute little time saver that enables you to move-on-rename files or folders in one go right from the file list. Simply enter a relative (to the current path of the file!) or absolute path\[name] into the inline rename box (F2) in the file list, and move-on-rename will happen. If the target folder doesn't exist yet it is created on the fly.

For example, select a file in the file list and enter this into the rename box:

`C:\new.png` = Move file to C:\ renaming it to new.png.

`C:\1\2\3\new.png` = Move file to C:\1\2\3 renaming it to new.png. If "C:\1\2\3\" does not exist it is created.

`..\new.png` = Move file one up renaming it to new.png.

`stuff\new.png` = Move file one down to subfolder "stuff" renaming it to new.png. If "stuff" does not exist it is created.

You can also move a file or folder without renaming it. Simply select the rename box, and enter a path with a trailing slash:

`C:\stuff\` = Move file to C:\stuff\.

`stuff\` = Move file one down to subfolder "stuff".

`..\` = Move file one up.

Or create a new folder and paste `Parent\Child` into the fresh rename box: The relative path "Parent\Child" is created in the current path.

Note that there is Batch Move, a move-on-rename for many files at once.

Tip: If Show name length while renaming is enabled a yellow warning icon is shown in the box when the new filename would trigger a move-on-rename.

Preview all Rename Special operations

Check to show a preview for each of the Rename Special operations, including detection of possible conflicts.

Auto-replace invalid characters

If ticked then invalid characters in pasted or typed input are replaced by the character defined at Character to replace invalid characters in dropped messages (in Templates). If no character is defined then the invalid input is simply ignored.

Applies to inline rename in Tree and List as well as to dialog based rename (see above Use dialog to rename single items).

Resort list immediately after rename

Resorts list after single and batch renames, and scrolls focused item into view.

Note that on Serial Rename (using the Tab key) the list will not be auto-resorted after each rename operation even if you checked "Resort list immediately after rename" in Configuration | Tree and List. It would be extremely confusing if the list resorted between each rename. The list will also not be auto-resorted if the current sort order is manual or random.

Also when a single rename is triggered by focus loss (e.g. when you click outside the rename box) the list does not auto-resort because, again, it might lead to confusion.

So, a single rename (aka inline rename) should be completed by pressing [Enter] in order to invoke an immediate resort.

Set archive attribute on folder rename

Tick to set the Archive attribute on all contents a renamed folder (including subfolders). This allows backup operations by programs like XCOPY to decide what has to be backed up.

Note: This setting makes renaming huge folders (> 100,000 items) quite slow because attributes have

to be retrieved from each file and then possibly set for each file.

Renaming Items in Tree and List (General Remark)

The above rename dialog as well as inline rename in Tree and List support the following

Multiline Paste: You can paste multiline clipboard contents into the edit box. The line feeds are replaced by a space before pasting, so all lines go into the edit box.

Refresh, Icons, History

Auto-Refresh

Auto-refresh

Automatic refresh (of tree and list) on file system changes. You don't need to press F5 (Refresh) anymore to get a fresh impression of reality.

In network locations as well

Auto-refresh network locations as well, including mapped network drives. Unticking this option will speed up network access on some systems.

Include removable drives

Auto-refresh removable drives as well. Applies to USB sticks, thumb drives, flash card readers, and the like, but NOT to floppies (which are excluded from all auto-refresh).

Include virtual folders

Auto-refresh Virtual Folders as well. Auto-Refresh in Virtual Folders works only *if* the common path or branch of the contained items supports it. That path is displayed in the Turkish Rose colored Information Bar; if no path is displayed then you get no Auto-Refresh.

Refresh during file operations

Check to auto-refresh the Tree and current List while file operations triggered within XYplorer are in progress.

Tip: File operations will generally run faster if you do not enable this setting because the permanent refreshing takes time.

Respond to file system notifications

Tick it to automatically refresh Tree and List on file system changes anywhere.

Depending on your system this can lead to more refresh events than you might like or need.

Detect portable devices

Tick it to automatically detect when portable devices are inserted or removed.

Tip: If you don't need that refresh on Portable Device plug in/out then untick "Detect portable devices". Your tree will be calmer.

Icons

Draw selected list icons dimmed

Check it to draw the icons of selected list items in a dimmed way. It was a common thing in XP and earlier versions of Windows.

Draw hidden icons ghosted

Check it to draw the icons of hidden items in a ghostly, semi-transparent way. The default is ON, as this is the Windows standard. Here you can turn it off to drive the ghosts away.

Cache specific icons

Enable it to obtain specific icons (e.g. for ICO files) from the shell icon cache. It's faster than accessing the files each time.

Use generic icons for super-fast browsing

Generic icons are determined by the extension (or type) of a file only, and not by its specific contents and hence need no file access to be determined, which makes retrieving them much faster, especially in network locations.

Note that only the following file types can have specific icons (which typically but not necessarily are embedded icons), so only files of these types will show different icons when you enable the use of generic icons: folders and *.exe *.lnk *.ico *.icl *.cur *.ani *.htm(l) *.url.

But only in network locations: Since the effects will be most notable in network locations, there is a second option that allows you to confine the use of generic icons to network locations.

Apply to all controls: Tick it to have super-fast generic icons everywhere (Catalog, Status Bar, Properties, various other lists), not just in the file list. Note that the Toolbar is an exception and will always show specific icons for Droppable User Buttons (DUB).

Show icon overlays

Show icon overlays (as for example used by TortoiseSVN, a Subversion client, implemented as a Windows shell extension).

Note that icon overlays are not retrieved for (UNC or mapped) network locations. Reason: Retrieving icons slows down network browsing even if there are no overlays to be found, and the latter is quite likely the case.

In tree as well

Tick to show icon overlays also in the folder tree. The idea of this refined control is that polling the icon

overlays can take a long time for a large tree and slow down startup considerably. But you might need those overlays only for the list.

In network locations as well

Tick to show icon overlays also in network locations. Note that this can slow down network browsing considerably.

Show shortcut overlays

Show standard icon overlays for LNK files, junctions, and symbolic links in Tree and List.

Show shared folder overlays

Show standard icon overlays for shared folders in Tree and List.

Show embedded icons on Properties tab

If enabled then any embedded icons are extracted from the following file types:

.exe .dll .cpl .ocx .scr .icl

Note that this operation updates the last accessed date of a file.

Show custom file icons

Tick to display any defined Custom File Icons. No restart required.

History

History without duplicates

This option makes the history work very similarly to an MRU (most recently used) list. Differences to the normal history:

Automatically avoids duplicates in the history.

Note that the History holds up to 256 items.

History per Tab

History per Tab means that Back and Forward move within each tab's individual history, as opposed to the global history over all tabs.

If History per Tab is enabled then:

(1) Back and Forward (Toolbar buttons and Keyboard Shortcuts) are confined to each tab's individual history.

(2) The Back and Forward buttons's "arrow drop-downs" are confined to each tab's individual history.

(3) In Tools | List Management | History... those history items belonging to the current tab are shown bold.

(4) If History without duplicates (see above) is enabled then duplicates on different tabs are tolerated.

(5) The Toolbar back/forward buttons are blue instead of green.

The per-tab-data are saved between sessions. They are even maintained and saved if you untick History per Tab. So you can enable it at any time and get the desired functionality right away.

History retains selections

If checked the selected items are stored with the History. If you return-by-history to a location you have visited during the current session, the location's previous selections are restored. The selections are NOT remembered between sessions. Factory default is OFF because it can take quite some memory.

History retains sort order

If checked the sort order is stored with History. If you return-by-history to a location you have visited during the current session, the location's previous sort order is restored.

Scripting

Remember permanent variables

Tick it to remember Permanent Variables across sessions.

Menus, Mouse, Usability

Menus

Native context menu

Tick it to show the lightning fast Native Context Menu by default, untick it to show the Shell Context Menu by default. Affects right-clicking items in tree and list.

The Native Context Menu has a couple of interesting features compared to the Shell Context Menu:

- It is lightning fast.
- It features a second open command, Open by Shell, which opens the selected items exactly as if they were opened via the Shell Context Menu. Especially when using Custom File Associations, you now have quick access to two alternative open commands. Note that even if both commands point to the same opening application, there may be subtle (and undocumented by Microsoft) differences in the opening process.
- When hovering the Open or Open by Shell commands the status bar displays the application that will be used to open the right-clicked item.
- It features the additional commands Selection Stats (summary stats on the current selection) and Metadata (all available Tags, Special Properties, and Shell Properties for the right-clicked item).

Hold Ctrl to invert the above selection

Tick it to replace the default context menu (Native Context Menu or Shell Context Menu) by the other one when you hold down the Ctrl key while right-clicking an item.

- CTRL can be combined with other function keys such as SHIFT and ALT.
- Holding down the Ctrl key can be replaced with one-handed "rocker-click" (Left-Mouse-Down + Right-Mouse-Click).
- When both Native context menu and Hold Ctrl to invert the above selection are off, holding down the Ctrl key still works in the tree to show the Native Context Menu.

Custom items in context menu

Check to show a couple of custom items at the top of the context menu of List items.

The buttons "Folder Tree..." and "File List..." allow you to further refine which custom context menu items you want to see in the Tree or List.

See here for descriptions of the individual commands.

Hide shell extensions from shell context menu

Check to have a clean context menu without all those funky shell extensions. For purists only.

Native drag and drop context menu

Check this option if you prefer to see XYplorer's native drag and drop context menu instead of the shell's original one. The native menu has an entry called "Shell Context Menu..." which allows you to show the shell menu from the native menu.

Tip: If you hold CTRL while releasing the right mouse button the the effect of this setting is inverted. That way you can easily switch between both drag and drop context menus without going first to Configuration. If you don't like the current menu hold CTRL to show the other one.

Navigation commands in List context menu

Check to have the items Up, Back, and Forward in the List's context menu.

Find Files commands in List context menu

Check to have the Find Files section from menu Edit in the List's context menu.

Cell Context Menu

Hold Ctrl to show cell context menu

Untick to pop the Cell Context Menu by a simple right-click without holding CTRL. That way the new cell-oriented functions are more easily accessible.

If unticked you still can pop the standard Edit context menu by holding CTRL (or by "rocker-click": hold the left mouse button down and then right-click simultaneously).

Use localized search and filter patterns

Tick it to use localized terms (eg "Taille: 1 Ko" instead of "Size: 1 KB") when triggering a Live Filter or Quick Search from the cell context menu.

Mouse

Single-click to open an item

Tick it to open list items by a left button single click. It will trigger the same action as if you double-clicked the item. The action is triggered on MouseUp.

Note: In Thumbnails view Single Click Open works only if Mouse Down Blow Up (On left mouse down) is OFF, else both features would conflict.

On the icon only

Tick it to have the single-click work only on the icon, not on the caption. That way you can still select an item by clicking the caption.

Folders only

Tick it to single-click open only folders, not files.

Point to select

Tick it to enable the Select On Hover mode: A list item is selected (and focused) when you hover it.

- Holding CTRL or SHIFT works just like with normal click-selection.
- The delay is controlled by "Configuration | Information | File Info Tips & Hover Box | Initial delay in milliseconds".

Tip: Untick "Configuration | Information | File Info Tips & Hover Box | Show info tips only when hovering file icon" so you still can see file info tips and Hover Box when hovering the caption of an item.

To the icon only

Tick it to have "Point to select" work only on the icon, not on the caption. That way you can still hover the caption without triggering selection.

Full name column select

Tick it to select an item in the file list not only by clicking the icon or caption but the whole Name column. Works in views Details, Details with Thumbnails, List, and Small Icons. Also the right-click and dbl-click will do what they normally do only on the caption. The effect is confined to left and middle click. That way right-click on empty space will still trigger the empty space context menu.

The setting affects the hot zone of File Info Tips & Hover Box. When this option is enabled, this zone is not limited to the label itself, but includes the entire width and height of the Name cell.

The focus line, selection rectangle, and color filter backgrounds are drawn as wide as the Name column in Details, List, and Small Icons views if this setting is enabled.

Tip 1: Drag-select still can be initiated in List and Small Icons views when you start it in the empty area of the Name column of a non-selected item.

Tip 2: You can force the List's white space context menu by Ctrl+Right-Click on the empty area of the Name column.

Allow dragging from a background window

Tick it to allow dragging items from the file list without moving the XYplorer window to the foreground on mousedown. You would usually want this behavior when dragging items to another, foreground, application.

Shift+Wheel scrolls horizontally

Check to map Shift+Wheel to horizontal scrolling (when possible). If unchecked, or if horizontal scrolling is not possible, Shift+Wheel does PageUp/Down.

Ctrl+Wheel scrolls through the list views

Check to scroll through the list views (Details through Details with Thumbnails #1) of the active pane by Ctrl+Wheel over the active pane's list (WheelDown=Next View, WheelUp=Previous View).

- Changing the views of the inactive pane is currently not supported.
- If this new option is enabled then logically "Configuration | Colors and Styles | Fonts | Enable zoom by Ctrl+mouse wheel" won't work over the active pane's list anymore.

Tip: There are two keyboard-shortcut-only commands that you can use as an alternative to the mouse wheel:

- Miscellaneous | List | Next View (Ctrl+9)
- Miscellaneous | List | Previous View (Ctrl+Shift+9)

Usability

Highlight hovered items

Tick it to highlight the hovered items in Tree, List, Catalog, and all small lists.

Show tooltips

Untick it to hide any tooltips when hovering Toolbars, Tabs, Breadcrumb Bars, Status Bar, or Catalog.

Note that this setting does not affect the File Info Tips (Configuration | Information | File Info Tips & Hover Box), and other tips inside Tree and List.

Show verbatim tooltips

Tick it to show what the mouse is pointing at in a tooltip. This option is aimed directly at screen readers that are reading tooltips aloud.

It's a radical setting that overwrites all other settings relating to mouse pointing in Tree, List, and Toolbar. If it is enabled, for example, no more Hover Box is displayed.

Tooltip zoom (%)

Enter a percentage (100 - 400) that will be applied to most tooltips in the main window. Can turn a tooltip into a kind of magnifying glass.

Scroll margin

Here you can set the distance in lines from top or bottom where scrolling sets in when you move around in a long list by arrow keys. The scroll margin gives some context to your cursor. It lets you see where you are going before you go there. Which is quite relaxing. Recommended value is 2.

Affects keys Up, Down, PageUp, and PageDown in Tree and List when there is a vertical scrollbar.

Note that the value also controls where the focused item is positioned when you do Ctrl+Dbl-Click on the white space (= Scroll focused item into view) in Tree or List.

Wheel scroll lines

Here you can set the number of lines that are scrolled for each notch that the mouse wheel is rotated.

- Set it to 0 (or leave it empty) to use the global system value.
- The actually scrolled lines will never exceed a whole page. So you can set it to a high value (eg 1000) to always scroll the whole page.
- The value -1 is reserved by Windows for "whole page", so that's another way to always scroll the whole page.
- Values -2 and less will scroll in reversed direction.

Thumbnails and Tiles: In Thumbnails and Tiles views the wheel does exactly one line per notch regardless of the *main* value you set here at Wheel Scroll Lines. However, there is an advanced way to separately specify the lines-per-notch for Thumbnails and Tiles views: Simply append it to the main value, separated by ; (semicolon). The default value for the second number is 1. Examples:

```
0     system value for all, but 1 for thumbs/tiles
3     3 for all, but 1 for thumbs/tiles
0;0   system value for all
5;2   5 for all, but 2 for thumbs/tiles
5;0   5 for all, but system value for thumbs/tiles
```

Custom Event Actions

Here you can define things to happen when certain events occur. These are events from the perspective of the application, e.g. a double-click on some area of the window, the use of a special mouse button, a click on a file or a folder, or the change of the current location. Everything here is organized within a large list with three columns:

- Event: The supported events are listed in the left column of the list.

- Action: The middle column offer an array of actions for each event, e.g.: *None, Go up, Go back, Scroll to top, Scroll current item into view, Refresh, Run script*. Click into the cell to select an action of your choice.
- Script: The right column shows the beginning of any script defined for the "Run Script" action. Click into the cell to edit the script. The column's right-click menu offers some useful commands.
  A green dot tells you that a "Run Script" action is enabled and a script is defined. A red dot tells you that "Run Script" is enabled but no script yet exists to be run.

Keyboard support

F11: Pop the actions menu for the selected event.
F12: Pop the Edit Script dialog for the selected event.

Notes on various events and actions

Action "Go up": Supports the same keyboard modifications as the "None" action:

Hold CTRL:	Scroll current item into view
Hold CTRL+SHIFT:	Scroll to top
Hold SHIFT:	Go down

Events "Double-click on white in file list", "Middle-click on white in file list", "Right-click on white in file list":

- The variable `<CEA_ClickedItem>` is set to the item (full path) when double-clicking anywhere on the row of an item. For example:
  `if (<CEA_ClickedItem>) {tab("new", gpc(<CEA_ClickedItem>, "path"));} //open the path of the dbl-clicked item in a new tab (useful in search results)`
- The variable `<CEA_ClickedColumn>` is set to the name of the clicked column (as it appears in the interface). Works in Details view only.
- The variable `<CEA_ClickedColumnCanonic>` is set to the canonic name (UI language independent) of the clicked column. Works in Details view only.
- The variable `<CEA_ClickedCell>` is set to the content of the clicked cell. Works in Details view only.

Events "Clicking on Line Numbers": Obviously, these events are only available in the Details View (including Details with Thumbnails), which is the only view with line numbers.

- The variables `<CEA_ClickedItem>` and `<CEA_ClickedItems>` are available to scripts triggered by "Run script".
- Left-click: Note that you get a 200ms delay before the action is triggered because we have to wait for a possible double-click before firing the left-click event.

Event "Right-click on tab": Actions Small menu and Large menu can be inverted by holding CTRL while right-clicking. Rocker-click (left button down, right button click) works as well.

Event "Middle-click on folder": Affects Tree, List, Catalog, and Breadcrumb.

Event "Middle-click on file": Currently implemented only for the file list.

Action "Sticky selection": Click on line number toggles list item selection. With "Sticky selection" activated you cannot start a lasso selection from the Line Numbers column anymore with that button.

Action "Clicked item context menu": Show a basic (= no shell extensions) shell context menu for the clicked item. That menu pops faster than the normal shell context menu.

Event "After painting the file list": Is also called after "New Tab" and "Clone Tab", even though the location is not changed. It is also called at startup for the startup location.

Event "After file operation": Is fired when a file operation, including refresh of the current listings, is complete. The summary report window of a Custom Copy/Move might still be showing at this time.

- Supported file operations are "Move", "Copy", "Delete", "Backup", "CustomCopy", "CustomMove", "Sync", and "Rename".
- See below for some special variables that can be used when scripting this event.
- You can use this event for example to do automatic check-ups after file operations and display status messages, or do some automated logging, or play a sound after a file operation has completed.

Action "Double-click on line numbers header | Autosize columns now": If you hold down CTRL while double-clicking, the automatic resizing only applies to the visible rows.

Event "Startup": It is fired when load is complete and the window is visible, right after any command line script is run.

Event "Exit": It is fired right before the automatic save settings on exit (if any). See below for how to cancel the exit process.

Scripting Custom Event Actions

Depending on the event there are some useful variables here.

Event "After painting the file list": For example, if you'd like a cup of coffee whenever you change to a folder called "Desktop" you could place this script into the "After painting the file list" event, and set the related action to "Run script". The `<newpath>` variable contains the new location, as you might have guessed:

```
if (<newpath> Like "*\Desktop") {
  makecoffee;
}
```

Tip 1: The `<newpathspecial>` variable contains the special path if the new location was one (eg "Documents\Guitar"), otherwise it returns the real path just like `<newpath>` (eg "C:\Users\Donald\Documents\Guitar").

Tip 2: The `<oldpath>` variable contains the path before the location change.

Events "Middle-click on folder" and "Middle-click on file": When these events run a script, the variable `<CEA_ClickedItem>` is set with the clicked folder or file. It always returns the item without a trailing backslash, whether file or folder.

Events "Middle-click on folder" and "Middle-click on file": When these events run a script, the variable `<CEA_ClickedItems>` (plural!) is set with a |-separated list of all currently selected and clicked items (you can middle-click multi-selections without losing them). Folders in that list have a trailing backslash.

Events in section "Clicking on Status Bar": The variable `<CEA_ClickedItem>` holds the index of the

clicked Status Bar section (1, 2, or 3).

Event "After file operation": The following variables can be used in the script:

```
<CEA_FileOp>          //type of operation (always in English)
<CEA_TimeMsecs>       //total duration of the operation in milliseconds (including all
waiting at prompts)
<CEA_SourcePath>      //source path, without trailing slash
<CEA_TargetPath>      //target path, without trailing slash
<CEA_SourceList>      //list of source items, |-separated
<CEA_TargetList>      //list of target items, |-separated
```

Event "Drive added or removed": The variable `<CEA_DriveAddedRemoved>` can be used in scripts called by this event (and only there). It is resolved to the drive root path prefixed by "+" or "-" depending on whether it was added or removed, for example "+G:\" or "-G:\".

Event "Exit": You can have the script cancel the exit process and keep the app open. To do this call the special scripting command `cancelexit`, for example like this:

```
if (confirm("Exit app?") == 0) {cancelexit;}
```

The command `cancelexit` does nothing visible when called, but when the script is completed, the app will stay open. Useful for prompts on exit.

Safety Belts, Network

Safety Belts

Show drag status box

Tick it to show the Drag Status Box, a color-coded information box shown at the mouse pointer when hovering over a drop target. It tells you how many items are being dragged, what's the type of operation (Copy, Move, Create Shortcut, Add to Paper Folder, Right-Drag), and what's the target path. When exactly one item is dragged the box shows the item name (without path).

The factory default colors are:

- Orange = Move
- Blue = Copy
- Purple = Create Shortcut
- Graphite = Add to Paper Folder
- Light Green = Right-Drag

The Drag Status Box is implemented for dragging onto Tree, List, Breadcrumb Bar, Catalog, Tabs, and Toolbar.

Disallow left-dragging from folder tree

If you feel shaky with the mouse this is your safety belt in the folder tree. Note that the right button is

always available for dragging.

Disallow left-dragging from file list

If you feel shaky with the mouse this is your safety belt in the file list. Note that the right button is always available for dragging.

Confirm drag and drop

Tick it to pop a confirmation prompt for every left-mouse drag and drop operation. A reliable protection from accidental drops.

Note: If "Disallow left-dragging from tree and list" is ticked then "Confirm drag and drop" is not applicable, of course.

Confirm copy and move operations

Tick it to get a confirmation prompt before any move, copy, or backup operation, be it from clipboard, drag and drop, or other means.

This setting includes and supersedes "Confirm drag and drop" whose setting is ignored when "Confirm copy and move operations" is ticked.

Confirm delete operations

Tick it to get an XYplorer native confirmation prompt before any delete operation, be it to recycler or permanent.

If only empty folders are deleted then "Empty Folders" is explicitly mentioned in the caption.

Remarks

Note the differences to the somewhat related setting "Configuration | File Operations | Miscellaneous | Suppress delete confirmation dialog":

- The latter is about suppressing the Windows Delete Confirmation.
- The latter will actively suppress a prompt, but unticking it will not necessarily show the prompt! This is controlled by Windows global settings (located in the Properties tab of the Recycle Bin context menu).

Of course, if the Windows Delete Confirmation is enabled in the system, AND showing it is NOT suppressed in XYplorer, then you will be double prompted if "Confirm delete operations" is enabled.

The XYplorer native confirmation prompt becomes particularly useful when you travel with portable XYplorer and board an unknown host system where the Windows Delete Confirmation might be turned off. Then you can easily turn on XYplorer's prompt and be back on the safe side without modifying the host system.

Deep Stats

The three above safety prompts (Confirm drag and drop, Confirm copy and move operations, Confirm delete operations) come with the deep recursive stats of the operation that's about to happen. For example, where Windows just warns you about deleting "1 folder" XYplorer will tell you how many files will be deleted inside this folder, and how many bytes we are talking about!

Gathering the deep stats might take a little longer if you select lots of folders, but the added safety makes up for it easily. Besides you will regain the lost seconds because the operation will run faster after the full stats have been retrieved due to Windows disk read caching.

Delete on key up

If ticked then all commands that delete files are triggered on KeyUp instead of on KeyDown if you use the keyboard to trigger them. All delete commands including Wipe are affected.

Disallow delete by key in folder tree

Tick it if you want to prevent accidental deletions-by-key in the folder tree.

Treat portable devices as read-only

Tick it to allow only copying from, and no other file operations to or from or on any portable device. You will get a warning message whenever you turn off this option. See also Portable Devices.

Directional formatting codes protection

Tick this to protect yourself from the so-called "Unitrix Exploit" where certain Unicode characters (Directional Formatting Codes, DFC) are used maliciously to disguise executables.

For example, U+202E (Right-to-Left Override): If you insert U+202E at the 2nd position of the file "Htxt.exe" it will be shown as "Hexe.txt" (in *every* Unicode-compliant software), an apparently harmless TXT file which in reality is a potentially dangerous executable. When you double-click this "TXT" file the executable will be run.

If you tick "Directional formatting codes protection" then "Hexe.txt" will be shown as "H*!*txt.exe", i.e. the U+202E character has not been resolved but replaced by "*!*", and the file is clearly recognizable as executable.

DFC protection is implemented in the following places:

- In the File List: each DFC in all filenames is replaced by "*!*".
- In the Status Bar: each DFC in the current filename is replaced by "*!*".
  NOTE: Because DFCs can make the Status Bar unreadable you get this replacement independently of the setting of "Directional formatting codes protection".
- When opening a file containing any DFC, you are warned.

Here is the list of the 7 DFCs handled by this safety belt:

```
U+200E = "Left-to-Right Mark"
U+200F = "Right-to-Left Mark"
U+202A = "Left-to-Right Embedding"
U+202B = "Right-to-Left Embedding"
U+202C = "Pop Directional Formatting"
U+202D = "Left-to-Right Override"
U+202E = "Right-to-Left Override"
```

Further Remarks:

- XYplorer allows you to search for files with such characters in the name. You can paste them into the Name field (you won't see them but they are there).
- The "Convert to ASCII" command in Rename Special is a quick way to remove such characters from a filename (but, of course, it will remove ALL Unicode characters).

Further information:

- http://www.unicode.org/reports/tr9/

Network

Assume that servers are available

If you are browsing the network with this option enabled then the check for existence of servers is skipped, with two possible effects:

(1) It might speed up browsing and start up.

(2) In some rare configurations the checks are not reliable (servers that do exist are not seen), so if YOU know the servers do exist, you don't need the check and can turn it off.

Note: When skipping the check, password protected servers won't pop up a logon dialog!

Cache network servers

Check if you want your servers to be stored between sessions; the Network node in the Tree will be populated much faster then.

You can edit the cached list in List Management | Servers in Network Folder....

Assume that mapped network drives are available

Check to avoid long delays when there are unavailable mapped network drives. But there is a price: You get the "available" icon for all network drives, even the unavailable ones.

If you don't mind losing 20 seconds here and there and prefer fresh and true information, uncheck this setting.

Skip calculation of free disk space for mapped network drives

Check to avoid long delays when displaying the This PC list and there are unavailable mapped network drives.

Controls & More

Drop-down Lists

Auto-complete recently used items

When you start typing into the field, it shows you a list of items (sorted alphabetically) you've used before that match what you've typed so far.

Move last used item to top

Check to have the last used item always at the top position in the drop-down list.

Select list items on mouse hover

When the list is dropped the mouse selects list items by mere hovering (no mouse buttons pressed).

Select all on focus by key

When the control gets focused by Tab key the current contents are auto-selected.

Select all on focus by mouse

When the control gets focused by clicking into the edit box the current contents are auto-selected.

Select all on item change

When you select a new item from the dropdown list by clicking or pressing [Enter], or you reset the current temporary item by pressing [ESC], the new edit box contents are auto-selected.

Select match on drop down

Tick it to auto-select the item in the dropdown list that matches the current item in the edit box (if any) and scroll it into view (if necessary).

Auto-Complete Path Names

Enable auto-complete paths for Address Bar (also affects the, Go to... dialog) and Find Files Location. See here for details.

Filter: Here you can choose whether the Auto-Complete match list should contain "Folders only", "Files only", or "Files and Folders".

Miscellaneous

Support overlong filenames

Tick it to support filenames with more than 260 characters.

Background: In NTFS, the total length can be up to 32,765 characters with each component limited to 254 characters. The Windows Shell, however, doesn't allow overlong filenames (> 260 characters, including the terminating null character) and Shell functions (e.g. Copying, Moving, Deleting) will fail with items having overlong filenames. However, since it is possible to create items with overlong filenames under NTFS, with Explorer you are stuck when meeting such item: You cannot delete, rename, or move it! Consequently, also XYplorer will fail wherever it uses Shell functions. When Support overlong filenames is enabled, however, XYplorer will resort to Kernel functions that can deal

with overlong filenames and thus work around the Shell limitations.

Note: XYplorer reports the filename length without the invisible terminating null character, but the maximum filename length for the windows shell is defined as 260 *including* the terminating null character. Therefore a filename reported as 260 characters by XYplorer is actually 261 characters for the shell and hence too long.

Here is a list of features with support for overlong filenames in XYplorer:

- Browse and search folders.
- Auto-refresh.
- Calculating the folder size.
- Create new files and folders within folders with overlong names.
- Inline Rename (Tree and List).
- Drag and drop overlong files, and cut/copy/paste them via clipboard. Note that the move/copy/delete support for overlong filenames is limited to operations on single items. Note also that the clipboard (tested on XP) will not handle items > 1027 characters.
- Image preview and thumbnails.
- Properties (Info Panel).
- Raw View (Info Panel).
- readfile, writefile (Scripting commands).
- Quick File View, Preview Tab, Preview Pane, Floating Preview, Full Screen Preview.
- Rename Special (menu File).
- Swap Names (menu File).
- Delete: If at least one of the items selected for deletion has an overlong filename you are automatically prompted to perform a special delete operation for such overlong items. Note that these deletions are permanent, they don't go to the Recycle Bin (it doesn't support overlong filenames).
  Note that the automatic detection only scans the selected items, not any contained items (in selected folders). If only contained items are overlong, the automatic detection will not see them and the deletion will fail (shell error message). In that case use the command Delete Long from menu File to successfully delete the items in question.
- All "Copy Here..." commands (menu File), but only for files, not for folders.
- All Backup operations: Note that Backup Operations are not affected by the setting of Support overlong filenames! They always support overlong filenames.

Support volume labels in paths

Tick it to support Volume Labels in path specifications, e.g. TheSystem:\Windows. See Named Drives.

Copy paths to the clipboard with a trailing slash

Tick/untick it copy paths from Tree and List to the clipboard with/without a trailing slash.

Resolve junctions

Tick it to automatically resolve junctions when entered through the Address Bar or similar ports (Catalog, Favorites, Go to ...).

E.g., it will silently convert "C:\Documents and Settings\<user>\AppData\" to "C:\Users\<user>\AppData\Roaming\" in Win7.

Open favorite files directly

Tick it to directly open files by selecting them from the Favorite Files menu. Custom File Associations are honored.

Tip 1: The toolbar button's tooltip reflects the current setting.

Tip 2: You can hold down CTRL while selecting a favorite file to reverse the current setting.

Allow zombies in the Mini Tree

Tick it to keep currently unavailable drives and folders (on currently unavailable drives) in the Mini Tree. Factory default is OFF, i.e. on loading a Mini Tree any unavailable drives and folders are silently removed from the tree.

Paste to selected list folder

When ticked and a folder is selected in the list then pasting from clipboard (Ctrl+V) goes into this folder. Also works in Branch View and Search Results. Of course, the list needs to have the focus.

Also works for shortcuts (LNK) to folders.

Show message when list is empty

Tick it to get a message ("This view is empty.") displayed right in the list when it is empty. There is some additional information if items are present in the folder but currently not shown.

Show version information in the Status Bar

Tick it to show the embedded version information of executables, DLLs, DRVs, etc. (if they have any) right in the Status Bar on selecting such a file.

Sunday is the first day of the week

If unticked (factory default and ISO standard) then Monday is the first day of the week.

This setting affects e.g. find files by week and color filters by week.

Play a sound on certain events

Tick it to get what it says. By default it plays a sound after Copy/Move operations, when an item is deleted, and when a file search has finished. These are the Default Event Sounds, embedded in the executable. You can override them by Custom Event Sounds of your free choice using the scripting command ces().

Enable surround selection

Tick it to enable surround selection in all edit boxes.

Surround selection works like this: Type the first or second character of the following character pairs and the selected part of the text will be surrounded by the pair: () [] {} <> "" '' // || \\ ** ## %%

Surround selection works in all edit boxes, including inline rename boxes, dropdowns, and filters.

The character pairs can be tweaked in XYplorer.ini at this key: `SurroundSelectionPairs=()[]{}<>""''//||\\**##%%`

Startup & Exit

Permanent startup path

Here you can set a path that will always be the startup path, even if you closed the app on another path and saved this other path to the INI file. If "Permanent startup path" is empty then everything is as it has always been: XYplorer starts where it was closed last time (if "Save settings on exit" is enabled).

Tip: Environment variables are supported. E.g. `%userprofile%\Desktop`. Also relative paths (to XYplorer.exe) and XYplorer native variables are supported. e.g.: `<xydata>\..\..`.

Note: If XYplorer was closed in Find mode, then the "Permanent startup path" will auto-set the mode to browse on next startup.

Dual startup path

You can state a dual startup path, i.e. a startup path for each pane individually. Both paths should be separated by a double pipe (||).

General Syntax:

`Startup Path Pane 1||Startup Path Pane 2`
Both parts are optional.

Examples:

```
C:\||D:\          Pane 1 -> C:\; Pane 2 -> D:\
||D:\             Pane 1 -> (keep last path); Pane 2 -> D:\
C:\||             Pane 1 -> C:\; Pane 2 -> (keep last path)
```

Remarks:

- Also the command line supports this syntax.
- Remember that a conventional single permanent startup path will always overwrite the last path of the *active* pane (which can be pane 1 or pane 2).

Expand in tree

If checked then the Permanent Startup Path (if any) will be guaranteed to be expanded on next startup. If unchecked it will only be expanded if on last exit it was the last path and expanded.

This setting only affects Maxi Tree, not Mini Tree.

Startup pane

Always start on a certain pane, or on the last pane (factory default).

Startup window state

Always start with a certain window state, or with the last window state (factory default).

Open command line start path in new tab

If checked then a start path given by command line (e.g. "xyplorer.exe e:\") will not overwrite the current tab (when XYplorer was unloaded last time) but open in a new foreground tab. If the start path happens to be identical to the current tab's location then, of course, no new tab is opened.

Allow multiple instances

If unchecked, then XYplorer will always re-use any previous instance (and restore it if minimized), even if no path is passed as command line parameter.

Note that if you use the /ini switch in the command line then you will ALWAYS get a new instance even if "Allow multiple instances" is unchecked.

Open new instance always

Uncheck to re-use an already running previous instance if the new instance is called with a start path in the command line.

Minimize to tray

When you minimize XYplorer's main window it will be removed from the task bar and its icon will appear in the system tray.

The tray icon's tooltip is identical to the main window's title bar (and hence also configurable).

Minimize to tray on X close

Check it to minimize the application to the system tray when using the X close windows button.

Show splash screen while loading

Tick it to show a splash screen while the app is loading.

Check for updates at startup

Check it to perform a quick online check whether your current version is still up-to-date. If there is a newer version available you get a message. The check is only performed if you are online already.

Note that if Admin Settings are set to eAPHide_Help_XYOnTheWeb (64), this option is hidden and the functionality disabled.

Note that the check can take a couple of seconds.

Include beta versions

Tick it to include beta versions when checking if there are any updates (aka beta channel). This affects the check at startup (Check for updates at startup, see above) as well the manual check via menu Help | Online Support | Check for Updates.

Check for language updates at startup

Tick it to automatically check for an update if your language file is out of date. If an update is available, you will be prompted to update to it.

No network browsing at startup

Block browsing network locations at startup. This will prevent delays when starting from an unavailable network location.

Reconnect mapped network drives at startup

If ticked then at startup all non-connected mapped network drives are reconnected.

Note that this can lead to notable delays if the network is unavailable.

Adjust to OS light/dark mode at startup

Tick it to automatically adjust to Windows light/dark mode at startup (Win 10 and later).

Save Settings

Save settings on exit

Automatically save all the settings when closing the application. This includes saving the current configuration to the current INI file, and all data settings to the various *.dat files located in the application data path.

Include most-recently-used lists on save

Check to save the most recently used data of History, Tabs, Address Bar and Go To, Visual Filter, Selection Filter, Rename Special, Opened files in Menu Files | Open, Find Files Patterns and Locations, Focused Item and Scroll Position of Tabs, and Last Tags. Pick your choices under the Apply To... button. Note that, of course, those lists are only remembered when you actually save the settings, either manually or automatically on exit (tick Save settings on exit).

Remarks on some of the choices:

- History: Uncheck to clear history on exit.
  Also the Mini Tree is not saved. On next startup, the Mini Tree defaults to the Favorite MiniTree (if there is defined any).
- Tabs: Uncheck to clear all tabs on exit, so that only the current tab will be there on the next restart. With this setting enabled also the last paths of each pane are forgotten and on next start will be initialized to Computer. If you want to forget the tabs but keep the last paths of each pane use this tweak: KeepLastPaths=1

- Tabsets: This concerns only the MRU lists of tabsets, not the tabsets data themselves, which will stay on disk, of course, even when the MRU lists are forgotten.
- Last Tags: Controls whether the last applied Label, Find by Label, and Tags are remembered across sessions.

Backup settings on save

If ticked then -- when the settings are saved -- the following files are backed up to the folder <xydata>\AutoBackup before it is modified if the last auto-backup is older than 24h:

- Cached Servers (server.dat)
- Catalog (catalog.dat)
- Configuration (typically XYplorer.ini)
- Custom Keyboard Shortcuts (ks.dat)
- Folder View Settings (fvs.dat)
- Tags (tag.dat)
- User-Defined Commands (udc.dat)

A smart little safety precaution to protect you from mishap. These are probably the data files you would be most sad about losing.

Save changes to disk immediately

Check this box to immediately and automatically write the settings to disk the moment they are changed. The "Apply to..." button allows you to specify which changes will be saved to disk. These options are available:

- Catalog (catalog.dat)
- Configuration (typically XYplorer.ini)
- Favorites (typically XYplorer.ini), see Remarks below!
- Folder View Settings (fvs.dat)
- Keyboard Shortcuts (ks.dat)
- Tabs (XYplorer.ini)
- Tags (tag.dat)
- User-Defined Commands (udc.dat)

Remarks:

- The assumption here is that changes in these categories are usually meant to be permanent across sessions, and/or are something you would not like to lose by hazard.
- Note, however, that the feature has advantages and disadvantages:
  Advantage: Your changes are fixed to disk and protected from loss by crash.
  Disadvantage: If you make a mistake your previous good settings are overwritten and nothing can bring them back (unless you still find a good backup).
- The "Favorites" option includes Favorite Folders, Favorite Files, Highlighted Folders, Boxed Branches.

And since there is no separate disk file for Favorites but they are kept in XYplorer.ini, XYplorer.ini will be written on changing a favorite.

- Unlike manual saving, automatic saving is silent: you won't see a message in the status bar when saving is complete.

Special remark on Tags:

If Tags is ticked then the tags database is auto-saved also on these conditions (additionally to tagging proper):

- On item copy IF "Configuration | Information | Tags | Copy tags on copy operations" is ticked.
- On item backup/sync IF "Configuration | Information | Tags | Copy tags on backup and sync operations" is ticked.
- On item move.

Colors

Setting Colors

Here you can set foreground and background colors for various interface elements. Click the labels to set colors via the Windows color selection dialog, or simply paste hex values into the text fields. Right-click the labels to reset to default colors.

The Selected Rows Color is also used for the background of selected thumbnails. This way selected thumbnails can be recognized even when "Show Caption" is off.

Tabs | Apply colors: Tick it to actually apply the colors defined above the checkbox.

Tabs | Match selected tab with breadcrumb bar: Tick it to color the selected tab of each pane like the breadcrumb bar of each pane. It will overwrite the colors in "Configuration | Colors and Styles | Colors | Selected Tab Text / Background". This option is internally set to True if "XYplorer Style" is selected at Configuration | Tabs and Panes | Tabs | Visual style.

Tabs | Preserve custom colors: Tick it to show any custom tab colors even when the tab is selected and "Match selected tab with breadcrumb bar" is enabled.

Breadcrumb Bar | Match breadcrumb bar with custom colored tab: Tick it to match the colors of the breadcrumb bar to the colors of the selected tab if it is a custom-colored tab, i.e. if its background and text colors have been set individually via View | Tab | Background Color... and View | Tab | Text Color... (both also available in each tab's right-click menu). Extending the color of the tab to the breadcrumb bar is a bold visual statement and a welcome clue in poor lighting conditions.

Reset Colors

The button pops up a little menu that offers two options:

- Reset Colors to Currently Active Values
- Reset Colors to Factory Defaults

Highlights & Dark Mode

Grid style

Here you can choose between grid styles:

- Grid Lines: Thin 1-pixel lines between the rows.
- Zebra Stripes: Alternate Rows (1): Every 2nd row is highlighted.
- Zebra Stripes: Alternate Rows (2): Every 2nd 2 rows are highlighted.
- Zebra Stripes: Alternate Rows (3): Every 2nd 3 rows are highlighted.
- Zebra Stripes: Alternate Rows (4): Every 2nd 4 rows are highlighted.
- Zebra Stripes: Alternate Rows (5): Every 2nd 5 rows are highlighted.
- Alternate Groups: Every 2nd group is highlighted. Groups of items are striped depending on the contents of the sorted column. This gives you immediate visual clues about groups of same file types, sizes, dates, etc.
- Alternate Groups (Smart Names): See below under Highlighted Groups (Smart Names).
- Highlighted Groups: Every group is highlighted. This style lets you spot groups (as opposed to single, non-group items) even easier than style Alternate Groups.
- Highlighted Groups (Smart Names): Extra-smartness for Names. For the Name column a smart algorithm is applied to bring some grouping into this column which otherwise would be without groups.

Notes:

- Grids require "Show Grid" enabled in the file list (menu Tools | Customize List). There is a Toolbar button for this as well.
- Grids are shown only in views "Details" and "Details with Thumbnails".
- File Dates (Modified, Created, Accessed, Deleted) are grouped by day (the time part is ignored).
- If the list is not sorted by any column no grid is shown on Alternate Groups.
- The grid color can be customized using the "Grid" item under "List" in Configuration | Colors.
- Grid Style is a global setting (not per Tab).

Borders

Here you can choose between various border styles for the main controls (Tree, List, Catalog).

Selections

Here you can choose how the selected items are drawn in Tree, List, Catalog, and the small lists all over the interface.

(1) Windows Theme Style: This choice is only meaningful in Vista and later because in earlier systems Windows Themes don't affect selections. Furthermore it is only effective when Themes

are enabled, of course.
If this option is selected and valid, then the color settings of Current Tree Folder and Selected List Items are ignored.

(2) XYplorer Style: Selections are drawn in a cool XYplorer style.

(3) XYplorer Style (Rounded): Selections are drawn in a cool XYplorer style, with rounded corners. Also controls the rounding of the hover highlight, the drive space bars, the custom copy progress bar and drive space bar.

Focus rectangle

The focus rectangle is drawn on the focused item of the focused control (folder tree, file list, other lists). Here you can choose how/if it is drawn: Solid, Dotted, or None.

Selections in focused controls (XYplorer Style only)

Here you can define the Selection Text and Selection Background color of selected items in focused controls. They will overwrite the colors of the XYplorer Classic selection style. If Windows Themes is the active selection style, they are ignored.

Tick Use to actually use the defined colors in the interface.

Selections in non-focused controls

Here you can define the Selection Text and Selection Background color of selected items in non-focused controls. They will overwrite the colors of the active selection style, be it Windows Themes or XYplorer Classic.

Tick Use to actually use the defined colors in the interface.

Tree Path Tracing

Here you can define the color used by Tree Path Tracing.

Match color with breadcrumb bar: Tick it to have the Tree Path Tracing in the same color as the breadcrumb bar of the current pane.

Mark intermediate nodes: Tick it to draw a fat square around each node of the current path.

Width of trace in pixels: Allowed range is 1 to 9 pixels.

Note: To turn on Tree Path Tracing tick menu Tools | Customize Tree | Tree Path Tracing.

Recent Location Pins

Here you can define the color of the Recent Location Pins.

Match color with tree path tracing: Tick it to match the pins color to the current Tree Path Tracing color.

Maximum number of pins: Here you can define the maximum number of pins to be shown in the tree.

Note: To turn on Recent Location Pins tick menu Tools | Customize Tree | Recent Location Pins.

Dark Mode

Here you can configure the Dark Mode. There is a small Dark Mode Preview to give you a first impression.

Level of darkness (0 is darkest): There are 51 levels of darkness, 0 - 50. The factory default is 25.

Text contrast: There are 31 levels of contrast, 0 - 30. The factory default is 15.

Color tint (0 is neutral): Select a color tint (1 - 360) for your dark mode, or set it to 0 to remove all color tint.

Adaptive colors: Tick it to automatically dim down all brighter colors (selections, filters, etc) to make them softer on the eye.

Styles

Styles

Apply list styles globally

When checked, the list styles are applied globally to all tabs in all panes. Otherwise they are only applied to the current tab of the active pane.

Note that list styles are not applied from or to tabs pointing to folders that are controlled by Folder View Settings (FVS), unless source and target tab share the same FVS.

Remember list settings per tab

Here you can control which list settings are remembered tabwise, and which are applied (inherited) cross-tab. The button "Apply to..." allows you to further refine which list properties you want to retain per tab.

For example, if you would like to have the same sort order in all tabs, but keep the view mode (Details, Thumbnails...) persistent in each tab, then you would check Remember list settings per tab, and then in the list under the "Apply to..." button you would check "View Mode" and uncheck "Sort Order".

Mirror tree box color in list

When checked, the list background color will adapt to the current tree boxed branch color.

Semi-transparent grid color

Tick it to display any grid semi-transparently, like half-transparent glass. This is probably preferable when you also ticked "Mirror tree box color in list" right above.

Underline selected rows

Allows you to further specify the "Highlight Selected Rows" list style: If enabled, all selected rows are

underlined with the color defined in "Selected Row", else all selected rows are fully backlighted with that color.

This is a global setting (not per Tab).

Sticky checkbox selection

Allows you to further specify the "Checkbox Selection" list style.

Untick it to have the behavior known from Explorer.

Tick it to enable the following checkbox selection behavior:

- Left-clicking an item does not select this item and unselect all others. Instead it moves the focus to the item, and displays it in the Info Panel, even if it's not selected.
- To select/unselect an item you have to click the checkbox. Or start a lasso select by dragging the mouse with the left button held down.
- Left-clicking the empty List space does not unselect anything.
- Moving the focus via arrow keys does not change any selections.
- Space key toggles the selection of the focused item (no need to hold CTRL).
- Right-clicking an item focuses and selects it (but leaves any other selections unchanged), then pops the context menu for all selected items.
- No slow double-click rename.

Translucent selection box

Available only for Windows 2000 Professional or later. If unchecked the old-school dotted focus rectangle is used instead of the translucent selection box.

Line Spacing

Here you can customize the general line spacing in Tree, Catalog, and all lists and drop-downs.

Note that Tree, Catalog, and List support individual line spacing customization on top of this general one, adjustable by Ctrl+Wheel over the respective control.

Overall Spacing

Here you can modify the general spacing (or airiness, or density) of the main window. The value stands for pixels that are inserted in different places.

In addition to a more relaxed airiness, a higher setting also gives you larger mouse targets.

Show Age maximum hours (0 = unlimited)

Here you can limit the Age display in the List. For example, enter 24 to show the age for all ages <= 24 hours, for larger ages it would show the absolute date (as if "Show Age" was turned off).

You can also enter any of the following values to achieve a logic that comes even closer to the way humans process age:

Value	Meaning
s	this second

n	this minute
h	this hour
d	this day (today)
w	this week (Monday to Sunday, or Sunday to Saturday) *
m	this month
q	this quarter of year
y	this year

* Depending on Configuration | Controls & More | Miscellaneous | Sunday is the first day of the week.

You can also specify factors with the date units, e.g.:

Value	Meaning
1s	1 second
2n	2 minutes
3	3 hours (hour is the default unit)
3h	3 hours
4d	4 days
5w	5 weeks
6m	6 months
7q	7 quarters of a year
8y	8 years

FYI, here is where Show Age is toggled: Tools | Customize List | Date Column Format | Show Age. You find it as well in the right-click menu of the Date columns (Modified, Created, Accessed) in the file list.

Clipboard Markers

Here you can define if and how items are marked that are currently in the clipboard.

- Cut and copied items are marked in folder tree and file list.
- The markers are always in sync with the clipboard (even if other processes modify its contents).

Dimmed icons

Tick it to show the traditional dimmed icon effect on items that are *cut* to clipboard (no effect on *copied* items).

Colored lines

Tick it to show the new colored lines effect (orange for cut, blue for copied).

Columns

Lighter text in details columns

Tick it to lighten the text of all columns in Details view except the Name column.

Vertical grid lines in details view

Tick it to have vertical grid lines in the Details view in all tabs on each pane.

- The color is controlled by the value in Configuration | Colors | List | Grid.
- The setting is independent of the horizontal grid (Tools | Customize List | Show Grid).
- The vertical grid is painted on top of any other stuff in the list.

Truncate filenames in the middle

Tick it to put the ellipsis in the middle of overflowing filenames, instead of to the end. It's a good option because the end of the name often has interesting info bits like dates or serial numbers.

This setting also affects the printing of multiline filenames in thumbnails views, and the printing of filenames in the Hover Box status area.

Autofit the width of the Name column

Tick it to let the Name column automatically take up all of the space not used by the other columns.

Minimum / Maximum Name column width: Here you can specify the minimum and the maximum width (in pixels) for the Name column when "Autofit the width of the Name column" is enabled. For the maximum it's 0 (zero), which means: there is no maximum.

Always autosize the Size column

The Size column is the one with the most variable contents, the one that's typically always either too wide or too narrow. This option will autosize the Size column when changing locations or switching tabs or when a new column layout is loaded.

On autosize disregard the column headers

Tick it to mind only the column contents, not the headers (= column captions), when autosizing. Can save a lot of horizontal space if you have long-named columns with short contents.

Note that a certain minimum width is guaranteed so that you can at least guess what the heading is saying, especially if the sort icon is also present in the column.

Autosize columns maximum width

Here you can set the maximum width for auto-sized columns in pixels. Valid inputs are "0" ("unlimited") up to "9999".

Note that the default setting "0" ("unlimited") is actually limited to 1000 pixels internally.

Autosize Name column minimum width

Here you can define a minimum width for the auto-sized Name column. It won't get smaller than this. Set it to "0" to not enforce a minimum width.

This setting affects both Details views and both Column views (List and Small Icons). Not affected are Tiles views and Thumbs views.

Autosize Name column right margin

Sometimes on Autosize Columns the name column springs to a width far beyond the window edge where you cannot see it anyway, and where you are forced to horizontally scroll the list to enable you to manually grab the column separator to make it smaller again. What a drag!

Now here you can define a minimal margin from the right window edge. The name column will not go beyond this margin on Autosize Columns.

Use empty cell defaults

Tick it to use certain default values for certain empty cells in the file list (applies to Details views only). The actual values can be freely configured under the button Configure...: To edit a value either click into the cell, or select an item and press F2. Serial rename by Up/Down is supported.

Cell	Factory Default	Description
Ext (Folder)	<DIR>	Ext column for folders
Ext (File)		Ext column for files
Size	--	Size column
Label		Label column
Other	--	All other columns

Color Filters

See also the main article on Color Filters.

Enable color filters

Enable color filters in Tree and List. This is the master control for all settings below.

Apply color filters to the List

Check this box to color code items in the List. If you're not in the mood for color you can easily switch back to black & white, without deleting all your color settings.

Apply color filters to the Tree

Color filters based on name or attributes can also be applied to the Tree.

Ignore diacritics

Tick it to match Koln with Köln, and vice versa. Applies to name: and dir: filters in Tree and List.

Apply text colors to the Name column only

Tick it to apply the text color of a Color Filter only to the Name column. Affects only text-color-only filters since filters with back colors are only applied to the Name column anyway.

Draw background colors as rounded rectangles

Tick it to round the corners of the background color rectangle (if a background color is defined).

You can exclude individual filters from the roundness by passing the "-r" switch: `len:>=260|-r // overlong items (sharp rect)`

This setting also controls how highlighted folders are drawn in the tree (Favorites | Toggle Highlighted Folder).

Draw background colors in distinctive shapes

Tick it to give a distinctive shape to each type of filter (name, size, date...), given it has a background color defined.

You can exclude individual filters from adding distinctive shapes by passing the "-s" switch: `len:>=260|-s //overlong items (no shapes)`

Draw background colors as wide as the column

Tick it to draw the background colors of a Color Filter as wide as the Name column (or as the thumbnail space in Thumbnail views). Makes Color Filters more visible, especially with selected items.

The Color Filters List:

Setting Color Filters

In the color filters list you may add as many filters as you like, and define a Text Color and/or a Back Color for each filter. For details on color filter syntax see the main article on Color Filters.

The checkboxes in the list allow you to in/exclude the filters individually.

The filters are processed from top to bottom: the first match makes it. However, in Details mode, the first matching filter *with* backcolor (if any) is displayed combined with the first matching filter *without* backcolor (if any).

If a Back Color is defined, it is only drawn behind the item name (not any other columns). The color block is drawn with a distinctive shape at the right end to immediately hint at the applied Filter Type (name, attributes, size, date, age...).

Note: When you completely leave out the color definitions (Text/Back), black on pink will be shown as default color combination. This is also true for SC colorfilter, e.g.: `colorfilter("+lent:>12");`

Handy keyboard shortcuts

Rename a pattern: F2

Recolor a pattern: Double-click the list

Delete a pattern: DEL

Tip: To reset text color or back color to the default value, simply hold CTRL when you click the Text Color or Back Color button.

Tip: Hold CTRL when clicking New to duplicate current item.

Editing Color Filters via List Management

Alternatively to the interface in the Configuration window you can edit your Color Filters menu Tools | List Management | Color Filters...

For your interest, the following filters are provided as factory defaults. They are listed here as copied from menu Tools | List Management | Color Filters... (Editor Mode) (the leading + means the filter is activated; the hex numbers are the color codes):

```
+len:>260 //overlong filenames>FFFFFF,FB4F04
+attr:junction>D500D5,
+attr:system>FF0000,FFFF80
+attr:encrypted>008000,
+attr:compressed>0000FF,
+ageM: d //modified today>FFFFFF,74C622
attr:d>5E738C,
+size:0>4199E0,E0E2ED
+*.exe;*.bat>D24257,
+*.htm;*.html;*.php>4287D2,
+*.txt;*.ini>38A050,
+*.png;*.jpg;*.gif;*.bmp>933968,
+*.zip;*.rar>CC6600,
+*.dll;*.ocx>7800F0,
+*.mp3;*.wav>FF8000,
```

Fonts

Main Contents: Configure the font (name, size, weight etc.) for Address Bar, Tabs, Tree, Catalog, and List.

Tip: If you have a wheel you can easily zoom the font by Ctrl+MouseWheel. The "Apply to..." button opens a checkbox list where you can tick the controls that shall adjust to the font size.

Buttons and Labels: Choose the font (name and size) used for buttons and labels in the main window, i.e. all elements apart from Address Bar, Tabs, Tree, Catalog, and List. Use font size with care since larger sizes will surely mess up the layout.

Tip 1: You can control the size of this font using Ctrl+Wheel over the Status Bar. This is not only very practical but also gives you some intermediate fractional sizes that the standard font dialog does not offer.

Tip 2: The font size here should not be set larger than 10.5 (or rounded 10). Else the layout breaks and some letters get hard to read.

Regular Expressions: Choose the font (name and size) to be used when entering a Regular Expression. A fixed space font like Courier makes things more easy here.The size is limited to a maximum of 12 points, else the application's layout would break. Defaults to "Courier New, 8.25".

Tip: This special setting is implemented for the Name field of Find Files, RegExp Rename (under

Rename Special), Visual Filter dialog, Selection Filter dialog, Go To dialog (for RegExp Quick Searches and Visual Filters), Catalog Item's Destination (same). In most of these fields Regular Expressions mode is turned on by prefixing the entered term with ">".

Edit Text: Choose the font (name and size) used in text boxes (e.g. in List Management's Editor Mode, or in Quick File View, or in the Preview tab for previewed text files). Defaults to "Courier New, 8.25". Note: This font can also be used in Raw View.

Note: Tooltips in the main window follow the font settings of their parent controls, i.e. you can have them as big or as small as the rest of the text.

Enable zoom by Ctrl+mouse wheel

Tick it to enable zooming the fonts or metrics of various interface elements by Ctrl+mouse wheel.

To reset the zoom to the startup values of the current session you can use the command Reset Zoom (Miscellaneous | Layout | Reset Zoom) with default keyboard shortcut Ctrl+0.

See Mouse Tricks.

Notes

- You can independently customize the font size of the Status Bar by Ctrl+Wheel over the Status Bar.
- Ctrl+Wheel anywhere over the Info Panel will modify the "Buttons and Labels" font size.

Templates

Filename affixes

Here you can configure various templates that may be used by XYplorer in "Copy Here with Increment" etc. operations, and/or when new files are created and need to be named, and/or generally when name collisions are to be avoided.

Incremental affix

Used e.g. by "Copy Here with Increment". Set the start value in the template and use "a" as the placeholder for letters and "0" for numbers. The factory default is "-01". Use e.g. "_000" to start with "_000", or "-a" to start with "-a". The suffix is always appended to the file base (i.e. before the extension).

Examples:

```
                  Templates
                  -01   -ab
               ---------------
      Numbers  1: -01   -ab
```

```
  2: -02   -ac
 25: -25   -az
 26: -26   -ba
100: -100  -dw
675: -675  -zz
1000: -1000 -bmm
```

You see, the hexavigesimal system saves space and is fun. :)

If you mix numbers and letters, the increment template will be interpreted as numerical, and the letters will be used unchanged. For example, a template `-copy01` will render increments *-copy01, *-copy02, *-copy03, etc.

Using the asterisk: While the default is to suffix the increment you can also define increments that are prefixed to the filename, or that have fixed parts (additionally to the increment part) that are prefixed or suffixed to the filename. The syntax is identical to that of the Date Affix (see below): The asterisk (*) stands for the base (= name without extension) of the original filename. This way you can e.g. mimic the Windows standard "Copy (#) of..." in Custom Copy. Here are some examples:

```
Copy (1) of *     (similar to Windows standard increment)
Copy of *-01
01-*
01-* (Copy)
aa-*
```

Other formats: Also these templates are possible and will yield names similar to Windows's own way to handle collisions:

```
Templates             Examples
-----------------------------------------------
Copy of *             Copy of File.png
                      Copy of File (1).png
                      Copy of File (2).png
 - Copy
                      File - Copy.png
                      File - Copy (1).png
                      File - Copy (2).png
```

Notes:

- If Custom Copy is turned off and you paste (Ctrl+V) a file into its own location, or you use File | Duplicate | Copy Here, then Windows controls the format of the increment and the incremental affix template is ignored. If on the other hand you use Custom Copy or File | Duplicate | Copy Here with Increment (Ctrl+D), then the incremental affix template is applied.
- If no asterisk (*) is present the increment is suffixed to the base of the original filename.
- You CANNOT combine the asterisk with fixed parts AND non-numeric increments because such a pattern is ambiguous and cannot be parsed without making too many assumptions:

```
Copy of *-aa          NO GOOD!
```

- The fixed parts may contain numbers, but be aware that always the right-most numbers in the pattern will be incremented:

```
Copy 2012 of *-01     Will work as expected
Copy (1) of * (2012)  Will NOT work as expected!
```

Date affix

Used e.g. by "Copy Here with Current/Last Modified Date".

The factory default is "*-<date yyyymmdd>". The date is always affixed (prefixed, circumfixed, suffixed) to the file base. The file base is represented by an asterisk in the pattern. If the asterisk is missing the pattern is by default suffixed to the file base.

Examples for copying a file called "Test.txt" using various affix templates:

```
*-<date yyyymmdd>          = Test-20100803.txt
<date yyyymmdd> *          = 20100803 Test.txt
Copy of *                  = Copy of Test.txt
Copy of * from <date yyyymmdd> = Copy of Test from 20100803.txt
```

The template also supports the variable <folder> that is resolved to the moved/copied file's parent folder (only the folder, not the whole path).

Example for a template:

```
*-<date yyyymmdd> (<folder>)
```

Note that the presence of this variable in the template is not reflected in the related commands' menu captions, so this variable is rather a very special option that you have to make a mental note about when you use it.

Dropped messages

When drag-dropping messages from Outlook or Outlook Express or Thunderbird, this template controls the makeup of the new filenames. The following fields are available for the template which you can define in Configuration | Templates:

- <from> e-mail address or, if given, name of the sender

- <to> e-mail address or, if given, name of the receiver

- <subject> subject line of the message

- <date> local (at yours) date/time when message was sent

A template may, for example, look like this:

```
<from>_<to>_<subject>_<date yyyy-mm-dd_hh-nn-ss>
```

You can freely change the order of the fields and pack any text strings before, after or between them. Do not add the extension to the template -- it is done automatically, and mind the space after "<date". If you just put <date> without specifiying the format, the format will default to yyyy-mm-dd_hh-nn-ss.

Note that it's your responsibility to ensure that your templates contain only characters that are legal for filenames. It is not validated inside the configuration window.

Show Edit Prompt: You can trigger an edit prompt for each field you desire. Simply insert a "?" at the end of the field and you'll be prompted. If you cancel the dialog the whole job is aborted. Examples:

```
<from>_<to>_<subject?>_<date>
<date yyyy-mm-dd_hh-nn-ss?>_<subject?>_<from?>_<to?>
```

Character to replace invalid characters in dropped messages

Here you can state a character that will be used to replace characters that are invalid in filenames (e.g. :?\/*).

For example if you set it to ~ then the email subject

Re: Verifying whether a CD/DVD is a true copy?

will be renamed to

Re~ Verifying whether a CD~DVD is a true copy~

to function within a filename.

If you leave it empty, invalid characters are replaced by "", i.e. they are simply removed.

Maximum length of generated filenames (0 = unlimited)

Here you can set a length limit in number of characters. The <subject> field is cut down first, then if still needed the end of the filename is cut off.

Auto-increment filenames on collision

Tick it to auto-increment filenames on collision when dropped. Else you get an overwrite prompt.

This setting is also honored when you copy-n-paste messages, not just when you drag-n-drop them.

Title Bar

Here you can define the layout of the main window's title bar. The factory default is:

```
<path> - <app> <ver>
```

You may add or remove what you like, in any order. There's only one rule: The <app> variable (resolved to the EXE base name, normally XYplorer; if XYplorer is not part of that name it is added to the title bar anyway) is mandatory.

Variables supported in the Title Bar Template:

- <path> current path
- <app> this EXE base name
- <ini> current INI file (without path)

- <ver> this EXE version
- <folder> current folder (without path)
- <title> the title of the current session (the title can be set by command line switch /title).
- <tabset> name of the tabset of the active pane
- <tabset1> name of the tabset of pane 1
- <tabset2> name of the tabset of pane 2
- And most other XYplorer native variables, e.g. <date>, <xydrive>, <get pane>, <get userrole>, <curfolder> etc.
- Environment variables.

Status Bar

Here you can define a template for the 3rd section of the status bar, aka Custom Status Bar Info (CSBI). The factory default is:

```
<s:dimension> <s:duration>
```

These are meta variables that only work in this context. They are internally resolved to the following XYplorer native variables (you could just as well use those native variables, the meta variables are just easier to handle):

```
<s:dimension>  -->  <prop #image.dimensions> (<prop #aspectratio>)
<s:duration>   -->  |s|<get LengthsSelected a 1>
```

The first part serves to show dimensions and aspect ratio for images and videos, the 2nd part will show the summary duration of all currently selected media files. The `|s|` tells XYplorer, to update the value on each selection change (only useful if the template pulls data from more than the currently focused file). Only the part right of `|s|` will be used and updated on each selection change.

If the template returns nothing useful (because e.g. an image has no duration) then the defaults are used for the status bar.

You can use all XYplorer native variables as or within a template. Here are some examples (see more example at property() and get()):

```
<prop #audio.bitrate>
<prop #audio.samplerate>
<prop #tag.composer>
<prop #mp3.year>
<prop #Image.Fuji.FilmMode>
<prop #image.exposurebias>
<prop #image.cameramodel>
<prop #date.created>
<prop #version>
<prop #hash.md5>
<prop #hash.sha256>
```

Also the drives listing ("This PC") supports CSBI but it needs the presence of the `|d|` switch: Everything right of `|d|` is used for drives, everything left of it for other items. If `|d|` is missing then the template is not used for drives at all. Examples:

```
|s|<get BytesSelected '' 3>|d|<get FileSystem>
|s|<get BytesSelected '' 3>|d|* - <get FileSystem>
|d|* (<get FileSystem>) |s|<get BytesSelected '' 3>
```

You can prefix various switches to the template, separated by ||, to modify the appearance of the info. The order of the switches is irrelevant. Currently there are two:

i = Use file icon (instead of the generic "i" icon).

c = Use marked color (Configuration | Colors and Styles | Colors | Marked Text 1).

Examples:

```
i||<s:dimension> <s:duration>
c||<s:dimension> <s:duration>
ic||<s:dimension> <s:duration>
```

Use status bar template: Tick it to actually use the template defined in the box above. Tip: The same toggle is available in the right-click menu of the status bar.

Command Line Interpreter

Here you can define a custom command line interpreter (CLI) and the arguments it should use. This CLI is used when you use the DOS command syntax in location ports like the Address Bar, e.g. `!dir`. It is also used by the command Open Command Prompt Here (Tree folder context menu).

Use custom command line interpreter: Check it to actually use the custom CLI. If unchecked the factory default CLI cmd.exe (usually in C:\WINDOWS\system32\) is used.

Executable: State a full or portable path to executable file.

Arguments: Define any arguments for the custom CLI.

- If your definition contains the placeholder <command> (case-sensitive!), it is replaced with the input from the Address Bar, if not, the input is appended to the end of the arguments definition.
- If your definition contains the placeholder <path> (case-sensitive!), it is replaced with the current path.

Tags

Show tags in file list

Uncheck it to stop showing tags (Label, Tags, Comment, and Extra) in the file list. Useful to speed up browsing when you have tagged a large number of items but do not need the tags at the moment. Note that reading and writing the tag database (tag.dat) is not affected by this setting.

Toggle tags by column click

Check to allow quick one-click tagging of items by left-clicking directly into the column. Three checkboxes let you control this setting individually for the Label, Tags, and Comment column.

- Label: Click in the column toggles Last Label / No Label.
- Tags: Click in the column toggles Last Tags / No Tags.
- Comment: Click in the column toggles Last Comment / No Comment.

Also on Full Row Select. Tick it to make Tag columns clickable also in Full Row Select mode.

Popup by tag columns right-click

If checked then right-clicking into the Label, Tags, Comment, or Extra column will popup a little context menu with some useful options relating to the respective tags of the clicked item.

Sticky Selection Deluxe. There's a command Select All "[Label Name]" in the context menu of toolbar button Labels and in the Label column right-click menu. This gives you a comfortable way to select a group of files based on their Labels and the Label of the currently focused item (or the clicked item in case of the column context menu).

The context menu command Select All "[Label Name]" can be modified by modifier keys:

- Hold None: Select all [Label Name] items, unselect all other items.
- Hold CTRL: Select all [Label Name] items, keep other selections.
- Hold SHIFT: Unselect all [Label Name] items, keep other selections.

Apply tagging to all selected items

Check to apply the commands in the right-click popup menu to all selected items instead of to the right-clicked item. If nothing is selected this setting is ignored and the tagging applies to the right-clicked item.

On sorting keep tagged items on top

If ticked then the tagged files are kept on top and the empty items are sorted to the bottom. Pretty useful when most of your items are not tagged so that your tagged items would vanish below the viewport instead of showing up when sorting by tags. Note that "tag columns" include Label, Tags, Comment, and Extra.

Copy tags on copy operations

If enabled then tags are copied from the source to the target items of copy operations.

Note: You should use Custom Copy if copying tags is enabled. It is more reliable than Shell Copy at

copying tags only for those items that have actually been copied (and not skipped).

Copy tags on backup and sync operations

If enabled then tags are copied from the source to the target items of backup operations and of sync folder operations.

Confirm copying tags

If ticked then you are prompted before any tags are copied from the source items to the target items.

The prompt is popped not more than once per copy or backup operation.

Notes on Copying Tags

- The tag.dat database will grow for each new target item.
- If the target is on a removable drives be aware that the tags do not actually end up on that drive but are just added to the local tag.dat where they merely point to that drive.

Auto-refresh tags

Tick it to auto-refresh tags on changes in a shared tags database, a database shared by several members of a team over a network.

With this setting enabled your tags in memory and display are auto-updated whenever the shared tags database is saved to disk by any member of the team. In other words you (and all members using this setting) are synchronized with the member that saved the database. Any dirty tags in your own instance are kept alive and are not overwritten by the shared tags.

Label captions and colors

Here you can edit the labels and how labeled items are displayed in the file list. You can add up to 31 label schemes to the list.

Note that the menu icons for labels in the main window will adjust to the back colors you have chosen. Also note the following extra functions of the buttons (also in the buttons' tooltips):

- Edit: Hold CTRL to reset all to factory defaults. Choose any caption you like. Note that some special variables are supported here (see Soft labels below).

- Text Color: Hold CTRL to reset to default color.

- Back Color: Hold CTRL to reset to default color. Hold SHIFT to remove any back color.

Tip: Double-click an item to edit its Back Color.

Note: You cannot move the list positions of the label schemes. Also deletion works only on the last item of the list. This way the index order of the existing labels is always maintained.

Soft labels

The following label-specific variables allow you to soften a label and use it as a kind of link to other stored information about the same item. This means you can have a virtually infinite number of label

captions in one (or more) of your 31 possible label definitions.

These are the variables. You can use them in the list below Label captions and colors, also in combination with literal strings and with each other:

- `<tag>` = Resolved to the (alphabetically) first tag of an item.
- `<tags>` = Resolved to all tags (comma-separated list) of an item.
- `<cmt>` = Resolves to the comment of an item.
- `<ex1>` ... `<ex16>` = Resolves to the "Extra 1" ... "Extra 16" field of an item.

Of course, since this is mainly about label captions, it makes the most sense when Label style (see below) is set to "Label column (caption)".

Further notes:

- Visual Filters and Live Filters support soft labels. To filter by a soft label regardless of its resolved content, you can refer to it by its numeric index using this syntax: `lbl:"#1"`, `lbl:"#2"`, `lbl:"#3"`, and so on. The quotes and the # are important here.
- Find Files and Quick Search support soft labels.To search a soft label regardless of its resolved content, you can refer to it by its numeric index using this syntax: `lbl:"#1"`, `lbl:"#2"`, `lbl:"#3"`, and so on. The quotes and the # are important here.

Label style

Select if and how the labeled items shall be colored in the list.

Rounded: Tick it to paint the labels with rounded corners.

Storage

Select one of several options for storing the tagged files in the tags database (usually "tag.dat"):

- Absolute paths: Not portable. `[= Storage: 0]`
- Relative to application drive: Using the "?" syntax, useful for portable installations on sticks. `[= Storage: 1]`
- Relative to application: Useful for backups where XYplorer installations have the same position relative to the tagged files. `[= Storage: 2]`
- Relative to tags database: Useful for in-place tags database files that are copied along with the tagged files (a scripting command is planned for loading those databases). `[= Storage: 3]`
- Use volume serials: The 8-digit hex volume serial is stored in place of the drive letter. Allows to identify removable drives regardless of the drive letter assigned by the host system. `[= Storage: 4]`

Remarks:

- The storage mode is stored within the tags database so it is ensured to be read the way it was written. The value is contained in a additional line in the header of the tags database, e.g.:

```
Storage: 3        (3 = Relative to tags database)
```

- The storage setting only affects *writing* the tags database. Reading a tags database is controlled by the value in the "Storage:" line (see above).
- On reading the tags database relative paths are resolved to absolute paths and kept like that in memory. Tools | List Management | Tagged Items always shows the resolved tags as they are in memory.
- After reading a tags database, "Configuration | Tags | Storage" is automatically set to the storage value found in that database. That way it is ensured that it will be written in the same mode as it was read.
- However, you may read a tags database stored in one mode, and then change "Configuration | Tags | Storage" to another mode. On next save the tags database will be saved in that new mode. On next read it will be read in that mode.

Options

The Options button pops a menu with several advanced options:

Database Check...

If other processes move or rename files or folders, the tag database (the file tag.dat) may contain items that don't exist anymore, the so-called "orphaned tags". This does not affect the functionality of the database at all, but it might slightly decrease performance if the number of orphans is huge.

Click this button to check whether there are any orphaned tags (and how many), with the option to remove them from the database. Along with the orphan removal a number of other things are performed as well:

- Sorting is checked and corrected if necessary.
- Any double entries are removed.
- Any wrong capitalization is corrected.
- Any orphaned entries (entries pointing to non-existing items) are removed.
- After checking for orphans and no orphans are found there is an option for an additional check for dupes and false capitalizations.

You can choose (checkbox at the bottom) whether to check all drives or just fixed drives. Choose the latter if you want to clean your database but leave tags from removable drives untouched.

Note: This action cannot be undone by cancelling the Configuration dialog. But, of course, you can undo it by exiting XYplorer without saving (Ctrl+Alt+F4) so that the tag.dat on disk is not touched.

Edit Tagged Items...

Offers backdoor access to the tags database in its current state in memory. You can edit the records (of course you should know what you do, there is no error checking concerning the syntax), remove records, add records. Using Editor Mode you can outsource the work to a text editor of your choice. This feature also comes in handy at checking out orphans and manually reviving them by updating the

paths.

Dirty items (items with unsaved tags) are shown with a yellow background color (brown in dark mode).

FYI, here is the record syntax: `Full filename|LabelID|Tags|Extra1|Extra2|Extra3|Extra4|Extra5|Extra6|Extra7|Extra8|Extra9|Extra10|Extra11|Extra12|Extra13|Extra14|Extra15|Extra16|Comment`

Edit Orphans (All Drives)...

Pops a list of all current orphans (entries pointing to non-existing items) (if any) which can be manually edited. Note that in this interface, you cannot add or remove items.

Edit Orphans (Fixed Drives Only)...

Same as above, but only checking tagged items on fixed drives.

Remove All Tags...

Removes all Labels, Tags, Comments.

Note: This action cannot be undone by cancelling the Configuration dialog! However, in case you made a mistake, if you exit XYplorer without saving (Ctrl+Alt+F4) then tag.dat will not be changed.

Tip: The label "Currently X items are tagged" shows the path of the current tags database in the tooltip.

Custom Columns

Custom column definitions: A list of all available column definitions. Press "Edit..." or double-click an item in the list to configure a column. Of course, only when you OK the Configuration dialog, the changes take effect and are applied to the list.

Reset Columns...: Pops two self-explaining options, Reset Columns to Currently Active Values and Reset Columns to Factory Defaults.

Edit...: Configure the column selected in the list.

See here for further details on Custom Columns

File Info Tips & Hover Box

File Info Tips

Show File Info Tips: Tick it to show a balloon tip with extensive file information when you move the mouse over a filename's name or icon in the List. There are some configuration options:

When hovering over the icon: If checked File Info Tips will pop up when you move the mouse over a file's icon.

When hovering over the filename: If checked File Info Tips will pop up when you move the mouse over a filename.

Tip: To use both of the above mentioned hover zones, you can check or uncheck both boxes, no difference.

Note: When the File Info Tips and the Hover Box compete for the same place, the Hover Box always wins.

Only while the shift key is held down: Tick it to show File Info Tips only while the Shift key is held down.

In Tree as well: Tick it to show File Info Tips also for the items in the folder tree.

For executables as well: Tick it to show informative file info tips for EXE files. It's optional because under Win10 file info tips for EXE files got extremely slow (probably due to anti-virus). If unticked, you'll still see all the information that can be gathered without triggering an AV-induced delay.

Show audio info and tags: Tick it to show basic audio info and tags at the speed of light on mouse over, i.e. without any clicking. Supports ID3v1 and ID3v2 tags in MP3 files, and Vorbis Comments in FLAC and OGG files. A nice feature for audiophiles. Tip: You can hold CTRL to force the normal file info tip instead of the special audio tip when hovering a FLAC, MP3, or OGG file.

Use standard shell file info tips: Tick to show the system's standard file info tips. Leave unchecked to show XYplorer's native file info tips.

Show these fields: Tick the checkbox to customize the file info tips using the list (button Select Standard Fields...) of all available properties (the list varies with your Windows version and the installed shell extensions). Leave unchecked to show the tips as they are shown in Explorer.

Extra fields: If ticked then the following XYplorer-only extra fields (button Select Extra Fields...) are appended (if they are ticked as well) to the standard file info tips.

Len (Full Path): Shows the hovered filename's length (including the full path).

Path: Shows the item's full path (search results only).

Version Information: Shows the following fields (if they exist) from the so-called "String Version Info", for example for XYplorer.exe:

FileDescription: XYplorer
CompanyName: Cologne Code Company
FileVersion: 27.10.0000
OriginalFilename: XYplorer.exe
LegalCopyright: Copyright © 1997-2025 by Cologne Code Company
Comments: www.xyplorer.com

Target (Shortcuts only): Shows the target of shortcuts (LNK files).

Junction Target: Shows the target of NTFS junctions.

Label: Shows the item's label (if any, else the field is not shown).

Tags: Shows the item's tags (if any, else the field is not shown).

Comment: Shows the item's comment (if any, else the field is not shown).

Hard Links: Shows the number of Hard Links of a file. Every file has at least one. The field is shown only if the return is not "1".

Color Filters: Shows which color filter definition made the match for the hovered item (if any, else the field is not shown).

Opens With: Shows the opening application for the hovered item, the application that runs when you double-click the item.

Bitness: Shows the bitness (32-bit or 64-bit) for all file types where it makes sense (EXE, DLL, DRV, TLB, etc.).

Encoding: For files the possible returns are: Ascii, DBCS, UTF-8, UTF-8 BOM, UTF-16LE, UTF-16LE BOM, UTF-16BE, Binary, File is empty, Could not open file, File too large. For non-files it's Drive, Server, Share, Folder.

Hover Box

Show Hover Box: If ticked the normal File Info Tip is replaced by the so-called "Hover Box", a popup window showing a small file or folder preview plus some file or folder data, e.g. in case of images: a thumbnail and basic file, image, and (if applicable) photo data. The image thumbnail has a maximum bounding box depending on the screen size, other previews can be slightly larger depending on the file type. The idea is to get a small preview without any clicking, a Zero Click Preview. It's kind of a pre-preview to the Mouse Down Blow Up, which is just a mouse down away and provides you with a larger preview (including animation and sound).

Tip: There is also a toolbar button "Hover Box" to enable/disable the hover box.

When hovering over the icon: If checked the Hover Box will pop up when you move the mouse over a file's icon.
Tip: If you tick this and leave "Show info tips only when hovering file icon" unticked then you can have both Hover Box and File Info Tips in the list at the same time (without holding any keys), one over the icon, the other over the caption.

When hovering over the filename: If checked the Hover Box will pop up when you move the mouse over a filename.

Tip: To use both of the above mentioned hover zones, you can check or uncheck both boxes, no difference.

Note: When the File Info Tips and the Hover Box compete for the same place, the Hover Box always wins.

Only while the shift key is held down: Tick it to show the Hover Box while the Shift key is held down.

See here for further details on the Hover Box

File Info Tips and Hover Box

In network locations as well: Unless you check this option File Info Tips and Hover Box are not shown for items in network locations (mapped and UNC). This is for performance reasons: depending on the connection they can be somewhat slow because the file has to be physically accessed to retrieve this information.

For junctions as well: Untick it to avoid spontaneous auto-downloads from the cloud when online-only files are hovered with "Show File Info Tips" or "Show Hover Box" ticked. If unticked you will see no Hover Box or extended File Info Tips when hovering junctions or symbolic links.

Initial delay in milliseconds: Time in milliseconds between initial mouse over and popup. Valid values are from 0 thru 9999.

Visible time in milliseconds: Here you can customize the time a File Info Tips or Hover Box is visible before it auto-destroys. The factory default is 10,000 (= 10 seconds). The maximum value is 32767, so the tooltips cannot be shown longer than about 32 seconds. Note: The Hover Box is not affected by this value; it stays visible until actively closed by some action or process.

Hover Blacklist

Advanced Tip: You can specify a "Hover Blacklist" containing any number of target paths to be excluded from the File Info Tips and Hover Box, i.e. excluded from file access on mouse hover. Usually you want to exclude slow paths, e.g. cloud folders. The data should be provided in a file placed in <xydata> and named "HoverBlacklist.txt". The paths should be listed one per line in any order. All sorts of variables and portable syntax are supported. You can use wildcards in the standard way for whole paths (identified by the presence of / or : in the resolved pattern) or just item names. Some examples (no surprises here, all pretty standard):

```
A:\*                  //match all items on drive A: (and A: itself)
A:\slow               //match A:\slow
A:\slow*              //match all items in and under A:\slow\ (and A:\slow itself)
A:\slow\              //match all items in A:\slow\ (but not in folders below)
A:\slow\*             //match all items in and under A:\slow\ (but not A:\slow itself)
%HOMEDRIVE%           //match %HOMEDRIVE% and all items in the root folder on %HOMEDRIVE%
F:\Sl?th\             //match all items in paths called F:\Slath\, F:\Sloth\ ... (but not
in folders below)
*\backup\*            //match all items in folders called "backup"
*.webp                //match all WEBP items
D:\*.webp             //match all WEBP items on drive D:
k*                    //match all items starting with k (or K)
readme.txt            //match all items named "readme.txt"
```

Clipped Items Tips

Show tips for clipped tree and list items (aka "in-place tooltips"): Will display a tooltip with the full item name when hovering an item with a partly clipped name. In case of the file list, these tips will only appear when the other sorts of tips (file info tips, audio info tips) do not.

Report & Data

Info Panel / Report

Classic Directory Dump

- Table width: Set the table width measured in characters (64-256).
- Line feed on oversized filenames: Wrap long lines.

CSV Field Separator

Select which separator is used in the CSV reports in Info Panel | Reports | Current Folder.

- System list separator: The System List Separator can be changed in the system regional settings. Depending on the Windows version this can be done around here:
  Start > Control Panel > Regional and Language Options
- Tab: That's the TAB character.
- Other: The size of "Other" is limited to 16 characters. If "Other" is set to "" (nothing) then it defaults to the System List Separator.

Tree structure

Select which data is included in the tree structure reports in Info Panel | Reports | Current Folder.

- Include files: Tick this to include files with Info Panel | Report | Tree structure. The files are marked with a > in front.
- Include basic item data: Tick this to include basic item data in CSV format: `Raw Bytes;Modified Date [;Version]`.

Output File Options

When you want to send your report to file you find some options here regarding the default filename (the name preset in the Save As dialog).

Default name to "[Current folder].txt": Automatically generates a default filename from the currently browsed folder, quite useful to automate archiving find results.

Date/time as filename suffix: Save the current report to a file whose name will carry the exact time when the report was created, eg Report_2009-12-23_21-09-33.txt.

Append to existing file: Appends the current report to the file you select in the Save As dialog. If the file does not exist it will be created.

Encoding: Choose the encoding of the output file:

- ANSI (the actual encoding, e.g. Windows-1252, depends on the active codepage)
- UTF-8
- UTF-8 BOM
- UTF-16 LE
- UTF-16 LE BOM (factory default and previously hard-coded)

Photo Data

Photo Data include Camera Model, Focal Length, F-Stop, Exposure Time, Exposure Bias, ISO Speed, and Date Taken.

Show photo data in the Hover Box

Tick it to show photo data in the Hover Box status.

Tip: While the Hover Box is showing an image you now can toggle whether photo data are shown in the Hover Box status by pressing the "P" key.

Show photo data in the Large Tiles view

Tick it to show photo data in the Large Tiles view.

Tip: The same setting is also found as "Show Photo Data" in the CTRL+Right-click menu of the text area of each tile in the Large Tiles view.

File Operations

Background Processing
Backup Operations
Custom Copy Operations
Custom Copy Blacklist
Configure Custom Copy
External Copy Handlers
Miscellaneous

Background Processing

Enable background processing

Check to launch file operations in a separate process.

The advantage of background processing: You can continue working in XYplorer while heavy copy jobs are processed in the background.

Technically this is achieved by passing the file operations to the background copy handler XYcopy (XYcopy.exe) which is contained in the distribution package and must be located in the same folder as XYplorer.exe.

Apply To...

Click the button "Apply to..." to configure which types of file operations are backgrounded. The factory default is:

```
[x] Backup             (can be queued)
[x] Copy               (can be queued)
[x] Move cross-volume  (can be queued)
[ ] Move intra-volume  (runs at once in parallel process)
[ ] Delete             (runs at once in parallel process)
[x] Sync Folders       (can be queued)
```

Notes

- Background processing is by default applied to Backup, Copy and Move (cross-volume) operations. Delete operations are not backgrounded. Use Apply To (see above) to change these default settings.
- Note on Move operations: Only moves across volume boundaries (= to a different drive) involve copying of bytes and hence take time, whereas moves within volume boundaries just update the MFT (master file table), so they are very fast and do not really need to be backgrounded.
- When you terminate XYplorer while XYcopy is still working, the job in progress will be finished, but any jobs still waiting in queue will not be triggered. You are prompted whether you really want to lose the jobs.
- Any file operations triggered by Undo/Redo are not spawned to XYcopy.
- Any file operations triggered by drag and drop to XYplorer from another application are not spawned to XYcopy.
- Of course, any file operations triggered by drag and drop from XYplorer to another application are not spawned to XYcopy because those operations are handled by the drop target.
- Any file operations triggered by drag and drop using the Shell context menu are not spawned to XYcopy.
- Background processing is not supported under Windows Me and Windows 98.
- In scripting background processing may lead to undesired results when the script flow overtakes a backgrounded file operation before it is completed. To avoid this you can temporarily disable background processing by inserting this line to the beginning of such a script:
  `setting "BackgroundFileOps", 0;`
  Background processing will be automatically re-enabled when the script is finished.

Queue file operations

Check to process background file operations sequentially, in a queue, where successive operations are lined up in sequence and completed one after the other. If unchecked then background file operations are processed in parallel.

E.g. if you initiate a job, say a Copy operation, while another one is still in progress, the new one will

be delayed until the other one is finished. One job is triggered after the next, until all pending jobs are completed. A copy queue, also known as asynchronous copy, avoids inefficient pseudo-parallel processing and undue stress on your HD head.

Notes

- This option is applied on a per-job basis in the moment the jobs are initiated by the user, so while one or more jobs are running in the background you can change the setting of "Queue file operations" to control how the *next* job is handled; running or pending jobs are not affected.
- There's no explicit limit applied to the number of parallel processes.
- Move operations within volume boundaries (see above Note on Move operations) are always processed in parallel (= not queued) regardless of the setting of Queue file operations, because they are usually fast and you rather expect immediate action.
- There is also a toolbar button for this setting.

Status of background jobs

See Background Jobs Dialog.

Backup Operations

See Backup for some general remarks on the nature of XYplorer's Backup.

Configure (button): Click to open the Configuration Dialog (see below) for Backup operations.

Custom Copy Operations

Custom Copy is a widely configurable alternative copy method based on Kernel API. It exceeds shell copy in lots of ways as you will immediately notice when you open its configuration dialog. Note that Custom Copy optionally also includes all sorts of move operations, so the term "Custom Copy" comprises "Custom Move"; see For all move operations below.

Configure (button): Click to open the Configuration Dialog (see below) for Custom Copy operations.

Use Custom Copy: Check to use Custom Copy in place of the standard Windows shell copy. This includes operations like Copy To, Paste (copy) from clipboard and drag and drop. If Copy operations are configured as backgrounded, then Custom Copy operations are backgrounded as well.

For all copy operations: Use Custom Copy for all copy operations.

No progress dialog on duplications: Tick it to prevent the progress dialog when copying files in place (source and target path are the same).

Note: If ticked then the copy operation is forced to run in the foreground even if Backgrounding is enabled.

For all move operations: Use Custom Copy for all move operations.

For cross-volume moves only: Tick it to use Custom Move only for cross-volume moves (between different logical drives) and not for intra-volume moves (where no bytes are copied at all, and which

technically are a rename rather than a move).

No progress dialog on intra-volume moves: These moves are extremely fast since no bytes are copied (see also Remarks below), so a progress is not really necessary. Instead you get a summary feedback in the status bar (at least on foreground jobs; on background jobs another status bar message will overwrite it -- but usually you will exclude intra-volume moves from background processing anyway because they are so fast). Note that this setting overwrites the setting of "Show progress dialog" in the general Custom Copy configuration. (However, if "Show progress dialog" is off, then unticking "No progress dialog on intra-volume moves" will not make it show, of course.)

Note: If ticked then the move operation is forced to run in the foreground even if Backgrounding is enabled.

Remarks on Intra-Volume Moves vs Cross-Volume Moves

Custom Move comprises intra-volume and cross-volume moves. Since intra-volume moves are just changes in the volume's Master File Table (MFT) and do not involve any copying of bytes, they are very fast and notably will ignore some of the configuration options, namely:

- Verification (not applicable/necessary: no bytes are copied)
- Safe overwrite (not applicable/necessary: no bytes are copied)
- Remove read-only attribute (unlikely to be useful with moves)
- Preserve all item dates (not necessary: they are preserved anyway)

Note, however, that both intra-volume and cross-volume custom moves optionally provide automatic rename-on-collision.

On cross-volume moves, each file is first copied and then (after optional verification of the copy's correctness) the source file is deleted. When all files are moved, the source folders are deleted (unless they are not completely empty for whatever reason).

Check beforehand whether there is enough space: Tick it check beforehand whether there is enough space.

- This setting applies to Custom Copy/Move, Backup, and Sync operations.
- Intra-volume moves do not check space anyway, so here the setting is irrelevant.

Default to repeat action on collisions: Tick it to initialize the "Do this also for the next collisions" checkbox to be checked.

- This setting applies to Custom Copy/Move, Backup, and Sync operations.

Tip: You can configure Custom Copy directly from the right-click menu of the toolbar button "Queue File Operations".

Note: Custom Copy supports Undo/Redo.

Custom Copy Blacklist

Advanced Tip: To exclude certain target folders from Custom Copy (and automatically use normal Shell Copy instead) you can list those folders in a manually created text file placed in the Application Data Folder and named "CustomCopyBlacklist.dat". The paths should be listed one per line in any order. All sorts of variables and portable syntax are supported. Append an * (asterisk) to a path to include subfolders. Here's an example for the contents of the file on 64-bit Windows (on 32-bit Windows you would drop the line "%ProgramFiles(x86)%*"):

```
%HOMEDRIVE%
%ALLUSERSPROFILE%*
%ProgramFiles%*
%ProgramFiles(x86)%*
%ProgramW6432%*
%WinDir%*
```

The typical reason for blacklisting folders is that Custom Copy will fail where UAC disallows writing to them, whereas Shell Copy will prompt for elevation.

The Configuration Dialog for Backup and Custom Copy operations

On name collisions: Here you can configure what happens on a name collision. You have the choice between the following options:

- Ask: [factory default for Custom Copy] Pops an overwrite prompt on each name collision, allowing you to decide case by case. The overwrite prompt tells you a bit about the conflicting files (differences are underlined in the source) and offers you not to be asked again during the current job.
  In the overwrite prompt itself, you find
  a) an Overwrite button to quickly overwrite the existing file with the source file. Ticking the checkbox "Do this also for the next collisions" also applies to the Overwrite button making it effectively a Overwrite All button when ticked.
  b) a Skip button to skip the current file from copying/moving. Ticking the checkbox "Do this also for the next collisions" also applies to the Skip button making it effectively a Skip All button when ticked.
  c) the source and target files have a small right-click menu that lets you copy the paths.
  d) If the file sizes are equal and below 100 MB (otherwise it would be too slow), then the contents are compared using SHA-384. If the contents differ, the size label turns red and " Contents differ!" is added.
- Overwrite if newer: [factory default for Backup] Copies only files that are younger/newer than same named files in the destination, otherwise does nothing.
- Overwrite if different size or date: Date here refers to the modified date. This setting provides a pretty good way to copy only what is different and skip the rest. Helps maximizing the lifetime of your SSD, and generally speeds up copy jobs.
  Note: Since the actual contents of the files are not compared (for performance reasons) there is the slight chance that two files with the same size and modified date still have different contents.

But that should be quite unlikely in real life.

- Overwrite if different contents: Overwrite only if the contents of both files differ. The contents are compared by comparing the SHA-256 hash of each file.
- Overwrite: Copies always, whether a file of the same name exists or not.
- Skip: Copies only files that don't exist in the destination, otherwise does nothing.
- Suffix increment to copy: Suffixes an increment (formatted as defined by the template in Configuration | Templates) to the files being copied.
- Affix current date to copy: Affixes the current date (formatted as defined by the template in Configuration | Templates) to the files being copied.
- Affix last modified date to copy: Affixes the file's last modified date (formatted as defined by the template in Configuration | Templates) to the files being copied.
- Suffix increment to existing: Suffixes an increment to the existing files.
- Affix current date to existing: Affixes the current date to the existing files.
- Affix last modified date to existing: Affixes the file's last modified date to the existing files.

The "existing file" is the one that would otherwise be overwritten. If the existing file cannot be renamed for some reason, the copy will not happen.

Note: Files that are copied in-place (copied onto themselves) are unconditionally auto-renamed (incremental affix) if the setting for "On name collisions" is "Ask" or "Overwrite if newer" or "Overwrite".

On failures: Here you can configure what happens on a failure.

- Ask: Pops a prompt on each failure.
- Continue: Just continues processing.
- Cancel: Cancels the whole job.

On "Ask" you get a prompt with four buttons:

- Retry: Try the same operation again. For example, when a file could not be moved because it was in use by another process, you can close that process and then retry the move.
- Continue All: Continues on all such failures (no more asking).
- Continue: Continues (ask again next time).
- Cancel: Cancels the whole job now.

The following failures are recognized:

- Verification failed: May happen if verification is enabled (see below).
- Decryption failed: May happen when copying an encrypted file to a medium that does not support encryption.
- Something else failed: Well, something bad happened that cannot be further specified.

Verification: Verify the correctness of each copy operation on the fly (file by file). If a verification fails you are prompted to continue or abort the operation. You additionally get the option to continue without further such messages. You can choose among several verification methods:

- None (no verification).
- Byte-to-byte: Most secure, but slower on huge files.
- MD5: Fastest of the hashing algorithms offered here.
- SHA-1: More secure than MD5 but slightly slower. Faster than Byte-to-byte on huge files.
- SHA-256: More secure than SHA-1 but slightly slower.
- SHA-512: More secure than SHA-256 but slightly slower.

Notes on verification:

- The verification is only done when bytes are copied, i.e. when a new file is created in the target or when an existing file is overwritten. No verification is done when there is no copying, e.g. when overwriting an existing target file is skipped because the user configured name collision behavior like this.
- Verification ensures that the target file is read from disk, not from the Windows write cache. This makes it slower, but much more useful.
- The reports (available from the Progress dialog) show the hash value of each successfully verified file.
- Verifying a file copy is not necessary under normal conditions. Windows copying is reliable. However, when copying over a network or on USB drives, things can get interesting. Especially if you are backing up files over a shaky network you might want to use Verification for ultimate reliability.

Junctions: How junctions and their contents are handled:

- Copy as folders: Copies/moves the contents of the junction target to the destination folder.
- Skip: Junctions and their contents are skipped.
- Copy as junctions (admin rights required): Copies/moves junctions natively as junctions, meaning they will remain junctions in the target location and keep their original target. This feature works for directly selected junctions, as well as those deeper within selected folders. Symbolic links are copied as symbolic links. If the source file is a symbolic link, the destination file is also a symbolic link pointing to the same file that the source symbolic link is pointing to. If ON, symbolic links are skipped.
  Since creating junctions and symbolic links requires administrative rights, you must run XYplorer as an administrator to use this feature, otherwise all junctions are skipped. Also if a particular junction fails to create for any reason, it is simply skipped (there is no fallback to copying the contents of the junction target recursively because this is probably not what you want).

Rename folders on collision: If checked then all copies of folders in your selection are auto-renamed (incremental affix) on collision, else they are overwritten. Note that this concerns only the top folders,

not any subfolders, of the items selected for the copy. Subfolders are never renamed on a Backup or Custom Copy.

Note: Folders that are copied in-place (copied onto themselves) are unconditionally auto-renamed (incremental affix) even if this option is not ticked.

Ask before merging folders: If checked then you are asked before any of the folders you are copying or moving are merged with same-named folders in the target location.

- The setting is ignored if "Rename folders on collision" is ticked! In that case no merging can happen anyway.
- The "Merge Folder?" prompt is popped for each folder to be merged (unless you tick "Do this also for the next cases" before pressing OK or Skip).
- All "Merge Folder?" prompts are popped before any files are moved or copied.

Ask before overwriting read-only files: Tick it to get prompted before a READONLY or SYSTEM file is overwritten. (SYSTEM is not mentioned in the caption because it would be too long.)

Remove read-only attribute: Tick it to remove the read-only attribute on target files if present on the source files AND if the source medium is optical (CD-ROM, DVD).

Preserve all item dates: Here you can decide whether all three item dates (created, modified, accessed) are preserved with the copy. Note: Also when the setting is not ticked then at least the Modified date of files (but not of folders) is preserved; this is Windows standard behavior.

Safe overwrite: Check it to minimize the risk of data loss due to failures while overwriting. See how it works compared to normal overwrite:

Normal Overwrite:

- Source overwrites target. If there is a failure (e.g. power out) while copying the bytes the target is lost and nothing will bring it back. So if the target was your only backup you are without any backup now.

Safe Overwrite:

- Copy: Source is copied to a new temporary target file in the target folder.
- Optional Verification (you set this under "Verification"): The temporary target is compared with the source.
- Rename: If the copy (and optional verification) succeeds then the target to be overwritten and the temporary target swap names.
- Delete: If the name swap succeeds then the target to be overwritten (which now has the temporary name) is deleted.

Note that due to its logic Safe Overwrite needs enough space on the target device to hold the target file to be overwritten and the temporary target file (same size as the source file) at the same time. Nevertheless: Applying Safe Overwrite is HIGHLY RECOMMENDED!

Skip verification on local hard disks: Check it to skip verification on local hard disks, i.e. when both sources and targets are located on local hard disk drives (aka fixed drives). Ticking it is recommended because copying between local hard disks is extremely error-safe nowadays, so the verification is not really necessary and just increases wear.

Note that portable external hard drives sometimes show up as fixed instead of removable, i.e. as hard disks. They would be excluded from verification as well if you tick this option.

Exclude hidden files and folders: Does exactly what it says. "Hidden" here refers to the file attribute "HIDDEN". Note that items with the "HIDDEN" attribute that are among the selected items (or among the items explicitly specified as source items) are NOT excluded. This setting now only applies to copying (Backup and Copy), not moving (it is difficult to imagine a situation where you would want to move files and folders while leaving hidden items behind).

Show progress dialog: Check it to show a progress dialog (see "The Progress Dialog" below) while the copying is in progress. If unchecked the progress is shown in the status bar if the operation runs in the foreground (else it is totally silent).

Keep progress dialog open: If checked the dialog stays open after the operation is completed. If unchecked the dialog is auto-closed. In the latter case, if Show summary report is checked the dialog is auto-closed only after 5 seconds, otherwise it is auto-closed immediately.

Show summary report: Check to pop up a summary message about what was copied / overwritten / not overwritten. Such a message is only shown when no progress dialog is shown, otherwise the setting has an effect described in the paragraph here above.

Create log file: If checked then all copy operations are logged to file. See details.

Log to default location: If checked (and Create log file is enabled) then the factory default locations are used for the log file location and name. If unchecked you are prompted for a name when triggering the copy operation. The factory defaults are:

- Backup: `<xydata>\Log\Backup_<date yyyy-mm-dd_hh-nn-ss>.txt`
- Custom Copy: `<xydata>\Log\CustomCopy_<date yyyy-mm-dd_hh-nn-ss>.txt`

Tip: The defaults can be changed by tweaking the INI file. See Tweaks.

The Progress Dialog

Backup and Custom Copy operations optionally come with a detailed progress dialog. The dialog contains information about the elapsed and remaining time, the average transfer rate, number of done and remaining files and bytes, etc. ... the usual progress dialog stuff. But there is also something interesting:

- There is a Pause button which allows you to Pause and Resume an operation.
  Tip: Now you are prompted before the process is paused. This prompt alone might often be already all the pause you need, so you need not "really" pause the processing (with the possible disadvantage that processing resumes with processing the current again from start).
- For files larger 10MB you get a per-file progress (numeric percentage only, no separate progress bar, to keep the interface as plain as possible).
- For files being verified you get a per-file verification progress (numeric percentage).
- The amount of used space (and implicitly free space) on the target drive is shown, graphically in a Used Space bar and in numbers. The Used Space bar tooltip shows the path of the actually polled drive. In some contexts (junctions, mounted drives, network drives) this can be interesting.
- Pressing ESC while the operation is in progress will Pause, pressing ESC while the operation is

not in progress (paused or cancelled or completed) will close the dialog.

- After Pause/Resume the process will resume with the file that was being processed when Pause was clicked. So it will be copied/verified again. (But see also next point.)
- Without Safe Overwrite, pausing or cancelling processing while a target file was being overwritten would effectively lead to the deletion of the target. Assuming that this can hardly be a desired outcome of pressing Pause or Cancel, an ongoing overwrite operation is allowed to finish before processing is paused or cancelled. Note that this is not necessary (and therefore not done) when Safe Overwrite is enabled.
- When the operation is paused or completed you can view the details (button Report / All Files). Note that this works also when Create log file is off.
- You also can view the log file from this dialog (button Report / View Log File) if you enabled Create log file. The contents are identical to the Report / All Files contents, but here they are shown in a file opened in the system default editor.
- In the unlikely case of failures (e.g. verification failed) you get a detailed list of all failed files (also under button Report / Failed Files).
- When the operation is completed, the dialog (optionally, see Show progress dialog above) stays up so that you can check the results.
- Showing the progress dialog is optional, see Show progress dialog above.
- The progress dialog remembers its position. The same position is also used for the Backup progress dialog.
- The progress dialogs are cascading, i.e. any non-first dialog is positioned in an offsetting (cascaded) position from the last dialog popped up.

Further Remarks on Backup and Custom Copy

- The available free space on the target drive is checked before each copy operation (per job or per file, depending on the circumstances). The operation is only initiated if enough space is available. This will save you frustration when copying very large files to already packed drives. If space is scarce you are asked how you wish to continue.
- Care is taken to prevent the system from entering sleep/standby during long copy or backup jobs. Windows is kept awake until the copy process is completed. Note: This function does not stop the screen saver from executing. This service needs Windows XP or later.

External Copy Handlers

You can let External Copy Handlers handle the Custom Copy and Custom Move operations. Currently the following products are supported:

- FastCopy
- TeraCopy

To use an External Copy Handler the following has to be ensured:

- Of course, the External Copy Handler has to be installed on your system.
- The setting Configuration | File Operations | Custom Copy Operations | Use Custom Copy has to be ticked.

Configure (button): The button opens a multiline edit box where you can specify any number of external copy handler definitions, one per line. Each line has to follow the same syntax which should agree to this general format:

```
Caption|[Executable]|[Switches]
```

- The Caption has to begin with either "TeraCopy" or "FastCopy".
- The Executable has to be the full path to the executable. The path supports full portability, and you can use all kinds of XYplorer native and environment variables. The path can be omitted in which case it will default to the standard path (e.g. "%ProgramFiles%\FastCopy\FastCopy.exe", or "%ProgramW6432%\FastCopy\FastCopy.exe" for 64-bit FastCopy.exe).
- The optional Switches can be used to modify the behavior of the copy handler (rename on collision, verify, overwrite only older, etc.). Form and function here totally depend on the respective copy handler. Look them up in the documentation of the copy handler.
  Optionally you can define separate switches for Copy and Move operations. For this use the following alternate syntax:
  `Caption|[Executable]|[Copy Switches]|[Move Switches]`

For instance, the contents of the edit box could look like this:

```
FastCopy (AutoClose)|%ProgramFiles%\FastCopy\FastCopy.exe|/auto_close
FastCopy (Verify)|%ProgramFiles%\FastCopy\FastCopy.exe|/verify /auto_close
TeraCopy (Rename All)|%ProgramFiles%\TeraCopy\TeraCopy.exe|/RenameAll
TeraCopy (Overwrite Older)|%ProgramFiles%\TeraCopy\TeraCopy.exe|/OverwriteOlder
```

Select copy handler: Here you can select the current copy handler. This contents of this dropdown are defined by the contents of the edit box that is opened by the Configure button. "XYplorer" is automatically added as first item in the list; select it to use XYplorer's native Custom Copy instead of any external copy handler.

The same can also be done via the toolbar. See next paragraph.

Toolbar Integration: All copy handlers listed here are also listed in the right-click menu of the "Use Custom Copy" toolbar button, where you can now select your preferred copy handler by a single click. Also listed here is "XYplorer" which stands for XYplorer's native Custom Copy.

The icon of that toolbar button turns green when an external copy handler is selected, and the tooltip displays the name of the current copy handler.

Auto-Elevation: The copy handlers are auto-elevated if necessary when copying to UAC-protected locations. You don't need to define any Custom Copy Blacklist for this. Actually, you should remove any Custom Copy Blacklist if you plan to use copiers because locations listed in this file will be handled by shell copy and thus never reach the copiers.

Differences to Custom Copy: Integration is very good (whenever anything is copied or moved the External Copy Handler will be used for it), but there are a few differences to using XYplorer's native Custom Copy:

- No Rich Copy. In a Rich Copy situation Custom Copy will be automatically used instead of the External Copy Handler. Reason: You cannot state multiple destinations (one individual destination for each source file) in the External Copy Handler.
- No undo/redo.
- No background processing via XYcopy (the External Copy Handler is another process anyway), and hence no queueing.
- Any tags will not be copied or moved along.

Special Remarks on particular copy handlers

FastCopy: When copying with FastCopy the mode "diff" (copy only if size or date are different) is used which is the default for FastCopy (see FastCopy documentation).

Miscellaneous

Suppress delete confirmation dialog

Check it to skip the confirmation prompt that usually precedes a deletion. Also controls the behavior when you empty the Recycle Bin.

Note that unticking this option does NOT ensure the confirmation prompt! If the delete confirmation prompt is turned off in the Windows system configuration then nothing will bring it back.

Preserve permissions on move operation

When checked then the security attributes of files are preserved when they are moved in the same volume. Uncheck it to have the moved files inherit the security attributes of their new folder.

File operation progress modeless

If checked the file operation progress window is shown modeless. Which effectively means that you can go on working while files are being moved or copied in the literal background: the progress dialog is not "always on top" of the main window. The setting also affects the confirmation prompt for Delete operations.

Notes:

- A modeless progress window can easily get lost behind larger windows.
- When file operations were triggered by drag and drop the source window is usually blocked until the operation is finished.
- This setting is ignored for Copy and Move operations if Enable background processing is ON.

Recreate source folder structure

Here you can decide whether the relative folder structure present in the source items should be recreated in the target location, aka "Rich Copy/Move". "Rich sources" typically come from recursive search results or branch views. The dropdown offers three choices: Ask, Always, and Never. Factory default is Ask.

Undo & Action Log

Log actions and enable undo/redo

Turn the whole thing on/off.

Remember the logged actions between sessions

If checked then the file action.dat is written on exit and read on startup. Note that it is even written when "Save settings on exit" is off because it must stay in sync with reality as much as possible.

Even on exit without saving

Tick it to remember the logged actions even on "Exit without Saving". After all, if the file system has been changed then "Exit without Saving" won't change it back. You would still like the option to undo them in the next session.

Allowed number of entries in the action log (maximum is 256)

The more entries you log the larger is action.dat, the more memory is used, the slower is startup. But don't worry: It's virtually not notable.

Allowed number of items per logged action (0 = unlimited)

Here you can limit the size of the jobs that are added to the action log (and that can be undone). If a job is larger than the limit it is not added to the log at all.

Purpose: If you frequently process large numbers of files the action log can become quite large (and it's completely loaded into memory on each startup). This setting allows you to set a limit to this.

Date format in action labels

Some options for date display.

Prompt before undo/redo

Pops a short description of what will happen when you click OK. Note: With cumulative undo you get only one prompt, but you get it always (independent of this setting) because cumulative undo can be a pretty terrifying experience...

There are three levels of safety:

* Always
  The safest.
* If action is older than 10 minutes
  You'll lose at most 10 minutes of work without being warned.
* Never
  You are tough.

Prompt before delete

If checked then you are prompted before undoing Copy or New, which both involve deletions. This prompting in accordance with Explorer behavior and the factory default is ON.

Delete to recycle bin

When undoing a Copy or a New operation, items will be deleted. If this option is checked they will be moved to the recycle bin instead of being permanently deleted. This is a safety net in case you pressed Undo without thinking. The price is that you don't get back the disk space. Note that redoing such an operation will not get the items back from the recycle bin but create them anew!

Toolbar Buttons

Show last actions in toolbar button menu

If checked then (after a restart) the Undo/Redo buttons have a small arrow that pops the previous/next actions (relative to the current position in the action log) in a menu.

The dropdown has further options:

* Allow only single step undo/redo
  The safe way. You can see your action history in the menu but you are not allowed to trigger any non-sequential operations. An easy protection from potential disaster.
* Allow cumulative undo/redo
  If selected then undoing a non-next action will not trigger (non-sequential) undo of this action alone but trigger undo for all actions from here to the selected one.

  As this can mean a tremendous (and devastating) operation by a single click, you'll get a safety prompt before it happens.
* Allow non-sequential undo/redo
  If selected then the toolbar menu allows for non-sequential undo/redo.

  When you trigger a non-sequential operation (one that is not next to the current position in the action log) from this menu the current position in the action log will NOT change. This makes it easier to do non-sequential undo/redo operations.

  If checked then inverse operations are tolerated, i.e. you can trigger a Redo by clicking the Undo button and vice versa! This is a necessary consequence of the ability to do non-sequential undo. So Undo/Redo are then Back/Forward in action log.

  Repeated warning: Non-sequential undo can mess up the action log (and hence the possibility of further undo) if you are not careful enough. It's safer to do it from the Action Log Window than via the toolbar menu because the former has more information for you.

Show options in menu

If checked then the above settings are also available in the Undo button's arrow menu. Gives you an idea about what will happen when you click a menu item.

Clipboard

Log clipboard contents and enable restore

Tick it to be able to restore the previous clipboard (Edit | Paste Special | Restore Previous Clipboard). Untick it if you don't care and want to save that memory and CPU.

Note: A clipboard full of files is only stored if it has 1000 or less files. Larger numbers are just too slow and too heavy on memory.

Find Files & Branch View

Find Files

Show relative path in Path column

Show only the relative path in the find results path column (relative to the search path). This is very useful when you do recursive file finds starting somewhere deep down the folder hierarchy when otherwise the path column would be stuffed with redundant information.

Synchronize tree with search location

Check it to always keep the tree in sync with the find results list.

Cache Search Results

Check to store and retrieve the results of searches tabwise and across sessions.

Where and when does it happen. Cached data are not held in memory but written to tab_*.ini files down in the Panes subfolder. They are written whenever a non-cached search tab is backgrounded, and read whenever a search tab is foregrounded (selected). In other words, caching only happens on tab selection, i.e. when you re-open a cached tab. Whereas pressing F3 or any other way to run a search will always trigger a live search.

What is cached. Only the item names are cached. All other data (size, file dates, attributes, etc.) are freshly polled and always up-to-date even in a cached search. Naturally, this polling takes time, and the efficiency of the cache highly depends on the nature of the search:

- The worst case is a search without any filters (e.g. find all items in all subfolders), where reading the cache and processing the data is likely to be even slower than a fresh live search.
- The best case is a search over a large area with relatively few results -- this is where the cache really shines.

Maximum number of items cached (0 = cache always)

Here you can set an absolute limit to the number of items cached. The factory default is set to 1000 and a much higher number is not recommened. There are several reasons for such a limit:

- The bigger the cache the slower it is compared to a live search. So caching is simply inefficient, at least if it is done for speed.

- Reading a cache may (depending on the sort order of the cached items) trigger lots of wild-jumping head movements.
- A cache takes space on the disk.
- A huge cache is more likely to be stale (out-of-date).

Note that this limit does not apply to Search Templates that are explicitly saved together with the search results (which internally is identical to caching).

Follow junctions

Tick it to allow following junctions on a search including subfolders.

Skip invisible subfolders

Enable it to not scan folders hidden by current settings (i.e. HIDDEN folders if "Configuration | General | Tree and List | Items in Tree and List | Select Items... | Hidden files and folders" is OFF, and SYSTEM folders if "Configuration | General | Tree and List | Items in Tree and List | Select Items... | System files and folders" is OFF).

Tip: The effect is identical to using the `/V` switch (see Search Pattern Switches).

Show search results in

Choose among these options:

- Current tab: Show search results right in the current tab.
- New tab: Show search results in a newly created tab.
- "Search results" tab (locked): Show search results in a locked tab named "Search results" (this default name can be changed by renaming the tab). If no locked tab named "Search results" exists it is automatically created. All following search results will be sent to this tab.
- "Search results" tab (unlocked): It's the same as above but -- you guessed it -- the tab is not auto-locked.

Show quick search results in current tab

Tick it to show the Quick Search results always in the current tab (unless it is locked), irrespective of the setting of the above "Show search results in".

Show search information in list

If checked then search information ("Quick Search: <pattern>" or "Find Files: <pattern>") is shown in a colored bar above the results list. There is a small right-click menu where you can edit or remove the search.

The right-click menu also has the toggle Always Sort Search Results This Way: Tick it to initially sort all coming searches just like the list is sorted now. Untick it to reset the feature and use whatever sort order the list is in before each search. Note that sort column, sort direction and even secondary sorting is supported.

Tip: Double-clicking the Search Information Bar removes the current Search.

Search results inherit current columns

Tick it to inherit the current Browse or Find mode columns, view, and sort order to the Search Results listing.

Enable extended pattern matching

Tick it to treat #, !, and [...] as special function characters: # as wildcard for digits, ! as logical NOT, and [...] as a group of characters to match.

Untick it to treat #, !, and [...] literally in search patterns, and not as wildcards or special function characters.

Examples:

```
Pattern       Match Unticked  Match Ticked
-----------------------------------------
#             ##KEEP##.jpg    1.jpg
[1984]        [1984].jpg      1.jpg
!a            !achtung.jpg    1.jpg
!#            !#3.jpg         a.jpg
```

Note that a single ! is not seen as unary Boolean NOT operator even if the setting is on, but as the normal "!" character.

Note that the setting also affects Quick Search.

Matching Characters by Their Ordinal Value

You can specify characters in a range by their Unicode number using the syntax U+HHHH (where H stands for a Hex digit from 0-F). The U and the H can be upper or lower case, it does not matter. All these examples are functionally identical and in a Quick Search they will match all filenames containing one or more characters beyond the ANSI range (0-255).

```
*[U+0100-U+FFFF]* /E
*[U+0100-U+ffff]* /E
*[u+0100-u+FFFF]* /E
*[u+0100-u+ffff]* /E
```

Ranges can be combined without any separator between them. This search pattern, for example, matches filenames with Hiragana or Katakana characters:

```
*[u+3041-u+3096u+30A0-u+30FF]* /E
```

Enable smart Boolean query parsing

Tick it to treat a query as Boolean expression if it looks like one. If ticked then:

```
" AND " is seen as Boolean AND
" and " is seen as Boolean AND
" & "   is seen as Boolean AND
" "     is seen as Boolean AND

" OR "  is seen as Boolean OR
" or "  is seen as Boolean OR
";"     is seen as Boolean OR   (NOTE: this one even works if "Enable smart boolean query parsing" is off because it's a de facto standard in Windows)
```

If unticked all the characters are just seen as normal parts of the filename, and to have them treated as Boolean operators you have to prefix the Boolean marker ":" to the query (or set Mode to "Boolean" in Find Files). But note that the shorthand operators " " (AND) and ";" (OR) are only recognized in Smart Boolean, not in Explicit Boolean mode.

Note that this setting affects the parsing of Quick Search and Find Files patterns for the Name (name:), Tags (tags:), Comment (cmt:), and Contents (cont: or content:) fields.

Note that the shorthand operators " " (AND) and ";" (OR) are only recognized when there are no wildcards or quotation marks. For example:

```
cmt:Amanda Lynda      //match all items with "Amanda" AND "Lynda" in the comment
cmt:*Amanda Lynda*    //match all items with "Amanda Lynda" in the comment
cmt:"Amanda Lynda"    //match all items with "Amanda Lynda" as the comment
```

Tip: If you are unsure how your query is parsed you can always use this script to reveal it (always returns the last-used query): `text <get find_queryparsed>;`

Persist quick search across folders

Tick it to directly repeat the same quick search on the next folder you go to within the current tab.

- If you click the current folder again in the tree the quick search mode is ended.
- The setting is also available in the context menu of the search information bar in the file list.

Branch View

For a detailed description of Branch View see here. All of the following settings are global to all tabs in Branch View mode.

Persist across folders

Keep Branch View mode active when changing folders within the same tab.

Toggle on same query

Applying the same query again, e.g. "? /flat" from the Catalog, turns the Branch View off.

Auto-refresh

Allow a list in Branch View mode to be auto-refreshed.

Level-indent

Indent the items in Branch View to show their hierarchical tree position. Hard-coded to 8 pixels per level.

The level-indented Branch View is most useful when you sort the file list by Path which will give you a tree-like picture. To automatically invoke such a sort order there is the option Default to tree-like sort order here below.

Level-indent width in pixels (1 to 64)

Customize the level-indent width in pixels. Allowed values are from 1 to 64. Factory default is 12.

Default to tree-like sort order

When invoking a level-indented Branch View sort the items like in a tree: All items sorted alphabetically under their direct parent folders. This is usually the best way to show a level-indented view so it's recommended to tick this option.

Let folders pass all filters

If ticked then on a mixed Branch View (showing files and folders), any Visual Filters or Quick Search patterns only apply to files, not to folders. So the folders always stay visible while their contents are filtered. A concept that's further discussed under the name Filtered Branch here.

Multi branch view lists top folders

Tick it to list the containing/top level folders in a Multi Branch View as well, not just the items contained in those folders.

Default branch view type

There are four types of Branch View:

- *Files and folders*
- *Files only*
- *Folders only*
- *Files and non-empty folders* (here any folders that contain nothing or only empty folders (or, in case of a search, do not contain any files that match the search) are not listed in the Branch View)

Here you can choose which type shall be used when Branch View is toggled via menu or toolbar.

Tip: You can as well select a Branch View of a certain type right from the right-click menu of the Branch View toolbar button. The Default Branch View Type is not affected by this choice.

Filters & Type Ahead Find

Visual Filters

Persist Visual Filters Across Folders

By factory default, Visual Filters will not persist across folders within the same tab; in other words, a Visual Filter is automatically cancelled when you change the location within the same tab. Tick this toggle to change this behavior.

Toggle On Same Filter

If checked then applying the same filter to a filtered list again will remove the filter.

Enable extended pattern matching

Tick it to allow ! for Boolean Not, and # for digits in Visual Filters. You can escape the ! by putting a \ before it.

Examples:

```
sweet     match anything sweet
!sweet    match anything not sweet
\!sweet   match anything !sweet
!\!sweet  match anything not !sweet
```

The setting is also found in the right-click menu of the Visual Filter toolbar buttons.

Show filter information in list

If checked then the filter information "Visual Filter: <pattern>" is shown in a Visual Filter Bar above the filtered list. There is a small right-click menu where you can edit or remove the filter (Edit Filter..., Repeat Filter, and Remove Filter).

Also any Live Filter is shown here.

Tip: Double-clicking the Visual Filter Bar removes the current Visual Filter or Live Filter.

Show filter information in tab headers

If checked then the Visual Filter pattern is shown right in the tab header in blue color (color "Marked Text 1"), separated from the caption by a "|" character.

Live Filter Box

Highlight matches

Tick it to highlight the current Type Live Filter matches in the List. Changing the setting will affect the next Live Filter (not any current).

Tip: The option "Highlight matches" is also offered in the context menu of the Live Filter Box icon. Here you can toggle Live Filter highlighting with immediate effect.

Auto-select first match

Tick it to auto-select the first match. Note that this also works for Type Ahead Find with "Redirect typing to Live Filter Box" enabled.

Persistent live filters

Tick it to make Live Filters persistent:

- Live Filters persist across folders, i.e. when you browse to a new folder in the same tab the filter remains active.
- Note that new tabs are opened unfiltered, but they inherit the last used filter from the current tab. So you can turn it on by Toggle Live Filter (Ctrl+Alt+F3).

Note that there is always some Live Filter persistence, independently of this setting:

- Live Filters survive tab switches, i.e. they are still active when you come back to a filtered tab.
- Live Filters survive across sessions, i.e. you can start up with a live filtered tab.

Note that the setting is also available in the context menu of the Live Filter Box icon.

Enable navigation keys

Tick it to allow keys Up, Down, PageUp, PageDown, Back, and Enter to function within the Live Filter Box as if they were press in the List.

Note that the setting is also available in the context menu of the Live Filter Box icon.

Enable extended pattern matching

Tick it to allow ! for Boolean Not, and # for digits in Live Filters. You can escape the ! by putting a \ before it.

Examples:

```
sweet     match anything sweet
!sweet    match anything not sweet
\!sweet   match anything !sweet
!\!sweet  match anything not !sweet
```

The setting is also found in the right-click menu of the Live Filter Box icon.

Delay before filter is applied

Here you can define a delay (after the last keystroke) before a typed in Live Filter is triggered. E.g. set it to 400 ms and it will reduce the browsing action if you type faster than 400 ms between the key strokes.

Tip: Tick Redirect typing to Live Filter Box (see below) to enable Filter-As-You-Type right in the file list.

Visual Filters and Live Filter Box

Apply to Files Only

Tick it to apply the visual filters only to files, not to folders, in other words to let pass all folders.

Note that any of the inline scope prefixes overwrite this setting. So "!\|*.jpg" will not list any folders even if "Apply to files only" is enabled.

Match case

Tick it to make all Visual Filters case-sensitive (a!=A). The factory default is OFF (case-insensitive, a==A).

Note that the setting also affects Global Visual Filters.

Ignore Diacritics

If checked then the filter matching ignores diacritics and a filter "Koln" will match a file "Köln.txt", and vice versa: a filter "Köln" will match a file "Koln.txt".

Note: This setting also affects the Live Filter Box.

Use space character for Boolean AND

Tick it to greatly simplify your Boolean filter terms. Instead of `Chuck AND Berry` or `Chuck & Berry` you can simply do `Chuck Berry` and it will match all items containing "Chuck" and "Berry", regardless of their order.

Exceptions: (a) if the entire pattern is enclosed in asterisks, any spaces in the pattern are not considered a Boolean AND; (b) if the entire pattern is enclosed in quotation marks, any spaces in the pattern are not considered a Boolean AND, and the entire pattern must match the given property:

```
cmt:black hair      //matches all items with "black" AND "hair" in the comment
cmt:*black hair*    //matches all items with "black hair" in the comment
cmt:"black hair"    //matches all items with the comment "black hair"
```

Tip: Note that the same option is also available for the Live Filter Boxes in various small dialogs (e.g. Help | List All Commands..., Go | Recent Locations..., all List Management dialogs): Right-click the filter icon in the box to pop a small menu where you can toggle the option "Use space character for Boolean AND".

Multi-column matching

Tick it to look at the Comment and Tags columns in addition to the Name column if the pattern is not qualified with a selector. If you qualify a pattern, e.g. by a prefix "Name:" no fallback will happen:

```
Linda           -> match *Linda* in Name OR Comment OR Tags
Name: Linda     -> match *Linda* in Name
```

If patterns are connected by Boolean operators (here loose Boolean AND by space) the fallback logic can cross columns:

```
Linda Paul      -> match (*Linda* in Name OR Comment OR Tags) AND (*Paul* in Name OR Comment OR Tags)
```

Type Ahead Find

Enable type ahead find

Check to enable Type Ahead Find (aka "Find as you type") in Tree and List.

Matching

"Match at beginning" of filenames is the factory default (and the only thing Windows Explorer can do), but XYplorer gives you four different matching options:

- Match at beginning: Check to count matches just at the beginning of the file names.
- Match anywhere: Check to count matches anywhere, not just at the beginning of the file names.
- Prefer matches at beginning: Only if there are no matches at the beginning, XYplorer starts looking for matches anywhere.
- Prefer matches at beginning (but only for single-characters): Only if there are no matches at the beginning OR if the pattern is longer than one character, XYplorer starts looking for matches anywhere. Sounds complicated but is the most popular matching mode.

Highlight matches

Tick it to highlight the current Type Ahead Find matches in the List (it's not implemented in the Tree). The color is hard-coded to black on yellow.

Ignore diacritics

Tick it to ignore diacritics in Type Ahead Find in Tree and List. Typing "ö" will match "o" and vice versa.

Note: This setting also affects Jump and Spot matching.

Allow repeated characters

Check it to allow jumping to "33" after pressing "3" twice. If unchecked, the second "3" will jump to the next item containing "3". Note that this setting affects Type Ahead Find in Tree and List.

Skip single spaces

Enable it (it's actually the factory default) to keep the default functionality of the space bar in the List: to select the currently focused item when it is not selected. To make the space bar work like any other key in Type Ahead Find, uncheck the option.

As the name suggests, it only affects single spaces. Within a fast-typed string of letters, spaces work like any other key, e.g. "y " or " y".

Use sorted column

Uncheck to let Type Ahead Find in the file list always use the Name column; check to use the columns Name, Ext, Type, Path, Label, Tags, or Comment if they are the sorted columns.

Paste and find

Tick it to enable Type Ahead Find by pasting text directly into the list. So when you have a bit of text in the clipboard, pasting this text into the List will attempt to find, focus, and highlight the next match,

just as if you used Type Ahead Find.

- Multi-line inputs are cleared of line feeds and then passed to the Paste and Find function.
- The pasted string is trimmed of leading and trailing spaces.
- Characters that are illegal in filenames are removed (if matching is done against the Name column).
- Paste and Find always uses "Match anywhere".
- You can paste a full URL to find and highlight the corresponding file in the current list. For example:
  Pasted Text: https://www.xyplorer.com/whatsnew.php
  Found File: whatsnew.php
- Paste and Find includes Paste and Go: If you paste a full path then XYplorer will simply go to it. This can also be a full path filename. In that case the file will be focused and selected in the list. Note that Paste and Go also works in the folder tree. Note that you can also paste relative paths. They will be resolved relative to the current list path.

Note: There are possible conflicts with pasting stuff from Outlook. If you get those conflicts turn off "Paste and find".

Redirect typing to Live Filter Box

Tick it to enable Filter-As-You-Type right in the file list.

Note that all other options under "Enable type ahead find" are ignored if the typing is redirected to the Live Filter Box.

Preview

Audio/Video preview:

Loop

Auto-restart medium when through.

Autoplay

If checked then Audio/Video files will start playing immediately. Else you have to click on the progress bar or press the space key.

Play only the first seconds

You can optionally play only the beginning of a sound or video file, and then stop, loop, and jump to the next file (depending on the Loop settings). The number of seconds to be played can be defined in the edit box right of "Play only the first seconds". Untick the checkbox or enter value 0 to play the whole file. The factory default is 3 seconds.

Note that for technical reasons the stop time cannot be exact to the millisecond but will usually be

plus-minus 30 milliseconds off.

Keep playing when info panel is hidden

Tick it to keep a running audio/video preview busy when you close the Info Panel.

Use native handling in the preview pane

Check it to play audio and video directly in the Preview Pane instead of going through an embedded Windows Media Player. If checked and the Info Panel (with the Preview Tab) is closed, you will see an interactive progress bar in the Status Bar while the audio/video is playing.

Audio preview:

Play also when info panel is hidden

Tick it to preview (listen to) your audio files while the info panel is down and/or the preview panel is not selected.

Seamless wave looping

Tick it to play WAV files in a special way that renders perfectly seamless looping.

- For technical reasons this mode has no progress feedback and no pausing (only full stop / back to zero).
- WAV files will be looped regardless of the setting of "Configuration | Preview | Preview | Audio/Video preview". It does not have to be "Play[ed] Again".
- This setting also affects "Configuration | Preview | Mouse Down Blow Up | Mouse Down on Thumbnails and Icons | Audio preview". Analog to above, WAV files will be looped regardless of the setting of "Configuration | Preview | Mouse Down Blow Up | Mouse Down on Thumbnails and Icons | Loop".

Video preview:

Preview as thumbnail

Tick it to quickly show the video thumbnails in the preview area, instead of slowly loading the video (and optionally playing it).

Skip [...] milliseconds

Check it to preview a single static frame of a video. You can further define the offset of that frame (skip intro) in milliseconds, which allows you skip any black/blank frames at the very beginning.

Note that Skip is supported only in the Preview Tab. Preview Pane and Floating Preview will show the shell preview in paused state.

Image/Video preview:

Zoom smaller originals to fit preview area

Images that are smaller than the preview area (under standard display settings 320x160 pixel), for example icons and cursors, are zoomed to fill up the space available.

Border style options: No Border, 2D Border, 3D Border, Shadow

Choose border style for image and Audio/Video preview (Shadow is only applied to image preview).

Affects the Preview Tab and the Preview Pane.

Image preview:

High quality image resampling

Here you can control the quality of the image resampling used in the Preview Tab, the Floating Preview, the Full Screen Preview, and in Mouse Down Blow Up (on Preview and on thumbnails).

The setting does not affect the thumbnails quality.

Limit original preview size

Tick it to limit the size shown as "original size" (or 100% size). Why would you want to do this? Because it can be much faster, especially with large RAW images, and still give you enough detail to evaluate an image.

- This limit only affects the preview of RAW images and other formats that need a special preview handler (eg WEBP). Normal image formats (JPG, PNG, GIF, BMP) and PDF are not affected (meditation in progress).
- This limit affects the "Original Size" in all previews that can show the original size, like Floating Preview and Mouse Down Blow Up.

Auto-rotate preview

Tick to enable Auto-Rotate for the Image Preview, its Mouse Down Blow Up, the Floating Preview, and the Full Screen Preview.

Digital cameras with orientation sensors allow auto-rotation of portrait images by reflecting the positioning of the camera with respect to the ground in the picture's EXIF data. So, the Auto-Rotate feature works only with picture files that contain EXIF data, usually JPEGs and TIFFs (extensions: jpg, jpeg, tif, tiff).

Transparency background

Here you can choose how transparent areas in images (PNG, GIF, and other formats supporting transparency) are displayed. You have four options, including the nice "neutral" option which melts the transparent areas with the surrounding background:

- Neutral
- Grid
- White

- Black

Color 1 and Color 2: You can customize the two checkerboard grid colors. Color 1 is the first top-left checker field. Left-clicking the color field opens the Choose Color dialog, right-clicking resets to the factory default color.

Tip: If you don't want the checkerboard grid but just one particular color simply set both colors to that same value.

Quick file view:

Modeless dialog

Tick it to have the Quick File View (and also the Metadata view) modeless similar to the Floating Preview: You can leave it open and select files in the file list to have them quick-viewed.

Web preview:

Enable server mappings

See Web preview.

Text preview:

Display Tabs as spaces

Here you can define how Tab characters are displayed in ASCII Raw View and Quick File View. Select 0 (factory default) to expand TABs by the system default tab stop value (usually 8). Select any other number to display TABs by this number of spaces (the files are not changed by this, of course).

Note: This setting also affects texts in Hover Box and Mouse Down Blow Up.

UTF-8 auto-detection

Tick it to attempt to auto-detect BOM-less UTF-8 files. This setting also controls UTF-8 detection in Find Files Contents search.

For your interest, the UTF-8 BOM (or signature) is "ï»¿" (EF BB BF), but often UTF-8 files come without the BOM.

Preview delay

Here you can enter the number of milliseconds that a preview is delayed after a file has been selected. This should make it a nicer experience to walk through a directory while the Preview tab or the Preview Pane is open.

Previewed Formats

XYplorer attempts to preview all checked file types you see in the lists. What formats actually work depends on the configuration of your local system (what's the Windows version, which preview handlers are installed). If you experience problems with a certain format, uncheck it in the list and it will not be previewed.

In the Categories list you can enable/disable the preview for a whole category, i.e. for a group of file types, e.g. for all Image files.

In the [File Types] list (contents depend on what is selected in the Categories list) you can enable/disable certain file types (identified by extension), or add new ones to the categories.

Order of Precedence

Note that there is an order of category precedence when previewing files, e.g. when a file format (=extension) is listed in the Text and in the Document Files category then it will be previewed as text (if possible). This is the order of precedence:

1. User-Defined Preview Handlers
2. Preview as Thumbnail
3. Text Files
4. Document Files

...

9. Video Files

User-Defined Preview Handlers

Here you can associate file types (identified by the extension) directly with the GUID of any installed 64-bit preview handler. It's your job to find out the GUID and whether the handler is fit to preview this type of files. So this is definitely an advanced feature.

The syntax is simple. For example, this definition would associate EXE files with the Quick View Plus handler:

```
exe>{8B1E92F5-AC8E-4DAB-9804-3C01AA112F17}
```

The displayed item in the list comes with some additional human-friendly information for your pleasure, for example:
*.exe, Executable File > {8B1E92F5-AC8E-4DAB-9804-3C01AA112F17}, Quick View Plus - Preview Handler

Notes:

- Category User-Defined Preview Handlers has precedence over all other categories. So this is the complete order of precedence (the order in which a given extension is checked against the categories): User-Defined Preview Handlers > Text Files > Document Files > Web Files > Font Files > Image Files > Audio Files > Video Files
- Contrary to other categories you can have more than one definition for the same extension in the list. So you can quickly de/activate a preview handler by changing some checkmarks. Fictitious example (- and + stand for (un)ticked checkboxes):

```
- txt>{1B1E92F5-AC8E-4DAB-9804-3C01AA112F17}
+ txt>{2B1E92F5-AC8E-4DAB-9804-3C01AA112F17}
- txt>{3B1E92F5-AC8E-4DAB-9804-3C01AA112F17}
```

The list is processed from top to bottom, unchecked items are ignored, the first match wins.

- You can have wildcards in the extension part of an entry to match whole families of items. To mark an extension as wildcarded extension prefix it with * (asterisk); this * is not part of the actual pattern but just the marker. Fictitious examples:

Pattern	Matches
*ab?>{1B1E92F5-AC8E-4DAB-9804-3C01AA112F17}	"ab" plus one other character
*??>{2B1E92F5-AC8E-4DAB-9804-3C01AA112F17}	any two characters
*###>{2B1E92F5-AC8E-4DAB-9804-3C01AA112F17}	three digits
*3##>{2B1E92F5-AC8E-4DAB-9804-3C01AA112F17}	all numbers from 300 to 399
3##>{2B1E92F5-AC8E-4DAB-9804-3C01AA112F17}	3## (# not treated as wildcard)

- With User-defined Preview Handlers you gain unprecedented control over the preview of particular file types. For example, if you have more than one image preview handler on the system, you can now easily define which is used for which image type, without going into the configuration of each preview handler or editing the registry.
- This feature also increases portability: You can re-associate preview handlers without touching the registry of the host system.

Preview as Thumbnail

This category is for files that are known to implement their preview not via Preview Handlers but via Thumbnail Providers. It simply saves time, energy, and climate to skip other preview methods that won't work anyway. All 64-bit Thumbnail Providers are supported.

Usage Tip: If you add the extension "pdf" to "Preview as Thumbnail" you get the fast thumbnail preview for PDFs everywhere (Preview Tab, Preview Pane, Floating Preview). It usually shows the first page of the PDF. Of course, a Thumbnail Provider for PDF must exist.

Buttons

Find...: Use this button to quickly jump to a particular extension in its category. Note that Find looks for the next match from here (current category, current extension). It turns around to the beginning when it has reached the end.

Tip: After using button "Find..." once, you can repeat the search from the current position by pressing Shift+F3.

Add...: Use this button to add new extensions. User-added extensions are colored blue (or color "Marked Text 2"). They can be excluded from preview by unticking them, or completely removed from the list using the Remove button.

For example, to add *.zig files to the text preview you first select the Text Files group in the upper list, and then add "zig" (without the quotes) as extension to the lower list.

You can as well add files without any extension as a separate file type to any of the file type groups. To do this state "<none>" (without the quotes) as the extension.

Keyboard shortcut: INS.

Edit...: Edit a user-added or user-defined file type. Keyboard shortcut: F2.

Remove: Remove a user-added or user-defined file type from a category. Keyboard shortcut: DEL.

Thumbnails

Width/Height

Thumbnail widths and heights are configurable. You can define three thumbnail types, for each you can choose between various sizes for each dimension.

Cache thumbnails on disk

While the processing on the fly is pretty fast (BTW, it can be aborted at any time by ESC), you can achieve speed of light if you enable the thumbnail caching. Contrary to Explorer, no hidden "thumbs.db" files will be scattered all around your hard disk, but (unless you opt out of this, see below) XYplorer stores all cache data in one central folder whose name and location you can choose yourself (button Browse...) (default is a folder "Thumbnails" located right under the application's data folder).

Include local disks: By default enabled, this option lets you exclude local disks from caching which can be quite useful when operating XYplorer from a removable media (e.g. USB stick) on an alien host system. Note that when you untick both "Include local disks" and "Include removable media and network locations" you will effectively get no caching at all.

Include removable media and network locations: Tick it to cache thumbnails for removable drives, network locations, and portable devices.

Include search results: Cache as well thumbs of Find Files results.

Show cached thumbnails only: Tick it to enable the following behavior:

- Only thumbnails present in the thumbnails cache are shown.
- No new thumbnails are created.
- The benefit: Instantaneous browsing in thumbnails views in any location.

Remarks:

- You still can explicitly create new thumbnails by using the various "Refresh Thumbnails" commands, and especially the "Create Missing Thumbnails" command in menu View | Caches (and in the right-click menu of all View related toolbar buttons). In that case the cache on disk will be updated. So, the "Show cached thumbnails only" mode yields more control to you: Now *you* decide which thumbnails are created and when.
- "Show cached thumbnails only" is ignored if "Cache thumbnails on disk" is OFF.

- When you untick "Show cached thumbnails only" then any missing thumbnails are created right away.

Cache path: Here you can specify the cache path, i.e. the location where the cache files are placed. Note that the Thumbnails Cache Folder is fully portable, i.e. you can define it in XYplorer's portable path syntax, e.g.:

```
Thumbs          : relative to application data path
Thumbs\         : same as above (backslash is optional)
..\..\Thumbs    : relative to application data path
?:\Thumbs       : relative to this XYplorer's drive
%temp%          : Temp folder
%temp%\XYThumbs\      : subfolder of Temp folder
<xydata>\Thumbs\      : subfolder of application data folder
DriveName:\Thumbs     : relative to the drive named "DriveName" (see Named Drives)
F:\Thumbs       : absolute
```

Tip 1: The tooltip of the Thumbnails Cache Folder edit box shows the resolved absolute path.

Tip 2: You can state a non-existing path in the field. It will be created on OK-ing the Configuration dialog.

Button "Clear...": Allows you to delete all *.dat2 and *.dbits files from the thumbnails cache folder. Note that the thumbnails cache folder is the one that is defined in the "Cache folder" edit field on Configuration | Thumbnails in the moment you press "Clear...". But there's a prompt anyway, before the actual deletion takes place.

Resolve cache path from current folder: Enable it to make a softly defined cache path location (i.e. one that is not a hard path) dependent on the current folder instead of the XYplorer app data path. The current folder is the folder for which you are currently showing thumbnails. Examples:

```
Cache path    Resolve cache path      Path is resolved to
              from current folder
-------------------------------------------------------------------------
?:\XYThumbs\  disabled                [XYplorer app data drive]\XYThumbs\
?:\XYThumbs\  enabled                 [Current drive]\XYThumbs\, eg E:\XYThumbs\
                                      [Current share]\XYThumbs\, eg \
\Server\Share\XYThumbs\
XYThumbs\     disabled                [XYplorer app data path]\XYThumbs\
XYThumbs\     enabled                 [Current folder path]\XYThumbs\
[empty]       enabled                 [Current folder path]\
The setting has no effect on a hard path, e.g.:
E:\XYThumbs\  disabled                E:\XYThumbs\
E:\XYThumbs\  enabled                 E:\XYThumbs\
```

With Paper Folders the cache path is resolved relative to the path of the paper folder file itself.

Using this setting makes your thumbnails cache even more portable. You can take it with you on a USB stick and use it on any other system that has XYplorer on it. The pattern ?:\XYThumbs\ doesn't care about any new drive letters Windows assigns to your removable drives.

If your cache path is a subfolder of the current folder (e.g. pattern `XYThumbs\`), you can do the following: move the cache along with the original files, rename the folder of the original files, or make a copy of the folder of the original files (including the cache subfolder) in any location. No new thumbnails will be created, the cache will not get stale, the copied cache will just work. Even if you do the above with an external program while XYplorer is closed.

Tip: Refreshing the Cache

To refresh the thumbnails cache click menu View | Caches | Refresh Thumbnails. However, it won't be necessary too often: while XYplorer expects and supports user smartness, the cache isn't dumb either and whenever an image file is updated the cached thumbnail is immediately updated too.

Create Missing Thumbnails: Menu View | Caches (and the right-click menu of View related toolbar buttons) features the command "Create Missing Thumbnails" if "Show cached thumbnails only" is ON. You can use to create those thumbnails that are missing from the cache, and have them added to the cache.

Also whenever you rename or move a folder within XYplorer (or with another application while XYplorer is running), any affected thumbs cache is automatically adjusted to the change in the background. This is accomplished with an index file called XYThumbs.txt which is created and maintained in the thumbs cache folder. The format of the index file is simple and self-explaining, so that you can even edit it manually or programmatically if needed (if you try to do this note that the folder list is expected to be sorted).

Create all thumbnails at once

Check it to have all thumbnails in the current list created at once (not just the ones that are currently visible in the viewport). Enabling this option is only advisable if also "Cache thumbnails on disk" is enabled because then the work will have to be done only once. Otherwise browsing large image folders will become slow again and again.

Note that even with "Cache thumbnails on disk" enabled this new setting makes browsing a bit slower on fully cached folders. So, if all or most of your folders are already fully cached, you can (very) slightly increase speed by disabling "Create all thumbnails at once".

Note that this setting is ignored in Find mode (Search Results and Branch View) for performance reasons.

Tip 1: The context menu of the Toolbar buttons related to list views features a command Create All Thumbnails Now. Here you can have all thumbnails in the current list created for this list, without having the global Create all thumbnails at once enabled in Configuration | Thumbnails.

Tip 2: The same context menu also features the commands Refresh Selected Thumbnails (High Speed), Refresh Selected Thumbnails (Fast), Refresh Selected Thumbnails (Crisp), and Refresh Selected Thumbnails (Smooth).

Show thumbnails for RAW files

Tick it to show the thumbnails of RAW image files generated by digital cameras (e.g. CRW, DNG, NEF, RAF, etc). Of course, the required RAW Codecs have to be installed in the system.

Show thumbnails for non-images

Tick this option to make use your Shell's capabilities (shell extensions, thumbnail handlers), and you will get thumbnails for PDF, HTML, MHT, URL, various document and video formats.

Background: The Shell provides a way to obtain preview images (aka thumbnails) for any file which has a Shell extension that supports this: typically images, videos, documents, and folders. Windows Explorer's thumbnails are created using this way.

Note: It depends on your operating system and the installed software which file types are actually supported. In the web you can find numerous "Windows Thumbnail Plugins" that will add more thumbnail formats to the Shell and hence to XYplorer.

Show folder thumbnails

Tick it to show thumbnails for folders if possible.

Folder Thumbnails. The file list supports Folder Thumbnails. When enabled, folders display a thumbnail drawn from one of the contained files in the following order of precedence:

1. desktop.ini (this file can define a folder thumbnail in various ways)
2. folder.jpg, folder.jpeg, folder.png, or folder.gif (in that order)
3. <foldername>.jpg, <foldername>.jpeg, <foldername>.png, or <foldername>.gif (in that order).
4. The first image file among the alphabetically first 50 files.
5. The first video file among the alphabetically first 50 files.
6. The first PDF file among the alphabetically first 50 files.

Show thumbnails in tiles views

Tick it to show thumbnails in Small Tiles and Large Tiles views. Else only system icons are shown.

Small size: Enter the size (width = height) for thumbnails in Small Tiles view. Minimum is 32, maximum is 512 pixels.

Large size: Enter the size (width = height) for thumbnails in Large Tiles view. Minimum is 32, maximum is 512 pixels.

Auto-rotate thumbnails

Tick it to enable Auto-Rotate for the Thumbnails and their Mouse Down Blow Up.

Digital cameras with orientation sensors allow auto-rotation of portrait images by reflecting the positioning of the camera with respect to the ground in the picture's EXIF data. So, the Auto-Rotate feature works only with picture files that contain EXIF data, usually JPEGs and TIFFs (extensions: jpg, jpeg, tif, tiff).

Align to bottom

Tick it to vertically align all thumbnails to bottom.

Zoom to fill

Tick it to fill all the available space with the thumbnail. Left/right parts or top/bottom parts may be cropped. Using this option makes better use of the available space at the expense of the cropped areas.

- Images are never stretched to fill the space, just cropped.
- The cropped image is centered, i.e. the cropping is done equally at both of the cropped sides.
- Mouse Down Blow Up on cropped thumbnails works fine just as expected.
- Changing this setting takes immediate effect on the currently shown thumbnails.
- Cropped thumbnails are cached independently from non-cropped ones, so once cached you can quickly toggle Zoom to Fill without any delays.

Tip: The command is also available from the context menu of thumbnails/view-related Toolbar buttons.

Zoom to fit

Tick it enlarge smaller originals as much as the current thumbnail size will allow (no cropping). For example, in the default Large Tiles view, a 16x16 icon is then displayed in 192x192 monster size.

Note that the setting only affects painting the thumbnails, the thumbnail cache in memory and on disk is not touched.

If the original is at least 4 times smaller (in any dimension) than the thumbnail or the original is 32x32 pixels or smaller, it will be resized without anti-aliasing, so you can really see the fat pixels of your little icons. With anti-aliasing it would just look awful.

Show film strip overlay on video thumbnails

Tick it to show video thumbnails with dark edges and sprocket holes.

Show file icon on thumbnail

Check to show the small file icon in the bottom-left corner of the thumbnail.

Show dimensions of original

Tick it to display the size of the original image at the bottom of each thumbnail.

For videos as well

Tick it to show dimensions also for videos. The main reason to turn it off speed. Retrieving this info can take time.

Show caption

Tick it to show captions (= filenames) below the thumbnails.

Note: When "Show caption" is disabled then no inline rename is possible in the file list.

Tip: On Thumbnails views the command *View | Columns | Autosize Columns Now (Ctrl+Numpad Add)* toggles the thumbnail captions visibility.

Overlay caption

Tick it to display the caption right on the thumbnail instead of underneath it. That way you save screen

space and still see the filenames.

- The overlaid captions are done in white text on a semi-transparent black background.
- It's done only in the three Thumbnails views, not in Tiles or Details with Thumbs.
- The max number of lines is controlled by "Configuration | Preview | Thumbnails | Caption lines". The overlay height will then adjust to the lines needed.
- The dimensions on "Show dimensions of original" are appended to the subtitle in parentheses.

Quality

Choose between four qualities:

High Speed: Check this box to have thumbnail creation run 3 up to 5 times faster than "Fast". The price you pay is a lower quality of the thumbnails. This setting needs Windows Vista or later. Otherwise the "Fast" algorithm is used internally.
Note: This setting uses the Windows thumbnail cache, if available; embedded thumbnails may not be synchronized with the actual image if the cache is out of date.

Fast: Check this box to have thumbnail creation run much faster than "Crisp" and "Smooth". The price you pay is a lower quality of the thumbnails.
Note: This setting uses embedded thumbnails if they are available; embedded thumbnails may not be synchronized with the actual image.

Crisp: Almost as good as smooth, with a little edge.

Smooth: The best quality.

Note: Changing the quality will NOT automatically affect thumbnails currently shown in the list or stored in a cache on disk. For this you have to explicitly recreate the thumbnails via "View | Caches | Refresh Thumbnails".

Transparency

Here you can choose how transparent areas in images (PNG, GIF, and other formats supporting transparency) are displayed. You have four options, including the nice "neutral" option which melts the transparent areas with the surrounding background:

- Neutral
- Grid
- White
- Black

Style

Choose between various display styles.

Padding

Here you can customize the distance between thumbnails. Note that depending on the thumbnails

Style some minimal padding is internally hard-coded and silently ensured. Only style "Plain" allows effectively zero padding.

Caption lines

Here you can customize the number of caption lines shown. This setting affects Thumbnails and Large Icons views.

Use (Thumbnails View Background)

Here you can customize the background color just for thumbnails and tiles views. Notes:

- The color is only applied to the active pane.
- The color overwrites Configuration | Styles | Mirror tree box color in list.

Mouse Down Blow Up

Mouse Down Blow Up is a unique feature in XYplorer: Mouse down on image preview and thumbnails pops up the image in original size.

General

The general settings affect Mouse Down Blow Up on Info Panel Preview, Floating Preview, and on thumbnails.

Use whole screen

If checked then the blow up will use the whole screen (if necessary), not only XYplorer's main window. The option "Shrink to fit" (above) will then apply to the screen, of course, i.e. it will work as "Shrink to fit screen".

Centered

Tick it to always center the blow up in the window or screen (depending on "Use whole screen"), no matter where the thumbnail/preview is located or where exactly you down your mouse. Applies to MDBU on thumbnails, on file icons, on the Preview tab, and on the Floating Preview, in other words: everywhere. And on left and right mouse down.

With border

Tick it to show a 1-pixel border around the blow up.

Shrink to fit

Check to avoid scrolling around larger images.

Fit width only

If ticked then the blow up is shrunk so that the full width fits in the view port, only vertical panning is possible, and the panning is restricted to what's necessary to show the full height of the blow up.

Allow panning

Tick it to allow panning the blow up even if it's shrunk to fit (and hence fully visible).
This setting does not affect the vertical-only panning on "Fit width only".

Movement

Lets you control the Mouse Down Blow Up behavior on mouse move. There are two options:

Loupe: The mouse pointer, the blow up, and the preview are always exactly aligned "over" the blown-up spot.

Touch Screen: Like with a touch screen the blow up is dragged around along with the mouse.

Apply zoom

Tick it and enter a percentage of your choice to show the Mouse Down Blow Up previews in a scale larger (or smaller) than the original size, aka Mouse Down Blow Up Zoomed (MDBUZ). Got small images, large screens, and weak eyes? MDBUZ is your ticket.

- You can also enter a value smaller than 100% to pop a preview smaller than the original. For what it's worth.
- Allowed values: 10% - 1000%.
- From 400% onwards you will see the real pixels, not any anti-aliased sludge which just does not look good anymore at these extreme enlargements.
- An internal upper limit is currently hard-coded to 50,000,000 square pixels. Otherwise zooming large images by 1000% (note that MDBUZ shows the whole image, not just a part) will destroy your computer... ;)
- The zoom applies to all MDBUs (thumbnails, icons, preview tab, preview pane, floating preview) as long as they show an image (including thumbnail previews of PDFs and similar).
- The blow up obeys to "Shrink to fit" if ticked.

Tip: If you use a high zoom (1000%) and combine it with "Shrink to fill" you end up with a "Zoom to fit" effectively for images that are larger than just an icon.

Mouse Down on Thumbnails and Icons

On left mouse down: Pop up image in original size on left mouse down. The popup stays up until the mouse button is released.

Tip: If it's unticked you can drag and drop thumbnails easier because you can use the whole thumbnail area for grabbing, not just the caption.

On middle mouse down: Pop up image in original size on middle mouse down. The popup stays up until the mouse button is released.

On right mouse down: Turn it on for right mouse down.

- Stay up: Tick to have the blow up stay up until clicked again or any button is pressed.
- Fit popup to screen: Tick to show the whole image using the whole screen. If unticked then the MDBU settings at Configuration | Mouse down blow up | General are used.

- Fit popup width only: Does the same as "Fit width only" above (slightly different caption to avoid confusion), but only on thumbnails and only on right-click. So that way you can have the full size popup on left-click, and the fit width one on right-click.

Allow dragging items by the thumbnail: Tick it to enable dragging items by the thumbnail. The price you pay for this is a 150 msec delay before you get the blow up (in case you don't drag). XYplorer uses this time to decide whether you intend a drag or a blow up.

If this is ticked a quick right-mouse-down-mouse-up on a thumbnail will show the shell context menu.

Note that if it is unticked you can still drag a file in thumbnails view by grabbing it by the caption.

Enable blow ups on file icons as well: Tick it to enable Mouse Down Blow Up on file icons in Details and List view. Works for image files (including PSD and RAW formats, and animated GIFs), Video files (showing stills), Document files, PDF files, Text files. Of course, for many document and media formats the necessary Codecs and shell extensions have to be installed on the system. The feature also works on shortcuts (*.LNK files) to such files.

- Remember relative position: Tick it to remember the relative position of the blow ups on file icons (relative to the point of mouse down).
  The location is only locked (= stored and remembered) if the blow up is beyond (not covering) the point of the mousedown.
  Consequently, you can stop any locking by dragging the current blow up over the mousedown.
  The main idea behind this setting is to get an unobstructed view of the file names an icons.

Audio preview: Tick it to enable Mouse Down Blow Up for audio files, also known as Quick Audio Preview.

- Loop: Tick it to loop the playback.

Mouse Down Blow Up of Text

You can as well blow up the textual contents (i.e. what can be shown of them in one view) of files. No need to open an application or a preview pane anymore. There's never been a faster way to check the contents of a file.

- This works for text files and, if the necessary IFilters are installed, also for Web and Document files. Windows 8 and later seem to support this by factory default.
- It also works on file types that are not listed as text files in Previewed Formats. Any non-binary file gets a blow up.
- Automatic UTF8 decoding.
- The blow up is always centered initially (else the top of the text is often out of view).
- Mouse Down Blow Up on Icons is supported.
- The font used is controlled in Configuration | Fonts | Edit Text.
- The height if the blow up adjusts to the actually needed space. It will be smaller if there is only little text.

Mouse Down Blow Up Hex View

You can also blow up the raw hexadecimal contents (i.e. what can be shown of them in one view) of any file by holding CTRL while you down the mouse on thumbnail or icon. Works with any file type (but not with folders, of course). The hex data are shown in blue, as opposed to normal text data which are shown in black.

Mouse Down Blow Up of Animated GIFs

Note that Mouse Down Blow Up is also available for the thumbnails Animated GIFs. The blow up will be animated.

Quick Audio Preview: Mouse Down Blow Up for Audio Files

If "Configuration | Mouse Down Blow Up | Audio preview" is ticked Mouse Down Blow Up works for audio files. On mouse down (on icon or thumbnail) it just plays the sound as long as you hold down the mouse. All previewable audio file types are supported (MP3, WAV, FLAC, OGG, MID, AIFF ... you name it) -- of course, the necessary CODEC has to be installed.

Once the audio preview starts a basic progress bar is shown in the main Status Bar giving you an idea of the current position within the piece. On a preview that's staying up: (see blow) the progress bar is interactive: You can left-click on it to change the position of the playback. Right-click the progress bar to toggle Pause/Play. There is also a tiny button to toggle Pause/Play. Also the SPACE key will toggle Pause/Play (while focus is in the list).

Tips:

- Tick these two and the audio preview starts on right-click and stays up on releasing the button:
  Configuration | Mouse Down Blow Up | On right mouse down
  Configuration | Mouse Down Blow Up | Stay up
- Tick this and you get audio preview via Mouse Down on file icons:
  Configuration | Mouse Down Blow Up | Enable blow ups on file icons as well
- To stop a preview that's staying up: Either left-click or press ESC, or click anywhere in the list, or start the next file's audio preview.
- Horizontal Scrubbing: While holding down the mouse (on the icon or thumbnail), you can move the mouse to the left or right to move to a new position in the playback file (fast forward/backward).
- Vertical Scrubbing: While holding down the mouse (on the icon or thumbnail), you can move the mouse up or down to change the playback volume.
- The playback loops if Loop is ticked.

Mouse Up on Folder Icons

Folder contents preview

Tick it to get a popup menu (scrollable if necessary) listing all items contained in the folder whose icon you just clicked (mouse-upped) in the folder tree or file list. *"Mouse Up Show Down"* is the catchy name for this revolutionary feature, which functionally is a mouse-driven instant non-invasive Folder Contents Preview. These are the main properties:

- The popup menu lists all contained folders and files (folders on top), without the full path.
- Click the same icon again and the menu disappears. You can also hide it by pressing ESC or by clicking anywhere else in the window.
- Works in all views.
- If tweak `MUSDonthumbs=1`: If there are no icons in a view but folder thumbnails, it works on the folder thumbnails. In that case the folder thumbnails lose their Mouse Down Blow Up skills.
- Clicking a folder item in popup menu will open that folder in XYplorer. It will zap you 2 levels down, cool motion on a single click.
- Clicking a file item in popup menu will run that file as if you double-clicked it in XYplorer.
  Caveat: Note that your Custom File Associations might not work as expected here, since there is a crucial difference to a *real* double-click: The file is not selected, it's not even visible in the current list. Not every function can handle that.
- General visibility settings are honored. So, when an item is hidden from the list then it's also hidden from the Folder Contents Preview. In the unlikely case that all items are hidden, you won't see "Folder is empty." but "[n] items are hidden."
- It supports File Info Tips and Hover Box. Hover the icon (or caption, depending on the settings in File Info Tips) of any item listed in the Folder Contents Preview and you get the File Info Tip or Hover Box just like in the normal list. Note: For the Hover Box this has to be enabled in *Configuration* | *Information* | *File Info Tips & Hover Box* | *Show Hover Box* | *Select Context*.
- If a folder is empty a menu pops up with just one disabled item "Folder is empty."
- Shortcuts (LNK files) to folders are supported (treated like folders).
- Each item in the popup menu has a little context menu itself featuring a couple of useful commands for files and folders.
- Tip: Note these additional options for clicking on folders in the popup menu (not featured in the context menu to keep it light):
  `Shift+Click`: Open Folder in a New Tab (also: `Middle-Click`)
  `Shift+Alt+Click`: Open Folder in New Tab in Other Pane
- Tip: Note these additional options for clicking on files:
  `Ctrl+Shift+Click`: Go To File in New Tab
  `Ctrl+Alt+Click`: Go To File in Other Pane
  `Ctrl+Shift+Alt+Click`: Go To File in New Tab in Other Pane
  `Ctrl+Shift+Click`: Go To File in a New Tab (also: `Middle-Click`)

In tree

Enable the Folder Contents Preview in the folder tree.

In list

Enable the Folder Contents Preview in the file list.

On left mouse up

Pop the menu on left mouse up.

On right mouse up

Pop the menu on right mouse up.

Sorted by

Here you can control the sorting in the Folder Contents Preview (including the Hover Box on folders).

Name	= Name, ascending
Modified	= Modified date, descending
Created	= Created date, descending
Accessed	= Accessed date, descending

Tip: While the Hover Box is showing for a folder you can cycle the sort order by pressing the "O" (letter o) key.

Tabs

New tab path

If you want to open new tabs always in the same path then enter this path here. Leave the field empty to always open new tabs at the current path.

Tip 1: You may enter "This PC" (or whatever your "This PC" node in the Tree is called) to open new tabs at "This PC".

Tip 2: The field supports environment variables and native variables, so you can enter e.g. `%userprofile%\Desktop` or `<xydrive>`.

Open new tab

Choose where a new tab is opened:

- Next to the current tab
- At the end

On closing the current tab

Choose which tab is selected when the current tab is closed:

- Activate the right tab
- Activate the left tab
- Activate the last used tab
- Activate the default tab
  If there is no default tab defined it will fall back to "Activate the right tab".

Cycle tabs in recently used order

Check to cycle the tabs based on the usage sequence (aka MRU) instead of the left-to-right position.

The cycle functions are these (as listed in Customize Keyboard Shortcuts):

Miscellaneous / Tab Functions /

Cycle Tabs Backward:	Ctrl+Shift+Tab; MouseWheel Up
Cycle Tabs Forward:	Ctrl+Tab; MouseWheel Down
Cycle Tabs Backward, Delay Browsing:	Ctrl+PageUp
Cycle Tabs Forward, Delay Browsing:	Ctrl+PageDown

Note that going by MRU has the usual "history" logic: If you go backward from A to B, then to return to A you have to go *forward* (not backward again).

Note that with "Activate left tab on closing current" and "Cycle tabs in recently used order" both enabled, the *previously used tab* is auto-selected on closing the current tab.

Reuse existing tabs when changing the location

Tick it to reuse existing tabs when changing locations through Favorite Folders, Favorite Files, Special System Folders, Hotlist, Recent Locations, Catalog, Tree, List, Breadcrumb Bar, or whatever other way.

Maximum number of tabs (0 = unlimited)

If the limit is reached, and no Default Tab is defined, and a new tab is to be created by you or by some implied process, then the last (right-most) tab will be used for it.

Tip: If set to 1 (which means you really don't like tabs) you won't get any message prompts when another tab is supposed to be opened. Everything will just go into the current tab without asking questions. So that's a way to turn off the whole Tabs feature.

Flexible tab width

If enabled tabs will only be as wide as necessary to show the full caption.

Special benefits for named tabs (Menu View | Tab | Rename Tab):

(1) You can save valuable screen space if you choose short names for your tabs.

(2) Named tabs' captions are never cropped if space gets scarce.

Minimum / Maximum tab width in pixels

Here you can define a minimum and a maximum tab width. Defaults are 25 and 250 pixels.

The hard minimum value for both fields is internally set to 25. You cannot go below that. This is internally validated and silently corrected if necessary.

If you set the same value to both fields the tabs will be totally fixed to that size.

While it says "pixels" the value is auto-adjusted to the screen scaling. E.g. on 150% a value of 80 will be internally converted to 120 pixels.

Show icons

Tick it to show icons in the tab headers.

If turned off iconized tabs will still show the icons.

Make selected tab bold

Tick it to display the caption of the selected tab in bold.

Prompt on closing a locked tab

Tick it to prompt on closing a locked tab, a tab with a home, or a default tab.

Auto-select tabs on drag-over

Tick it to auto-select tabs when being dragged-over.

Delay before a dragged-over tab is auto-selected (in milliseconds)

The delay before a dragged over tab is automatically selected. Allowed values are from 0 to 9999.

Also auto-select tabs in the inactive pane

Untick it to keep the inactive pane's tabs passive on being dragged-over. This can be quite practical.

Add tabs via drag and drop on tab bar

Tick it to allow adding new tabs by drag-dropping folders onto the empty part of the tab bar or onto the gaps between two tabs.

Works also on the tab bar of the inactive pane, and for folders drag-dropped from other apps.

Tip: Holding down the CTRL key while dragging / dropping will open the new tab.

Tab captions

Here you can customize the tab captions. Choose among three options:

Full path: Show the whole path.

Folder only: Show only the last child folder name.

Custom: Use a template that can be defined using the Custom button. The default pattern is `<drive>: <folder>` but you can do anything that crosses your mind, e.g. `<folder> (<drive>)` or `<folder> on <drive>:` etc.

Supported placeholders:

- `<drive>`: drive letter
- `<folder>`: folder name
- `<parent>`: parent folder name
- `<path>`: full path of the tab's folder
- `<volumelabel>`: volume label of the tab's drive

- `<drivetype>`: drive type of the tab's drive.

Tip: The placeholders can also be specified per-tab using menu View | Tab | Rename Tab.

Custom Extended: Same as Custom but also applies the custom definition to Special Folders (OneDrive, Download, Documents, etc.) and Rapid Access Folders.

Show X close buttons on tabs

Choose whether/how X close buttons are displayed on the tabs.

Visual style

Choose the style of the tab headers:

(1) Windows Themes Style: Only effective when themes are enabled.

(2) XYplorer Style (Archaic): Retro Windows 95 look.

(3) XYplorer Style: Tab headers with square corners, more air, less lines. Captions disappear if there's not enough space (instead of ellipses).

(4) XYplorer Style (Rounded): Like XYplorer Style, but with rounded corners.

Show 'New Tab' button

Tick it to show the 'New Tab' button. It is positioned right of all tabs on each pane and provides a mousy way to open a new tab.

Show 'Tab List' button

Tick it to show the 'Tab List' button. It is positioned right of all tabs (but see next paragraph) on each pane and provides a mousy way to pop a list of all open tabs. That way you can select tabs that are not visible in the tab bar because of lacking space.

When space becomes scarce and tab headers start to shrink the Tab List button jumps from the right end to the left end of the tab bar, i.e. to a place that will always be visible even under the worst conditions (millions of tabs).

[Tab Bar Auto Scroll] (not an option, it just works): When the 'Tab List' button is shown it's ensured that the header of the current tab is always visible within the view port. To achieve this the tab headers are horizontally scrolled to the left internally as necessary.

Buttons position

Here you can choose where in the tab bar to show the "New Tab" and "Tab List" buttons. Factory default is "Flexible" which keeps them visible most of the time even if space becomes extremely scarce.

Auto-save tabsets on switch

Check it to automatically save the current tabset when switching to a new one. If unchecked the closed tabset will be opened in its last saved state next time it is opened.

Tabsets can revert after saving settings

Check it to allow "Tabsets | Revert to Saved" to revert to an explicitly saved version of a tabset (Tabsets | Save) even after using "Save Settings" with a modified state of the tabs. This allows you to always return to your favorite tab layout.

Going home also restores the list layout

Tick it to restore the full list layout (view mode, sort order, column layout, list style) when selecting the "Go Home" command. Otherwise going home only changes the location.

Remember tree scroll position per tab

Does what it says. Some notes:

- Applies to switching between tabs and also between panes.
- Works across sessions and also within tabsets.
- It's possible now to have the current node out of view after opening a tab which can be very good thing.
- This setting overrules "Configuration | General | Tree and List | Tree | Scroll selected folder to the top" when switching between tabs or panes.
- It just remembers the scroll position, not the whole tree. So when the tree changes between tab openings you might end up in a different place than before.

Dual Pane

Shade inactive pane

Check to shade the background of the inactive pane. The color is configurable in Configuration | Colors.

If ticked then the inactive pane will not color the sorted column anymore.

Tab key

Select the functionality of the Tab key.

Resizing the window

Select the automatic resizing of the panes when the window size is changed.

Always keep 1st pane visible

Check it to ensure that the first pane is always visible. E.g., when you are in single pane mode / 1st pane, and do "Toggle Active Pane", then dual pane mode is auto-enabled before activating the 2nd pane so that the first pane is still visible. In other words, tick this option to never see only the 2nd pane.

Sync Select

Honor relative paths

If ticked (= factory default) then in search results, Sync Select automatically honors the *relative* paths of the items (relative to the root path of the search, or of the Branch View).

Note that the settings also takes effect if only one of the panes is in Find mode.

Sync Browse

Auto-select matching items

Tick it to auto-select any matching item in the other pane when you select an item in this pane.

Auto-create any missing folders

Does what it says without further questions. When you dive into a folder that does not exist in the other location it will be created as an empty folder and then you go into it.

Shell Integration

Here you can integrate XYplorer with the Windows shell.

Default File Manager

Scope

Here you control the scope of the shell integration. You have two options:

- Only for the current user
- For all users of this computer (you need admin rights to alter the following settings!)

Note that changes in the following checkboxes will take immediate effect and modify the registry of the host system.

XYplorer in shell context menu

Tick this box to add the item "XYplorer" to the shell context menu for drives and directories. By selecting "XYplorer", XYplorer will open this drive or directory.

To remove XYplorer from the shell context menu simply untick the checkbox.

XYplorer is default file manager

Tick this box to have drives and directories opened in XYplorer by system wide default and thus replace Windows Explorer as default file manager. Untick the box to go back to Explorer at any time.

Note that this option is logically and technically subordinated to the one here above: You cannot have this without the other, but you can have the other without this.

Drag and Drop

Use standard shell drag and drop

Tick it to provide full support for certain more sophisticated drop targets.

Why would anybody turn this off? Well, when using standard shell drag and drop there are no custom mouse pointers anymore when hovering certain drop targets within XYplorer. e.g. a white-on-black plus when hovering a Catalog category.

Default action on drag and drop to same drive

Here you can change the default drag and drop behavior for intra-volume operations:

- Copy
- Move (Windows Standard)

Default action on drag and drop to different drive

Here you can change the default drag and drop behavior for cross-volume operations:

- Copy (Windows Standard)
- Move

Extended compatibility for clipboard and drag and drop

Tick it to create the "Shell IDList Array" data structure along with the files copied to the clipboard or dragged. This is only needed when pasting or dropping files into another app, and even then it usually works fine without it. However, creating this structure takes time. So, if you frequently work with large numbers of files (in clipboard or drag and drop) leave this option unticked to get a ton of speed.

Features

Here you can control some of the advanced functionality of XYplorer and disable features which you do not use or wish to see. Disabling a feature will remove the related elements from the GUI and may improve overall resource usage.

2.2 Multilingual Support

Multilingual Support

XYplorer supports interface languages other than English. A list of all available interface translations can be found on this web page:
https://www.xyplorer.com/languages.php

How to Select a Language

Use the command Select Language... in menu Help to download the language file of your choice from the server and load it into the interface in one go. No restart required.

To quickly go back to English simply tick Back to English in the same menu (this command will always stay in English). Untick Back to English to go back to the loaded language. Again, no restart required.

Tip: You may need to change the system locale

Loading a language works seamlessly when the language matches the language of the operating system, or more exactly: when the character sets match. So, there is no problem to run XYplorer in e.g. French under a German Windows. However, on a German Windows you won't be able to see, for example, Korean characters in the main menu but just question marks. In that case you should set the system locale to Korean.

Here is how to change the system locale in Windows 7: Open the Control Panel and click the Clock, Language, and Region link. In the Clock, Language, and Region panel, click on Region and Language. This opens the Region and Language. There, go to the Administrative tab. In the Language for non-Unicode programs section, you can see the currently set language. To change it, first click on Change system locale. This opens the Region and Language Settings window. Click on the Current system locale drop-down list and select the language you need to be used. When done selecting the language, click OK. You are now informed that you need to restart your computer, so that the change gets applied. Close all your open applications and documents and click on Restart now. When you log back in, the new language is applied for non-Unicode programs. IMPORTANT WARNING: The change of the language used for non-Unicode programs gets applied to ALL non-Unicode programs. Therefore, if you need to run another non-Unicode program which uses a completely different character set, you need to change the non-Unicode program language yet again.

How to Edit the Interface Language

You can easily edit a translation or create a new one yourself using the Interface Translation Tool. If you want to contribute to XYplorer and translate it into your language please see this:

https://www.xyplorer.com/languages.php#translators

Interface Translation Tool (ITT)

The command Interface Translation Tool in the Help menu is only enabled when a language is loaded. It opens a dialog where you can edit every single text bit in the interface.

Here is a short overview over the main features. A more detailed description of the Interface Translation Tool (online!) is found in the XYplorer User Forum.

- At the top of the list you can edit some of the header data. Then the translation begins with "OK", the caption of the "OK" button.
- Original and translation are WYSIWYG-previewed where applicable. Variables are resolved using "Dummy" values, number variables are resolved using a set of test numbers (0, 1, 2, 5, 21).
- Apply & Next: Pressing <enter> in the Edit box (Ctrl+Enter in a multi-line box) will apply the changes in the current item, set it to "translated" state internally, make it green in the text items list, and move on to the next item.
- The right-click menu of the main items list offers a traffic light system for tagging the items:

Translated	Green
Mostly Translated	Yellow
Partially Translated	Red
Not Yet Translated	White

- Suggestions are given if similar items have already been translated. Click on a suggestion to place its value into the Edit field.
- Assign Missing Accelerators: This command in the Tools menu automatically adds accelerators to all items meeting a set of conditions.
- A progress bar at the bottom colorfully reflects the overall translation state (red, yellow, green).
- If you save the changes they will be immediately applied to the application interface after closing the ITT.
- Upgrade Language File (Download): Downloads the latest reference file from the XYplorer server, stores it locally and upgrades the currently loaded language file to the latest state. You do this when the MultiLingualSupportVersion is not up-to-date (the ITT will tell you in red letters).
- Apart from translation you can use this tool to modify the original English, e.g. add helpful remarks to captions or tooltips. In other words, you now have a revolutionary way to personalize the application.
- Of course, you can as well work directly in the LNG file using any editor. The ITT is just a GUI for this job.

2.3 Keyboard Shortcuts and Mouse Tricks

Index

Menu Command Default Shortcuts
Other Shortcuts
Floating Preview and Full Screen Preview
Edit Boxes
Hover Box
Mouse Tricks

Menu Command Default Shortcuts

Shortcut keys allow you to quickly trigger menu commands without having to display the menu and then select an item. The most commonly used commands can be triggered using the following shortcut keys. Note that all shortcuts can be changed using the Customize Keyboard Shortcuts dialog in menu Tools.

File Menu

Enter	Open Selected Item(s)
Ctrl+Alt+Enter	Open with...
Ctrl+Enter	Open Focused Item
Ctrl+O	Open...
Ctrl+D	Copy Here with Increment
Ctrl+Shift+Alt+D	Copy Here with Current Date
Ctrl+Shift+D	Copy Here with Last Modified Date
Ctrl+S	Copy Here As...
Ctrl+Shift+C	Copy Here to New Subfolder...
Ctrl+Shift+X	Move Here to New Subfolder...
Del	Delete
Shift+Del	Delete (No Recycle Bin)
Ctrl+Del	Delete (Skip Locked)
F2	Rename
Shift+F2	Batch Rename
Alt+Enter	Properties

Ctrl+Q	Quick File View
F11	Floating Preview
Ctrl+Alt+F4	Exit without Saving
Alt+F4	Exit
Ctrl+P	Copy Item Path/Name: copy name(s) with path to clipboard (of all selected items in List)
Ctrl+Shift+P	Copy Item Name: copy name(s) (w/o path) to clipboard (of all selected items in List)

Edit Menu

Ctrl+Z	Undo
Ctrl+Shift+Z	Redo
Ctrl+Alt+Z	Action Log
Ctrl+X	Cut
Ctrl+C	Copy
Ctrl+V	Paste
Ctrl+Shift+V	Paste Here to New Subfolder...
Shift+F7	Move To...
Ctrl+F7	Copy To...
Ctrl+Shift+F7	Backup To...
Ctrl+Alt+N	New Path...
Ctrl+A	Select All: select all items currently listed
Ctrl+Shift+A	Deselect All: deselect all items currently listed
Ctrl+Shift+I	Invert Selection
Ctrl+M	Selection Filter...
Ctrl+Alt+M	Select By Selected Type(s)
Ctrl+Shift+M	Select Items...
Ctrl+Shift+Alt+M	Select All Files
Ctrl+F	Find Files: open find files tab if not visible anyway
Ctrl+Alt+F	Find Now
F3	Quick Search
Ctrl+Shift+F3	Toggle Quick Search
Shift+F3	Repeat Last Quick Search
Ctrl+F3	Show All Items In Branch

Ctrl+N	Create New Folder
Ctrl+Shift+N	Create new Text File

View Menu

F5	Refresh
Ctrl+Shift+R	Auto-Refresh
F4	Refresh Tree
Shift+F4	Refresh Current Folder
Ctrl+Shift+F4	Reset Tree: rebuilds the whole folder tree and then carries you back to the location where you were before, while closing all other open branches
Ctrl+F5	Refresh List: update the current list data, but keep any selections and scroll position.
Ctrl+Shift+F5	Reset List: update the data, scroll back to top, set focus to the first item (if any), unselect any selections.
Shift+F5	Calculate Folder Sizes: refresh current file list with folder sizes shown.
Ctrl+Shift+H	Show / Hide hidden files and folders

Ctrl+T	New Tab
Alt+Home	Go Home
Ctrl+Shift+L	Lock Location
Ctrl+W, Ctrl+F4	Close Tab
Ctrl+Shift+W	Close Other Unlocked Tabs
Ctrl+J	Set Visual Filter...
Shift+Alt+J	Filter by Selection(s)

Ctrl+Numpad Add	Autosize Columns Now (In thumbnails, this command toggles "Show Captions".)
Ctrl+Shift+Numpad Add	Grow Name Column (In thumbnails, this command toggles "Overlay Captions".)
Ctrl+Shift+Numpad Subtract	Shrink Name Column

Ctrl+Alt+R	Sort by Random Order

Go Menu

F7	Previous Location: jump to the previous tab/mode/location (useful to zap back and fore between two locations)
Shift+Alt+F7	Go to Previous Item in List: jump to the previously focused and selected item in the current file list.
Ctrl+Alt+F7	Go to Last Target
Backspace	Up
Shift+Backspace	Down
Ctrl+Backspace	Breadcrumb...
Alt+Left Arrow	Back
Alt+Right Arrow	Forward
Ctrl+H	Hotlist...
Ctrl+Shift+T	Tab List...
Ctrl+Shift+G	Go Now
Ctrl+G	Go to...: enter/paste a location to jump to
Ctrl+Shift+Alt+L	Go to Line...: enter/paste a line number to jump to
Ctrl+Shift+Alt+G	Go to Application Folder

Favorites Menu

Ctrl+B	Toggle Favorite Folder: toggle favorite status of current folder

Tools Menu

F9	Configuration: show configuration window
Shift+F9	Customize Keyboard Shortcuts...
Ctrl+F9	Customize File Associations...
Shift+Alt+F9	Customize File Icons...
Ctrl+Shift+F9	Customize Toolbar...
Ctrl+Shift+E	Toggle Age (date display format)

Panes Menu

F10	Dual Pane
Ctrl+F10	Horizontal Panes
Ctrl+Alt+F10	Toggle Active Pane
Shift+F6	Move to Other Pane
Ctrl+F6	Copy to Other Pane
Ctrl+Shift+F6	Backup to Other Pane

Window Menu

Ctrl+Shift+F12	Show Address Bar
Ctrl+F12	Show Toolbar
F8	Show Navigation Panels
Shift+F8	Show Tree
Ctrl+F8	Show Catalog
F12	Show Info Panel: toggle the Info Panel's visibility

Help Menu

F1	Contents and Index: display this help file

Other Shortcuts

Tree

[alphanum. key]	Select the next visible folder whose name starts with that letter or number
Numpad - (subtract)	Collapse selected node
Numpad + (add)	Expand selected node
Numpad / (divide)	Fully collapse selected node
Numpad * (multiply)	Fully expand selected node
Ctrl+Numpad / (divide)	Fully collapse drive
Shift+Numpad / (divide)	Optimize tree
Return	Toggle collapse/expand
Ctrl+Alt+Numpad Subtract	Go to Next Sibling (Miscellaneous | Tree | Go to Next Sibling)
Ctrl+Shift+Alt+Numpad Subtract	Go to Previous Sibling (Miscellaneous | Tree | Go to Previous Sibling)
Backspace key	Browse to parent folder
[arrow keys etc.]	All common navigation keys just like Explorer
Ctrl+Down	Select next sibling
Ctrl+Up	Select previous sibling
Alt+Return	Show Properties dialog of the focused item
Ctrl+Home	Jump to the top folder (Drive, \\Server, Desktop, MyDocuments) of the current folder
Shift+Alt+F6	Move the focused item into view
Ctrl+Alt+P	Open Command Prompt Here

Menu Key	Shell Context Menu (64-bit)

List

[alphanum. key]	Select the next file whose name starts with that letter or number
Backspace key	Browse to parent folder
Del	Delete all currently selected files (Recycle Bin)
Shift+Del	Delete all currently selected files (NO Recycle Bin)
Ctrl+Left	Jump to currently focused file in its folder (find results only)
Ctrl+Down	Move focus downwards
Ctrl+Up	Move focus upwards
Alt+Return	Show Properties dialog of the focused item (AltGr+Return to avoid beep sound)
Ctrl+Numpad Add	Autosize Columns Now
Esc	Abort calculating folder sizes or a running search
Shift+Alt+F6	Move the focused item into view
Menu Key	Shell Context Menu (64-bit)
Scroll Lock Key	If Scroll Lock is ON then the Up and Down arrow keys scroll the list keeping the current item (the focused and selected item) in the same screen position. To enable this feature you need to tweak the XYplorer.ini file. Set the MindScrollLock key to 1: `MindScrollLock=1`

Catalog

Del	Remove selected item from Catalog ("sure?"-prompt)
Shift+Del	Remove selected item from Catalog (no prompt)
Ins	Add New Item Here...
Ctrl+Ins	Duplicate Item
Shift+Alt+F6	Move the focused item into view
Ctrl+Shift+Enter	Open Selected List Item(s), using the clicked application
Ctrl+Shift+Alt+Enter	Show a Catalog item's location in the Tree, without selecting the tree folder nor displaying its contents in the file list.

Tabs

Ctrl+T	New tab
Alt+Home	Go Home
Ctrl+W, Ctrl+F4	Close current tab
Ctrl+Tab	Cycle thru tabs (forward)

Ctrl+Shift+Tab	Cycle thru tabs (backward)
Shift+Click	Close that tab

Tabbed Info Panel

Ctrl+1 - Ctrl+8	Jump to Panel tabs by number

File Find

Return	Start File Find when any control on the find files tab is focused (except the textbox on the Contents tab, where this key just creates a new paragraph).
Esc	Abort a running File Find or Folder Report process

Audio/Video Preview

Space bar	Start/Pause playing.
Shift+Space bar	Go to beginning and stop.
Left Arrow	5 sec backward
Right Arrow	5 sec forward
Shift+Left Arrow	30 sec backward
Shift+Right Arrow	30 sec forward
Ctrl+Left Arrow	1 sec backward (Video: Frame step backward)
Ctrl+Right Arrow	1 sec forward (Video: Frame step forward)

Floating Preview and Full Screen Preview

F11	Open current image file in Floating Preview
Shift+F11	Open current image file in Full Screen Preview
Shift+F11, Esc	Close Full Screen Preview
PageUp/Shift+Space/WheelUp	Previous image
PageDown/Space/WheelDown	Next image
Shift+Up, Shift+Down, Shift+Left, Shift+Right	Pan cropped image
Home/End	First/Last image
A, Numpad Multiply	Fit All
W	Fit Width
H	Fit Height
I	Zoom to Fill
O, 1, Numpad Divide	Original Size
D, 2	Double Size

Numpad Add	Zoom In
Numpad Subtract	Zoom Out
Shift+Numpad Add	Fine Zoom In
Shift+Numpad Subtract	Fine Zoom Out
K	Lock Zoom
X	Lock Zoom Position
G	Toggle Zoom
Z	Zoom to Fit
V	Top-align if Vertically Cropped
S	Show Status Bar
M	Mouse Down Blow Up
L	Rotate Left
R	Rotate Right
F	Flipped
Ctrl+Alt+Enter	Open With...
Del	Delete File
F2	Rename File
F5	Refresh
F12	Full Screen
Advanced Options	Ctrl+Right-Click
F11, Esc	Close Floating Preview

Floating Preview and Full Screen Preview: Advanced Options

N	Navigate by Click
M	Mouse Down Blow Up
Ctrl+M	Mouse Down Blow Up: Shrink to Fit
Q	High Quality Image Resampling
Ctrl+T	Cycle Transparency Background
Ctrl+W	White Border
B	Cycle Background Color
Shift+B	Cycle Background Color Backwards
S	Show Status Bar
P	Show Photo Data

T	Show Tag Bar
F8	Run Script
Ctrl+H	Show Histogram
Ctrl+R	Color Histogram
Ctrl+I	Invert
Ctrl+G	Grayscale
C	Copy Original
Ctrl+C	Copy Preview

Configuration Dialog

F3	Jump to Setting...
F7	Jump back and forth between this and the previous page
Ctrl+Right-Click	On most settings: Display a menu with variations of their names that can be copied to the clipboard.

Customize Toolbar

DEL	Remove selected item
INS	Insert selected item
S	Add separator
Backspace	Remove the item right before the currently selected item
F3	Focus filter box (if no filter box is focused, else toggle filter)
F6	Switch between the two lists.
Alt+V	Focus left list.
Alt+B	Focus right list.

Edit Boxes (incl. Address Bar and Live Filter Box)

Ctrl+A	Select all.
Triple-Click	Select all.
Ctrl+F	Opens Find Text dialog.
F2	Select toggle all/nothing.
F3	Find next occurrence of the search string specified in Find Text dialog. Opens Find Text dialog if there wasn't any previous search in that edit box.
Shift+F3	Find previous occurrence of the search string specified in Find Text dialog.

	Opens Find Text dialog if there wasn't any previous search in that edit box.
Up	If selection: caret to start of selection and unselect else: caret to start of text.
Down	If selection: caret to end of selection and unselect else: caret to end of text.
F5,Ctrl+Alt+U	Convert selected text to Title Case (converts the first letter of every word in string to upper case).
Shift+F5, Ctrl+Shift+U	Convert selected text to UPPER CASE.
Ctrl+F5, Ctrl+U	Convert selected text to lower case.
Ctrl+Backspace	Delete the previous word.

Live Filter Box

ESC	Remove the current live filter.
Ctrl+Alt+F3	Toggle live filter.
Ctrl+Z	Undo.
Ctrl+Down	Drop the current match list.

Dropdown Boxes (incl. Live Filter Box)

Alt+Down, Shift+Down	Drop the list.
Alt+Up, Shift+Up	Undrop the list.
F4	Toggle drop/undrop.

Small Lists

Ctrl+Drag	Create a clone of the dragged item in a new position. Works only in lists where you can add new items, e.g. in Color Filters.

Hover Box (the keys work only while the hover box is displayed)

F1	Hover Box help
B	Cycle Background color
C	Copy image to clipboard
D	Reset to Default size (512 x 512)
F	Cycle image Frame
G	Toggle Grayscale
H	Toggle real/special patH
I	Toggle Icon display

K	Toggle Key scrolling
L	Toggle scroLling
O	Cycle sort Order
P	Toggle show Photo data
R	Toggle Resizing
S	Cycle Status display
T	Cycle Transparency background
U	Toggle roUnded corners
W	Toggle Word wrap
X	Trigger hotkeys with/without CTRL
Z	Zoom to fit

Numpad Add (or Wheel Up)	Increase size by 8 pixels
Numpad Subtract (or Wheel Down)	Decrease size by 8 pixels

Hold down SHIFT to make it 64 pixels

Hold down CTRL to make it 1 pixel

Space	Select item
Ctrl+Space	Toggle selection

Other

ESC	All time-consuming processes (finding files, calculating folder sizes...) can always be stopped by pressing ESC.
F6	Cycle focus in this sequence: Address Bar > Tree > List > Catalog.
Alt+D	Focus Address Bar.
Ctrl+L	Focus Address Bar.
Ctrl+Up/Down	In lists with shiftable positions: shift selected item up/down.
Ctrl+Numpad Div	Fully collapse current drive. The drive root will automatically be selected.
Hold Shift	Open the location you are going to in a new tab. Exception: The location change is triggered by a script.
Ctrl+0	Reset Zoom (Miscellaneous | Layout | Reset Zoom)
Ctrl+Shift+Alt+F12	Toggle Main Menu (Miscellaneous | General Functions | Toggle Main Menu)

Mouse Tricks

Various Places

Wheel	If there's a scrollbar: 3 lines up/down (3 is the Windows default; it can be changed).
Shift+Wheel	If there's a scrollbar: PageUp/Down (unless scrolling horizontally is enabled and possible).
Ctrl+Wheel	Over Status Bar: Zoom font size of Buttons and Labels. Elsewhere: Zoom font size of main controls.

Tree

Ctrl+Dbl-Click	On the white space: scroll focused item into view.
Ctrl+Shift+Dbl-Click	On the white space: Scroll to Top.
Right-Click	On non-folder area: pops up the Favorites context menu.
Ctrl+Click	Open in new background tab.
Ctrl+Shift+Click	Open in new tab (and focus it).
Shift+Click	Open in new tab (and focus it).
Shift+Right-Click	On empty: open the submenu Tools | Customize Tree as a popup menu. Unless Custom Event Actions | Right-click on white in tree is set to "Run script".
Alt+Click	Open in inactive pane.
Shift+Alt+Click	Open in a new tab in the inactive pane.
Middle-Click	Open in new tab (equivalent to Shift+Click).
Shift+Wheel	Scroll horizontally (if possible).
Ctrl+Shift+Wheel	Increase/Decrease row height (line spacing).
Shift+Alt+Wheel	Increase/Decrease indent.
Click	On expansion icon: Collapse/Expand selected node.
Ctrl+Click	On expansion icon: Fully Collapse/Expand selected node.

Catalog

Ctrl+Click	Open in new background tab.
Shift+Click	Open in new tab (and focus it).
Ctrl+Alt+Click	Open in this tab (even if it is locked).
Middle-Click	Open in new tab (equivalent to Shift+Click). Needs Custom Event Action "Middle-click on folder" set to "Open in new foreground tab".
Ctrl+Shift+Click	Open Selected List Item(s), using the clicked application.
Ctrl+Shift+Alt+Enter	Show a Catalog item's location in the Tree, without selecting the tree folder

	nor displaying its contents in the file list.
Drag	Move dragged item to new position.
Drag holding Ctrl	Copy dragged item to new position.
Ctrl+Shift+Wheel	Increase/Decrease row height (line spacing).
Click	Left-most 8 pixels: select item but do not trigger action.

List

Ctrl+Left-Click	In Details view on any cell in the row: Toggles the selection of items even if Full Row Select is turned off. Quite useful if the Name column is currently scrolled left out of view.
Ctrl+Hover	In Details view on any cell in the row: Shows a tooltip with the name of the item. Quite useful if the Name column is currently scrolled left out of view.
Dbl-Click	On the line numbers column header ("#"), or on the area to the right of all column headers: Autosize Columns Now.
Ctrl+Dbl-Click	On the white space: scroll focused item into view.
Right-Click	On the line numbers column header ("#"): opens the Sort By menu. On the column headers: opens the Show Columns context menu.
Ctrl+Right-Click	On the line numbers column header ("#"): opens the Column Layouts menu.
Shift+Right-Click	On the line numbers column header ("#"): opens the Views menu
Middle-Click	Open a folder in a new tab (equivalent to Shift+Dbl-Click). Needs Custom Event Action "Middle-click on folder" set to "Open in new foreground tab".
Shift+Wheel	Scroll horizontally (if possible).
Ctrl+Shift+Wheel	Increase/Decrease row height (line spacing).
Ctrl+Shift+Dbl-Click	On a folder: browse the folder in the inactive pane. On a file in Search Results: the containing folder is opened in the inactive pane and the file is selected. On the white space: Scroll to Top.
Shift+Right-Click	On empty: open the submenu Tools | Customize List as a popup menu. Unless Custom Event Actions | Right-click on white in file list is set to "Run script".
Ctrl+Wheel	Scroll through the list views (Details through Details with Thumbnails #1) of the active pane. Needs to be enabled here: Configuration | General | Menus, Mouse, Usability | Mouse | Use Ctrl+mouse wheel to scroll through the list views.

Toolbar

Ctrl+Shift+Wheel	Toolbar zoom.

If button captions are shown:

Ctrl+Wheel on upper half	Button texts font size.

Ctrl+Wheel on lower half	Button captions font size.

If button captions are not shown:

Ctrl+Wheel	Button texts font size.

Address Bar

Wheel	Scroll through the dropdown list items when it's collapsed.
Shift+WheelDown	Expand the list.
Shift+WheelUp	Collapse the list.
Shift+Alt+Wheel	Change the width of the Address Bar (if it shares a row with the Toolbar).

Breadcrumb Bars

Ctrl+Shift+Wheel	Increase/Decrease bar height.

Status Bar

Ctrl+Wheel	Status Bar font size.
Shift+Alt+Wheel	Section width.

Live Filter Box

Shift+Alt+Wheel	Change the width of the box.

Dropdown Boxes (incl. Live Filter Box)

Wheel Up/Down	Browse through the collapsed dropdown list.

List showing thumbnails

Left-Click on thumb	Mouse Down Blow Up, while mouse button is down. Note: hold the mouse down longer than 150ms. Quick clicks will simply select the item.
Right-Click on thumb	Mouse Down Blow Up, stay up until you hit any key or click it again.

Drag and Drop

Ctrl+Left-Drag	Force copying the dragged items.
Shift+Left-Drag	Force moving the dragged items.
Alt+Left-Drag	Create shortcuts of the dragged items.
Ctrl+Shift+Left-Drag	Fake a right-drag (and pop the drag and drop context menu). Useful when a real right-drag isn't possible.
Right-Drag	Pop the drag and drop context menu.

Tabs

Wheel over Tab heads	Scroll through the tabs
Dbl-Click	Close tab (background tabs are shortly selected, then closed).
Middle-Click	Close tab (background tabs are closed without selecting them first).

Info Panel

Dbl-Click on Splitter	Toggle Info Panel min/max size.
Ctrl+1, Ctrl+2, etc.	Select Tab 1, 2 etc.
Ctrl+Wheel	Buttons and Labels font size.

Find Files Tab

Ctrl+Click Tab heads	Toggle search filter On/Off
Right+Click Tab heads	Toggle search filter On/Off

Copying Information to Clipboard

As rule of thumb, any text or icon you see anywhere on the info panel or the status bar can be copied to the clipboard by double-clicking it. Some special cases are extensions to this rule:

Dbl-Click on the Objects-section on the status bar (the left-most section) copies summary information on the current folder, eg:

D:\Download\, 145 object(s), Total: 13.29MB (13,935,798 bytes)

Dbl-Click on a Key label on the Properties and Version tabs copies the key and the value. eg:

Created: Wednesday, January 12, 2000 20:57:21

Dbl-Click on a Value label on the Properties and Version tabs copies just the value. eg:

Wednesday, January 12, 2000 20:57:21

Image Preview

Mouse actions directly on the preview image:

Left Mouse Down	Mouse Down Blow Up: Popup image in original size (aligned to the bottom-left). If image is larger than the main window move the mouse while holding it down to scroll the image in any direction (or enable "Shrink to fit" in Configuration | Preview).
Ctrl+Mouse Down	Toggle zoom/original size.
Right Mouse Click	Pops up the image preview context menu.

Audio/Video Preview

Mouse actions directly on the progress bar:

Left Mouse Down	Go to mouse position.
Right Mouse Down	Start/Pause playing.
Shift+Left Mouse Down	Go to beginning.
Shift+Right Mouse Down	Go to beginning and stop.
Ctrl+Left Mouse Down	Set start of sub-loop.
Ctrl+Right Mouse Down	Set end of sub-loop.

Ctrl+Alt+Mouse Down	Reset sub-loop.

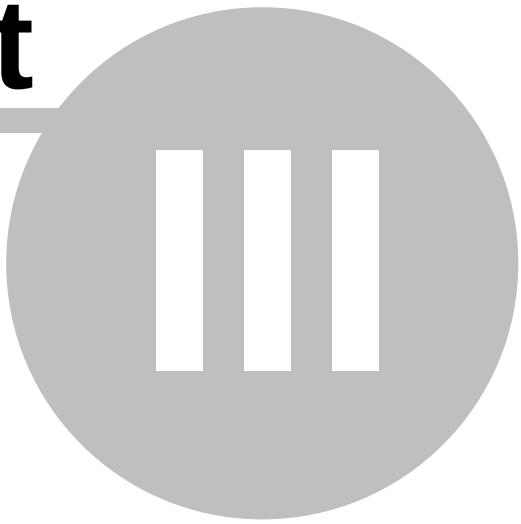
Floating Preview and Full Screen Preview

Right Mouse Click	Show context menu.
Ctrl+Left Mouse Drag	Pan a cropped image.
Right Mouse Drag	Pan a cropped image.
Middle-Click	If Navigate by Click is enabled: Close preview. If Navigate by Click is disabled: Middle-Click goes to the next, and Shift+Middle-Click goes to the previous image.
MouseWheel Up	Previous Image.
MouseWheel Down	Next Image.
Ctrl+MouseWheel Up	Zoom In.
Ctrl+MouseWheel Down	Zoom Out.
Ctrl+Shift+MouseWheel Up	Fine Zoom In.
Ctrl+Shift+MouseWheel Down	Fine Zoom Out.

Main Menu

Hold Shift	While selecting a view from menu View | Views | ...: Apply the view to all tabs.

Part



Main Menu

3 Main Menu

3.1 File

To Clipboard (Submenu)

Copy various file information to the clipboard.

Section 1:

The following commands are applied to all selected list items (files or folders) no matter where the focus is. Exception: The focus is in the folder tree, or there are no selections in the file list: then the commands apply to the current path.

If more than one item is selected, each item in the copy string is terminated by a line feed (thus generating a vertical list). If only one item is selected, no line feed is appended to it.

Item Path/Name(s)

Copy the selected item name(s) with path, eg:

C:\Program Files\XYplorer\XYplorer.exe

Shortcut: Ctrl+P

Item Name(s)

Copy the selected item name(s) (no path), eg:

XYplorer.exe

If the focus is in the list but no items are selected the current folder name is copied to the clipboard.

Shortcut: Ctrl+Shift+P

Item Path(s)

Copy the selected item path(s) (no name), eg:

C:\Program Files\XYplorer\

Shortcut: Ctrl+Shift+Alt+P

Item Base(s)

Copy the selected item base(s) (no path, no extension), eg:

XYplorer

Item Short Path/Name(s)

Copy the selected item short (DOS 8.3 format) name(s) with path, eg:

C:\DOKUME~1\ALLUSE~1\STARTM~1\PROGRA~1.LNK

Item UNC Path/Name(s)

Copies the full path/name of all selected Tree or List items, using the UNC path for items located on a mapped drive.

If the item is on a normal local (non-mapped) path then the UNC version of the path is returned.

Name(s), Bytes, Modified[, Version]

Copy item name (no path) and basic info, eg:

XYplorer.exe, 6.029.312 bytes, 2014-01-02 09:35:12, ver 13.50.0019

Name(s), Bytes, Modified[, Version], Path

Copy item name and basic info including path, eg:

XYplorer.exe, 6.029.312 bytes, 2014-01-02 09:35:12, ver 13.50.0019, E:\XY\XYplorer\

Name(s), Bytes, Modified[, Version], Path, MD5

Copy item name and basic info including path and MD5 hash, eg:

XYplorer.exe, 6.029.312 bytes, 2014-01-02 09:35:12, ver 13.50.0019, E:\XY\XYplorer\, 5ecc1e347db8e789fd97d5f59f9aa491

Section 2:

The following commands are applied to only the focused item (the file or folder marked by the dotted focus line).

Compact File Info

Eg: XYplorer.exe, 1.539.072 bytes, 16.01.2006 09:56:06, ver 4.40.0047

This is equivalent to Name, bytes, modified[, version] (see above)

Extended File Info

Example

Large Icon

Copy bitmap of the large icon (32x32) of the focused item. Transparent areas are set to window background color (typically white).

Small Icon

Copy bitmap of the small icon (16x16) of the focused item. Transparent areas are set to window background color (typically white).

Section 3:

Text

Copies the textual contents of the currently selected ANSI or Unicode (UTF16 LE) text file to the clipboard. It also works on complex file types like DOC, DOCX, PDF, or HTML, in which case the text is extracted using IFilters (matching IFilters need to be installed).

Tip: The copied text can then be pasted to a new file using Edit | Paste Special | Paste Text Into New File (Ctrl+Alt+V).

Image

If the current file is an image file this command will copy the image directly to the clipboard.

Section 4:

Selected List Rows

Copy all selected rows of the List as currently shown in the list

All List Rows

Copy all rows of the List as currently shown in the list

Rename Special (Submenu)

The following rename routines apply to all currently selected items in the file list, or to the current folder if the folder tree has the focus. It is recommended that you use the built-in Preview function before actually applying the rename.

Batch Rename...

Provides a simple way to rename a bunch of files using a common template consisting of a fixed part and an auto-incremented counter. You can define the starting value and the number of leading zeroes. Examples:

```
File<#1>.ext    =  File1.ext, File2.ext, ... File74.ext ...
File<#00>.ext   =  File00.ext, File01.ext, ... File235.ext ...
File<#055>.ext  =  File055.ext, File056.ext, ... File9574.ext ...
```

You can as well add strings to the beginning and/or the end of the original name (extension excluded). In the template the original name is referred to by "*" (asterisk) -- the self-referential wildcard (see also below). For example, the template "2006-*-(backup)" will trigger the following renames:

```
Convdsn.exe   -> 2006-Convdsn-(backup).exe
Drvspace.vxd  -> 2006-Drvspace-(backup).vxd
Comdlg32.ocx  -> 2006-Comdlg32-(backup).ocx
```

You can as well datestamp the selected files. <date will used the current date, <datem/c/a will use each item's modified/created/accessed date:

```
*_<date yyyy-mm-dd_hh-nn-ss>  -> OldName_2006-08-16_15-49-03.txt
<date yyyy>_*   -> 2006_OldName.txt
```

```
<datem yyyy>_*  -> 2004_OldName.txt (OldName.txt was last modified 2004)
<datec yyyy>_*  -> 2002_OldName.txt (OldName.txt was created 2002)
<datea yyyy>_*  -> 2006_OldName.txt (OldName.txt was last accessed 2006)
```

Note that fractions of a second (up to 7 decimal digits) are supported. The placeholder is "f". The following example will use the milliseconds of the current time for the new filename:

```
<date yyyymmdd_hhnnss.fff>_*   -> 20130610_103214.106_test.txt
```

The files are renamed one after the other in their current order. In case of a name collision you get a message with the option to continue or abort the batch job. If only one list item is selected you can force the Batch Rename interface by pressing Shift+F2.

Move on Rename (Batch Move)

Batch Rename supports move on rename. This feature allows you to automatically move masses of files by certain criteria to certain folders, for example by their modified date using the following Batch Rename pattern:

```
<datem yyyy-mm-dd>\*              = by day
<datem yyyy-mm>\*                 = by month
<datem yyyy>\*                    = by year
```

A more popular use case would probably be to automatically move photos to folders based on each file's embedded exif date taken:

```
<dateexif yyyy-mm-dd>\*           = by day
<dateexif yyyy-mm>\*              = by month
<dateexif yyyy>\*                 = by year
```

You can even state absolute paths in the patterns. Examples:

```
E:\<datem yyyy>\*
F:\Photos\<dateexif yyyy>\<dateexif mm>\*
```

And, of course, using the powerful <prop> variable you can achieve quite some cool automation here, e.g. distribute photos to folders according to their Aspect Ratio, their Film Simulation Mode, or many many other properties.

Notes:

- "Configuration | Sort and Rename | Rename | Allow move on rename" has to be is ticked to enable this functionality.
- The necessary subfolders are automatically created under the parent folders of the files.
- Works cross-volume on Win 8.1 and later. In earlier Windows it only works within the same drive.
- Not supported on Portable Devices.
- Nice plus: The Rename Preview lets you see the target paths before anything actually happens.

Self-referential wildcards

The wildcard (*) is replaced by the original file base (file title excluding extension). Additionally, there's a 2nd self-referential wildcard (?) that stands for the original file extension. Examples:

```
Old     Pattern                 New
```

```
a.jpg   ?                       jpg.jpg
a.jpg   *.?.bak /e              a.jpg.bak
a.jpg   ?-<#001>                jpg-001.jpg
a.jpg   ?-<#aa>                 jpg-aa.jpg
a.jpg   *.?.<date yymmdd>       a.jpg.081023.jpg
a.jpg   *.?.<date yymmdd> /e    a.jpg.081023
```

You can as well use other variables in the pattern, e.g. the property variable. This appends the dimensions to all selected image files:

```
*_<prop #image.dimensions>
```

Special variables

The variable <folder> is resolved to each item's parent folder name. For example, this pattern will prefix the item's parent folder name to each renamed item:

```
<folder>-*
```

The variable <name> can be used for otherwise difficult self-referencing batch renaming tasks.

```
Syntax: <name [length=0] [start=1] [part=b]>
        length: Number of characters; 0 = all; negative numbers for portions from the right
end.
        start:  Start position of returned string.
                1 = 1st position; negative numbers for the position from the right end.
                Can be *xxx or xxx* for the part left or right of xxx.
        part:   Part to return; a = all, b [default] = base, e = ext, n = name (base.ext), p
= path.
```

For example, if you have a bunch of files named "yyyy-mm-dd - xxx.xxx" this pattern will move them into new folders named by the first four characters of the file's base name (yyyy):

```
<name 4>\*
```

This would use the last four characters of each file's base name:

```
<name -4>\*
```

This would characters 2 and 3:

```
<name 2 2>\*
```

This would use the last two characters of each file name including the extension:

```
<name 0 -2 n>\*
```

Note that "Configuration | General | Sort and Rename | Rename | Allow move on rename" has to be enabled for the above examples to work.

You can also move parts of the filename, e.g. move the first four characters to the end:

```
<name 0 5><name 4>
```

For example, this will move files to a subfolder named by the part of the filename to the left of the first "-":

```
<name 0 *->\*
```

Batch Rename Switches

/e for Extensions: Note that the original file extensions remain unchanged. However there's an easy way to actually change the extensions: append /e to the pattern! The /e switch simply means: drop the original extension. Then it depends on the pattern whether a new extension is supplied or not. Examples:

Original name:	test.txt
*-<datem yyyymmdd>	test-20070607.txt
*-<datem yyyymmdd>.htm	test-20070607.htm.txt
*-<datem yyyymmdd>/e	test-20070607
*-<datem yyyymmdd>.htm/e	test-20070607.htm
New<#01>	New01.txt, New02.txt, New03.txt ...
New-<#aa>	New-aa.txt, New-ab.txt, New-ac.txt ...
New<#04>/e	New04, New05, New06 ...
New<#01>.csv	New01.csv.txt, New02.csv.txt, New03.csv.txt ...
New<#01>.csv/e	New01.csv, New02.csv, New03.csv ...
ping.png/e	ping.png

/s for skip: If you want to smartly avoid any collision that might arise from using auto-increments: append /s to the pattern! Examples:

Old	Pattern	New	If b-3.jpg exists
a.jpg	b-<#3>	b-3.jpg	collision!
a.jpg	b-<#3> /s	b-4.jpg	collision auto-avoided
a.jpg	b-<#3>.png /s	b-3.png.jpg	no collision anyway
a.jpg	b-<#3>.jpg /e	b-3.jpg	collision!
a.jpg	b-<#3>.png /es	b-3.png	no collision anyway
a.jpg	b-<#3>.png /se	b-3.png	no collision anyway

The last two examples merely illustrate that the order of switches is irrelevant.

/i for "Increment on collision": If you want to automatically avoid collisions with existing files, append a "/i" to your pattern.

For example, you have these files, both modified on 2008-07-29:

hilite.png

hilite-20080729.png

If you now apply the pattern *-<datem yyyymmdd> to hilite.png, renaming is not possible because of collision with the already existing hilite-20080729.png. However, if you use the pattern *-<datem yyyymmdd>/i the file will be renamed to hilite-20080729-01.png (or hilite-20080729-02.png if hilite-20080729-01.png exists, etc). The format of the incremental suffix is determined by the setting of Incremental suffix in Configuration | Templates.

Note that some patterns avoid collisions anyway by default, so appending the switch is not necessary here (but harmless):

- patterns with explicit increment, e.g. "New-<#01>"

- simple patterns, e.g. "New"

/u for "Uppercase the first character": Leaves all other characters unchanged. Example:

```
* /u
```

/o for "Overwrite on Collision": If you want to automatically overwrite same-named existing files without further questions, append "/o" to your pattern! Note that nothing can restore file once they have been overwritten. No return ticket.

/r for "Process in reverse order": This switch is especially interesting when automatic increments are added when renaming multiple items at the same time. It allows you to number the items from the bottom up which is nice and natural when the list is sorted by Modified descending. Example:

```
Larry /r
```

Note that pattern switches can be combined in any order: If you need to change the extension (/e) AND want to increment on collision (/i) then simply append /ei or /ie.

Shorten Filenames

Here is a simple way to cut down a batch of filenames. You can make them shorter from both ends by stating a max length and a start (left offset) by way of the switch `/cut [start,]length[,keep]`. Pass a negative length value to cut off a certain number of characters from the right end of the base name. You can also keep a certain number of characters at the right end of the base.

Examples:

```
/cut 10          cut everything right of the first 10 characters
/cut 2,10        cut everything left of the 2nd character, then cut everything
                    right of the remaining 10 characters
/cut 2,0         cut everything left of the 2nd character, take all the rest
/cut 2,          (same as above)
/cut -5          cut the last 5 characters (of the base)
/cut 1,10,4      keep the last 4 characters, take the first 10 characters
/cut 3,10,4      keep the last 4 characters, take 10 characters beginning at pos 3
/cut 1,-1000,4   keep the last 4 characters
/cut 1000,0,4    keep the last 4 characters
```

- The space right after /cut can be omitted.
- This switch cannot be combined with any other switch or pattern. So you just use this switch, nothing else in the name field.
- Only the base of the filenames is affected, never the extension.
- You must state "start" as well if you want to use "keep".

Cut and Add

The `/cut` switch can be combined with serial numbering. This is especially useful when you want to replace old numbers by new ones. For example, you have these files:

```
DSC00032 Banana.jpg
DSC00047 Blue.jpg
DSC00123 Passion.jpg
```

Now applying this pattern will cut the first 9 characters (start at position 10, keep all the rest), then prefix the new number:

```
DSC<#00001> */cut 10,0
```

Result:

```
DSC00001 Banana.jpg
DSC00002 Blue.jpg
DSC00003 Passion jpg
```

Prefix-Overwrite and Suffix-Overwrite

For this both | and * must be present in the pattern. The |-character is used to mark the cut, and the * character stands for the original filename (which is cut and affixed).

1) Prefix-Overwrite: For example, the pattern `NEW-|*` means: Overwrite the beginning of the base name with "NEW-". Result when applied to above files:

```
NEW-0001 Banana.jpg
NEW-0002 Blue.jpg
NEW-0003 Passion.jpg
```

2) Suffix-Overwrite: For example, the pattern `*|-NEW` means: Overwrite the end of the base name with "-NEW". Result when applied to above files:

```
DSC00001 Ba-NEW.jpg
DSC00002 -NEW.jpg
DSC00003 Pas-NEW.jpg
```

RegExp Rename...

Allows you to rename selected filename(s) using Regular Expressions. Format:

```
RegExpPattern > ReplaceWith
```

Mind the spaces around the separator (>).

The default operation is case-insensitive (a=A). To search case-sensitively add a back-slash to the end of ReplaceWith:

Examples:

```
.html$ > .htm    = change extensions .html to .htm
.html$ > .htm\   = change extensions .html (but not eg .HTML) to .htm
```

Move on rename

RegExp Rename and Replace supports move on rename. Example:

```
Pattern:  ^(\d+)-(\d+)- > $1\$2\$1-$2-
From:     E:\Test\2015-12-21_test.png
To:       E:\Test\2015\12\2015-12-21_test.png
```

Note Renaming includes file extensions.

Tip For details on XYplorer's variety of regular expressions, search the web for: vbscript regular expressions.

Search and Replace...

Here you can enter a list of characters to be removed or replaced from selected filename(s). You can

- define no replace entity, in which case all characters in the list will be removed, or
- define a common replace string (can be longer than one character) at the end of the list after '>', or
- define a replace list at the end of the list after '>>' (remove-list and replace-list must have same length), or
- define lists with units larger than one character before and after '>>', or
- define a string to be replaced by another string, both strings separated by '/'.

Switches:

Matching by default is case-insensitive (a=A) and extensions are excluded from the replacement. This can be changed by appending switches (format: one space, one slash, one or more letters):

```
/c    = case-sensitive
/e    = include extensions
/ce   = case-sensitive, include extensions
```

Examples:

[]{};,!	= Remove all these characters
[]{};,!>_	= Replace each of these characters with _
[]{};,!>off	= Replace each of these characters with 'off'
äöü>>aou	= Replace all ä with a, ö with o, ü with u
Ä|Ö|Ü|ä|ö|ü>>Ae|Oe|Ue|ae|oe|ue	= Replace all Ä with Ae, Ö with Oe, etc.
before/now	= Replace all 'before' (or 'beFOrE') with 'now' (case-insensitive: a=A)
before/now /c	= Replace all 'before' with 'now' (case-sensitive)
txt/text /e	= Replace all 'txt' with 'text' (include extensions)
before/	= Remove all 'before'

Note: The original file extensions are preserved (unless you append switch /e).

Move on rename

Search and Replace supports move on rename. Example:

```
Pattern:  Delta/Omega\Plus
From:     E:\Test\b\Saturn.exe\Delta2.jpg
To:       E:\Test\b\Saturn.exe\Omega\Plus2.jpg
```

Note that first the replacement is made (in the example: "Delta" is replace by "Omega\Plus"), then the file is moved according to the resulting target name.

Keep Particular Characters...

Here you can enter a list of characters to be kept in selected filename(s). The default list is:

abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 _-.()

You can

(1) define no replace entity, in which case all other characters will be removed, or

(2) define a common replace string (can be longer than one character) at the end of the list after '>'.

Examples:

abc = Remove all characters that are not a, b, or c

abc>_ = Replace all characters that are not a, b, or c, with '_'

Note The original file extensions are preserved.

Set Extension...

Sets a new extension to all selected files (replacing the old one, or adding a new one if there was none before). Enter nothing to remove any extensions.

Folder names are skipped.

Examples:

txt	On files with extension: Replace the current extension with ".txt".
	On files without extension: Append ".txt".
	On folders: Do nothing.
<empty>	On files with extension: Remove extension.
	On files without extension: Do nothing.
	On folders: Do nothing.

You can as well add or remove an extension regardless of whether an extension already exists or

whether the item is a file or a folder. To invoke this functionality prefix a dot to the entered pattern:

.txt Append ".txt".

. Remove any extension.

Edit Item Names...

Here you can edit the names of the selected List items in a multiline text box. Depending on your rename tasks this offers an elegant way to handle them.

You might even copy the names to a full blown editor, do some fancy things with them, and then paste them back to the dialog and perform the rename.

Notes:

- Empty lines or missing lines at the end of the list will be taken as "don't rename this item"; excessive lines at the end of the list will be ignored.
- The Rename Preview (see below) is unconditionally invoked before the actual rename is performed.
- Move on rename is supported here, i.e. you can provide relative or absolute paths in the edited names.

Capitalization

A* A*.aaa Capitalizes each word in the filename(s), but leaves any upper case letters within the words untouched (so-called Camel Case is preserved)). The extension is put to lower case.

Aaa Aa.aaa Capitalizes each word in the filename(s), sets all non-first letters to lower case (also known as Title Case). The extension is put to lower case.

Note In Title Case there are some exceptions in the English language, words that will not receive a capital first letter unless they are in the first position of the title: a;an;and;at;by;for;from;in;of;on;or;out;the; to. The characters . : ? ! - when followed by a space are recognized as title-separators.

aaa aa.aaa All lower case.

AAA AA.AAA All upper case.

*.aaa Set extension to lower case.

*.AAA Set extension to upper case.

Examples:

```
XYplorer is the best.ini ->
          A* A*.aaa: XYplorer Is the Best.ini
         Aaa Aa.aaa: Xyplorer Is the Best.ini
         aaa aa.aaa: xyplorer is the best.ini
```

```
AAA AA.AAA: XYPLORER IS THE BEST.INI
     *.aaa: XYplorer is the best.ini
     *.AAA: XYplorer is the best.INI
```

Spaces to _ and _ to Spaces: Replaces spaces in long filenames by underscore (and vice versa).

Remove Diacritics: Strips any diacritical marks from the letters in the selected item names.

Convert to ASCII: Converts the filename to 7-bit ASCII using a various smart and language-specific conversions. In German e.g. it converts ä to ae, Ö to Oe, ß to ss, etc. Cyrillic characters are romanized. Non-convertible characters are replaced by "_" or whatever is defined in Configuration | Templates | Character to replace invalid characters in dropped messages.

UrlEscape: Converts " " to "%20" etc."%20" is often found in the names of downloaded files, it is the escape sequence for a space used in URLs. Apart from this there are other escapes for unsafe characters in URLs, eg "%FC" for "ü". Unsafe characters are those characters that may be altered during transport across the internet.

UrlUnescape: Converts "%20" to " " etc. (reversal of UrlEscape).

Unicode to UTF-8: Converts non-latin characters to ANSI characters using the UTF-8 encoding scheme.

UTF-8 to Unicode: The same backwards.

Mp3 Special (Submenu): Here you find some convenient renaming schemes for MP3 files ("ID3 Tag to Filename") and also schemes to alter the ID3 tag from the file name ("Filename To ID3 Tag"). The placeholders Artist, Year, Album, Track, and Title are supported. The factory default presets can be altered by editing the INI file.

Preview All: Check to show a preview for each of the Rename Special operations, including detection of possible conflicts.

Rename Preview

The Rename Preview shows the item names resulting from your rename pattern. It points out potential conflicts:

- Dupe: Duplicates among the new names.
- Exists: An item of the new name exists already.

It also tells you which items remain unchanged.

The footer shows some statistics for the current job, including the hidden state for unchanged items. If there are conflicts, the entire footer is displayed in red.

The context menu of each list contains the Copy Conflicted Items command. Copies the items highlighted in red, as they appear in the right-clicked list, to the clipboard.

Tip: You can hide all unchanged items. Right-click any of the lists in the dialog (or press the Context

Menu Key) to toggle the setting Hide Unchanged Items (F8) via a small popup menu. The setting is remembered across sessions.

Move/Copy/Backup To (Submenu)

Move To / Copy To / Backup To

Recent Locations...: Move/Copy/Backup the currently selected items to a recently visited location.

Favorite Folders...: Move/Copy/Backup the currently selected items to a predefined favorite folder.

Tabs: Move/Copy/Backup the currently selected items to any of the currently existing tabs (if they are suitable as destinations).

Move To / Copy To / Backup To...: Open the internal Move/Copy/Backup To dialog to each of the submenus. Duplicates the commands in menu Edit for the sake of usability.

Browse for Move Destination...: You can target folders not listed by selecting the Browse... command; they'll be added to the top of the list. Move To/Copy To/Backup To all share the same list.

Tip: Double-clicking a destination in the destination list triggers the default action (Move, Copy, or Backup) with that destination.

Other Menu Commands

Open Selected Item(s)

Open all currently selected files at once.

Note: All items are opened in one call (in a single instance of the opening application) if they are of the same type and hence share the same associated application, be it XYplorer-associated (CFA), or system-associated.

Shortcut: Enter

Open with...

Pops up a menu showing all applications associated (Custom File Associations) with the current (focused+selected)* List item. Select one of them to open all selected List item(s) with it.

* If none of the selected items is focused, then the first selected item (from top) is used to match the CFA patterns against.

See also Portable Openwith Menu (POM).

Shortcut: Ctrl+Alt+Enter

Open with Arguments...

Open the selected list item(s) with command line arguments of your choice. The last used arguments are saved between sessions.

Special Case: If a single non-executable file is selected then this function opens it with the associated application, using the entered arguments plus the selected file itself as arguments.

Example:

```
Selected file:    E:\Test\b\Moon.txt
Entered argument: /r
```

Now the following command is run if EmEditor is associated with TXT files:

```
C:\Program Files\EmEditor\EMEDITOR.EXE /r "E:\Test\b\Moon.txt"
```

General form:

```
associated_application arguments "selected_file"
```

Notes:

- Of course, the entered argument must match the rules of the associated application, so you have to know what you do.
- Custom File Associations are honored if "Open Items by First 'Open with...' Match" is ticked. If the default Custom File Associations is a script then the OS-associated application is taken.

Open Focused Item

Open the currently focused file.

Shortcut: Ctrl+Enter

Open...

Here you can enter, paste, or select an item (file, folder, drive) to open, without the need to go there first. Recently opened items are stored in a MRU list that is saved between sessions. The dialog is preset to the last opened item.

In the input dialog you can choose between "Open" which will open the item using the default application (taking XY's portable associations into account), and "Open with..." (Ctrl+Enter) which will pop the Portable Openwith Menu (POM) for the item you have entered.

The POM features the extra commands "Go to Item" and "Copy Item". Note that both commands (like all other commands in the menu) work on the current contents of the edit field. So you can type or paste any existing item name, and go to it or copy it (the item, not just the name) to the clipboard. Nice!

What is added to the MRU: Single files (not folders) opened thru the List by Enter/Dbl-Click or via the POM are automatically added to the MRU (and thus made available in the new Open dialog. Files opened via scripts, or User-Defined Commands, or Catalog, or shell context menu, or within a group of files (i.e. non-single), or other means are not auto-added.

You can edit the MRU in Tools | List Management | Recently Opened Files.

Shortcut: Ctrl+O

Duplicate (submenu)

Copy Here

Makes a copies of the selected items. On collision items are renamed in the Windows standard way ("Copy of...").

Copy Here with Increment

Creates auto-serial-numbered/lettered (template defined in Configuration | Templates) duplicates of the selected items. Examples:

My.txt => My-01.txt

My.txt => My-02.txt (if My-01.txt already exists in the target folder)

My.txt => My-03.txt (if My-01.txt and My-02.txt already exist in the target folder)

etc.

Shortcut for Copy Here With Increment: Ctrl+D (think "Duplicate")

Copy Here with Current/Last Modified Date

Makes a copies of the selected items and automatically appends the last modified date [or current date] of the copied item in a format defined by a template. Date-stamped copies are very, v-e-r-y, useful when working with versions. If an item by the new name already exists, auto-incremented numbers are suffixed.

Shortcuts (think "Duplicate Dated"):

Ctrl+Shift+D: Last Modified Date

Ctrl+Shift+Alt+D: Current Date

Copy Here As...

Allows you to create renamed copy of the selected item in the same path.

It also works for multiple selected items. The newly generated names are distinguished by suffix numbers in the current order of listing, for example:

Copy `china.jpg and japan.jpg` Here As "asia" will generate the following copies (assuming your incremental suffix template is "-01", see Configuration | Colors and Styles | Templates | Filename Affixes):

```
asia-01.jpg          (copy of china.jpg)
asia-02.jpg          (copy of japan.jpg)
```

However, if only one file is selected (and this will be the typical case with this command) then no suffix number is appended: Copy `china.jpg` Here As "asia" will generate a copy named `asia.jpg`.

You can also enter absolute or relative paths along with the name. Any missing subfolders will be created silently:

```
Perseus.txt
moq\Perseus.txt
..\moq\Perseus.txt
```

```
Q:\moq\Perseus.txt
..\moq\              (will use the source filename as target filename))
```

Wildcards in Copy / Move Here As...

You can as well state wildcards to define the target names of the operation:

Wildcard * stands for the file base.

Wildcard ? stands for the file extension (without dot).

When items with multiple extensions are selected (or right-mouse dropped) the extension is to "?" which ensures that each of the copied items retains its original extension.

Apart from wildcards also XYplorer native variables are supported. For example, the pattern (the date variable refers to the current date)

```
*_backup_<date yyyymmdd>.?
```

applied to files

```
motherearth.txt
funlicker.jpg
```

would create copies named

```
motherearth_backup_20100117.txt
funlicker_backup_20100117.jpg
```

in the same folder. If you'd do the same thing again on the same day the following copies would be created:

```
motherearth_backup_20100117-01.txt
funlicker_backup_20100117-01.jpg
```

FYI, the format of the incremental suffix is defined in Configuration | Templates.

Also supported are date variables that refer to the selected items individually. For example, to append the modified date of each file to the copy:

```
Pattern:  *_<datem yyyymmdd>.?
Sources:  Che.txt, Fidel.txt
Targets:  Che_20100918.txt, Fidel_20101111.txt
```

Shortcut for Copy Here As: Ctrl+S (think "Save As...")

Copy Here to New Subfolder...

Copy the selected list item(s) to a new subfolder of the current folder. An input dialog opens to enter the desired new folder name.

Note that this command only works on the selected *list* item(s), and is disabled if the file list is not focused.

Shortcut: Ctrl+Shift+C

Move Here to New Subfolder...

Move the selected list item(s) to a new subfolder of the current folder. An input dialog opens to enter the desired new folder name.

Note that this command only works on the selected *list* item(s), and is disabled if the file list is not focused.

Shortcut: Ctrl+Shift+X

Delete

Delete all currently selected files.

Shortcut: Del

Rename

Renames the currently selected items. If only one item is selected and focused, inline rename is invoked. Otherwise Batch Rename is invoked.

Serial Rename: When in single item rename mode, pressing the Tab key will apply the changes and immediately open the rename box for the next item in List. Shift+Tab will do the same for the previous item in the List. The function wraps around the upper and lower ends of the list.

Note that on Serial Rename the list will not be auto-resorted after each rename operation even if you checked "Resort list immediately after rename" in Configuration | Sort and Rename.

Shortcut: F2

File Special (submenu)

Create Shortcut(s)

Creates a shortcut (*.lnk) to each selected tree or list item(s) in the current folder. After creation the shortcuts are auto-selected.

Delete (No Recycle Bin)

Delete all currently selected files, skip the Recycle Bin.

Shortcut: Shift+Del

Delete (Skip Locked)

Normally, when deleting multiple files and a locked file (a file currently used by any application) is met, the OS sits about 2 seconds doing nothing, and then you get a message that a file can't be deleted because it is in use, and finally the deleting stops at this point. All other files that are not in use aren't deleted. Very annoying, and you have surely experienced this when you wanted to clean up in your temp folder last time. "Delete (Skip Locked)" comes in handy here. It simply deletes everything that's not locked and does not contain anything locked.

It takes the following two-step approach:

(1) Look for any locked files within the current selection (including any selected folders). Then delete all

other (unlocked) files (if any).

(2) Look for any empty folders (where empty means: no *files* contained in the whole branch) within the current selection. Then delete those folders (if any).

In step (1), if any locked files are detected, you'll get a confirmation dialog where you can decide how to proceed.

In step (2), if any empty folders are detected, you'll get a confirmation dialog where you can decide how to proceed.

Shortcut: Ctrl+Del

Shortcut: Ctrl+Shift+Del (No Recycle Bin)

Delete Long

Use this command to delete the selected Tree or List item(s) even when they have overlong paths (> 260 characters). The deletion is permanent since the Recycle Bin does not support items that long. The action is listed in the Action Log but cannot be undone.

Note: This command is only available if Configuration | Controls & More | Support overlong filenames is ticked.

See also Overlong Filenames.

Wipe

Wipe is a method to securely delete files beyond recovery (at least by conventional software-based forensic tools). This sort of deletion is also known as Shred, Erase, Secure Delete, Nuke, etc. XYplorer's Wipe doesn't perform paranoid stuff like multiple passes or random patterns, because instead of increasing security this is just stressing your hardware. The following steps suffice to destroy a file permanently:

1. Reset file attributes.
2. Overwrite file with zeros (single pass).
3. Set file size to zero.
4. Rename file.
5. Reset the three file dates to random dates.
6. Delete file.

You can trigger Wipe in four ways:

- First, there is a normal menu command: File | File Special | Wipe.
- Second, there is the Wipe toolbar button.
- Third, there is the Nuke toolbar button which has configurable properties in its right-click menu. Tick "Wiping Beyond Recovery" to make the Nuke button a very dangerous tool.
- Fourth, there is a Miscellaneous command Nuke (#1054) which can be assigned a keyboard shortcut and acts the same as if the Nuke toolbar button is pressed.

Notes:

- You will always get a confirmation prompt before a Wipe operation starts.
- Wipe supports overlong filenames (> 260 chars).
- Wipe does support files > 2GB.
- Wiping huge files takes a while (the "Overwrite file with zeros" part, of course). You will see a little progress indicator in the status bar, and a chance to ESC the operation.
- You can also wipe whole folders. All files found within are individually wiped. Then the folders are removed.
- Toolbar | Nuke: When "Wiping Beyond Recovery" is ON, then "To Recycler" is automatically disabled. Obviously the two cannot coexist.
- Note that wiping (specifically overwriting a file with zeros) should only be performed on HDs (hard disks), but not on SSDs (solid state drives, e.g. flash drives) where the success of overwriting the bytes with zeros cannot be assured due to the logic of those devices, and where it in fact would only increase wear.
- Take care: There is no undo for wipe. :)

Swap Names

Swaps the names of two selected List items. You have to select exactly two items.

Shortcut: Ctrl+Shift+F2

Set Modified Date to Current

Sets the Modified Date of all selected List items to the current date/time. Also known as timestamp.

Set Modified Date to Exif

Sets the Modified date of each selected image file to its Exif date. If an EXIF date cannot be retrieved the shell property "Date taken" is tried as a fallback.

Set Created Date to Exif

Sets the Created date of each selected image file to its Exif date. If an EXIF date cannot be retrieved the shell property "Date taken" is tried as a fallback.

Display Hash Values

Generates and displays the most common hash values for the current file: CRC-32, MD5, SHA-1, SHA-256, SHA-384, SHA-512.

Extract Here

Extracts the selected Zip archive(s) here. Also extracts *.rar and other WinRAR formats if WinRAR is installed on the system, and *.7z and other 7-Zip formats if 7-Zip is installed on the system.

Add to Zip...

Adds the selected items to a Zip archive. You are prompted for a name for the archive.

Properties

Display the shell properties dialog for the current item (Tree, List, Catalog, Address Bar). Works on the focused list item when there are no selections, else it works on any selected list items.

Shortcut: Alt+Enter

Metadata

Show all available metadata (all available Tags, Special Properties, and Shell Properties) of the current item (Tree, List, Catalog, Address Bar) in a popup text box. Also works on the focused list item when it is not selected. Note that also a toolbar button exists for this function.

The window shares size and position with the Quick File View, and remembers both between sessions.

Tip: Tick Configuration | Preview | Quick file view | Modeless dialog to have the dialog modeless: You can leave it open and select files in the file list to view their metadata.

Shortcut: Shift+Enter

Quick File View

Opens a Quick File View window for the currently focused file (even if it is not selected).

(1) There's no editing, just viewing. But you can select and copy.

(2) Of larger files only the first 256 KB are shown. If you need more, use the Raw View on the Info Panel.

(3) Non-binary file are displayed in the usual text mode. All sorts of line feeds (DOS, Mac...) are recognized correctly.

(4) Binary files are displayed in the usual hex editor mode. For performance reasons only the top and bottom lines of binary files are shown.

(5) Of course, this function does not work for folders.

Tip: Tick Configuration | Preview | Quick file view | Modeless dialog to have the dialog modeless: You can leave it open and select files in the file list to have them quick-viewed.

Shortcut: Ctrl+Q

Note that the Floating Preview (F11) and Full Screen Preview (Shift+F11) will automatically open the Quick File View if the previewed file is not supported by them.

Note also that the same keyboard shortcut that opens the Quick File View window can also be used to close it.

Floating Preview

Toggles the Floating Preview.

Note that the Quick File View is auto-opened on a file that's not supported by the Floating Preview.

There is a toolbar button to toggle the Floating Preview.

Shortcut: F11

Save Settings

Save the current configuration to the current INI file. This command also saves the keyboard shortcuts (ks.dat), the User-Defined Commands (udc.dat), the Folder View Settings (fvs.dat), the Tags (tag.dat), the Catalog (catalog.dat), and (if enabled) the cached servers (servers.ini).

The command also writes the current Action Log from memory to disk (action.dat) if "Remember the logged actions between sessions" is enabled.

Tip: Now the Save Settings toolbar button's tooltip tells you how long it's been since you last saved the settings.

Settings Special (submenu)

An asterisk (*) is prefixed to all dirty configurations (that have been changed since startup).

Load Configuration...

Loads a previously saved (see below Save Configuration As...) configuration (*.ini).

Tip: In the title bar of the Configuration window (F9) you can see the currently active INI file.

Save Configuration

Save the current configuration to the current INI file.

Save Configuration As...

The current configuration will be saved to a new INI file located in the app's path. This new INI will become the currently active one.

Save Copy of Configuration As...

Backups the current configuration under a new name (defaults to the present date suffixed to the INI base name) without making it the active INI file.

Create Shortcut to this Configuration...

Creates a shortcut file to XYplorer containing a switch to the current INI-file on your Desktop. All you have to do is to give the shortcut a name.

Save Catalog

Save the catalog database (catalog.dat).

Save User-Defined Commands

Save the User-Defined Commands database (udc.dat).

Save Keyboard Shortcuts

Save the keyboard shortcuts database (ks.dat).

Save Folder View Settings

Save the Folder View Settings database (fvs.dat).

Save Tags

Save Tags, Labels, and Comments to the database (tag.dat).

Save Servers

Save (if Cache Network Servers is enabled) the cached servers database (servers.ini).

Backup Application Data Folder...

Backups all contents of the Application Data Folder to a folder of your choice. The hard-coded operation type is "Backup" with "Overwrite only older files" and "Preserve item dates".

The last used target path is remembered and preselected when you use the command the next time.

Restore Application Data Folder...

Use it to restore the application data from a folder to which you previously backed up the data. You will be prompted before the actual operation begins.

Note the difference between Restore Application Data Folder and the /restore switch:

- /restore reads the configuration from a backup application data folder. It does not copy any files.
- Restore Application Data Folder copies all files and folders from the backup folder to the current application data folder (reckless overwriting, no questions asked in case of collisions), then restarts the application (without saving, it would be overwritten anyway).

Open Throw Away Clone

Opens a clone of the current XYplorer instance (as saved on disk) in read-only mode. You can use such an instance for quick in-between jobs that should not affect your main XYplorer configuration.

- The clone configuration is read from the configuration files on disk, not from XYplorer's current unsaved state in memory.
- Exception: The clone always opens in the current path (even if that's not the start path written on disk).
- Tip 1: To open a clone of the exact current state of the main instance use command "Save Settings" right before using "Open Throw Away Clone".
- Tip 2: Hold CTRL while clicking the command to open a fresh instance in the current path. So it's the same as running the script "freshhere;".

- Tip 3: Hold CTRL+SHIFT while clicking the command to open a fully enabled (non-throw-away) clone of the current window. So it's identical to a Throw Away Clone, but without READ-ONLY.
- Tip 4: The clone can save settings via scripting: Use scripting command savesettings with the "ini" argument.
- The command lets you open as many clones as you like.
- [READONLY] in shown in the window title bar of a clone.

Restart without Saving

Restart program without saving settings.

Exit without Saving

Exit program without saving settings.

Keyboard trick: hold CTRL while closing the app via Window X-button to force exit without saving.

Shortcut: Ctrl+Alt+F4

Exit

Exit program. If Configuration | Startup & Exit | Save Settings on Exit is ticked then all settings are saved before exiting.

Shortcut: Alt+F4

3.2 Edit

Edit Menu Commands

Undo

Undo the previous action (if possible). See Undo...

Shortcut: Ctrl+Z

Redo

Redo the previously undone action. See Undo...

Shortcut: Ctrl+Shift+Z

Action Log...

Show the Action Log dialog. See Action Log...

Shortcut: Ctrl+Alt+Z

Recent File Operations...

Pops a menu by which you can apply recent file operations (Copy and Move) to the currently selected files, using the same operation and target path as recently. The menu draws its data from the Action Log and is organized in two sections:

- Section 1: Recent file operations from here, chronologically ordered from youngest to oldest (up to 10).
- Section 2: recent file operations from elsewhere, chronologically ordered from youngest to oldest (up to 10).

Copy and Move operations are mixed, Move is indented for clearer distinction.

Tip: Hold CTRL to go to the selected target path (instead of repeating the file operation).

Note: There is a toolbar button available with the same name and functionality.

Shortcut: Alt+F7

Cut

Cut selected items to clipboard.

Shortcut: Ctrl+X

Copy

Copy selected items to clipboard.

Shortcut: Ctrl+C

Append

Append selected items to clipboard.

- Keeps the effect ("copied" or "cut") of the items already in the clipboard.
- If no items are in the clipboard the effect defaults to "copied".

Shortcut: Ctrl+Alt+D

Tip: There is also a toolbar button "Append".

Paste

Paste selected items from clipboard.

Shortcut: Ctrl+V

Tip: There is also a toolbar button "Paste". It serves as a quick and basic clipboard viewer when you hover it. It shows text and file items contained in the clipboard, i.e. the stuff that will be pasted if you click the button.

Paste Special (Submenu)

Paste Here to New Subfolder...: Lets you paste any files currently found in the clipboard to a newly created subfolder of the current folder (the one that's displayed in the main title bar).

You are presented an array of up to four possible default names for the new subfolder about to be created. The first three are (can be) defined in the [NewTemplates] INI section. Here's an example:

```
[NewTemplates]

Folder0=\N\e\w \F\o\l\d\e\r

Folder1=yyyymmdd

Folder2=yyyy-mm-dd
```

The fourth suggested name is made from the first item to be moved [copied, pasted] into the new subfolder, i.e. the item that was focused in the moment of cutting, copying, dragging. The index of the last selection among the first three names is remembered between sessions.

Shortcut: Ctrl+Shift+V

Paste Here with Path...: A copy command that offers you smart choices regarding the target location. Makes it very easy to mirror copy individual items from/to deeply nested locations. Same functionality as "Copy/Move Here with Path..." in the Native Drag and Drop Context Menu.

Paste Here As...: Lets you to paste item(s) in the clipboard under a new name. The functionality, interface, and syntax are analogue to File | Duplicate | Copy Here As... (including advanced syntax with wildcards and variables), only that the source(s) are in the clipboard.

Paste (Move) and Paste (Copy): Paste moving/copying the items currently found in clipboard, independent of the way the files have been send to the clipboard before (cut or copied).

Paste (Backup): Trigger the Backup operation for items on the clipboard with the current folder as

target.

Paste As Shortcut(s): Use it to create shortcuts to the items (files or folders) on clipboard, in the current location. It also works for more than one item at a time.

Paste As Hard Link(s): Creates a new file hard linked to the file currently on the clipboard, in the current location. It also works for more than one file at a time. Notes:

- Hard links can only be created for files, not for folders.
- All hard links to a file must be on the same volume (same drive letter or network share).
- Hard links can only be created on NTFS volumes.
- The maximum number of hard links that can be created is 1023 per file.

Paste As Symbolic Link(s): Creates a symbolic link (aka symlink or soft link) to the item currently on the clipboard, in the current location. It also works for more than one item at a time. Notes:

- This command needs Vista or later!
- You may need administrator rights to create symbolic links.
- Contrary to Paste As Junction(s) this command can be applied also to files (not just folders).
- Symbolic links can point to non existent targets because the operating system does not check to see if the target exists.
- Currently no relative symbolic links are supported. It might come later...
- The targets of Symbolic Links are shown as "Junction Target" in the interface (File Info Tips, List context menu) just like the targets of Junctions proper, in order to keep things a bit tighter.
- Also the Color Filters make no difference between Symbolic Links and Junctions, since all share the FILE_ATTRIBUTE_REPARSE_POINT file attribute.
- And, consequently, when you uncheck "Show Junctions" in Configuration | Tree and List, then also any Symbolic Links are hidden.

Paste As Junction(s): Creates a junction (aka NTFS junction point or directory junction) to the folder currently on the clipboard, in the current location. It also works for more than one folder at a time. Notes:

- The command works only on folders, not on files.
- Windows does not support junctions to directories on remote shares.

Paste Extracted: Lets you extract the Zip archive on the clipboard to the current folder. Also extracts *.rar and other WinRAR formats if WinRAR is installed on the system, and *.7z and other 7-Zip formats if 7-Zip is installed on the system.

Paste Zipped: Lets you paste the items in the clipboard to a newly created zip file. To make it maximally streamlined you are NOT prompted for a name. The zip name defaults to the first item found in the clipboard (auto-suffixed on collision). After completion the new zip file is auto-selected.

Note: This command will never remove the source items, even if they were *cut* to the clipboard. So, "cut" and "copy" make no difference here.

Paste Folder Structure: Use it to paste the folder structure (without any contained files) from folder(s) on clipboard into the current list path. Functionally equivalent to "Create Branch(es) Here" in the drag and drop context menu.

Paste Text As Item(s): Pastes a copy of the item(s) referred to by textual clipboard contents.

- More than one item is possible, separated by | or by line feed (CRLF). Both separators can be mixed. Empty lines/elements are auto-removed.
- XYplorer native and environment variables are allowed.
- Relative paths are resolved relative to the current path.
- Leading and trailing spaces are trimmed on each item.
- Folders are copied with their contents (of course).
- URLs are supported as well. But only one at a time.

Usage Examples:

- A colleague mails you a path to some file on the intranet. You copy the path from the email to the clipboard, then use "Paste Text As Item(s)" to create a copy of that file in your current folder.
- A colleague mails you the URL to some package in the web. You copy the URL from the email to the clipboard, then use "Paste Text As Item(s)" to download that file into your current folder.
- Equally you can fetch download links on the web via clipboard and grab those items by "Paste Text As Item(s)" without going via the Browser Download to the Download folder.

Paste Text Into New File: Creates a new text file filled with the current clipboard contents. The pasted text depends on the actual data format on the clipboard; it is identical to what variable <clipboard> would

return:

Clipboard contents	Text
Text	The text
File items	One item per line, CRLF ends a line
Else (e.g. images)	Nothing

The new file is called something like "Clipboard- 20080302.txt". The "-20080302" part is the date now (format defined in Configuration | Templates under Filename affixes. The file is appended to the current file list (aka "lazy sort"), and after creation of the file, rename-mode is invoked.

Shortcut: Ctrl+Alt+V

Paste Image Into New [EXT] File: Creates a new image file filled with the image (if any) currently on the clipboard. The standard image format is PNG. But you can change this to JPG, GIF, BMP or TIF using a new INI Tweak:

```
[General]
ImageFromClipFormat=jpg
```

Allowed values are: png (default), jpg, gif, bmp, and tif. In the case of JPG you can control the quality by appending a number from 1 (worst) to 100 (best) inclusively, for example:

```
ImageFromClipFormat=jpg100
```

or:

```
ImageFromClipFormat=jpg50
```

The default value is 85.

Note that the GIF quality is very bad (dithered) and practically unusable. The other formats, however, are of high quality.

This feature requires GDI+.

Shortcut: Ctrl+Shift+Alt+V

Mark Files in Clipboard as 'Cut'/'Copied': here you can set/change the "drop effect" of the items currently in the clipboard. On paste, "cut" items will be moved, and "copied" items will be copied. Also the current "drop effect" is signaled by a checkmark.

Edit Clipboard...: Allows you to view what's currently sitting in the clipboard and to manually edit it. Works for files as well as for text.

Tip: There is also a toolbar button "Edit Clipboard". The button tooltip serves as a quick and basic Clipboard Viewer. It shows text and file items contained in the clipboard (cropped after 2048 characters, or after 32 lines, whatever comes sooner). For text it shows the line count and the character count. Yes, it even shows images in the clipboard. By the way, the "Paste" button does it, too.

Restore Previous Clipboard: The previous state of the clipboard is automatically stored whenever the clipboard changes, and you can toggle the current and the previous state by using the new command.

- This has to be ticked to enable the feature: Configuration | File Operations | Undo & Action Log | Clipboard | Log clipboard contents and enable restore
- New clipboard contents are only auto-stored while XYplorer's main window is in the foreground.
- When the clipboard is empty "Restore Previous Clipboard" will toggle the two previous non-empty states.
- You can peek into the previous clipboard contents by holding SHIFT while you hover the "Edit Clipboard" or "Paste" toolbar buttons.

Clear Clipboard: Allows you to remove any files (and, en passant, everything else) from the clipboard. Of course, the actual files are not touched by this operation, even if they have been "Cut" to the clipboard.

Move To..., Copy To..., Backup To... (Dialogs)

Opens an interface to move/copy/backup the selected items to a destination folder. The destination folder can be typed, pasted, selected through a "Browse for folder" dialog, or selected from a list of the most recently used destination folders.

The list (shared between Move/Copy/Backup) of the most recently used destination folders can be edited in Tools | List Management | Move/Copy/Backup To...

Move To	Shortcut: Shift+F7
Copy To	Shortcut: Ctrl+F7
Backup To	Shortcut: Ctrl+Shift+F7

Relative paths

Relative paths are allowed.

You may state paths relative to the current path. So e.g. you can state a target like "..\sister" to move a file to a folder called "sister" under the same parent as the moved items. If this folder does not exist yet it is created!

Destination paths may contain Date Variables

The destination path is allowed to contain date variables that will be resolved into the date now! The syntax is identical to the one used in Batch Rename, namely "<date [date spec]>". Examples:

```
<date yyyy>                 =  2006
<date yyyymmdd>             =  20061004
<date yyyy-mm-dd_hh-nn-ss>  =  2006-10-04_09-08-04
```

Together with the feature described in the next paragraph (non-existing paths are created for you), this is an extremely valuable addition! For example, you now can easily make regular

dated backups by using a destination like this:

"D:\My Regular Backups\<date yyyy-mm-dd_hh-nn-ss>"

When you now Copy (or Backup; it's the same in this case, since

the destination is new) anything to that destination

1. The folder "D:\My Regular Backups\2006-10-04_09-08-04" is created.

2. The selected stuff is copied there.

Note, that you even can have more than one date variable in the string. So you also can do something like:

"D:\My Regular Backups\<date yyyy-mm-dd>\<date hh-nn-ss>"

A folder for the day, a subfolder for the time.

Non-existing paths are created on the fly

You may enter non-existing paths, and you will be prompted whether you wish to create them on the fly before continuing with the file operation. Note that all parts can be new, not just the last subfolder. So "D:\new\new too\also new" will work just fine.

Catalog Categories can be backed up as well

The Catalog (Items and Categories) is supported as source for backup operations. To cope with distributed source items (i.e. items in multiple locations), the paths of the items are *fully* recreated in the target folder, including the drive letter (or server name). For example, if the selected Catalog Category

contains these items...

```
C:\project\code\
D:\www\project\code\
\\cary\grant\hat\black.ico
\\cary\grant\suit\black.ico
```

... and you backup them today (2007-10-26) to `E:\Backup\<date yyyy-mm-dd>`, then the resulting copies will be:

```
E:\Backup\2007-10-26\C\project\code\
E:\Backup\2007-10-26\D\www\project\code\
E:\Backup\2007-10-26\cary\grant\hat\black.ico
E:\Backup\2007-10-26\cary\grant\suit\black.ico
```

This is the only way to guarantee that no merging/overwriting of homonymous items happens. And by the way, it's also a nice way to tell you quickly and exactly where a backup came from. Of course, you have to take care with huge path names, else you will hit the 260 char limit sooner than you think. The Backup Report will state as source: "[multiple source locations, fully recreated in target]".

So, the capability to backup whole Catalog Categories provides a one-click backup of distributed sources.

Compare (Submenu)

Compare Current File with Previous File

Allows you to determine whether the currently selected file is same or different from the previously selected file. The files need not be in same folder. A file is "selected" once it is displayed on Info Panel and Status Bar.

Logic & Remarks: See next paragraph.

Shortcut: Ctrl+K

Compare Current File with File in Clipboard

Sort of a "Quick Compare" that allows you to check whether the currently selected file is same or different from the file currently on clipboard.

Steps

(1) Copy one file to the clipboard. Normal file copy (Ctrl+C), or copy of file name (Ctrl+P), both works. Copying of the file's contents will not work.

(2) Select another file in the file list.

(3) Now these two files can be compared using the Compare ... command.

Logic

First the file sizes are compared. If they are identical, the contents themselves are compared. The result is simply a "same" or "different" without any further information. If you need more details use a specialized comparison application (which can easily be automated from XY using User-Defined Commands or Scripting).

Remarks

For comparisons of files larger 1 MB you get a progress indication and a break chance by ESC.

Shortcut: Ctrl+Shift+K

Compare Current File on Both Panes

Checks whether the currently focused file is same or different from the file currently focused on the other pane.

Logic & Remarks: see previous paragraph.

Shortcut: Ctrl+Alt+K

New (submenu)

New Folder, New Text File

Create new folder or new text file in current folder. Choose the date-named variants of the commands to auto-name the created folder or text file to the current date, eg:

2000-03-05.txt

New Shortcut...

Displays the Create New Shortcut dialog. Completing the dialog creates a shortcut at the current location.

New Path...

Here you can create a multi-part path in one go. Any slashes "/" are auto-converted into backslashes "\". After creation you will be transported to the newly created location.

Shortcut: Ctrl+Alt+N (= AltGr+N)

New Folders...

Here you can enter a list of new folders or whole new paths that will be created in one go.

- The folder paths can be absolute or relative (to current).

- They can be slashed or unslashed.
- They can be whole new paths, or just new subfolders to existing paths.
- They can be stated in any order.
- Leading or trailing spaces in each line are tolerated (and ignored).
- Any slashes "/" are auto-converted into backslashes "\".
- XYplorer native variables and environment variables are supported.
- The last list is remembered within the session.
- This interface does not support auto-elevation in UAC-protected locations.

Example for a list:

```
backup
code
docs
materials\%username%
materials\pics
materials\audio
materials\video
New <date yyyymmdd>
```

New Files...

Here you can enter a list of new files (one per line) that will be created in one go.

- Paths can be absolute or relative (to current path).
- Non-existing paths are created on the fly.
- They can be stated in any order.
- Leading or trailing spaces in each line are tolerated (and ignored).
- XYplorer native variables and environment variables are supported.
- Files that already exist are silently ignored (and not overwritten).
- The last list is remembered within the session.
- This interface does not support auto-elevation in UAC-protected locations.
- This command can also create new folders. Any item ending with a backslash is considered a folder.

Example for a list:

```
newtextfilehere.txt
MoonLanding\TopSecret\new.txt
\MarsLanding\TopSecret\new.txt
D:\Testpath\new.txt
%username%.txt
New <date yyyymmdd>.txt
```

Remember Last Input: Check "Remember last input" in the dialog if you want to remember the last input the next time you open the dialog. The setting applies to both dialogs, New Files and New Folders.

New Items (submenu)

The New Items submenu is a fully and easily configurable "New"-menu that vastly surpasses the possibilities of Explorer's "New" menu.

You can freely place any files or folders (with subfolders and contents if you like) of your own choice and making in an application data folder called "NewItems". On startup, XYplorer scans this folder and fills the submenu Edit/New Items with all items found in this folder.

Example: How to add a new Word Document to the NewItems folder

1) Create a new Word Document and copy it to the clipboard.

2) Enter this into XYplorer Address Bar and press ENTER: `<xydata>\NewItems`

3) Paste the Word Document in to the NewItems folder.

4) Now you have your Word Document available in Edit | New Items.

Using the menu works as expected: You select one of the menu items, and the respective file or folder (incl. contents!) item is created in (copied to) the current folder, and the rename mode is immediately invoked. Which makes for instance the repeated creation of a project's complex folder structure a one-click-affair!

Tags: Any tags in the source item are copied to the new target item. This works silently and independently of the setting of "Copy tags on copy operations".

The number of items is not limited.

Date variables are supported in the item names. For example: (1) Create a file named "Log {yyyy-mm-dd}.txt" in the NewItems folder in app path. (2) A file called "Log 2010-09-17.txt" will be created when you select this file in the NewItems menu! (If today is 2010-09-17.) Note that date variables are not resolved in any subitems of a folder in NewItems.

Also comments are supported in the item names. The comment is added as part of the source item name and is separated from the target item name by the sequence space-space-singlequote (singlequote = ASCII 39). When a new item is copied the comment is dropped and only the part left of the comment separator is used in the target. For example, and item named `"targetname  'comment"` in the New Items folder will be copied to the new item `"targetname"`. The comments are visible in the New Item menu.

With *folders* the comment is simply appended to the name. With *files* the comment has to be placed left of the extension, for example:

Source: `<newitemspath>\targetname  'comment.txt`

Target: `<targetpath>\targetname.txt`

Go to New Items Folder: Brings you to the folder where you can add or edit the items that will be shown in the New Items menu.

Select (submenu)

Toggle Selection

Toggles the selection of the focused list item (the list has to be in focus as well).

Note that multi-selection is supported: Any other selections are not automatically unselected. So functionally it's identical to the hard-built-in Windows standard shortcut Ctrl+Space.

Shortcut: INS

Select All

Selects all items currently listed.

Shortcut: Ctrl+A

Deselect All

Deselects all items currently listed.

Shortcut: Ctrl+Shift+A

Invert Selection

Selects all unselected items, unselects all selected items.

Shortcut: Ctrl+Shift+I

Restore Selection

Use Restore Selection to restore the last selection when you inadvertently lost it. The "last selection" is any selection right before the last time all items in the list have been unselected, or when a multi-selection has been directly replaced by a single or multi-selection.

Notes:

- A plain change of selection from one item to another (e.g. by single click on another item) does NOT store the previously selected item and thus will not affect the currently stored selection.
  This means: You can have a complex selection, then single-select some items (e.g. to preview them), and then restore the complex selection. Nice!
- When you do "Restore Selection" while there is another selection then doing "Restore Selection" again will restore that other selection. In other words, you can toggle between two selection patterns.
- When in Find mode (Search Results tab), then the full paths of the last selected items are (re)stored.
- The command does not restore the previous focus.
- Each pane remembers its own last selection (but not each tab).
- To save resources selections are only retained in lists of no more than 32,767 items, and only selections of no more than 4,096 selected items can be restored.
- The last selection is not stored between sessions.

Selection Filter...

Select a subset of the currently listed items via pattern matching or Regular Expressions (prefix term with >). The first selected item is automatically focused and scrolled into view.

You can enter more than one wildcard patterns (separated by ;) to select all matching items.

For example: '*.jpg;*.gif;*.png'.

Shortcut: Ctrl+M

Column-wise Selection Filter: You can as well select items by comparing your pattern with *any* column (not just the Name column). The comparison that results in selections on a match is a case-insensitive *string* comparison (not a *number* or *date* comparison). To select by columns other than the name column, simply prefix the pattern with '[column name]:' Note: the column identification is case-insensitive (e.g. 'type' works for the Type column), and there's no space after ":".

Examples (results partly depend on the column display format):

```
size:? KB            -> selects all items < 10 KB
modified:*2006*   -> selects all items modified in year 2006
type:vis*            -> selects all items of type starting with "vis"
len:2??              -> selects all items with a filename >= 200 characters
```

The name column is the default, so "name:" can be omitted. If you type an invalid column name, the name column will be used.

The column prefix can be partial. If no column is identical to the prefix, then the first column with a partial match (from beginning: "Modified" matches "mod*") is used.

Limiting the selections to files or folders: You can restrict the selections to files or folders by attaching one of the following prefixes to the pattern:

```
*:  => select files only
\:  => select folders only
```

Examples:

```
new*        Select all items starting with "new"
*:>new*     Select all files starting with "new"
\:>new*     Select all folders starting with "new"
```

Also works with RegExp:

```
>^new       Select all items starting with "new" (RegExp)
*:>^new     Select all files starting with "new" (RegExp)
\:>^new     Select all folders starting with "new" (RegExp)
```

The prefix `*:` or `\:` must precede any other prefix:

```
*:!>^new     Select all files NOT starting with "new" (RegExp)
*:!type:doc  Select all files where the Type does NOT have "doc"
```

Adding to / removing from the current selections: You can add to or remove from the current selections very easily:

- Hold CTRL while clicking OK in the Selection Filter dialog = Add matches to the current selections.
- Hold SHIFT while clicking OK in the Selection Filter dialog = Remove matches from the current selections.

Select By Selected Type(s)

Selects all list items with the same type(s) as the currently selected item(s). Also useful when you want count the items of a certain type.

Shortcut: Ctrl+Alt+M

Select All Files

Selects all files in the List, not the folders.

Shortcut: Ctrl+Shift+Alt+M

Select Items...

Here you can select items in the current list by way of a list of item names. These are the rules for such a list:

- One item name per line.
- Empty lines are allowed and ignored.
- Full paths are allowed in the list and will be ignored at the matching. However, when you pass items with paths and the list is in Find mode, then the matching takes the full path into account.
- Encoded URLs will be decoded (%20 is decoded to space, etc.).
- URL queries are allowed and ignored at the matching.
- The sequence of the item names does not matter.
- Leading and trailing spaces around each name are allowed and ignored.
- Matching is case-insensitive (A=a), so you can state the names with wrong capitalization and still have them selected.
- You can use wildcards * and ? to match bunches of items.
- XYplorer native and environment variables are allowed.

As an example for the parsing, the following item names would all select the file "Arial Unicode MS.php" in the current list:

```
Arial Unicode MS.php
c:\files\A\Arial Unicode MS.php
\\192.168.0.2\Fonts\Arial Unicode MS.php
file:///c:/files/A/Arial Unicode MS.php
http://www.sofontes.com.br/files/A/Arial Unicode MS.php
http://www.sofontes.com.br/files/A/Arial%20Unicode%20MS.php
Arial Unicode MS.php?f=3&t=5502&p=51346
```

Some further options and properties:

- Ignore extensions: If checked then matching is done only against the base of the given item names, and one item name can potentially generate any number of selections in the List.
  The state of the checkbox is remembered within the session.
- The first selected item from top is focused and moved into view.
- The last used list is remembered within the session.

Example: A small list of item names could look like this (and would select these three files in the current List if they are present):

```
install.log
Readme.txt
License.txt
```

Example with wildcards: This small list of patterns would select all PNG files, all items beginning with "car", and all items containing the letter "z":

```
*.png
car*
*z*
```

More Interesting Use Case: This command enables you to perform a potentially large, complex, and not-rule-based selection in a very easy and convenient way. There is no other way to do what Select Items... can do for you. Here's an example: Say you are going to overwrite 250 files in a folder of 5,000 files with newer versions. But you'd like to backup the original versions first. How will you grab those 250 out of the 5,000 without spending a tedious time with manual selection? Using Select Items... this is a snap: Go to the folder with the 250 new items and copy their names the clipboard (Ctrl+Shift+P); then go to the folder with the 5,000 items, click Select Items..., paste the 250 names into the text box, and click OK; all original items about to be overwritten are selected and ready for backup! This will take you less than 10 seconds.

Shortcut: Ctrl+Shift+M

Selection Stats

Displays statistical information on the currently selected list items (including any folders). When there are no selections then the stats are on all listed items.

When the Tree has the focus, the stats on the current tree folder are returned.

Find Files

Open the Find Files tab on the Info Panel, and place the cursor into the Name field. If that is already the case then the command hides the Info Panel.

Shortcut: Ctrl+F

Find Now

Starts File Find with the current parameter setting, starting in the current folder.

Shortcut: Ctrl+Alt+F

Repeat Last Search

Runs File Find at the last searched location using the current find settings.

Search Templates...

Opens the Search Templates.

Quick Search...

Opens the Quick Search dialog.

Shortcut: F3

Toggle Quick Search

Toggles between no Quick Search and the last one (if any) in the current location.

Shortcut: Ctrl+Shift+F3

Repeat Last Quick Search

Repeats the last Quick Search (if any) in the current location.

Shortcut: Shift+F3

Show All Items In Branch

Browses the current folder including subfolders and lists everything in it independent of the current Find Files settings.

Note that the more recent feature Branch View offers a superior way to achieve this.

Shortcut: Ctrl+F3

3.3 View

View Menu Commands

Views (Submenu)

Here you can choose between a number of display styles (you can also open this menu by Shift+Right-Click on the Line Numbers Column Header (#)):

Details: show detailed file information column-wise.

List: List is a wrapped column of icons plus names, as opposed to "Small Icons" which is a wrapped row of icons plus names.

Small Icons: Just icons (16x16) and names. The column width matches the current Name column width.

Large Icons: Just icons (32x32 or 48x48 depending on the Windows version) and names.

Small Tiles: Extra large icons (48x48) or thumbnails (default: 64x64), 3 info lines: Name, Size, Modified Date.

Large Tiles: Extra large icons (48x48) or thumbnails (default: 192x192), 4 info lines: Name, Size, Modified Date, Extra Info (depending on file type, see below).

Extra Info:
Images: Dimensions and Aspect Ratio
Media: Length
Executables and other binaries: File Version

Tip: To enable thumbnails in Tiles views tick *Configuration* | *Thumbnails* | *Show thumbnails in tiles views*.

Tip: The width of the text area in Tiles views depends on the width of the Name column in Details view, so you can adjust it via:

View | *Columns* | *Grow Name Column (Ctrl+Shift+Numpad Add)*

View | *Columns* | *Shrink Name Column (Ctrl+Shift+Numpad Subtract)*

or by drag-sizing the Name column header.

Note that you can minimize the Name column width so that no text at all is visible along with the tiles.

Tip: On Tiles views the command *View* | *Columns* | *Autosize Columns Now (Ctrl+Numpad Add)* toggles the size of the Name column only (i.e. the width of the tiles data area) between optimal size (minimal size to fully show the longest filename) and minimal column size (to totally hide the tiles data).

Tip: In Large Tiles view you can show some basic photo data (Camera Model, Focal Length, F-Stop, Exposure Time, ISO Speed, Exposure Program, Exposure Bias, Date Taken) along with the other info. Hold CTRL and Right-Click the text area of any tile and select Show Photo Data from the context menu. Of course, this only works for photo files, i.e. JPEG, PSD, and all RAW formats that you have Codecs for on your system. Other files will not display the extra fields. The feature needs Win7 or later.

Thumbnails #1 / #2 / #3 (Width x Height): thumbnails plus names. The thumbnails' size can be

defined by the settings in Configuration/Thumbnails.

Note: Views are stored tab-wise and retained between sessions.

Tip: Hold Shift while selecting a view to apply the view to all tabs.

Details with Thumbnails #1: Details plus a Thumbs column. The thumbnails' size is defined by the settings in Configuration/Thumbnails. Default size is 64 x 64.

Configure Thumbnails...: open the Configuration dialog for thumbnails.

Branch View: Toggle Branch View.

Touchscreen Mode: Toggle Touchscreen Mode, i.e. bigger icons in Tree, List, and Catalog. Big enough to be touched with big fingers. And optionally bigger, more readable fonts. You can customize the Touchscreen Mode in the context menu of the Toolbar button called "Touchscreen Mode".

Dark Mode: Toggle Dark Mode.

Sort By (Submenu)

[Column Name]: Set currently sorted column in file list.

Unsorted: Show order as is in the FAT (file allocation table), which is usually alphabetically ascending by name, files and folders mixed.

In a recursive search results listing, the order is in each folder: files first, then folders. The result is a tree-like sort order.

Random Order: Shuffle the file list. Press it repeatedly to keep on shuffling!

- The random sort order will survive a tab switch, an explicit or automatic refresh, and even a restart.
- The random sort order is indicated by a small "dice" character (Unicode U+1F3B2) in the Name column header. It's hard to recognize in point 9 fonts but at least you can see the list not normally sorted and it's not manually sorted.
- Only the order of the files is randomized whereas the other items (folders or drives) are kept on top in alphabetically ascending order (but see the following point).
- The toolbar button "Random Order" has a context menu that features the toggle "Include Folders". Tick it to also shuffle the folders. This setting also affects the command "View | Sort By | Random Order".

Shortcut: Ctrl+Alt+R

Sort Again: Refresh current sort.

Previous Order: Resort the list using the previous sort order. Quite useful to toggle between the two recent sort orders.

Reverse Order: Reverse the current sort order. Handy to reverse the current sort order by keyboard, and the only way to reverse a custom sort order.

Selected Items to Top: Click it to show all selected items at the top of the list (superseding whatever

is the current sort order), and scroll the list to the top. Note that this sorting is not permanent. It will not survive a session, a tab switch, or even a list refresh. The idea is to quickly get the selected stuff together in one spot to easier do what has to be done with it.

Sort Folders Apart: Check this to make XYplorer behave like Explorer in sorting folders apart from files. Uncheck it if you want everything in one alphabet. (Mirrors the same setting in Configuration | Sort and Rename | Sort.)

Show Sort Headers in All Views: Tick it to have sort headers (column headers) also in non-Details views, e.g. in Thumbnails. You can click the column headers to sort the list by this column. Mirrors the same-named setting in "Configuration | Sort and Rename | Sort" and makes it accessible by keyboard and scripting.

Columns (Submenu)

Line Numbers: Show line numbers in the left-most column. Its width auto-adjusts to the number of items in the list, and it got its own fully configurable color in Configuration | Colors. The right border color of the line numbers area is derived from its backcolor by darkening.

[Column Name]: Show/hide this column in List.

Add Column: Add a new Soft Column to the list. The column is inserted left of the currently sorted column. The newly added column is called ""Right-click here...". Right-click its header to give it some definition (see Soft Columns).

Show Columns...: Pops a checkbox list of all available columns in the current list mode, and this offers an alternative way to control column visibility.

Show All Columns: Show all columns.

Skip Custom Columns: Check it to not fill custom columns in the current list. Custom columns are often time-consuming, especially when scripts are involved. This option allows you to temporarily disable this service when the extra information is not needed. Note that you can still manually populate a custom column by clicking "Refresh Column" in the context menu of the column header.
Tip: The toggle is also available in the context menu of the Custom Column headers.

Autosize Columns Now: This command allows you to adjust the width of *all* columns. Columns are auto-sized to a max width of 600 pixels. In thumbnails, this command toggles "Show Captions".

Tip 1: You can adjust each file list column's width by double-clicking the little gap right of the column header. You can adjust the width of *all* columns by one double-click on the dead-zone right of all column headers, or by double-click on the Line Number column header. To automatically adjust the width of *all* columns when listing the contents of a new location, check Tools | Customize List | Autosize Columns. This command is also found in the Ctrl+Right-Click context menu of each column header.

Tip 2: On Tiles views this command measures the width of the data below the filename instead of the filename itself. That way a few XXL filenames will not affect the layout of the whole list, and the data are shown uncropped even on short filenames.

Tip 3: On Thumbnails views this command toggles the captions visibility. Might not be very logical given the name of the command, but once you got used to the similar functionality in Tiles views you quickly come to expect this behavior.

Shortcut: Ctrl+Numpad Add

Grow Name Column: Grows the name column by 4 pixels (in all views apart from thumbnails). In thumbnails, this command toggles "Overlay Captions".

Shortcut: Ctrl+Shift+Numpad Add

Shrink Name Column: Shrinks the name column by 4 pixels (in all views apart from thumbnails).

Shortcut: Ctrl+Shift+Numpad Subtract

Load Column Layout...: Loads a new column layout (all visible and invisible columns, their sequence, their captions, their widths) from a small TXT file in the application data path <xydata>\Columns.

Save Column Layout As...: Saves the current column layout (all visible and invisible columns, their sequence, their captions, their widths) to a small TXT file in the application data path <xydata>\Columns.

Tip: There is a toolbar button called "Column Layouts" that lets you load predefined and user-defined column layouts.

Set Line Number Column Width: Set the minimum width of the line numbers column measured in digits. The column will enlarge as necessary but never shrink below the user-defined minimum size.

Tab (Submenu)

New Tab: Create a new tab. The new tab will be a clone of the current tab (but without Home, Locked status or Visual Filter). The new tab keeps the location of the current tab unless configured otherwise in Configuration | Tabs and Panes | Tabs | New tab path. If in Find mode it switches to Browse mode.

Shortcut: Ctrl+T

Clone Tab: Opens a perfect clone of the current tab.

Default Tab: Make this tab the default tab. Any passively (implicitly) opened new tab will open in this tab instead. There's a visual indicator (a green icon overlay) to show which tab is the default tab (if any). There can be only one default tab.

Iconized Tab: Tabs can be iconized, i.e. shrunk to the size of an icon.

Rename Tab...: You can give the tabs any name of your choice. Once a tab is named that name will stay fixed when you change the current directory inside the tab. A tab name can have any characters, even those that are illegal for filenames. To un-name a tab rename it to nothing.

Tip 1: You can specify the same variables as in Configuration | Tabs | Tab captions | Custom, so you can have per-tab custom captions like e.g.: `<drive>: <folder>`.

Tip 2: You can as well specify icons of your free choice for each tab. This is described here.

Text Color...: Set the text color for this tab. This custom color has to be specified in hexadecimal format (RRGGBB). Use the Select Color... button to pick a color. To go back to the default colors simply specify nothing.

Background Color...: Set the background color for this tab. See notes here above at Text Color...

Copy Location Term: Puts the tab's location, including any Visual Filters, up to the clipboard. If the tab is renamed, the command will still copy the location, not the caption.

Relocate Tab...: Here you can enter a new location for the current or any non-current tab (when opening the menu via right-click on a tab header). If you do it for the current tab, it will browse to the new location.

Of course, the location of an unlocked tab is usually set to the new path simply by going to a new location via Tree or Address Bar or whatever; however, this path is always a hard-coded one. Now Relocate Tab allows you to enter a soft path (aka Portable Path), like %TEMP%, or %USERPROFILE%, or <xydata>. So for example, if you have a tab pointing to %USERPROFILE%, it will always be resolved to the current user profile folder, no matter where it is located in the system you are currently working on! This revolutionary concept is called Portable Tabs.

Tab History...: Pop up a list containing all history items pertaining to the current tab.

Set Home...: Each tab can have its own "Home", defined by a browse/find location and mode, and the usual list settings. To set (or update) the current tab's home click "Set Home".

Tip: Note that also Quick Searches are supported. All these terms can be homes:

```
E:\Test\Sort\Ignore Articles?a*
C:\Program Files;C:\Program Files (x86)? /n
E:\Test\SyncSelect? /flat
This PC?pop /T
Desktop\Desk?*.txt /maxdepth=1 //search this level plus one
```

Go Home: To go to the current tab's home click "Go Home".

Shortcut: Alt+Home

Lock Home Zone: First you have to define a home, then you may lock your tab to the Home Zone, which includes the home path and all its subfolders (in other words, the home branch). You can freely move inside the home zone and stay inside the tab, but if you browse to a destination outside the home zone, the tab stays where it is and a new tab will be automatically opened. Home-zoned tabs are recognizable by a dotted underline. Note: a tab that's home-zoned and tight-locked (Ctrl+L; locked to a specific path) is treated as tight-locked.

Lock Location: Lock the current tab. A locked tab will never leave its current location but instead open

new tab automatically.

Shortcut: Ctrl+Shift+L

Set Visual Filter

Set a Visual Filter to the current file list. See Visual Filters for syntax.

Shortcut: Ctrl+J

Toggle Visual Filter

Toggle between no Visual Filter and the last used Visual Filter or Power Filter.

Filter By Selection(s)

The command makes the list show only those items that are selected in the moment you apply the command. The filter is hard-coded to the caption "sel". A nice thing about it is that the "Toggle Visual Filter" command will restore the filtered view even when the items are not selected anymore. Even across sessions. Even in other tabs or folders if their contents share the same filenames.

Shortcut: Shift+Alt+J

Type Stats and Filter...

Pops a menu of all file types contained in the currently listed items (or of the currently selected items if more than one item is selected), ordered by count (most frequent on top) or by extension (ascending). Click any of the types to filter (Visual Filter) the list by this type. Works also on Branch View and Search Results.

When there are 32 or more items the interface is a list instead of a menu. Better to handle.

Note: There is a toolbar button available with the same name and functionality. You can modify the sort order of the popup menu via right-clicking the button. There are two options, Sort by Count and Sort by Extension.

Tip: You can use the menu to (un)select files by type.

- Hold CTRL: Add all files of this type to the selection.
- Hold SHIFT: Remove all files of this type from the selection.

Toggle Live Filter: Toggles the Live Filter (No Filter / Last Filter) keeping the input focus where it is. The Live Filter Box is shown if necessary when the filter is toggled on.

- The last Live Filter is remembered per tab.
- If there is no last Live Filter the command does nothing.
- The functionality is identical to pressing F3 when the focus is in the Live Filter Box.

Shortcut: Ctrl+Alt+F3

Restore Last Closed Tab: Restores the tab that was last closed in the current session (not remembered across sessions). The restored tab is automatically selected.

Close Other Unlocked Tabs: Close all unlocked tabs apart from the current tab.

Shortcut: Ctrl+Shift+W

Close Other Tabs: Close all tabs apart from the current tab.

Close Tabs to the Right: Does what it says. If the selected tab was closed (if it was to the right of the right-clicked tab), the rightmost tab is now automatically selected.

Close Tab: Close the current tab.

Shortcut: Ctrl+W, Ctrl+F4

Mini Tree (Submenu)

See also Mini Tree.

Mini Tree. Toggles between Maxi Tree and the last used Mini Tree. A Mini Tree will show only the tree paths that have been actually used.

When toggling from Maxi to (last) Mini Tree, the current path is kept even if it is not part of the last Mini Tree. So, no change of location will happen using this command.

Mini Tree from Here. Radically reduces the tree to just the current location. Turns on Mini Tree mode if it is currently off.

Mini Tree from Recent Locations. Reduces the tree to the recently visited locations.The history is scanned backwards until the 12 deepest paths are collected that are not in a mutual parent-child relation. Turns on Mini Tree mode if it is currently off.

Mini Tree from Current Tabs. Reduces the Mini Tree to show just the paths of all tabs in both panes. Turns on Mini Tree mode if it is currently off.

Load Previous Mini Tree. When in Mini Tree mode, toggles between this and the previous Mini Tree. When in Maxi Tree mode, loads the Mini Tree that was in use before switching to Maxi Tree. The previous Mini Tree is remembered across sessions.

Set as Favorite Mini Tree... Stores the current Mini Tree as Favorite Mini Tree. Overwrites any previously stored Favorite Mini Tree (after OK-ing a prompt). Only available/applicable in Mini Tree mode.

Load Favorite Mini Tree. Loads any previously stored Favorite Mini Tree. Turns on Mini Tree mode if necessary.

Hide Current Folder. Hides the current folder from the Mini Tree (not from your hard disk). Since there must be a selection the parent folder is auto-selected.

Hide Siblings. Hides the current folder's siblings from the Mini Tree.

Paper Folders (Submenu)

See also Paper Folders.

New...: Create new empty Paper Folder.

Open...: Open existing Paper Folder.

Save: Save Paper Folder to file. You normally don't have to do this -- saving is fully automatic. But there might be reasons to perform an explicit save now, so here is the command to do it.

Save As...: Save current Paper Folder under a new name and load it.

Save Copy As...: Save current Paper Folder under a new name but keep the current one loaded.

Remove Selected Items: Well, it does what it says. Note that this does not delete the items from the file system! Nevertheless you are prompted before the items are removed.

Empty Paper Folder: Remove all items from the Paper Folder. Again, this does not delete the items from the file system! Nevertheless you are prompted before the items are removed.

Toggle Paper Folder: On a normal folder, create a Paper Folder from the current list. Only Browse and Find tabs can be converted to a Paper Folder. On a Paper Folder, go back to the last used normal folder in this tab.

Folder View Settings (Submenu)

See also Folder View Settings.

Enable Folder View Settings: Check to enable the Folder View Settings feature, i.e. make the commands available and auto-restore folder-specific view settings.

Save Folder View: Save and remember the current folder view settings. If a folder view is currently active it will be updated to the current view settings.

Restore Folder View: Reverts the current folder view to the saved settings.

Edit...: Edit the active folder view.

Remove: Remove the active folder view.

Apply this Folder View Also To...: Apply the current folder view settings also to another folder name/ pattern.

Define this Folder View as Default: Save the current folder view settings as default folder view (used for any folder with no specific folder view).

Automatically Apply Default Folder View: Tick it to automatically apply the default folder view (if one was defined) if a new location does not have its own folder view defined.

Apply Default Folder View: Applies the default folder view (if one was defined) to the current list.

Apply Previous Folder View: Re-applies the folder view that was active just before one of the following commands was triggered:

- Apply Default Folder View
- Restore Folder View
- Apply Previous Folder View (sic, the command itself is one of them)

The command allows you to toggle between the last two folder views. Note: The previous folder view is not remembered across sessions.

Manage Folder Views...: Opens a list of all stored Folder Views. If a Folder View is currently active it is preselected in the dialog.

You can directly jump to the "Edit Folder View" dialog from the list: Select a Folder View and click OK, or double-click a Folder View. The dialog will automatically reopen when you're done with the Edit Folder View dialog you opened from there.

You also can delete Folder Views from here: Select a Folder View and press DEL (or use the item's context menu). Note that deletions are only performed when you OK the dialog.

Tip: You can use this dialog to apply any of the defined Folder Views to the current folder (just ad hoc and temporarily, without making it the permanent view for this folder): Hold CTRL while you OK the dialog. That way you can have an arsenal of Folder Views from which you can quite easily select one for instant use.

Refresh

Refreshes Tree and List.

Note that the List is not reset (i.e. any selections and the scroll position are kept). To reset the file list use "Reset List" (Ctrl+Shift+F5). The Tree is only refreshed when it is visible.

Shortcut: F5

Auto-Refresh

See "Auto-refresh" in Configuration.

Shortcut: Ctrl+Shift+R

Suspend Auto-Refresh

Will immediately and temporarily stop (or reactivate) any auto-refreshing for the current location, without changing the general Auto-Refresh setting. Auto-Refresh is automatically reactivated (only if Auto-Refresh is ON, of course) when the current location is changed.

Refresh Tree

Updates the whole Tree.

Shortcut: F4

Reset Tree

Completely rebuilds a fresh Tree.

The expansion state of the whole tree is preserved when it is a locked Maxi Tree and Configuration | Tree and List | Remember state of tree is enabled.

Shortcut: Ctrl+Shift+F4

Refresh Current Folder

Updates just the current folder and its child folders (i.e. the current branch) in the Tree.

Shortcut: Shift+F4

Refresh List

Updates the current list data, but keeps any selections and scroll position. Also refreshes the tab icon.

Shortcut: Ctrl+F5

Reset List

Updates the data, scrolls back to top, sets focus to the first item (if any), and unselects any selections. Also refreshes the tab icon.

Shortcut: Ctrl+Shift+F5

Calculate Folder Sizes

Calculates the folder sizes of all folders currently listed. Further, if any folders in the list are selected, then only the sizes of selected folders are calculated. Cached folder sizes (that might be stale) are refreshed (but see Tip below).

Empty folders will display a "[Empty]" in the file list and are thus distinguished from non-empty folders with zero bytes content, which display a "0" ("zero" char). Empty folders will sort before 0-bytes folders.

Shortcut: Shift+F5

Show Items (Submenu)

Show Floppy Drives: See under Configuration.

Show CD-ROM Drives: See under Configuration.

Show Hidden Drives: See under Configuration.

Show Hidden Files and Folders: See under Configuration.

Show System Files and Folders: See under Configuration.

Hide Protected Operating System Files: See under Configuration.

Show Junctions: See under Configuration.

Show Folders in List: Uncheck it to show only files in the List.

Set Global Visual Filter: Set the Global Visual Filter.

Toggle Global Visual Filter: Toggle the Global Visual Filter.

Ghost Filter: Toggle the Ghost Filter.

Edit Ghost Filter...: Opens the dialog to edit the patterns defining the items to be hidden by the Ghost Filter.

Show Custom File Icons: Show/Hide custom file icons everywhere. Mirror of Configuration | General | Refresh, Icons, History | Icons | Show custom file icons.

Shortcut: Alt+F9

Caches (Submenu)

Refresh Thumbnails

Refreshes all thumbnails in the current list, no matter the setting of "Show cached thumbnails only" and "Create all thumbnails at once".

Recreates the thumbs cache of the current folder (when thumbs caching is enabled).

If a list does not contain any items with thumbnails, but a pair of thumbnail cache files still exist for them (for images that are no longer here), invoking "Refresh Thumbnails" will permanently delete those thumbnail cache files.

Refresh Selected Thumbnails

Recreates the thumbs cache for all selected thumbnails.

Create Missing Thumbnails

Creates all missing thumbnails in the current list.

This command complements the setting Configuration | Thumbnails | Show cached thumbnails only and is only enabled if that setting is ticked.

Refresh Icons

Refreshes all displayed file and folder icons.

Update New Items Menu

Rescan the "NewItems" folder.

3.4 Go

Go Menu Commands

Go to Previous Location

Jump to the previous tab/mode/location (useful to zap back and fore between two locations). If the previous location has been on another tab, this tab is opened (if it still exists). If the previous tab does not exist anymore, the location is opened in a new tab. Any previous selections, the scroll position, and the focused position are restored.

Note that "previous" means within the session. So right after startup the command (and the corresponding toolbar button) will do nothing.

Shortcut: F7

Go to Previous Item in List

Jump to the previously focused and selected item in the current file list.

Shortcut: Shift+Alt+F7

Go to Last Target

Beams you to the most recent target of a move/copy file operation that was performed inside the current XYplorer instance and session. That's good news for the paranoid: a quick [Ctrl+Alt+F7] (or right-handed [AltGr+F7]) will carry you to the place you've just moved/copied some files to, to check personally whether the files arrived in a good condition. Dual pane without the space-waste... ;)

As a special service the copied or moved items found in that target location are auto-selected when you go there using this command. Every new copy/move will overwrite the previous last target data. And, just like the target folder itself, these selections are not remembered between sessions.

Shortcut: Ctrl+Alt+F7

Top

Jump to the top folder (Drive, \\Server, Desktop) of the current folder.

Shortcut: Ctrl+Home (only if the Tree is focused)

Up

Jump to parent folder of the current location.

Shortcut: Backspace, Alt+Up

Down

Takes you down to the most recently used subfolder of the current folder (if any).

Shortcut: Shift+Backspace, Alt+Down

Breadcrumb...

You could also call it "Up-Down-History", "Popup Breadcrumb", "Vertical Breadcrumb", or "Relative History".

You'll immediately grab the idea: When you click this command (or click the arrow besides the "Up" toolbar button, or right-click this button) a menu is popped up presenting you a selection of destinations directly related to your current location. The Up-part is simply generated by parsing the current path. The Down-part is drawn from the recent history.

- It's a breadcrumb that takes no screen space at all if you don't want it to. It works without toolbar, address bar, whatever bar. Simply press [Ctrl+Backspace] to have it pop up.
- It's a breadcrumb that's manageable by keyboard-only. Even one-handedly when using the right Ctrl-key.
- It's a breadcrumb that goes not only back up but also back down.

Tip: Right-clicking the icon of a tab pops a breadcrumb menu for the tab's path.

Shortcut: Ctrl+Backspace

and: Ctrl+Alt+Backspace; AltGr+Backspace (to avoid the ding sound)

Drives...

Pops up a menu featuring all currently available drives. Empty removable and unaccessible network drives are skipped.

Back

Step back in history.

Shortcut: Alt+Left

Forward

Step forward in history.

Shortcut: Alt+Right

History...

Pops up a list with your recent browse history. The first/oldest entry is at the bottom, the last/newest at the top. The history holds up to 256 entries and remembers them between sessions.

Recent Locations...

Shows the MRU list of locations from most recent (at the top, with number 1) to most ancient. Double-click an item to jump to it. The MRU holds maximally 128 items.

Note that there's a toolbar button of the same name that pops a menu of the most recent locations.

Hotlist...

The Hotlist is a brand-new interface to your history, with the intention to present the history in a way that makes it as easy as possible to quickly go back to one of the places you have recently been -- as

opposed to the traditional presentation of the history which makes it easy to go back and forth on your recent track.

The Hotlist is made like this:

(1) All its items are taken from the Recent Locations list.

(2) Computer and Network Places are removed (to keep it tight).

(3) It's ordered alphabetically.

(4) It's fresh: only existing paths are shown (the history can be outdated in this aspect).

(5) It's compressed: only the *leaves* of the branches are displayed. If your history contains

C:\

C:\Program Files

C:\Program Files\XYplorer

then only the latter is shown in the Hotlist. It is assumed that the skipped parent items have just been transit stations on the way to the final "leaf" item, C:\Program Files\XYplorer, and hence aren't interesting (or "hot") targets.

(6) It's grouped by separators, one section per drive.

Shortcut: Ctrl+H

Tablist...

Pop up a menu containing all currently existing tabs.

Shortcut: Ctrl+Shift+T

Aliases...

Here you can select an alias from a list of all aliases you have defined. If you don't remember what an alias stands for, you can check in Tools | List Management | Aliases, and maybe find a better name for it.

Go Now

Same as pressing ENTER in the Address Bar. The advantage: You don't have to focus the Address Bar, just press Ctrl+Shift+G wherever you are. Highly useful when running small scripts through the Address Bar, e.g. scripts that work on the currently selected item.

Shortcut: Ctrl+Shift+G

Go to...

Enter/paste a location (a path) to jump to. You can as well enter a path/file name to jump directly to a specific file in that path.

Shortcut: Ctrl+G

Go to Line...

Enter/paste a line number to jump to.

Shortcut: Ctrl+Shift+Alt+L

Go to Application Folder

Well, brings you to the application folder.

Shortcut: Ctrl+Shift+Alt+G

Go to Application Data Folder

It will beam you to the application data path.

3.5 Favorites

Favorites Menu Commands

Favorite Folders

All locations you have previously defined as favorites (e.g. by pressing Ctrl+B = Toggle Favorite Folder).

Note: The Favorite Folders submenu pops up when you right-click anywhere in the free area (not on a folder) of the Tree.

Toggle Favorite Folder

(Un)marks a folder as favorite. Favorite Folders are added to the Favorites Listing within this menu. When you rename a favorite folder the favorites are automatically updated so that the link is kept alive.

In the Tree Favorite Folders can be shown bold if you tick "Tools | Customize Tree | Mark Favorites (Bold)".

Shortcut: Ctrl+B

Dual Locations are supported here (and generally in all places that can take a location). In a Dual Location both folders are separated by || (double pipe). The first folder will be opened in the active pane, the second in the inactive pane, quite useful if you work in dual pane mode. For example:

```
\\VEGA\Users\Donald||D:\
```

You can use this syntax to only browse the inactive pane:

```
||D:\download
```

Note that dual pane mode is automatically enabled when necessary, and dual location favorites are not bolded in the tree.

With Dual Locations you can also use fancy stuff like filters and searches in both locations:

```
D:\Test|a*||E:\Test|b*        //visual filters for both panes
D:\Test?cat||E:\Test?dog      //quick searches in both panes
|cat|||dog                    //filter active pane by cat, inactive pane by dog
?*.txt||?*.jpg                //search active pane for *.txt, inactive pane for *.jpg
```

Of course, you can do different things on each side of ||. Filter here, search there, etc.

Even dual scripts work. Note that internally the inactive pane becomes active will being processed, so <curpath> works on both sides:

```
echo "active: <curpath>";||echo "inactive: <curpath>";
```

Dual locations have their own default icon in the address bar, lists, and menus.

Tip: The right-click menu of the "Favorite Folders" toolbar button lets you add/remove a Dual Location to/from your favorite folders with a single click on the command Toggle Dual Favorite Folder.

Manage Favorite Folders...

A shortcut to menu Tools | List Management | Favorite Folders. Here you can add or remove favorite folders, and also reorder the list and add menu separators by inserting items with a single hyphen (minus sign) as caption.

Captions: You can also define custom menu captions for the items. Simply prefix the caption in quotes (optionally followed by any number of blanks). These will work identically:

```
"caption" location
"caption"location
"caption"      location
```

This allows you to attach catchy aliases to long and hard-to-read paths! For example:

```
"Transparent PNGs" E:\VB-Don\TestFiles\Media\image-formats\trans\png\ARGB 32bit\
```

Custom Icons: You can append a custom icon specification to the caption, separated by a | character:

```
"caption|[icon spec]" location
```

The icon spec can be an icon file (path resolved relative the XYplorer icon path `<xyicons>`), any other file, a toolbar button key (prefixed with :), or some icon resource with icon index (exe; dll; cpl; ocx; scr; icl; bpl; wlx; wfx; wcx; wdx; acm):

```
"caption|Kiss.ico" location
"caption|C:\WINDOWS\notepad.exe" location
"caption|:dice" location
"caption|%winsysdir%\shell32.dll /160" location
```

Scripts as Favorites: Here is a custom favorite folder where the "location" is a quick script (prefixed by script marker ::):

```
"Script to append modified date" ::rename , '*-<datem yyyymmdd>'
```

Will show "Script to append modified date" (without quotes) in the menu; when you click it, the script is executed. So you now have another, very comfortable way to place some heavy-rotation scripts right under your mouse button!

Fuzzy Favorites: Favorite Folders and Favorite Files support the wildcards * (stands for "any number of characters including none"). With a Fuzzy Favorite the first match will be used. This can be useful when you need to reference Favorites with variable names.

Toggle Highlighted Folder: Throws a highlight (color configurable) on the current folder. It's 8 pixels wider than the selection rectangle to keep it visible when the folder is selected.

Set Highlight Color: Define the highlight color for the current tree folder. Enables Highlighted Folder for this folder if not yet enabled anyway.

Toggle Boxed Branch: Draws a colored rectangle on the background of the whole branch.

Set Box Color: Define the box color for the current tree folder. Enables Boxed Branch for this folder if not yet enabled anyway.

Favorite Files

All files you have previously defined as favorite files. You can directly jump to any file by a single click.

Tip: Tick Configuration | Controls & More | Miscellaneous | Open favorite files directly to open the files instead of jumping to them,

Toggle Favorite File

(Un)marks a file as favorite. Applies to the currently focused file.

Manage Favorite Files...

A shortcut to menu Tools | List Management | Favorite Files. Here you can add or remove favorite files, and also reorder the list and add menu separators by inserting items with a single hyphen (minus sign) as caption.

Special System Folders

Browse to various special system folders.

3.6 Tags

Tags Menu Commands

Note: If this menu is not visible, you can enable it in Configuration | Other | Features.

For general information about file tagging see Labels, Tags, Comments, and Extra Tags.

Labels (Submenu)

Here you can assign any of the currently defined labels to all currently selected items, or remove them by selecting "None".

Note: If Configuration | Information | Tags | Toggle tags by column click | Label is ticked, the commands in this menu also toggle the labels (apply the same label again to toggle it off).

Apply Last Label

Apply the last used label to all currently selected items.

Tags (Submenu)

Add Tags...

Add more tags to the tags of all selected items.

Add Tags by List...

Add more tags to the tags of all selected items (select from the Tag List).

Edit Tags...

Edit the tags of all selected items (based on the tags of the current item).

Edit Tags by List...

Edit the tags of all selected items via a checkbox tag list. The checkboxes are pre-ticked based on the tags of the current item.

Remove Tags by List...

Remove particular tags from all selected items (select from the Tag List).

Remove All Tags

Remove all tags from all selected items.

Find by Tags...

Find items by particular tags that you can enter into a text box.

Note that the scope of the search (Everywhere, This Branch, Here) can only be set via the right-click menu of the toolbar button "Find by Tags".

Find by Tag List...

Find items by particular tags that you can select from a list of all used tags.

Note that the scope of the search (Everywhere, This Branch, Here) can only be set via the right-click menu of the toolbar button "Find by Tags".

Update Tag List

Update the tag list to the tags currently used in the tags database (usually tag.dat).

Add Last Tags

Add the last added tag(s) to the tags of all selected items.

Comment...

Here you can assign a comment to all currently selected items, or remove any comments by entering nothing.

Remove Items from Database

Select it to remove all selected list items from the tags database, i.e. remove all their labels, tags, comments and extra tags.

Export Local Tags

Stores all local tags (i.e. tags in this branch) in a new local database called "XYplorerTag.dat" which is created in the current path. The paths of the tags are stored relative to "XYplorerTag.dat" (so you can move or rename the branch without destroying the tags). The tags are not removed from the main tags database.

Import Local Tags

Imports local tags from the database "XYplorerTag.dat" (which should exist in the current path, usually created by a previously performed "Export Local Tags"). The tags are merged permanently into current tags database.

Load Tags Database...

Load a different tags database. The 16 recently loaded tags databases are remembered across sessions in an MRU list.

You can also specify the name of a new not-yet-existing database file. After OK-ing the dialog all current tags are removed from memory, and Labels are reset to the 7 default colors.

Tip: You can use this function to perform a "Revert to Saved". Simply press OK on the DB that's prefilled in the input dialog. It's the current tags database. You will lose any dirty tags.

Reload Tags Database

Reload the tags database from disk, aka revert to saved. Especially useful when XYplorer is run in a company where many users share read-only access to a common tags database which can be altered only by the administrator (see Admin Settings).

Note that reloading the current database leaves your unsaved ("dirty") tags untouched. In detail:

- Non-dirty items are fully replaced with the imported items: All current fields are set to the imported values. So non-empty fields might end up empty.
- Dirty items are merged with the imported items: Only if the current field is empty and the imported field is not, then the current field is set to the imported value. So non-empty fields will not be changed. Reason: Your not-yet-saved tags are protected this way. Otherwise the auto-refresh of the tags (Configuration | Tags | Auto-refresh tags) would destroy them.

3.7 User

User Menu Commands

Note: If this menu is not visible, you can enable it in Configuration | Other | Features.

Manage Commands...

Open the Manage User-Defined Commands interface.

Tip: Dbl-clicking an item in the Commands list will trigger the "Browse..." or "Edit..." button (if there is any).

Shortcut: Ctrl+Alt+F9

Goto, Open, Open With... etc

These submenus contain all the User-Defined Commands that you have defined before. See next paragraph.

User-Defined Commands

Introduction

User-Defined Commands (UDCs) are basically menu commands (all located in menu User), where you decide (1) what they do when clicked, (2) and what their caption is. Optionally, as with all menu commands in XYplorer, you may assign keyboard shortcuts to them.

Example 1: You currently work on a project with distributed locations you need quick access to. Define a UDC of category Go To for each of the Locations and assign a KS (Keyboard Shortcut) to each of them. This will take you maybe 20 seconds, and then, say, Ctrl+1 will take you to \\yourserver\theshare\projectdata\ and Ctrl+2 will take you to E:\project\images\.

Example 2: You like to open certain image files either with Photoshop or with ACDSee. Define two UDCs of category Open With, one pointing to Photoshop, the other one to ACDSee. Again, assign a KS (Keyboard Shortcut) to each of them, and you are done. Ctrl+Alt+P (for example) will open all selected files with Photoshop, Ctrl+Alt+A (for example) will open all selected files with ACDSee.

Example 3: You regularly have to do certain rename operations for which you need a complex Regular Expression you always tend to forget. Define a UDC of category Rename, enter your Regular Expression into the Pattern field, and give it a descriptive caption (so that you later know what it does). Again, assign a KS (Keyboard Shortcut) to the UDC, and you are done. From now on, a single key stroke, e.g. Ctrl+Shift+8, will apply the rename operation to all selected items.

Example 4: You quickly want to list only the *.exe and *.dll files present in the current folder. Define a UDC of category Go To and set the Location to |*.exe;*.dll – this will activate a Visual Filter for the

current location. As you see in this example, UDCs know the full richness of XYplorer's location syntax. The possibilities are endless.

How to create a User-Defined Command

Open the Manage Commands (Ctrl+Alt+F9) interface.

Select a Category.

Click the "New" button and select Add New Command, or simply press INS.

Now enter the Command Properties. First you specify the main argument: that's the critical piece of data. It's where you specify the object of the Action which can be (depending on the Action) a Location, an Item, an Application, or a Pattern. Often it will be a file or folder, in which case a Browse button to the right of the edit field is enabled.

Required fields are marked by a bold label.

Optionally enter the Caption (for the menu item). If left empty, the main field (bold) will be used for the caption.

The changes are auto-applied when you OK the dialog, or select another category, another command, or create a new command. To undo any changes in the current Command Properties (an asterisk is shown in the frame title) press ESC. To undo all changes you made since opening the dialog simply Cancel the dialog.

Repeat for all UDCs you need.

Finally press OK to apply your changes to the current session of XYplorer. The new menus will be generated in a blink. Open the User menu and see yourself...

Now you can assign a Keyboard Shortcut to your new UDC. Either click the Assign Keyboard Shortcut button and choose one, or open Customize Keyboard Shortcuts (Shift+F9) from menu Tools (the UDCs are listed under category "User").

Of course, as always in XY, the UDCs will be saved to file on exit unless otherwise wanted. This file is called udc.dat.

On KeyUp

For UDCs with a Keyboard Shortcut assigned, this checkbox gives you control about the exact point of action. When checked, the command is triggered when the key is released, else (default) the command is triggered when the key is pressed down.

It is actually recommended that you enable "On KeyUp" on UDCs that pop a menu (and have a keyboard shortcut assigned, of course); otherwise you might experience that you trigger one of the menu items on releasing your keyboard shortcut because the released key is interpreted as an accelerator key!

How to quickly edit an existing User-Defined Command

You can directly open the Manage User-Defined Commands dialog for a particular user command by holding CTRL while you click the menu item pointing to the command. The command will not be triggered then, of course.

How to smartly pre-fill the dialog when creating a new User-Defined Command

The Toolbar's Manage User-Defined Commands button's right-click menu enables you to directly create a new UDC with the argument field preset to some current context value. How? Simply hold CTRL while clicking one of the menu items! The preset values adjust to the command category:

- "Go To", "Open", "Open With", "New", "Load Script File" -> current file (if any is selected)

- "Rename" -> [empty]

- "Run Script" -> [empty]

- else -> current path

How to quickly duplicate a User-Defined Command

You can duplicate the current command by pressing "New" or by pressing Ctrl+Ins. If the current command has a caption, then the new command's caption will have "Copy of" prefixed.

Command Line Parameters are supported

The Open and Open With commands, when applied to applications, support the usage of Command Line Parameters. Of course, it is the called application that controls (1) which parameters are recognized, (2) what they do, (3) what format they must have, and (4) in which order they are expected. So, to use this feature, you must know the command line syntax of the application you want to open files with.

The only rule you have to know is: If you use Command Line Parameters, then the application term has to be in quotes!

The possibilities arising from the support of parameters are absolutely astounding. For the following examples, I chose WinZip as the called application.

Example 5: Adding selected items to an archive. Open menu User | Manage Commands, and select category "Open With". Set the option to "Pass all the items to the same instance...". Then enter this line into the Application field in UDC | Open With:

```
"winzip32" -a UDCmade.zip <items>
```

<items> is a XYplorer variable, that in this context is set to a blank-separated list of all currently selected list items. Triggering this UDC will create (if not already existing) the archive UDCmade.zip in the current path and pack all currently selected files into it. If want the *.zip in some other place than the current path, state it with the desired full path.

Example 6: You may as well use XY date variables, e.g.:

```
"winzip32" -a UDCmade-<date yyyymmdd>.zip <items>
```

will create UDCmade-20071107.zip if today is 2007-11-07.

Example 7: Create a password protected archive and name it depending on the current folder:

```
"winzip32" -a -s"Secret password " <curfolder>-Secret.zip <items>
```

Remember that the quoting of the blank-containing password is part of WinZip's command line syntax! XYplorer is just passing this on to WinZip. What XYplorer does, however, is resolving the new XYplorer variable <curfolder>: it is set to the name of the current folder (not the whole path).

Example 8: Extracting items from an archive to some folder:

```
"winzip32" -e <items> newfolder
```

Will extract all items of the selected archive file into a newly created folder "newfolder" under the current path. Of course, you can as well give a full path for the target folder. Note that <items> here stands for the archive to extract! This example will only succeed when just *one* archive file is selected. This is a restriction of WinZip's command line syntax, not of XYplorer.

Example 9: Extracting items from an archive to some auto-named folder:

```
"winzip32" -e <items> <curbase>
```

The new XYplorer variable <curbase> will be set to the base (= name without extension) of the currently selected file (here: the archive to extract) -- it is always the file that is displayed in the status bar. For the case that the base contains blanks, you should put it in quotes else WinZip will choke:

```
"winzip32" -e <items> "<curbase>"
```

Of course, you can combine variable as you like:

```
"winzip32" -e <items> "<curbase>_<date yyyy-mm-dd>"
```

If the archive is called "test.zip", this would create a subfolder named "test_2007-11-07" and unpack the archive into it.

Example 10: With any compare-tool accepting two-way comparisons via command line you can easily set up a one-click file comparison. Assuming you have a 3rd party software installed and registered called "compare.exe" then you could do:

```
"compare" <item1> <item2>
```

This will your tool with the first two of all selected items in the given order. Any other selected items are ignored.

Some interesting examples for the "New" category

If the currently focused file is E:\Pics\ChristinaAguilera.jpg, dated 20070602:

```
============================================================
Name field:                New empty file created:
------------------------------------------------------------
<curfolder>.txt            Pics.txt
<curtitle>.txt             ChristinaAguilera.jpg.txt
<curbase>.txt              ChristinaAguilera.txt
<curbase>-<curext>.txt     ChristinaAguilera-jpg.txt
<curbase>-<datem yyyymmdd>.txt ChristinaAguilera-20070602.txt
<curbase>-<date yyyymmdd>.txt ChristinaAguilera-20071209.txt
============================================================
```

The variable <curver> will extract the version number of the current file (if it has one). If the currently focused file is E:\XYplorer.exe, v6.60.0053:

```
------------------------------------------------------------
<curbase>_<curver>.txt    XYplorer_6.60.0053.txt
------------------------------------------------------------
```

If you set the Source field to <curitem>, you can emulate the behavior of "Copy Here With Last Modified Date" (see below). Or you can "Copy Here With Version":

```
============================================================
Name field:                   Clone of Source created:
------------------------------------------------------------
<curbase>_<datem yyyymmdd>.<curext>  XYplorer_20071207.exe
<curbase>_<curver>.<curext>     XYplorer_6.60.0045.exe
============================================================
```

If a source is defined then the following variables in the Name field refer to the source: <srcbase>, <srcext>, <srctitle>, <srcver>, <srcdatem yyyymmdd>, <srcdatec yyyymmdd>, <srcdatea yyyymmdd>. They obviously correspond to <curbase>, <curext>, <curtitle>, <curver>, and <datem yyyymmdd>, which always refer to the current item.

Passing the full path

You may as well pass a full path/name in the Name field (only if no path is passed, the current path is taken as default). If the target folder does not exist it is created on the fly. The "last target" path (used by Ctrl+Alt+F7) is set to the target folder.

Further remarks

No limit: There's virtually no limit to the number of commands, although, being menu-based and having a finite number of keys on your board you will hit a usability wall at some point.

Accelerators: You can create an accelerator simply by prefixing the letter to be used by the ampersand sign (&). If you want to use the & character itself in your caption, simply double it, e.g. "This && That".

Separators: If you want to organize your menus into sub sections you can use separators. Simply create a new command, and put the minus sign (-) as argument.

Short forms for applications: You will have noted the use of "winzip32" above. If the EXE to run is registered (known by the registry) it is not necessary to state the full path. The base name of the EXE file is enough. Both ways will work the same (UDC category OpenWith):

```
"C:\Program Files (x86)\WinZip\Winzip32.exe" -a UDCmade.zip <items>
"winzip32" -a UDCmade.zip <items>
```

Of course, the latter version is more portable, because it will call WinZip successfully on systems where it has a different installation path from your home system.

Specifying Menu Icons: You can optionally pass a pointer to an icon along with the caption. The icon

will be shown in the User menu. The general form is MenuCaption|MenuIcon, where MenuIcon can have various formats:

```
    MenuCaption|C:\Beta\                    full path/file to file or folder
    MenuCaption|fun.ico                     path relative to the Icons Path (default:
<xydata>\Icons)
    MenuCaption|<xydata>\Icons\fun.ico      support of XY native variables
    MenuCaption|:minitree                   internal icon (toolbar buttons resource)
    MenuCaption|                            nothing (no icon shown)
```

Options: The Options button pops a small menu with two options:

- Show Menu Icons: Tick it to show icons in the User-Defined Commands menus.
- Show Status Bar Message: Tick it to show a status bar message before and after each triggered User-Defined Command.

Variables: See Variables.

3.8 Scripting

Scripting Menu Commands

Note: If this menu is not visible, you can enable it in Configuration | Other | Features.

Run Script...

Enter and execute a script directly. The script can be multi-line, and you can enter more than one script (in which case a menu will pop up).

The last entered script is remembered between sessions. Note that the dialog has context sensitive help: F1 will open the Scripting Commands Reference.

Run Script Again

Runs the script previously entered in Run Script... or Try Script... again.

Step Mode

Toggle Step Mode. Stepping through scripts is recommended for debugging scripts, and for carefully testing scripts that you have received from other authors.

Syntax Checking

Tick it show a warning on dubious syntax. Recommended for debugging scripts.

Tip: You can suppress further validation for the current script by holding Ctrl while pressing "Continue" in the "Dubious Syntax" dialog.

Try Script...

Like Run Script... above, but: Step Mode is always enabled for the script you run from the Try Script window. The previous state of Step Mode is restored when done.

Shortcut: Ctrl+Alt+S

Permanent Variables

Pops a list of all Permanent Variables that are currently in memory. Tip: The list has a right-click menu.

You can double-click a variable in the listing to show its value. Here you can edit the variable values after double-clicking them (or right-click / Show Value) in the list. You can as well unset these variables individually directly from the right-click menu in the variable list (command Unset).

Load Selected Script File

Allows you to directly load a script resource from the file list. It's your responsibility to ensure that the selected file is a valid script file.

Load Script File...

Here you can load a script file (*.xys) from application data path. If it contains exactly one script it will immediately executed, else a menu is popped up.

Go to Scripts Folder

Brings you to the "Scripts" folder, a subfolder of the application data path.

3.9 Panes

Panes Menu Commands

Note: If this menu is not visible, you can enable it in Configuration | Other | Features.

Dual Pane

Show/hide the inactive pane. In other words: Toggle between Dual Pane and Single Pane mode.

Shortcut: F10

Horizontal Panes

Toggle Horizontal Panes (Top/Bottom) and Vertical Panes (Left/Right).

Shortcut: Ctrl+F10

Toggle Active Pane

Activate the inactive pane.

Shortcut: Ctrl+Alt+F10

Note that also focusing the inactive pane will activate it: by clicking into it, or by Tab key.

Note that you can also toggle panes in Single Pane mode. Which means you can easily switch back and forth between two completely independent sets of tabs.

Move to Other Pane

Move all selected item(s) of the active pane to the inactive pane.

Shortcut: Shift+F6

Copy to Other Pane

Copy all selected item(s) of the active pane to the inactive pane.

Shortcut: Ctrl+F6

Backup to Other Pane

Backup all selected item(s) of the active pane to the inactive pane.

Note: This command comes pretty close to a nice Two-Way Folder Sync: Do it left-to-right and right-to-left, and you end up with two identical folders, and - depending on your Backup configuration - with all the newest versions from either side.

Shortcut: Ctrl+Shift+F6

Move/Copy Tab to Other Pane

Move/Copy the current tab to the inactive pane. All properties of the tab are taken along, including current selections and scroll position. The newly created tab is auto-selected and the target pane is

activated. If there's only one tab left on a pane it will not be removed, of course.

The position of the new tab is the same as if you'd created it within the same pane.

The command also works in single pane mode.

Tip for mouse users: You can move/copy tabs from one pane to the other by dragging them to their new position. The default action is move (the source tab is removed from its pane), but you can force a copy by holding CTRL.

Go to Other Location

Use it to go to the inactive pane's location, in other words to make the active pane's location the same as the inactive pane's.

Go Here in Other Pane

Use it to make the inactive pane's location the same as the active pane's. It's the reverse of the above "Go to Other Location".

Tip: Hold ALT when clicking an item in the Tree to go to it in the inactive pane.

Swap Locations

Swaps the locations of both panes.

Tip: Hold SHIFT to open the locations in a new foreground tab on each pane.

Sync Browse

Here you can toggle synchronous browsing of both panes. When enabled the inactive pane will auto-change its location in sync with the active pane.

Usage

- Enable Dual Pane (F10). Point both panes to locations that you want to sync-browse. Tip: Hold ALT and you can use the Tree to navigate the inactive pane.
- Turn on Sync Browse.
- Now when you change locations in one pane, the other pane will attempt to do the same relative move.
- When Sync Browse is not possible (because it would lead to a non-existing location) the status bar displays "cannot sync browse this location" and the synced pane will not change its location.
- Auto-Select Matching Items: When in Sync Browse mode selecting "A.txt" in the active pane will optionally auto-select "A.txt" in the inactive pane (if it exists there). Useful when both panes have different sort orders, differing file listings, or different view modes. To enable this feature tick "Configuration | Tabs and Panes | Dual Pane | Sync Browse | Auto-select matching items", or "Auto-Select Matching Items" in the right-click menu of the "Sync Browse" toolbar button.

Notes

- When you change to another tab in any of the panes, Sync Browse is suspended until both original tabs are active again.
- Sync Browse also works when the inactive pane is invisible.

- There's also a toolbar button "Sync Browse".
- Sync Browse is not stored between sessions.

Sync Browse is a great time saver when performing moves/renames and other reorganizations on mirrored drives. Another usage is to browse one location in two different simultaneous views, e.g. Details and Thumbnails.

Sync Scroll

Here you can toggle synchronous scrolling of both panes. When enabled then the relative top index of the active lists/tabs of both panes is stored. When you then scroll the list in either of these tabs the top index relation is restored (if possible) by auto-scrolling the other list.

Notes

- Sync Scroll includes Sync Sort: If Sync Scroll is enabled, and you then resort any of the lists the other list is automatically sorted the same way.
- When you change to another tab in any of the panes, Sync Scroll is suspended until both original tabs are active again.
- Also horizontal scrolling is synchronized.
- If both panes have different view modes than scrolling is synced as "percentage of window scrolled" instead of absolute line counts. That way you can perfectly sync scroll a Details view with a Thumbnails view of the same location.
- Sync Scroll also works when the inactive pane is invisible.
- There's also a toolbar button "Sync Scroll".
- Sync Scroll is not stored between sessions.

Sync Select...

Here you find various commands to modify the selection in one pane based on items in the other pane. They are best used for comparing two locations which have similar content.

- Select Matches: Select all items that are also listed in the other pane (i.e. all name matches).
- Select Uniques: Select all items that are only listed in this pane.
- Select Newer: Select all name matches with a more recent modified date.
  Newer is a file or folder where the modified date is at least one full second newer than that of the other item.
  If Ignore extensions is ticked then the first match (both lists are compared from top to bottom, with the active pane as the outer loop) is taken to the file time comparison.
- Select Different: Select all name matches with a different size or modified date.
- Select Uniques and Newer: Select all items that are unique or newer.
- Select Uniques and Different: Select all items that are unique or different.
- Select Selected: Select all items that are selected in the other pane.
- Select Different Content: Select all name matches with a different content. By default the content is

compared using SHA-256 hashing.

Additionally there are the following checkboxes to modify the functions:

On both panes: Check to apply the chosen command to both panes, i.e. to have items on both panes selected.

Ignore extensions: Check to ignore the extensions when matching the names.

Further Remarks

- Matching is case-insensitive (A=a).
- Sizes are only compared for files, not for folders.
- If a folder and a file happen to be have the same name and modified date, they will count as different (regardless of the size).
- You can add to / remove from the current selections:
  - Hold CTRL while pressing OK (or ENTER) = Add to the current selections.
  - Hold SHIFT while pressing OK (or ENTER) = Remove from the current selections.
- The last used command and the state of the checkboxes are saved between sessions.
- The dialog remembers its size and position.

Sync Folders...

Opens a dialog that lets you synchronize the folder in the inactive pane (target folder) to the folder in the active pane (source folder). Afterwards the target folder will be identical to the source folder (depending on the configuration). The source folder (active pane) is not modified. In other words it's a one-way sync, aka mirror sync.

See Sync Folders for the details.

Tabs (Submenu)

Listing of the current tabs on both panes.

Close this Pane

Right-click the empty part of the tab bar to pop a menu that features a command "Close this Pane". Does what it says.

3.10 Tabsets

Tabsets Menu Commands

Note: If this menu is not visible, you can enable it in Configuration | Other | Features.

New: Start a virgin tabset. You are prompted for a name.

Open...: Open/Load a stored tabset. A list of all available tabsets in the default tabsets folder (<xydata>\Panes) is popped. The current tabset of each pane is marked as [In Use - Pane #]; if you select one of them nothing happens.

Open As...: Open/Load a tabset as clone. The name for the clone is prompted for (it defaults to "Clone#", where # stands for the number of the current pane). Note that any existing tabset of the same name as the clone will be overwritten without any warning.

Revert to Saved: Reverts the current tabset to its saved state.

Tip: Selecting the current tabset from the Tabsets button drop-down list, i.e. reloading it, is interpreted as a "Revert to Saved" operation.

Save: Save the current tabset. Only useful if Configuration | Tabs | Auto-save tabsets on switch is off, otherwise tabsets are saved automatically on tabset switch.

Save As...: Save the current tabset under a new name. The new tabset becomes the current one. The old tabset is kept on disk.

Save Copy As...: Save a copy of the current tabset under a new name. The old tabset remains the current one.

Rename...: Rename the current tabset. If a tabset of the same name already exists in the parent folder of the current tabset you are prompted for overwriting.

Go to Tabset Folder: Brings you to the folder of the current tabset.

For more details on tabsets see here.

3.11 Tools

Tools Menu Commands

Configuration...

Open the Configuration window.

Shortcut: F9

Open Configuration File...

Opens the current INI file in whatever application is associated with *.ini files. Of course, any changes you do will be applied only after a restart without saving.

Customize Keyboard Shortcuts...

Note: If this feature is not visible, you can enable it in Configuration | Other | Features.

Open the keyboard shortcuts configuration interface.

Shortcut: Shift+F9

Customize File Associations...

Open the interface to customize file associations. See file associations.

Shortcut: Ctrl+F9

Customize File Icons...

Open the interface to customize the file icons. See Custom File Icons.

Shortcut: Shift+Alt+F9

Customize Toolbar...

Open the toolbar configuration interface. See Toolbar.

Shortcut: Ctrl+Shift+F9

Trick: Dbl-Click an item to shift it to the other side.

Remove all buttons to restore the default configuration.

Customize Tree (Submenu)

Adjust the folder tree to your preferences: Show Icons, Show Expansion Icons, Show Lines, Solid Lines, Mark Favorites (Bold), Narrow Tree, Rename On Slow Double-Click, Apply Highlighting, Full Row Select, Tree Path Tracing, Recent Location Pins, Lock Expansion State, Lock Scroll Position, Lock Tree, Show Section Colors, Show Glider.

Tip: You can open this menu by right-clicking anywhere in the white space in the tree while holding down the Shift key.

Show Icons: Show the folder icon with each tree node.

Show Expansion Icons: Show a plus/minus icon in front of each expandable node (each node with child nodes). There are a number of options for the shape of this icon, which can be accessed by Ctrl+Right-clicking on any of the currently displayed expansion icons: Plus Minus, Plus Minus Flat, Triangle, Square, Corner, Diamond, Chevron. Note that the color of these icons is slightly different between Maxi Tree and Mini Tree, giving you a hint as to which mode the tree is in (in case of the Chevron it's the size that is slightly different).

Show Lines: Show the node connecting dotted lines.

Solid Lines: If ticked then the node connecting lines are drawn solid instead of dotted, i.e. they are visually stronger.

Mark Favorites (Bold): Show favorite folders in bold type.

Narrow Tree: Reduces the tree level indent from 19 to 10 pixels. Saves a lot of horizontal space if you work in deeply nested folders.

Rename On Slow Double-Click: Tip: With Configuration | Tree and List | Expand tree nodes on single-click ON and Rename on slow double-click OFF, you can also collapse the current tree node by a single click on the name. Could save you many clicks.

Apply Highlighting: Turn it on to apply Highlights and Box Colors.

Full Row Select: Note that you can still trigger right-click on empty when you hit the area left of the nodes.

Tree Path Tracing: Turn it on to highlight the current path in the Tree, i.e. each component of the current path. It's a visual aid to always know your place in the folder tree, without even consciously looking at the tree. The color can be customized in Configuration | Highlights & Dark Mode | Tree Path Tracing. Note that there is a toolbar button "Tree Path Tracing" to quickly toggle it on and off. The button's right-click menu offers all tree styles.

Recent Location Pins: Shows pins at the recent locations in the folder tree. The color and number of pins are customizable in Configuration | Colors and Styles | Highlights & Dark Mode | Recent location pins.

- The current location pin has a slightly bigger head with a white center.
- The current location pin is drawn fatter than the others.
- No pin is drawn at "This PC" (not enough space).

Lock Expansion State: Check it to completely disable expanding/collapsing nodes in the Tree. Note that the Expansion Icons take a special form if the expansion state is locked.

Lock Scroll Position: Check it to prevent automatic scrolling of the selected folder into view. Note that the lock only applies to newly selected folders after a change of location. Of course, you can still scroll the tree manually by scrollbar, wheel or keyboard.

Lock Tree: Ever since, the Tree shows basically the same path as the Window Title, the Address Bar

and the List: If you change the current path by any of the numerous ways available in XYplorer, the Tree is automatically adjusted to the new location. Even, if you don't really need that service... "Lock Tree" allows you to turn off that automatic syncing of the Tree.

With Lock Tree on, "Refresh Tree" (F4) will re-sync the Tree without changing the mode. Also turning Lock Tree off again, will instantly re-sync the Tree. The setting is remembered between sessions. When you startup with Lock Tree on, the tree will be initialized to a default state with only the top nodes shown.

Here are some possible reasons for turning Lock Tree on once in a while:

(1) It offers a uniform maximum browsing speed that's totally independent of the nesting depth of the target locations. For example, if you quickly need to check size and version of a particular system file, it is totally pointless to expand the tree down to system32, because all you need is a quick glance at that file in the file list. Or, if you have a script that needs to visit some location -- why should the tree get busy here?

(2) It keeps the Tree in a stable state and position while browsing. For example, this can be valuable when collecting stuff from various locations via drag+drop into a couple of target folders -- no need to scroll the tree anymore: the Tree just sits there and waits.

(3) It makes consequent use of XYplorer's many ways of going to a new place: Address Bar, Tabs, Catalog, Favorites, History, Hotlist, Breadcrumb, GoTo, User-Defined Commands, Scripts, etc. ... they all work flawlessly without the Tree, and if you just need to go somewhere to work with certain items, you may well be completely uninterested in their position in the file system -- no need for a Tree to show you.

While the Tree is an ingenious interface element one must admit that it almost always shows much more information that you need to have at any given time. So, for the sake of informational economy, you now get a "stand-by" button with it.

Note the Tools | Customize Tree | Lock Expansion State toggle (here above), an optional radicalization of Lock Tree by which you can prevent any expanding or collapsing of the currently visible nodes. Makes the Tree indestructible.

Show Section Colors: Tick it to show section specific colors in the tree, the Tree Section Colors. Coloring different sections is meant to improve orientation in a long tree.

The sections as they appear in the tree from top to bottom:

- Special Folders: This PC and any special folders.
- User Folders: [Reserved for future use]
- Drives: All drives with a drive letter (local and mapped).
- Portable Devices: Portable devices.
- Recycle Bin: Recycle Bin
- Network: All network locations with UNC paths.

Notes:

- If the color is missing, the section uses the default tree backcolor.

- Dark mode colors are automatically derived from the light mode colors.
- The section text color is used only when no Color Filter color applies.
- You can customize the section text and background colors using the scripting command interfacecolors.

Show Glider: See Glider.

Customize List (Submenu)

Adjust the file list to your preferences: Show Icons, Column Drag, Multiple Select, Full Row Select, Highlight Sorted Column, Show Line Numbers, Autosize Columns, Show Grid, Highlight Focused Item, Highlight Selected Rows, Rename On Slow Double-Click, Hide Extensions, Sticky Selection, Manual Sorting, Checkbox Selection, Ignore Articles When Sorting, Show Folder Sizes, Show Folder Row Colors, Size Column Format (Submenu), Date Column Format (Submenu).

Note that, by factory default, the list styles are applied globally to all tabs in all panes. This setting is configurable in Configuration | Styles under "Apply list styles globally".

Tip: You can pop this menu by Shift+Right-click anywhere on the empty space in the List.

Some notes on particular options:

Autosize Columns: It is not done on startup, on tab switch, or on view change. In those cases the previous column widths are preserved. The main moment for Autosize Columns to happen is when changing the location within the current tab.

Show Grid: The colors for the grid are defined in Configuration | Colors | List | Grid. In Configuration | Highlights & Dark Mode you can as well choose a Grid Style.

Rename On Slow Double-Click: When this setting is enabled, you can do a slow double-click (two consecutive clicks, but slower than a double-click) on a file list item to invoke the single-rename interface. Note that, contrary to Explorer, this will only happen when no other than the clicked item is selected, else other selected items are unselected first.

Highlight Focused Item: Paints a light background to the focused item. The color can be defined here: Configuration | Colors and Styles | Colors | Focused Item.

Hide Extensions: Applies to all views.

Sticky Selection: This is a selection style known from many apps with multiple selection lists. The effect is pretty much identical to holding the CTRL key while you click on an item or hit the space key on the focused item: The previous selections are not touched and the current item's selection state toggles. Also clicking on the white space will not unselect the selected items.

Sticky Selection is obviously very practical when you work on long lists and want to pick out particular items manually: You won't easily lose your work by the slip of a finger.

Tip: To unselect all items using the mouse start a drag-select (hold mouse button down and drag a bit), or use Deselect All (Ctrl+Shift+A).

Note that you won't have the "Rename On Slow Double-Click" functionality when "Sticky Selection" is

active. Note also that obviously "Sticky Selection" will only come to effect when "Multiple Select" is on (which no known user ever turned off).

Manual Sorting: In Manual Sorting mode you can directly manipulate the displayed order of items by dragging them to a new position. You can drag single as well as multiple items -- the latter even when the selections are non-contiguous in which case they are inserted as a contiguous group to the new position.

Remarks:

- Manual Sorting just concerns display. The files are not touched at all.
- When Manual Sorting is enabled you cannot drag items from the list to any other controls using the left mouse button. Right button drag however works as usual.
- Manual Sorting also works with Search Results, Branch View, and other list modes (not necessarily extremely useful though with any mode).
- Since v16.60 there is Permanent Custom Sort Order (PCSO): The custom sort order is remembered per tab, across tab switches, and across sessions, and on an explicit List Refresh. Note, however, that Auto-Refresh is internally suspended while a custom sort order is in effect to protect the order from an unwanted List Refresh.
- Tip: You can restore a Custom Sort Order via "Go Home" if you previously saved it using "Set Home", and if "Configuration | Tabs | Going home also restores the list layout" is ticked.
- PCSO is limited to lists up to 5000 items. Larger lists forget their custom sort order on a tab switch or across sessions.
- A manual sort order is indicated by a small "hand" character (Unicode U+270B) in the Name column header.

Checkbox Selection: A checkbox is shown at each list item. Now you can individually select/unselect multiple list items simply by ticking/unticking checkboxes with the left mouse button.

Remarks:

- Checkbox Selection works in all list views and list modes.
- Selecting an item via checkbox does not move the focus to this item.
- Right-clicking any checkbox pops the Select submenu from menu Edit.
- The checkbox has an invisible 2-pixel safety margin that protects shaky clickers from missing the target too easily. In other words, you can miss it by 2 pixels and still make a hit.

Ignore Articles When Sorting: Tick it to ignore certain leading words when sorting files. Typically these words are articles, i.e. semantically near-empty words accompanying nouns in various languages. Since they are usually preceding the nouns and there are usually two or more of them they affect the sort order in a probably undesired way. A case in point are files named after song titles.

- The factory default are English articles "a", "an", and "the".

- Words are matched case-insensitively (A==a).
- They must be followed by a space (unless they end with an apostrophe).
- They are matched only at the beginning and only once per string.
- Tip: To change the factory default you can modify the tweak SortLeadingWordsToIgnore in the INI file.
- Additionally to spaces (default) also "." and "_" are supported as word separators. For this to happen you have to tweak this key as follows:
  `SortLeadingWordsToIgnore=a;an;the;a.;an.;the.;a_;an_;the_`

Show Folder Sizes: This setting is logically OR-ed with the global setting "Configuration | Tree and List | List | Always show folder sizes", i.e. if at least one of them is ON then the folder sizes are shown for the current list. What you gain by this additional way to turn on "Show Folder Sizes" is more control over this potentially time-consuming service. You can now enable it specifically per tab and per folder. Both presupposes, of course, that the global setting is turned OFF:

- Per tab: If you turn on "Configuration | Styles | Remember list settings per tab" and apply it (via the "Apply..." button) to "List style (line numbers, auto-size, grid...)" then you can show folder sizes individually per tab.
- Per folder: If you turn on "Tools | Customize List | Show Folder Sizes" for a particular folder and save the view using "View | Folder View Settings | Save Folder View", you can show folder sizes just for this folder.
- There is a toolbar button "Show Folder Sizes" available for this toggle.

Tip: If the file list shows folder sizes, the Size column heading's right-click menu has some useful related commands like "Cache Folder Sizes" and, if that is enabled, more commands and toggles related to the folder size cache. By "Calculate Folder Sizes" you can update individual folder sizes when you select these folders in the list. Note that this applies to all selected folders, or all folders if none are selected.

Show Folder Row Colors: Tick it to show folder row colors, i.e. mark folders with a special text and background color to make them easier to distinguish from files. The colors can be customized using scripting command interfacecolors.

Size Column Format (Submenu)

Set file size display format for the size column in List.

Show Size on Disk: When checked, all file sizes in the list are showing the disk space wasted by each file, taking into account compressed and sparse files. The column heading will be changed from "Size" to "Size on Disk". Of course, this item is also available in the context menu of the size column header.

Note that the Status Bar always shows the real uncompressed size of the selected files.

Note that Live Filter and Quick Search from the Cell Context Menu (right-click a cell) currently don't work with this column.

Raw Bytes Count: Pure byte count, no suffix, no thousand separators. This format is useful when you create reports that shall be further processed by software that cannot deal with thousand separators.

Bytes: Exact bytes, no suffix (would just take away space).

KB (Rounded Up): KBs are rounded up (as in Explorer).

KB, MB, GB, TB, PB: Size with 2-digit accuracy.

Flexible: Byte counts are given in the most easy-to-read unit (i.e. "bytes", "KB", "MB", "GB", "TB", "PB" depend on the actual number).

Flexible (Rounded Up): Like Flexible but with all values rounded up.

Clusters: Number of clusters actually used by a file on this drive.

No Graphics: Don't show any size related graphics.

Circles: Shows circles. Blue for files, khaki for folders. Diameter and darkness correlate with the item size. Maximum diameter adjusts to row height (which adjusts to font size). Note that this command is a toggle: Call it again to turn it off.

Tip 1: If Circles are enabled, the Donut, a donut-colored and donut-shaped circle, is displayed in the Size column cell for each unpolled folder. A single left-click on the donut triggers a deep folder size calculation for that folder, or for all selected folders if the clicked donut is from a selected folder. The size is freshly calculated, not taken from the cache. The folder does not have to be selected, and any current selections will not be changed.

Tip 2: You can click the size circle of a cached folder size to update that cache. The circle's tooltip says "Click to update cached folder size" at the bottom.

Bars: Shows bars whose sizes (filled area) correlate with the item sizes. The overall maximum width the bars take is 50 pixels. Note that this command is a toggle: Call it again to turn it off.

Tip 1: The graphic's tooltip shows the raw bytes count and the size in flexible format if it's 1 KB or larger.

Tip 2: Showing size graphics works nicely with Size Column Format Flexible (Rounded Up).

Date Column Format (Submenu)

Adjust date and time display format in List to your preferences.

Tip: These commands are also available in the right-click menu of any of the Date columns in the file list.

Show Age: Show the age of items relative to now, expressed in years, months, days, hours, minutes, and seconds.

Shortcut: Ctrl+Shift+E

Show Weekday: If enabled the day part of the date is shown as the weekday name ("Monday, Tuesday ...") if the day lies within the last week. If it's today (yesterday) it is shown as "Today" ("Yesterday"). If the file date is within the last hour, you get "Now" instead of "Today".

Remarks

(1) The option applies to all date formats except, of course, "ISO Week".

(2) Like with all date formats, the setting is per List mode: you can e.g. show the weekdays in Search Results while having the normal Date format in Browse mode.

(3) If Show Age is enabled then Show Weekday is ignored.

Show Milliseconds: Tick it to show all file times with milliseconds precision.

Show Times in UTC: Tick it to show all file times as UTC, i.e. time zone-independent as they are internally stored under NTFS. Untick it to show file times auto-converted to the local time zone (as defined in Windows settings), which is the factory default and the usual way file times are shown in file managers and similar applications.

UTC file times are easily recognizable by the suffixed "Z", which is the official way to mark UTC times according to ISO 8601. Note that not only the file list, but also the Properties tab and all sorts of Reports support showing UTC times and hence are affected by this setting.

No Graphics: Don't show any date or age related graphics.

Circles: Shows Age Circles. Note that this command is a toggle: Call it again to turn it off.

Age Circles

The so-called "Age Circles" are little visual helpers that let you instantly grasp the rough age of a file. No more brain power wasted on deciphering date strings.

This is achieved by mapping the age to colors. By factory default there are 10 age classes (can be customized, see below):

Future	turquoise, empty
Up to 5 Minutes	light green
Up to 1 Hour	middle green
Today	dark green
Yesterday	pink
This Week	light brown
This Month	dark brown
This Year	deep purple
Last Year	light grey
Even Older	light grey, empty

The internal definition of the above age classes looks like this:

```
< s>6BCAA9,b2//future
<= 5 n>DCF487//up to 5 min
<= 60 n>B7DC6F//up to 60 min
d>79BB53//today
1 d>DD7BB4//yesterday
w>D0AB63//this week
m>BE7C43//this month
y>866286//this year
1 y>ABAEB4//last year
> 1 y>ABAEB4,b1//even older
```

You can customize the age classes by right-clicking any age circle in the list while holding CTRL.

General Format:

```
Age Condition>[RGB hexadecimal color value][,border width][//comment]
```

Rules:

- The age condition syntax is identical to the one used in Color Filters By Age. Remember: There must be a space between counter and unit ("1 d", not "1d").
- You can specify an "Else" case by stating * instead of an age: `*>866286//this color for all remaining cases`
- The RGB value is in the usual format RRGGBB. If omitted then nothing is drawn: `> y>//last year or older`
- Border width is "b" followed by a number that specifies the width in pixels. If omitted then the circle is drawn filled.
- Comments are optional and prefixed with //. You can also comment out whole lines in the definition by prefixing // to the line.
- The definitions are processed left to right (top to bottom), first match wins. They are compared with the Modified date of the item in question.
- You can define as many age classes as you like.
- Tip: Enter nothing to reset the definition to factory defaults.

Notes:

- All circles are drawn on a slightly wider white background circle.
- Age always refers to the Modified date.
- Hovering age circles pops the exact item age in a tooltip. Additionally to the age the absolute date is shown including the week of the day.
- By default the circles are shown in the Name column, or near the filename (depending on the list view). You can optionally show the circles in the Modified Date column (if in Details view). To toggle

this behavior right-click any age circle and toggle the setting "Show Age Graphics in Date Column" in the context menu.

- The "rocker-click" (Left Mouse Down + Right Mouse Click) as a one-hand alternative to Ctrl+Right-Click.

List Management (submenu)

Here you can manage various lists like History, Favorite Folders, Address Bar & Go To, Find Files: Name, Find Files: Location, etc.

Tip: All lists (e.g. also in Customize Toolbar Dialog) support a simple Type Ahead Find where you jump to the next item beginning with the letter you typed.

Live Filter Box: The various lists in List Management (and some other dialogs featuring a list) are provided with a live filter box.

- Matching is case-insensitive (A=a).
- Partial matching is supported. E.g. a filter pattern "o" will match "dog". However, if the pattern has wildcards * or ? then the pattern is taken as is, so "o*" will not match "dog".
- In List Management | Editor Mode no filtering is applied.
- Patterns are not retained between calls.
- Patterns don't support Boolean logic.
- Line numbers in filtered lists still refer to the position in the unfiltered list. Cool!
- Sorting is supported in filtered lists.
- Click the icon to toggle between "no filter" (stores the current filter as "last filter") and "last filter". The last filter is shared across all lists, and retained across sessions.
- Pressing up/down arrow keys inside the Live Filter Box will propagate the keys to the list, so you can navigate the list without focusing it.
- The global "Ignore Diacritics" setting (Configuration | Find and Filter | Filters & Type Ahead Find | Visual Filters and Live Filter Box | Ignore diacritics) can be changed via the right-click menu of the filter icon in the filter box.

Sticky Section: Various most-recently-used (MRU) lists can have an optional Sticky Section. Items in the Sticky Section are permanent members of this list and will never be pushed out when new items are added. You can use this section for your all-time favorite items and carefully crafted patterns. The Sticky Section can be added via List Management, simply add this dummy item as section separator:

```
-----> mru
```

Everything above this line is in the Sticky Section. Everything below this line is in the MRU Section. So the Sticky Section is always displayed on top of the whole list.

Many MRU lists support a Sticky Section, e.g.: Address Bar & Go to, Find Files: Name, Find Files:

Location, Rename Special: (...), Visual Filters, Selection Filters, Move/Copy/Backup To, Recently Opened Files, Recent Catalogs, Recently Included Catalogs.

Notes on Sticky Section

- In dropdown lists the Sticky Section is distinguished from the MRU section by a special text and back color. The colors are currently hard-coded, later this might be made configurable. (In List Management there is no special coloring.)
- No shifting or sorting: When you pick an item from the Sticky Section it will not be shifted to the top but stays in its position.
- When a "Match List" (an alphabetically ordered subset of items that match what is currently typed into the edit field) is dropped the items pertaining to the Sticky Section are not marked in any way but mix with the "normal" items. Reasons: There would be clashes between coloring rules; it would be visually disturbing; and it simply does not make a lot of sense in a Match List.
- The size of the Sticky Section is not limited and adds to the max size of the MRU section (e.g. 64 for "Rename Special" patterns).
- The section separator makes managing the lists quite easy and intuitive. In List Management you simply drag items to above the separator to add them to the Sticky Section. The separator is visible only in List Management. In the GUI, in a dropdown list it is hidden from the user's eye, in a menu (e.g. Visual Filters in right-click menu of the toolbar button) it is replaced by a menu separator.

Tools Special (submenu)

Control Panel...

Opens the Control Panel.

Open Recycle Bin...

Opens the Recycle Bin.

Empty Recycle Bin...

Empties the Recycle Bin.

Recycle Bin Stats...

It's just a little message box that tells you relatively quickly the total number and size of items currently in the recycle bin. Just in case you wanted to know...

Map Network Drive...

Opens the Map Network Drive wizard. Note that there's also a Toolbar button for this function.

Disconnect Mapped Network Drive...

Opens the Disconnect Network Drive wizard. Note that there's also a Toolbar button for this function.

Reconnect All Mapped Network Drives

After a fresh system boot this will (if necessary) revive all your mapped drives in one go. To revive the drives manually and one-by-one: select a sleeping drive in the tree and press Shift+F4 (Refresh Current Folder).

Note: Reconnecting Mapped Network Drives may need a live network connection depending on your system setup.

Browse for Network Server...

Allows you to browse the network and select an accessible computer via the Shell interface.

3.12 Window

Window Menu Commands

Show Address Bar

Show/hide the Address Bar.

Shortcut: Ctrl+Shift+F12

Show Toolbar

Show/hide the Toolbar.

Shortcut: Ctrl+F12

Show Tab Bar

Show/hide the headers of the Tabs on each pane.

NOTE: If you hide the tab bar(s) then opening locations in new background tabs will open them in new "hidden" tabs, hence the results of such an operation may not be immediately visible.

Show Breadcrumb Bar

Show/hide the Breadcrumb Bar on each pane.

Show Status Bar

Show/hide the Status Bar. Note that if the Status Bar is hidden then also the Status Bar Buttons go away.

Show Status Bar Buttons

Show/hide the Status Bar Buttons.

Show Navigation Panels

Show/hide the tree (with catalog).

Shortcut: F8

Show Tree

Show/hide the tree.

Shortcut: Shift+F8

Show Catalog

Note: If this feature is not visible, you can enable it in Configuration | Other | Features.

Show/hide the catalog.

Shortcut: Ctrl+F8

Show Live Filter Box

Show/hide the Live Filter Box. When turned on then the input focus is auto-set to the box.

Shortcut: Alt+F3

Show Preview Pane

Show/hide the Preview Pane.

Shortcut: Ctrl+F11

Show Info Panel

Show/hide the Info Panel.

Shortcut: F12

Arrangement (Submenu)

Address Bar and Toolbar Stacked: Show Address Bar and Toolbar one above the other.

Toolbar First: Show the Toolbar at the top or at the left of the Address Bar.

Show Button Captions: Toggles the display of toolbar button captions.

Shortcut:Shift+F12

Allow Multiple Button Rows: When unchecked, only the first button row of your Multi-Row Toolbar set is displayed. This way, you can have optional rows of not-so-important buttons below your main row of buttons, and quickly toggle them on and off. Any menu-only rows before the first button row are not counted, so you can toggle between menu row plus one button row, and menu row plus more than one button row.

Switch Toolbar Button Set: Cycles through the button sets. The number of button sets can be set in Tools | Customize Toolbar | Options.

Tree and Catalog Stacked: Show Catalog and Tree one above the other. Setting is ignored if List Centered is selected.

Catalog First: Show the Catalog at the top or at the left of the Tree.

List Right: Show the List pane at the right of Tree and Catalog.

List Centered: Show the List pane between Tree and Catalog.

List Left: Show the List pane at the left of Tree and Catalog.

Wide Tab Bar: Show the Tab Bar (including the Breadcrumb Bar) over the whole window width (single pane mode only). Note that it works in single pane mode only. In dual pane mode the setting is ignored because it would be geometrically impossible to do this trick for two panes.

Wide Info Panel: Show the Info Panel over the whole window width.

Live Filter Box in Status Bar: OFF = Show the Live Filter Box right of the Address Bar; ON = Show the Live Filter Box in the left end of the Status Bar. Note that if the Address Bar is hidden then the Live Filter Box automatically goes to the Status Bar.

Preview Pane to the Left: Tick it to show the Preview Pane to the left of the List.

Load Layout...

Loads a previously saved layout (see Save Layout As...).

Save Layout As...

Saves the current layout. The layout is everything you can configure in menu Window, plus "Dual Pane" and "Horizontal Panes" from menu Panes, plus the visibility of the main menu.

Remarks:

- The layouts are loaded/saved from/to small TXT files in <xydata>\Layouts.
- The active pane will never be changed by changing the layout.
- The main window state, size, and position is not stored/restored. Rather the loaded layout is adjusted to the current window size.

3.13 Help

Help Menu Commands

Contents and Index

Displays this help file.

Shortcut: F1

Help on Keyboard Shortcuts

Displays all keyboard shortcuts and mouse tricks.

Help on Scripting Commands

Displays the Scripting Commands Reference section of this help file.

Keyboard Shortcuts on Menu

Here you can toggle whether hints for keyboard shortcuts are displayed in the main menu (factory default), or not.

Command IDs on Menu

Tick it to show the command IDs in the main menu (and, where they are featured, in context menus).

Tip: All commands that can be assigned a keyboard shortcut have an internal command ID which can be used in scripting to refer to the command.

List All Commands...

Shows a list of all commands with their keyboard shortcuts (if they have any). In the list you can select a command and click OK to call it directly from there. If the list shows only one item OK will trigger it even if it is not selected.

Unless scripting is disabled, the command IDs are shown in front of each command. These IDs can be used in scripts to refer to the commands.

The last filter is remembered between calls.

Tip 1: Hold CTRL when clicking the command to hide the command IDs shown along with the commands.

Tip 2: The list's right-click menu provides the option to copy the list to the clipboard.

Online Support (submenu)

Check for Updates: Retrieves the latest official release version online (if you are connected), compares it with the currently running version, and shows the result in a message box. Takes a couple of seconds on first check.

If a new official release is found you are offered to download and install the update right away if there is one.

Tip: Hold CTRL to update to the latest beta version. Or tick Configuration | General | Startup & Exit | Include beta versions.

Links to various XYplorer-related websites.

Trial Version Info...

Tells you what you need to know when running then 30-day trial version.

Unlock Trial Version... (if registered: Update Registration Details...)

Opens a dialog where you can enter your name and license key to unlock the trial version.

Environment Variables

Shows the current state of the Environment Variables.

Various Information

Pops up some contextual data.

Tip 1: Hold CTRL to get some speed related information about your current settings.

Tip 2: Hold CTRL+SHIFT to get detailed information about the application load times.

Select Language...

Downloads a language file from the server and loads it into the interface in one go. No restart required.

The downloaded language files are saved in the application data path.

Once a file is downloaded you can use CTRL+Select Language... to load the language (e.g. "French.lng") into the running application. If you have downloaded different language files you can switch languages on the fly. No restart required.

Select Local Language File...

Opens the standard Open File dialog to load a locally stored *.LNG file.

Back to English

To quickly go back to English simply tick Back to English (this command will always stay in English). Untick Back to English to go back to the loaded language. Again, no restart required.

Interface Translation Tool

The Interface Translation Tool (ITT) lets you edit a translation or create a new one yourself. See here for details.

About XYplorer

Opens the About window.

Tip: You can hide your personal license data. Right-click the license name (e.g. "Lifetime License") and

toggle "Show License Data" in the context menu. This allows you to stay under cover in live streaming contexts.

Part IV

Main Topics

4 Main Topics

4.1 Rapid Access Folders

Rapid Access Folders

You already know about the system-wide so-called special folders "Desktop, Documents, Downloads, etc", which are just short versions of real paths, placed at the top of the folder tree for quick access. For example, "Desktop" usually points to "C:\Users\[username]\Desktop". Rapid Access Folders work exactly the same way, only that you can decide which paths you want to have rapid access to. These Rapid Access Folders (RAF) work anywhere in the application (but not outside of it). For example, you could have "Config" point to "C:\Users\Donald\AppData\Roaming\XYplorer". In this case, the RAF path for "C:\Users\Donald\AppData\Roaming\XYplorer\Scripts" would be "Config\Scripts".

In the folder tree the Rapid Access Folders are listed under a new super-section, Rapid Access, located above This PC and Network. Unlike simple shortcuts (as in the Links folder), Rapid Access Folders are fully expandable within the Rapid Access section and share all properties and behaviors of the corresponding real paths. To show the Rapid Access section, check Configuration | General | Tree and List | Items in Tree and List | Rapid Access.

With Rapid Access Folders, you can focus on the areas of your file system that are important to you by moving them to the top of the tree. Shorter paths, less scrolling, no more irrelevant information cluttering your view. Makes your daily file management a whole lot easier.

Properties of Rapid Access Folders

Rapid Access Folders can point to:

- Real local paths, eg: `C:\, C:\Users\Donald\AppData\Roaming\XYplorer`
- Real network paths, eg: `\\VEGA, \\VEGA\shared, \\VEGA\shared\Test`
- Relative paths, eg: `..` (relative to XYplorer.exe)
- Portable paths, eg: `?:Temp` ("?:" stands for the drive this XYplorer is running on)
- Paper Folders, eg: `paper:Trip`
- Virtual Folders, eg: `vi:|t|<pick 8.m> //"Last 8"`

Additionally, Rapid Access Folders support:

- Environment Variables, eg: `%TEMP%`
- XYplorer Native Variables, eg: `<xydata>`

Note that these variables are resolved at startup or when the RAF are created or updated, not when a RAF is clicked.

Creation and Management of Rapid Access Folders

There are several ways to to create a RAF:

- Drag one or more folders onto the Rapid Access node in the tree or onto the Rapid Access listing in the list pane. The display name is automatically generated from the path.
- Paste one or more folders into the Rapid Access listing in the list pane. You can also paste textual clipboard contents (paths).
- Add it from the right-click menu of the Rapid Access node in the tree.
- Add it from the right-click menu of the Rapid Access toolbar button.
- Add it via Tools | List Management | Rapid Access Folders...
- You can also add RAF on-the-fly by prefixing "raf:" to the path you want to add. If it's not already present in RAF it will be added. Then you will be taken to the (new) RAF. Optionally prefix the desired display name, separated by ">". Examples:
  `raf:Documents`
  `raf:C:\Users\Donald\AppData\Roaming\XYplorer`
  `raf:iCloud>E:\iCloudDrive`

Internally (and in List Management, and in the INI file) each RAF is defined by a string of the format "Name>Path", e.g. "Config>C:\Users\Donald\AppData\Roaming\XYplorer". The names must conform to the rules for filenames. Specifying a display name is optional, if missing the last component of the path is automatically used.

Further Remarks

- Note that "Rapid Access" is an abstract node, and like the other super-nodes "This PC" and "Network", it is not part of the path. All folders under these nodes are in the same namespace, so duplicates are not allowed, see next point. The node's right-click menu has some useful stuff.
- To avoid duplicate folder names (which would create unresolvable ambiguities), the display names of the RAF are automatically renamed by appending numbers if necessary.
- The folders under the Rapid Access node are not sorted, so you have full control over their order (via List Management).
- The Rapid Access Folders can be inline-renamed in Tree and List.
- When you go up from a top RAF you don't end up in the virtual folder "Rapid Access", but in the real path parent folder of the RAF.
- The Rapid Access Folders can be removed directly from Tree or List by pressing DEL. You can also remove them via their context menu or via Tools | List Management | Rapid Access Folders...
- The number of Rapid Access Folders is not limited.

- The Mini Tree is supported here, just like anywhere else in the tree pane.
- In the tree, Rapid Access Folder file info tips show the real path.
- The toolbar offers a "Rapid Access" button. Clicking it takes you to "Rapid Access", the right-click menu has some useful stuff.

4.2 Quick Search

Quick Search

The Quick Search dialog offers a very simple interface to finding files by their name (and more, see below). The command "Quick Search..." (F3) is found near to the bottom of menu Edit.

The obvious idea is to offer an ultra-smooth access to the most common find task: the search by filename. No more need to open the Info Panel, select the mighty Find Files tab, reset any previous filters, enter a pattern, and press Find Now. Instead you completely bypass the Find Files tab leaving its current settings unused and unchanged.

Notes:

- By default, Quick Search is case-insensitive. If you want it to be case sensitive, you can add the "/c" switch (see below) or click the "c" button in the mini toolbar.
- By default, Quick Search performs a partial match. If you want it to match whole words, you can add the "/w" switch (see below) or click the "w" button in the mini toolbar.
- By default, Quick Search performs minds diacritics. If you want it to ignore diacritics, you can add the "/u" switch (see below) or click the "u" (umlaut) button in the mini toolbar.
- By default, Quick Search includes subfolders. If you don't want to include subfolders, you can add the "/n" switch (see below) or click the "n" button in the mini toolbar.
- Quick Searches always search the current location (as shown in the Title Bar, the Address Bar, and in the unlocked Tree).
- There is a handy command Toggle Quick Search (Ctrl+Shift+F3) to toggle between no Quick Search and the last one (if any) in the current location.
- There is another handy command Repeat Last Quick Search (Shift+F3) to repeat the last Quick Search (if any) in the current location.
- The last 64 patterns are stored in an MRU (most-recently-used) list. They are remembered across sessions if "Configuration | Startup & Exit | Include most-recently-used lists on save" is ticked and "Find Files Patterns and Locations" is ticked in the Apply list. The MRU list can be edited in List Management.
- You can use the mouse wheel to scroll through the MRU (most-recently-used) list items when it's collapsed.
- Quick Searches are displayed in the Address Bar, but not added to the Address Bar MRU (the dropdown list).
- Quick Search also supports more complex patterns like Boolean terms, RegExp, and Multi Field Search. Simply everything that's supported in the Address Bar right of the ?, the marker for Quick Searches through the Address Bar. See examples here below.
- Alternatively, you can run Quick Searches directly through the Address Bar (see Quick Searches via

Address Bar). These are not added to the Quick Search MRU list.

- There is a "Quick Search" button in the main toolbar.

Search Pattern Switches

There are freely combinable switches to modify a quick search and override the global search settings:

/n	Non-recursive, i.e. not including subfolders. (Quick Search defaults to include subfolders.) In a Multi Branch View this switch will also suppress listing the top folders of each branch.
/r	Recursive, i.e. including subfolders. (Obsolete since v15.60 where this behavior has become the default.)
/k	Follow Folder Links. (Quick Search defaults to not follow folder links.)
/f	Find only files, not folders.
/d	Find only folders, not files.
/e	Find only empty files (empty = zero bytes) and empty folders (empty = no items contained).
/E	Enforce Extended Pattern Matching (treat # as wildcard for digits and ! as Boolean Not), even if "Configuration | Find and Filter | Find Files & Branch View | Find Files | Enable extended pattern matching" if OFF.
/a	Find all items (regardless of any GUI settings that may hide them from the list).
/l	Explicit Search In List.
/p	Return the parent folders of all matching items. The depth of the returned paths can be optionally specified, e.g.: /p=3
/P	The full path of each item is matched against the search pattern. It corresponds to the Path checkbox on Info Panel | Find Files | Name & Location. See also "Examples for implicit path matching" below.
/x	Return all folders that don't contain any matching items (reverse of switch /p). The depth of the returned paths can be optionally specified, e.g.: /x=3
/b	Branch-filtered: name filters in the search or branch view are only applied to branches (= folders and all of their contents), not to files.
/c	Search is case-sensitive (a!=A). Corresponds to the "Match case" checkbox on Info Panel | Find Files.
/v	Match verbatim. No automatic wildcards are added, and passed wildcards are treated as normal characters (forbidden in filenames, but allowed in tags).
/w	Match only whole words. Corresponds to the "Whole words" checkbox on Info Panel | Find Files.
/i	Invert the search results: Return all items that do NOT match.
/u	Ignore diacritics (think umlaut vs ümlaut).
/h	Hide the information bar for this search (even if

	"Configuration | Find and Filter | Find Files & Branch View | Find Files | Show search information in list" is ticked.)
/t	Search the tags database only.
/T	Search terms are interpreted as "tags" even when they are not prefixed with "tags:".
/L	Search terms are interpreted as "labels" even when they are not prefixed with "lbl:".
/V	Scan only visible subfolders of the search location, not those currently hidden by user settings.
/flat	See Branch View.
/flatdirs	See Branch View.
/flatfiles	See Branch View.
/limit=n	Limit the number of search results.
/limitperdir=n	Limit the number of search results per directory.
/maxdepth=n	Limit the search depth to n levels. n=0 will search only the current location without any recursion into child folders.
/matching=[e|n]	Enable/disable extended pattern matching. The switch overwrites the global setting of "Configuration | Find Files & Branch View | Find Files | Enable extended pattern matching".
/fuzzy=n	Invoke a fuzzy search and set the degree of fuzziness.
/fld=[selector]	Set a default field for search patterns.
/excl=	Exclude the contents of certain paths from the search results.
/exclfile=	Exclude certain files from the search results.
/silent=	Prevent message boxes from popping before the search finally happens.
/types=	Setup a custom type filter via a list of ;-separated extensions (jpg;png;raf). Wildcards supported.
/itypes=	Setup a custom type filter via a list of ;-separated extensions (jpg;png;raf) that shall NOT be returned by the search. Wildcards supported.
/paths=	Filter items by the name of their parent folder or full path.
/contents=[string to search]	Run a simple content search.
/contflags=[mode=n];[type=t];[options]	Modify the behavior of /contents (see below).
/?	Help: Pattern will be parsed and displayed in a human-friendly form. No search will happen.

Note 1: All search pattern switches can optionally be appended to the search pattern in the name field and have the general format "blank-slash-letter(s)", e.g. " /r" or " /rf" (the order of letters is irrelevant). If you use more than one one-letter switch they must be combined after one slash, so it is eg /rf, not /r /f.

Note 2:They can be freely combined with all the other search filters (Name, Size, Date, Attributes, Tags, Contents, Dupes). Exception: On Dupes search the switches /p and /x are ignored (the results would be just useless).

Note 3: Switches /T and /L logically imply switch /t (search tags database only).

Examples:

?* /n	Find all items (not including subfolders).
?* /r	Find all items (including subfolders).
?*	Same as above (including subfolders is the default).
?* /nf	Find all files (do not include subfolders).
?* /f	Find all files.
?!s* /bE	Find all items, but exclude any branches starting with s*. Note: The E switch enforces extended pattern matching to interpret ! as NOT. Alternatively tick Configuration | Find and Filter | Find Files & Branch View | Find Files | Enable extended pattern matching.
?!(s* OR p*) /bE /flat	Run a filtered branch view excluding any branches starting with s* OR with p*.
?* /n /flat	Run an unfiltered non-recursive (i.e. subfolders not included) branch view.
?ultra /w /types=jpg	Find all JPG files with "ultra" as whole word in the name.
?b* /itypes=zip;rar	Return all items starting with "b" but not any of those types.
?Koln /u	Find Köln and Koln.
?Köln /u	Find Köln and Koln.
%commondesktop%\;%desktopreal%\?/nh	Show a merger of two desktop location without information bar.
?a* /e	Find all empty items starting with "a".
?a* /ef	Find all empty files starting with "a".
?a* /ed	Find all empty folders starting with "a".
?/ed	Find all empty folders.
?* /a	Find all items (even if they are hidden by visibility settings like "Hide protected operating system files").
?readme.txt /p	Find all folders that contain a file readme.txt.
?readme.txt /p=2	Find all folders that contain a file readme.txt, return parents of depth level 2.
?green /Lp	Find all folders that contain a file labeled "green".
?/p	Find all folders that are not empty (= that are parents).
?*.jpg /x	Find all folders that don't contain any JPG file.
?size: > 50KB /x	Find all folders that don't contain files larger 50KB.
?prop:#empty:1 /x	Find all folders that don't contain any empty files.
?prop:#nosubs:2	Find all folders without subfolders.
?prop:#contains.*.txt:1	Find all folders that contain TXT files.
?name:*.png & rock & british /T	Find all items named *.png and tagged "rock" and "british".
?*.txt & (cont:rock | punk)	Find all items named *.txt and containing the words "rock" or "punk".
?:{:image} AND !tags:*	Find all images that are not tagged.

Examples for Limited Search Results:

* /limit=10	Limit the number of search results to 10 items.
* /l=10	(same as above in short form)
* /limit=0	Return nothing.
* /limit=-1	Return all (default, equal to omitting the switch).
* /limitperdir=1	Limit the number of search results per dir to 1 item.
* /lpd=1	(same as above in short form)
* /limitperdir=0	Return nothing per dir.
* /limitperdir=-1	Return all per dir (default, equal to omitting the switch).

Examples for Depth-Limited Search:

*.txt /maxdepth=0	Only search the current location.
*.txt /maxdepth=1	Only search the current location and one level deeper.
*.txt /maxdepth=2	Only search the current location and two levels deeper.

Examples for Extended Pattern Matching:

[!a-zA-Z0-9 ._()&-] /matching=e	Match all files not containing any of a particular list of characters.
# /matching=n	Match all files containing the character "#".

Examples for Fuzzy Search:

Piper.png /fuzzy=20	Matches "Paper.png".
Koln.txt /fuzzy=10	Matches "Koeln.txt".
Koln.txt /fuzzy=100	Matches everything.

The fuzziness and can be a value from 0 thru 100. Value 0 is equivalent to non-fuzzy so it's the same as omitting the whole switch.

The similarity measurement of XYplorer's fuzzy search is based on the Ratcliff/Obershelp pattern recognition algorithm (Ratcliff 1988), also known as "gestalt pattern matching".

Examples for the Exclude switches:

Use the /excl switch to exclude the contents of certain paths from the search results. It's the inline equivalent of the Excluded tab on the Find Files panel.

- Multiple paths can be stated, separated by | or ;.
- The syntax of the paths is identical to the one used in the Excluded tab. E.g. "Private" would match any path containing the string "Private" at any position. "\Private\" would match any path containing a folder named "Private" at any position. "E:\Test\Private" would exclude this specific folder (and all its subfolders).

- However, the "file:" prefix is not supported because of a potential ambiguity in parsing.
- Like with the Excluded tab, the excluded paths themselves are shown (if they match the pattern), but not their contents.
- If switch /excl is used then the Excluded tab is ignored.
- Switches /excl and /exclfile support variables in their values:
  `*.jpg;*.avi;*.mov;*.mp4;*.mpg;*.wmv /excl="*<date mmdd>*"`
- This lists all JPGs in E:\Test\, excluding subfolder E:\Test\Private:
  `E:\Test\?*.jpg /excl=E:\Test\Private`
- This lists all JPGs in the current path, excluding subfolders containing the string "Private" or "Secret":
  `?*.jpg /excl=Private;Secret`
- This is a Branch View excluding the contents of certain paths (all versions are equivalent):
  `E:\Test\b\Pluto? /excl=Private;Secret /flat`
  `E:\Test\b\Pluto? /excl=Private;Secret :flat`
  `E:\Test\b\Pluto? /excl=Private;Secret:flat`
- List all files beginning with a* but not PNGs and GIFs, in the current path:
  `?a* /exclfile=*.png;*.gif`
- Same as above, but exclude everything with folders called "Private".
  `?a* /exclfile=*.png;*.gif /excl=\Private\`
- Switches /excl and /exclfile now allow quoting a single path. Can be useful to avoid ";" is seen as a list separator, e.g.:
  `?*.jpg;*.png /excl="E:\Test\a;b"`

Example for the Silent switch:

Use the /silent switch to prevent message boxes from popping before the search finally happens. Currently only one value (one message box) is supported.

```
E:\Test\here;E:\Test\gone?*.txt /silent=1   No message if some of the locations of a multi
                                            location search are not available.
```

Examples for the Types switch:

Use the /types switch to setup an inline custom type filter.

- If this type filter is set, only files are returned, no folders.
- Generic file types are supported, e.g. {:Image}.
- Variables are supported in the value, e.g. <date mmdd>.
- Wildcards are supported. This gives you nice options when looking for items by their extensions.

Examples:

- Search two folders for all files not older than 14 days of types JPG, PING, or PONG:

```
E:\Test\a;E:\Test\b?ageM: <= 14 d /types=jpg;ping;pong
```

- Search the current folder for all files "a*" or "b*" of types JPG or PNG:
  ```
  a*;b* /types=jpg;png
  ```
  Note that without this switch you had to do something like this to achieve the same:
  ```
  (a* or b*) and (*.jpg or *.png)
  ```
- Combine extensions and generic file types:
  ```
  ageM: <= 14 d /types=txt;{:Image};pop
  ```
- Using wildcards:
  ```
  /types=????*      Find all files with extensions of length 4 or more
  /types=??;????    Find all files with extensions of length 2 or 4
  /types=i*;j*      Find all files with extensions beginning with i or j
  ```

Examples for the Paths switch:

Use the /paths switch to filter items by the name of their parent folder or full path.

```
/paths=path;path;path
```

Remarks:

- Paths are separated by ; or | (the latter takes precedence).
- Wildcards (*, ?, #, and [groups]) are supported.
- If path contains no "\" (backslash) then it is compared with the immediate parent folder of the item in question.
- If path contains at least one "\" (backslash) then it is compared with the whole path (including trailing backslash!) of the item in question.
- Paths can be quoted (useful if path contains ";").

Examples:

- Return all items under parents "a" or "b":
  ```
  * /paths=a;b
  ```
- Return all TXT items under parents "a*" or "b*":
  ```
  *.txt /paths=a*;b*
  ```
- Return all TXT items under parents with exactly one character:
  ```
  *.txt /paths=?
  ```
- Return all items under parents "a;b" or "c;d":
  ```
  * /paths="a;b";"c;d"
  ```
- Return all items under parents "a;b" or "c;d" (alternative method):
  ```
  * /paths=a;b|c;d
  ```
- Return all items with "\test\" anywhere in the path:
  ```
  * /paths=*\test\*
  ```
- Return all items with "\test\" anywhere in the path, or with a parent three characters long:

```
* /paths=*\test\*;???
```

- Variables are supported in the value:

```
*.jpg;*.avi;*.mov;*.mp4;*.mpg;*.wmv /paths="*<date mmdd>*"
```

Examples for implicit path matching:

Path matching (ticked Path checkbox in Find Files, or /P switch) is implicitly turned on whenever the search term contains backslashes that are not just escapes for certain characters that would otherwise seen as special characters (eg " \!", "\[", " \(", "cat \and dog", ...), i.e. when the search term looks like a (part of a) path:

```
PN\P              //match all items under folder ending in "PN", beginning with "P"
\PN\P             //match all items under folder "PN", beginning with "P"
\PN\P | \PN\Q     //match all items under folder "PN", beginning with "P" or "Q"
```

Examples for the Contents switch:

Use the /contents switch to run an inline contents search.

- The search uses the vanilla setting of the Find Files | Contents tab: First option in the drop downs, all checkboxes unchecked. So, it's a text search only, binary files are not scanned!
- Of course, all settings on the Contents tab are completely ignored if running a Quick Search for contents.
- You can quote the string, and should do so if it contains something that looks like a switch (see 3rd example below).

Examples:

- Find all text files containing "em 2016":

```
* /contents=em 2016
```

- Find all *.txt files containing "em 2016":

```
* /contents=em 2016 /types=txt
```

- Find all text files containing "/limit":

```
* /contents="/limit"
```

Examples for the Contflags switch:

Use the /contflags switch to modify the behavior of the /contents switch (here above).

Syntax:

```
/contflags=[mode=n];[type=t];[options]

mode (choose one)
  n = normal (Default if missing)
  w = whole words
  c = wildcards
```

```
  r = regexp

type (choose one)
  t = text (Default if missing)
  b = binary
  tb = text and binary

options (any combination)
  c = match case
  i = invert
  h = hex string
  m = metadata (also search embedded metadata in image and media files)
```

Remarks:

- /contflags covers all options available on the "Find Files | Contents" tab, so you now have the full power of XY's content search available as inline parameter of Quick Search.

Examples:

- Find all text files containing "EM 2016" case-sensitive:
  `* /contents=em 2016 /contflags=;;c`
- Find all files with hex characters FF E0 at position 2. Note that the contents string has to be quoted in this case else the ":" will confuse the parser:
  `* /contents="2: FF E0" /contflags=;;h`
- Find all files with contents matching regexp pattern "\x61" (= "a"):
  `* /contents="\x61" /contflags=r;;`
- This works as well:
  `* /contents=\x61 /contflags=r`

Example for the Help switch:

Use the /? switch to check your pattern when in doubt. Like all switches it can be combined with other switches in any order.

```
!a* or b /?                    Check this search pattern.
```

On entering the pattern in Quick Search this will be shown in a message box (no search will be performed):

```
----------------------------
Query:  !a* or b /?
Parsed: (NOT "a*") OR "*b*"

Search: Quick Search
Mode: Boolean
```

```
FindBoolSmartDetect: Yes
ExtendedPatternMatching: Yes
MasterInvert: No
---------------------------
```

After "Parsed" you see the pattern in a human-friendly form. At the bottom you find the current state of those settings that affect the parsing of the pattern.

Examples for combined flags:

* /d /l=20 /maxdepth=1	Directories only, limit to 20, search this level plus one.
* /l=25 /lpd=2 /flat	Limit to 25 total, 2 per dir, Branch View.
* /lpd=1 /flat	Limit 1 per dir, Branch View.
* /dr /maxdepth=2	Find all directories in the current location and two levels deeper.
* /maxdepth=2 /dr	(same as above; sequence does not matter)

Notes:

- If you use more than one one-letter switch they must be combined after one slash, eg: /na, not /n /a.
- The maxdepth switch (like all other switches here) also works with the Name pattern in Find Files. Simply append it to the name pattern.
- /maxdepth=-1 is equal to unlimited depth, i.e. equal to omitting the switch.
- Stating a maxdepth > 0 implies recursion (Include Subfolders); you don't need to pass the /r switch as well.
- There is a short form for the /maxdepth switch: /md=n
- Tag database search is not affected by this switch.
- The switches /p and /x will not return drive roots but just folders.
- The switches /p and /x are ignored on a Dupes search.
- There is a short form for the /limit switch: /l=n
- When used together with the /p switch (return parents) the limit is applied to the children, not to the returned parents.
- There is a short form for the /limitperdir switch: /lpd=n
- Tag database search is NOT affected by the /limitperdir switch.
- In Branch View with "Let folders pass all filters" enabled the folders don't count when checking the limit.
- Quick Search Information Bar (in the list, above the search results): When the Match Case, Whole Word, or Non-Recursive buttons are pressed, it is indicated by appending [c], [w], or [n] to the search pattern display.

Examples for Default fields:

```
"Done" /fld=cmt                 Finds all items with comment "Done"
5 /fld=rating                   Finds all items with 5-star rating
:4 | 5 /fld=rating              Finds all items with 4-star or 5-star rating
"" /fld=rating /t               Finds all items in the tags database without a rating
```

Notes:

- In the case of custom columns and extra tags the field selector is identical to the column caption (case-insensitive).
- The /fld switch can be especially useful in Click and Search categories.

Single File Search

Sometimes you just want to see one particular file in the list, to check out its properties, preview it, or drag it somewhere. You don't care for the other thousands of files that exist in the same location, they just distract you and listing them wastes your precious time and adds to global warming.

Time for the revolutionary Single File Search: Simply enter the full path of the file or folder you want to see into the Quick Search interface, or -- prefixed with a "?" -- into the Address Bar, or into the Name field on the Find Files tab, and run the search. You will see your file within a very short time, and in splendid isolation.

Notes:

- The current location is completely ignored in this search. And when using the Find Files interface all other filters are ignored as well.
- The current location and bread crumb does not adjust to the path of the searched item. So the whole operation is pretty isolated from the rest of the interface.
- The Tree does not change at all.
- Single File Search does not include subfolders.
- In Address Bar and Quick Search also XYplorer native variables are supported.

Examples:

```
C:\Windows\System32\imageres.dll
<xypath>\XYplorer.exe
```

Using wildcards: Basic wildcards (*, ?) are allowed in the last component of the specified path/file. Note that in this case more than one result might be returned (although it's called Single File Search).

For example, this will probably just find notepad.exe:

```
C:\Windows\System32\Not*
```

This will find all matching DLL files:

```
C:\Windows\System32\na*.dll
```

This will list all JPGs in a certain network location:

```
\\VEGA\Users\Donald\Pictures\*.jpg
```

List all items in E:\ (excluding subfolder contents; kind of *treeless browsing*):

```
E:\*
```

Tip: Remember that Single File Search does not include subfolders. This is also true with wildcards in the pattern.

Variables

XYplorer native variables are resolved in the name pattern. So you can search for things like <U+202E> or <clipboard>.

Search Pattern Prefixes

While the default field in Quick Search is the filename you can match your pattern also against a number of other fields. The details are described under Find Files:

- Find Files by Size
- Find Files by Date and Age
- Find Files by Tags (Labels, Tags, Comments)
- Find Files by Columns
- Find Files by the Size Column Contents
- Find Files by Properties
- Find Files by Contents
- Find Files by Contained Characters
- Find Files by Multi Field Search
- Find Files by Meta Properties
- Find Files by the Length of their Name

Comments

You can append comments to a Quick Search term, separated by //. Must be right of any switches.

```
*.png //All PNG files
dateM: dw 6-7 /r //Modified on a weekend
```

If the Quick Search term has such a comment then the Quick Search bar will display the comment instead of the term. So you can use the comment as caption.

Examples

Wildcards and partial matches are supported, for example:

readme.txt	Find all files called readme.txt
black	Find all items containing 'black' in the name
*	Find all items
a*	Find all items beginning with a
a	Find all items containing a in the name
*.	Find all items without extension
a*.	Find all items without extension and beginning with a
a.	Find all items without extension and containing a in the name
*.png	Find all PNG files
* /fr	Find all files (no folders) (include subfolders)
*.txt /n	Find all TXT files (don't include subfolders)
ageM: 1 d	Find all items modified yesterday
ageM: 0 d	Find all items modified today ("ageM: d" works as well)
ageM: -1 d	Find all items modified tomorrow
ageM: -2 d	Find all items modified the day after tomorrow

Also tags can be searched for, for example:

lbl:red	Find all items labeled 'red'
lbl:#1	Find all items labeled with label #1
cmt:chicago	Find all items with 'chicago' in the comment
tags:cats|dogs	Find all items tagged 'cats' or 'dogs'
!tags:*	Find all items without tags

Also Boolean (:) and RegExp (>) patterns, for example:

:!#	Find all items with no number in the name
:*.jpg | *.png	Find all JPGs and PNGs
:*.jpg or *.png	Find all JPGs and PNGs
:tags:cats AND name:a*	Find all items tagged 'cats' and beginning with a*
>.*(\.jpg|\.png)$	Find all JPGs and PNGs

Note: The Boolean marker (:) is not necessary (but still tolerated) if you enable *Configuration* | *Find and Filter* | *Find Files & Branch View* | *Find Files* | *Enable smart Boolean query parsing*. Boolean expressions like these will then work in a Quick Search:

```
a* OR b* /n
size: > 5MB AND size: < 6MB
4 | 5 /fld=rating
dateM: dw 6 | dw 7
```

Quick Search Dialog: Quick Find Files

The Quick Search dialog has a little toolbar featuring, among some other buttons, a Find Files button. When this button is pressed (use Ctrl+F to toggle it) the dialog acts as a remote control for the Find Files tab in the Info Panel (F12), kind of a Quick Find Files. The entered pattern and the current location are injected into the Find Files interface (even when the tab is not visible), and all the current settings of the Find Files tab are effective, just as if you had entered that name pattern right into the tab. So you can quickly do a Find Files search without first opening that tab and without having it in the way of the search results listing.

If Name & Location happens to be unticked on the Find Files tab it will be silently ticked.

An obvious use case would be to do a Quick Search while honoring all the settings on the Name and Location tab (Ignore diacritics, Whole words, Include subfolders), and also the Excluded folders.

Note: The "Find Files" button in the Quick Search dialog only affects searches triggered from this dialog. Other Quick Searches (e.g. via Address Bar, or scripting) are not affected and run normally.

4.3 Visual Filters

Index

- Visual Filter Syntax
  - Wildcards
  - OR
  - AND
  - NOT
  - No-Extension
  - Special Operators
  - Captions
  - Comments
- Matching
- Usage
- Variables in Visual Filters
- Visual Filters by Particular Properties
  - Visual Filters by Attributes, Size, Date, Age, Length, and Properties
  - Visual Filters by Tags
  - Visual Filters by Columns
  - Visual Filters by Custom Columns
  - Visual Filters by Properties
  - Visual Filters by Generic File Types
- Must-Match-Patterns

- Power Filters
- Global Visual Filters
- Global Power Filters

Visual Filters

Visual Filters let you control what you see in the file list by stating simple wildcard patterns like *.txt. The Visual Filter is per-tab and, of course, saved between sessions. The current filter is displayed on the tab headers in a distinct color.

The main commands for Visual Filters are found in menu View | Tab, which is also available by right-clicking the tab headers. There is also a toolbar button "Visual Filter" to quickly toggle the last used filter.

Configuration Options

All configuration options for Visual Filters are also located in the right-click menu of the toolbar button "Toggle Visual Filter". See here for the details.

Visual Filter Syntax

Visual Filters and Live Filters (Live Filter Box) share exactly the same syntax. So everything below is also true for Live Filters.

Wildcards: You can use the wildcard "?" for any single character, "*" for any number of characters (including none). Visual Filter pattern matching is not case-sensitive: A==a. If Enable extended pattern matching is enabled you can use "#" for any single digit (0-9); to match the character # enclose it square brackets ([#]). To match opening square brackets ([ ) enclose them in square brackets ([[]).

```
Desktop|*.txt      = List all TXT files on Desktop
Desktop|???.txt    = List all TXT files on Desktop with exactly 3 characters in the base name
Desktop|???*.txt   = List all TXT files on Desktop with at least 3 characters in the base name
Desktop|*##*.txt   = List all TXT files on Desktop with at least 2 digits in the base name
Desktop|*[[]*      = List items on Desktop containing an open square bracket in the name
Desktop|*[#]*      = List items on Desktop containing # in the name
```

OR: You can state an unlimited number of wildcard patterns separated by ";" (logically treated as OR).

```
E:\Test|*.gif;*.jpg;*.png      = List all GIF, JPG, and PNG files in E:\Test
```

Note that you can also OR-connect Visual Filters by OR or | (both surrounded by spaces; OR is not case-sensitive, so or works as well).

```
E:\Test|*.gif | *.jpg | *.png      = List all GIF, JPG, and PNG files in E:\Test
E:\Test|*.gif OR *.jpg OR *.png    = List all GIF, JPG, and PNG files in E:\Test
```

The | has priority over ; so using | you can filter items containing the ; character:

```
Desktop|; | ,      = List all items on Desktop containing ";" or ","
```

If you state only one pattern and this pattern contains a ; then you have to add another dummy | (mind the surrounding spaces) to avoid that the ; is treated as separator:

```
Desktop|; |        = List all items on Desktop containing ";"
Desktop|;a |       = List all items on Desktop containing ";a"
Desktop|*; * |     = List all items on Desktop containing "; "
```

AND: Boolean AND is supported. The operator can be AND or & (both surrounded by spaces; AND is not case-sensitive, so and works as well). Examples:

```
cat AND dog AND pig              = all of these animals must be in the name
cat & dog & pig                  = all of these animals must be in the name
water and ageM: w                = "water" in the name, modified this week
size: < 1KB AND *.txt; size: < 100 AND *.gif          = TXT < 1 KB, or GIF < 100 bytes
"Image Files 2013" {:Image} and 2013                  = image files with "2013" in the
```

name

If you want to show items containing " and " you have to escape it with "\", or quote the whole pattern:

```
cat and dog   //matches items with "cat" and "dog" in the name
cat \and dog  //matches items with "cat and dog" in the name
"cat and dog" //matches items named "cat and dog"
```

NOT: A leading ! (exclamation mark) will invert the filter, i.e. show only items *not* matching any of the patterns. If you want to show items starting with "!" you have to escape it with "\":

Examples:

```
a*;b*    -> show all items starting with "a" or "b".
!a*;b*   -> hide all items not starting with "a" or starting with "b".
a*;!b*   -> show all items starting with "a" or not starting with "b".
\!a*;b*  -> show all items starting with "!a" or "b" (! is escaped by \).
water and !ageM: w  -> "water" in the name and not modified this week.
```

To logically reverse an entire pattern, wrap it in parentheses and prefix !.

```
!(a*;b*)            -> hide all items starting with "a" or "b".
!(Size: 529 KB | Name:*.txt)      -> NOT (Size 529 KB OR Name *.txt)
```

Note: A filter pattern that consists of a single ! is not seen as unary Boolean NOT operator but as the normal "!" character.

Also the verbal NOT-operator "NOT " (or "not ", or "Not ", it's case-insensitive) is supported:

```
Size: 529 KB or not Ext: txt
not Size: 529 KB or Ext: txt
not (Size: 529 KB or not Ext: txt)
dateM:NOT dw 6-7
dateM: NOT dw 6-7
NOT dateM:dw 6-7
NOT dateM: dw 6-7
```

No-Extension

Patterns ending with a dot match files that have no extension. Folders never match such a pattern.

```
*.     -> show all files without extension.
a*.    -> show all files beginning with "a" and without extension.
```

Special Operators

You can define a filter to apply only to files but show all folders. Simply prefix "\|" to the pattern(s). For example:

```
\|*.txt   -> show only *.txt files and all folders
```

```
\|!*      -> show all folders and no files
```

To hide all folders simply prefix "!\|" to the pattern(s). For example:

```
!\|*.txt  -> show only *.txt files and hide all folders
!\|*      -> show all files and no folders
```

You can also define a filter to apply only to folders but show all files. Simply prefix "*|" to the pattern (s). For example:

```
*|a*      -> show only folders beginning with "a" and all files
*|!*      -> show all files and no folders
```

To hide all files simply prefix "!*|" to the pattern(s). For example:

```
!*|a*     -> hide all files and show only folders with a*
!*|*      -> hide all files and show all folders
```

Captions

You can also define a caption for each Visual Filter. Simply prefix a quoted string followed by at least one space to the filter:

```
"Images" *.gif | *.png | *.jpg
"Items ending with g" >g$
```

The tab captions will show only the (unquoted) caption of a Visual Filter with caption.

The caption of a Visual Filter can have a short and a long version, the former used for tab captions (where space is scarce), the latter for menu captions (where space is abundant). The short and the long version are separated by "|" (pipe). This feature has been added especially to improve the appearance of the new Powers Filters which otherwise take too much space in the tab headers. Here are some examples:

```
"30 Mins|Modified In The Last 30 Mins" ageM: <= 30 n
"3 Hours|Modified In The Last 3 Hours" ageM: <= 3 h
"Today|Modified Today" ageM: d
"Week|Modified This Week" ageM: w
"Year|Modified This Year" ageM: y
```

Comments

Trailing comments are allowed after "//", e.g.:

```
ageM: w //modified this week
```

Matching

Loose matches are supported, i.e. wildcards are auto-added on both ends if no wildcards are passed with the pattern. For example, to show all items containing the letter "e" in the name simply state "e" as filter:

```
e      -> show all files containing "e" in the name
       -> show all files containing " " (one space) in the name
 e     -> show all files containing " e" (one space followed by "e") in the name
```

To force exact matches (as opposed to loose matches) you can quote a filter. If you state multiple filters then each filter that should be matched exactly should be quoted.

```
"readme.txt" | "setup.txt"        -> show only files readme.txt and setup.txt
"sel" "readme.txt" | "setup.txt"  -> the same with caption "sel"
```

Regular Expressions

Visual Filters also support the Regular Expressions syntax. Like elsewhere in XYplorer to activate the regular expressions mode simply prefix the pattern with the character ">". For example:

```
>\.doc$                          -> show only *.doc files
>.*(\.png|\.jpg)$                -> show only *.png and *.jpg files
```

For details on XYplorer's variety of regular expressions, search the web for: vbscript regular expressions.

Usage

Note that Visual Filters in combination with tabs allow you to show completely different listings ("filtered views") of the same location. You could for example have a tab "DLLs" (Visual Filter set to *.dll) and a tab "OCXs" (Visual Filter set to *.ocx) and have both pointing to Win\System32.

If a Visual Filter is active, the item count on the status bar tells you the total number of items, and it does it in blue (or whatever you set the Marked Color to) as a further indication that a filter is active and effective.

Variables in Visual Filters

Visual Filters support environment variables and XYplorer native variables. Using variables can make filters portable (adaptive to their host system).

For example:

- `<date yyyy>`
  Shows only items with the current year in the name.
- `%username%`
  Shows only items with the current user name in the name.
- `%pathext%`
  A special case: %pathext% happens to return a list of patterns which is quite compatible with XYplorer's Visual Filter syntax:

%pathext% => .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH
So you'd see a list of what the host system considers executable files.

Notes:

- The tab headers show the resolved values of the variables.
- The MRU list under the Visual Filter Toolbar button shows the variables.
- The "Toggle Visual Filter" toolbar button's tooltip shows the variable form of the current VF.
- Visual Filters are stored as variables (e.g. "%username%") only when passed through the Set Visual Filter interface. Visual Filters passed through the Address Bar are resolved (e.g. "Donald") *before* being passed to the Visual Filter engine and stored as the hard invariable value.
- The Visual Filter toolbar button's tooltip and the Status Bar first section's tooltip both show the current Visual Filter pattern, which is the active pattern after any variables have been resolved. This is useful when you have a VF like this and want to know what your filter actually looks like: `"Matching Clipboard" <clipboard>`

Visual Filters by Particular Properties

Visual Filters by Attributes, Size, Date, Age, Length, and Properties

Besides Name, also other item properties can be used for visual filtering. The syntax is very similar to the Color Filters syntax in that a selector is prefixed to the pattern:

Filter Type	Selector	Example	Notes
Name	[name:]	*.txt (or name:*.txt)	
Attributes	attr:	attr: readonly	
Attr. List	attrlist:	attrlist: hsr	
Size	size:	size: > 100 KB	Size filters don't support directories, only files.
Date Created	dateC:	dateC: 07.12.2012	
Date Modified	dateM:	dateM: 2012.12.01 - 2012.12.31	
Date Accessed	dateA:	dateA: 07.12.2012	
Age Created	ageC:	ageC: > 5 y	
Age Modified	ageM:	ageM: d	
Age Accessed	ageA:	ageA: <= 5 d	
Full Name Length	len:	len: > 260	Full Name is the file name with path.
Title Length	lenT:	lenT: <= 12	Title is the file name without path.
Property	prop:	prop:imagex: > 1500	Property filters need sub selectors to specify the desired property.
Path	path:	path:*\hello\	Path is the full path excluding the item name itself.

Examples:

*.txt	named *.txt
name: *.txt	named *.txt
attr: system	SYSTEM attribute set
attrlist: hsr	HIDDEN AND SYSTEM AND READONLY attribute set
size: >= 1.5 GB	1.5 GB or bigger
size: 0	empty
dateM: 2010.01.01 -	modified 2010 or later
dateC: dw 6-7	created on a weekend
ageM: y	modified this year
dateM: m 5 & dateM: d 1	modified on the 1st of May (of any year)
dateM: dy 51	modified on the 51st day of the year (the 20th of February of any year)
ageA: < 5 d	accessed less than 5 days ago
len: > 260	longer than 260 characters
lenT: <= 12	12 or less characters
path:hello	match all items with a path containing the string "hello"
path:\hello\	match all items with a path containing a folder "hello"
path:*\hello\	match all items with a direct parent folder "hello"
path:!.git	exclude the whole sub-branch ".git" with all contents from a Branch View (needs "Enable extended pattern matching" ticked)
!.git\	(same as above)
Modified: 2025	modified in 2025 (using the English column name as selector; needs ISO 8601 date format)
Created: 2025-07	created in July 2025 (using the English column name as selector; needs ISO 8601 date format)

Examples for Property filters:

prop:dimensions:638 x *	638 pixels width
prop:imagex: > 1500	a width larger 1500 pixels
prop:CameraModel:*Canon*	"Canon" in the camera model name
prop:fileversion:20.40*	file version number 20.40*
prop:System.Comment:love	system comment "love"
prop:#ShortcutTarget:?*	a non-empty shortcut target
prop:#mp3.composer:*Haydn*	Haydn in the MP3 composer tag
prop:#mp3.title:no*	an MP3 title starting with "no"
prop:#Image.Fuji.FilmMode:Classic*	Fuji Film Simulation "Classic"
prop:#AspectRatio: 16:9	an aspect ratio of 16:9
prop:#AspectRatio: v2	an aspect ratio of v2
prop:#AspectRatio: >= v2:1	an aspect ratio same or wider than DIN 476 (aka Din A) landscape
prop:#Hash.MD5:1092b0ce7d106d3d72e737c66ea20fa1	this MD5 hash

```
prop:#HardLinks: > 1                          more than one hardlink
```

Examples for Property filter dimensions which can be used to filter (or search) by area (width x height):

```
prop:dimensions: 1000            match area 1000
prop:dimensions: == 1000         match area 1000
prop:dimensions: 1000 x 1        match dimensions 1000 x 1
prop:dimensions: == 1000 x 1     match dimensions 1000 x 1
prop:dimensions: >= 1000         match area >= 1000
prop:dimensions: >= 1000 x 1     match area >= 1000
prop:dimensions: >= 50 x 20      match area >= 1000
```

Like Name filters also these filters can be inverted by a prefixed "!":

```
ageM: d         //modified today
!ageM: d        //not modified today
```

Note that there MUST be a space between counter and unit:

```
ageC: <= 3 n     //created not more than three minutes ago
```

And they can be freely combined separated by ";" or "|" (both meaning logical OR), for example:

```
size: <= 10 KB; name: *.png; ageM: w
```

Scope Suffixes:

Visual Filters with selectors also support scope suffixes.

For example, this filter will show only folders created today: `ageC d: d`

And this filter will show only files modified yesterday: `ageM f: 1 d`

Notes:

- The date has to be given in a form that is recognized as a valid date format on your locale. The time part can be skipped in which case it is internally set to 00:00:00.
- The selector is not case-sensitive: Datem: would work as well.
- The selectors date:/age: default to dateM:/ageM: ("Date/Age Modified").
- Spaces after the selector are optional: "ageM: d" is equivalent to "ageM:d"
- The following settings apply to the "name:" and "path:" matching behavior in Visual Filters and Live Filter Box:
  - Configuration | Find and Filter | Filters & Type Ahead Find | Visual Filters | Match case
  - Configuration | Find and Filter | Filters & Type Ahead Find | Visual Filters | Ignore diacritics

Visual Filters by Tags

You can as well filter items by Tags and Extra Tags. Again, a selector is prefixed to the pattern:

Selector	Value
lbl:	#n, where n = index of Label (#1 = first label). OR: Name of Label, wildcard pattern, case-insensitive (A==a).
tags:	A comma-separated list of tags, case-insensitive (A==a). All tags must be present in item (item can have more tags). Wildcard patterns allowed.
cmt:	Wildcard pattern, case-insensitive (A==a).
ex1 - ex16:	Wildcard pattern, case-insensitive (A==a).

Notes:

- All matching is case-insensitive (A==a).
- Item must be in tag DB to match. This regards patterns like "lbl:0" (see examples below).
- Extra columns can also be referred to by their caption, again case-insensitive (A==a).

Example	matches all items...
lbl:green	with label "green"
lbl:g*	with a label beginning with "g"
lbl:#1	with label #1
lbl:?*	with any label
lbl:	with or without any label (but being in tag DB)
lbl:*	with or without any label (but being in tag DB)
lbl:""	without any label (but being in tag DB)
tags:Jazz	with tag Jazz (and possibly any other tags)
tags:Rock,Pop	with tags Rock and Pop (and possibly any other tags)
tags:r*	with tags beginning with "r" or "R" (and possibly any other tags)
tags:r*,p*	with tags beginning with "r" or "R" and tags beginning with "p" or "P" (and possibly any other tags)
tags:?*	with any tags
tags:	with or without any tags (but being in tag DB)
tags:*	with or without any tags (but being in tag DB)
tags:""	without any tags (but being in tag DB)
cmt:*ow	with a comment ending in "ow"
cmt:	with or without any comment (but being in tag DB)

```
cmt:*             with or without any comment (but being in tag DB)
cmt:""            without any comment (but being in tag DB)

ex4:Ch*           where extra column #4 value begins with "Ch"
ex4:""            where extra column #4 is empty
country:Ch*       where extra column "Country" begins with "Ch"
-----------------------------------------------------------------------
```

Visual Filters by Columns

You can also filter the file list by the data shown in any other columns. All columns (apart from "Index") are supported. Simply use the column header (in your interface language) caption has selector.

Examples:

```
Ext: jpg
Ext: j*g
Size: >= 500000
Size: 529 KB
Size: 528.49 KB
Size: <= 528.49 KB
Modified: 2014-09-21 12:55:09
Label: Green
Tags: Charlie
Comment: A cool and easy to use feature!
Len: 108
Attr: R           (R must be among the attributes)
AttrList: HSDLI   (attributes must perfectly match)
```

Visual Filters by Custom Columns

You can also filter the file list by the data shown in Custom Columns.

- You can reference the columns by their canonic name and also by their caption (the latter can be ambiguous).
- Like with other filters you can use numeric operators (>=, <, ==, etc) and wildcards (*, ?). What makes sense when, only you can know since you wrote the Custom Columns.
- Note that the filtering also works for columns that are not currently present in the list. So you now have handy shortcuts to ever so complex filters.
- Examples:

```
Filter Pattern        Used Column Reference    Matches
-----------------------------------------------------------------
cc18: >= 2            Canonic Name             all numbers >= 2
Date Done: *25 days   Caption                  1 month  25 days
```

Visual Filters by Properties

You can also filter by Shell Properties and Special Properties. Works independently of any currently visible columns. Simply use the property name as selector, for example:

```
Item type: TXT File
MD5: d41d8cd98f00b204e9800998ecf8427e
Aspect Ratio: 2:3
```

Visual Filters by Generic File Types

You can filter certain file types, e.g. image, using the syntax `{:Type}`, for example `{:Image}`. This spares you typing long list of extensions (*.png; *.jpg; *.psd; ...). The following generic file types are supported:

```
{:Text}       *
{:Image}      *
{:Photo}      *   = All image formats that may contain Exif data.
{:Audio}      *
{:Video}      *
{:Media}      *   = Audio & Video
{:Font}       *
{:Vector}
{:Web}        *
{:Document}   *
{:Archive}
{:Executable}
```

Examples that can be pasted right into the Address Bar:

```
|{:Image}       Show only image files in the current location
D:\|{:Text}     Show only text files in the root folder D:\
```

Note:

The types marked * in the list above can be customized in Configuration | Previewed Formats.

Must-Match-Patterns

Matching a Must-Match-Pattern is necessary but not sufficient if there are more patterns to be checked. There are two special prefixes that go in front of any other prefixes and turn the following pattern into a Must-Match-Pattern: nec and no.

```
nec:ageM: y; *.jpg; *.raf    //show all JPGs and RAFs modified this year
*.jpg; nec:ageM: y; *.raf    //show all JPGs, and all RAFs modified this year
```

```
no:ageM: y; *.jpg; *.raf     //show all JPGs and RAFs NOT modified this year
*.jpg; no:ageM: y; *.raf     //show all JPGs, and all RAFs NOT modified this year
```

Tip for understanding: nec and no mean to AND this pattern with everything coming afterwards:

```
nec:a;b;c;d                   = a AND (b OR c OR d)
no:a;no:b;c;d                 = NOT a AND (NOT b AND (c OR d))
nec:a;b;nec:c;d               = a AND (b OR (c AND d))
```

More examples:

```
XY Visual Filter Syntax     Equals in Boolean Notation
-------------------------------------------------------------------------
nec:ageM: y; size: > 50KB   modified this year AND larger than 50KB
nec:*.txt; a*; b*           "*.txt" AND ("a*" OR "b*")
x*; nec:*.txt; a*; b*       "x*" OR ("*.txt" AND ("a*" OR "b*"))
no:ageM: y; size: > 50KB    NOT modified this year AND larger than 50KB
no:*.txt; a*; b*            NOT "*.txt" AND ("a*" OR "b*")
no:*.txt;no:*.ini;ageM: y   NOT *.txt AND NOT *.ini AND modified this year
```

Power Filters

The "Toggle Visual Filter" toolbar button got an arrow dropdown featuring a set of predefined visual filters called "Power Filters", e.g. "Image Files", "Modified Today", "Empty Files" and many more. These filters can be customized in List Management.

Using Power Filters to Select Files

You can as well use the Power Filters menu to (un)select files by type, just like you know it from the "Type Stats and Filter" menu:

- Hold CTRL: Add all files of this type to the selection.
- Hold SHIFT: Remove all files of this type from the selection.

The actually applied selection filter patterns are displayed in the Status Bar.

Note, however, that more complex Power Filters (age, size, attr) are not supported here.

Global Visual Filters

A Global Visual Filter (GVF) is applied to all Lists in both panes.

Any local Visual Filters (VF) are applied after the GVF. Effectively any local VFs then work on top of the GVF, i.e. they further narrow down what has passed the GVF.

With Global Visual Filters you can tell your file manager to list only TXT files, or only files smaller than 1KB, or to list only files modified this month. Globally, in each tab. This can be immensely useful.

Usage notes:

- GVF syntax is identical to VF syntax.
- Both GVFs and VFs support Master Invert (show all items that do NOT match the filter) by a prefixed "!", for normal patterns as well as for RegExp.
- Both GVFs and VFs support their own private scope prefixes (show all folders, hide all folders, etc).
- You can add captions and comments to both GVFs and VFs.
- Mixed separators (| and ;) are okay: GVF can be "*.txt|*.xys" and VF at the same time can be "a*; b*".
- Mixed syntaxes are okay: You can combine a GVF RegExp with a VF RegExp or with a VF wildcard pattern (or pattern list) or vice versa.
- A GVF will not show filter information in the list or in tab headers. So it's more stealth than a local VF.
- The active GVF pattern is displayed in the Status Bar tooltip (first section of the bar).
- Just like VFs, GVFs apply only to the List, not the Tree, and they don't affect the dropdowns in the Address Bar (both as opposed to Ghost Filter).
- The GVF is remembered between sessions.
- There is a Toolbar button to toggle the last GVF.

Global Power Filters

The "Toggle Global Visual Filter" toolbar button got an arrow dropdown featuring a set of predefined visual filters called "Global Power Filters", e.g. "Image Files", "Modified Today", "Empty Files" and many more. These filters can be customized in List Management.

4.4 Live Filter Box

Live Filter Box

Located to the right of the Address Bar or the left of the Status Bar, the Live Filter Box (LFB) offers Filter-As-You-Type. Ultra-simple, it just works.

Usage

- To show/hide the Live Filter Box use menu Window | Show Live Filter Box.
- Show the box to the left of the Status Bar: tick Window | Arrangement | Live Filter Box in Status Bar. Note that if the Address Bar is hidden then the LFB automatically goes to the Status Bar.
- Syntactically and functionally it's identical to Visual Filters, so you have all the power and need not learn anything new about patterns. Of course, wildcards * and ? and Regular Expressions are supported.
- Each tab internally remembers its last used filter even across sessions.
- Toggle the current filter by clicking the filter icon in the box (it changes to an X when a filter is active). Alternatively, use "View | Tab | Toggle Live Filter" (Ctrl+Alt+F3).
- When the LFB is shown it gets the focus, when it's hidden any live filter is removed from the list.
- When the LFB is shown and the focus is in the list (single pane), press TAB to move the focus to the LFB.
- Pressing ESC in a filled box clears the filter.
- When the filter is removed, the last scroll position of the unfiltered list is restored (it's even remembered across tab switches and sessions). Exception: If an item is focused AND selected at the moment the filter is removed, that item will be scrolled into view.
- Hiding the LFB will automatically clear any filter.
- When a live filter is active the filter information is shown in the list if this is ticked (highly recommended): "Configuration | Filters & Type Ahead Find | Visual Filters | Show filter information in list". The bar has a right-click menu.
- Dbl-click the filter information bar in the list to remove the live filter quickly.
- There is a MRU (Most Recently Used) drop-down list that is shared with the Visual Filters MRU list. Since both filters have the same syntax, it's possible and makes sense.
- You can use the Up, Down, PageUp and PageDown keys to scroll through the MRU items in a dropdown list when it's collapsed.
- Up and PageUp will clear the box if they would go beyond the top item of the list. This allows you to quickly move to the first MRU item and back to the unfiltered list using Down / Up in the box.

- You can use Alt+Down or Shift+Down to drop the list, and Alt+Up or Shift+Up to undrop it.

List navigation right from the Live Filter Box

Some extra functionality allows pretty comfortable keys-only List navigation right from the box:

- Keys Up, Down, PageUp, PageDown work to move the focus within the list.
- Ctrl+Backspace will remove the last word in the LFB.
- Hitting Enter on a folder navigates into the folder and resets the live filter back to nothing.
- Hitting Enter on a file runs the file and doesn't reset the live filter.
- Hitting Enter even works on the focused item even if it is not selected. In that case the focused item is auto-selected before processing continues. Saves a key stroke in some situations.

Filter-As-You-Type right in the file list

Tick "Configuration | Filters & Type Ahead Find | Type Ahead Find | Redirect typing to Live Filter Box" to redirect list input to the box. The behavior now changes to something even more radical and comfortable. If focus is in the List then:

a) All typing in the List works as if typing in the Live Filter Box.

- Pressing any character key will append it to the current filter. The box is shown if not yet visible.
- Pressing Backspace will remove the last character of the current filter.
- Pressing ESC will remove the filter if there is any.

b) The focus will stay in the List all the time.

- The Live Filter Box becomes visible when you type in the list and is filled with the characters you type, but the focus stays in the list.

So the Live Filter Box is under these conditions not much more than a visual feedback of the current filter. You can still, however, move the focus into it and work with it like normal.

Live Filter Box Right-Click Menu

The "Filter"-icon of the Live Filter Box has a right-click menu that offers quick access to a couple of useful settings and commands:

- Toggle Live Filter: Quick access to the command View | Tab | Toggle Live Filter.
  Tip: This command is also triggered when you single-click the "Filter"-icon!
- Show Live Filter Box: Quick access to the command Window | Show Live Filter Box.
- Live Filter Box in Status Bar: Quick access to the command Window | Arrangement | Live Filter Box in Status Bar.
- Quick access to the settings in Configuration | Find and Filter | Filters & Type Ahead Find | Live Filter Box.
- Toggle Favorite Live Filter: By this command (which only exists here in the menu) you can add or remove the current Live Filter pattern to an array of Favorite Live Filters which is displayed at the

bottom of that same menu. This array can hold up to 32 patterns. It's automatically sorted alphabetically. Clicking such a Favorite Live Filter will feed the pattern into the box and filter the list accordingly.
Whether these Favorite Live Filters are saved in the INI-file is controlled by the same setting that controls the saving of the Visual Filters MRU: Configuration | General | Startup & Exit | Save Settings | Include most-recently-used lists on save: Visual Filters.
Note that the Favorite Live Filters are global (same list in all tabs), whereas the last used Live Filter (used by "Toggle Live Filter") is stored per tab.
Tip: You can remove a Favorite Live Filter by holding CTRL while you click on it in the right-click menu of the "Filter"-icon.

Filter Syntax

See Visual Filters.

Width of the Live Filter Box

You can adjust the width of the box in a snap by hitting Shift+Alt+Wheel over the box. The width is increased or decreased in steps of 10 pixels. Minimum is 40 pixels, maximum is screen width.

Live Filter Box in Small Lists

Unlike the main file list, many of the small lists found throughout the application interface may contain an asterisk (*) in the data, a character that acts as a wildcard by default. To match an asterisk, you can and must escape the single asterisk by doubling it:

```
*.exe     //matches any string (incl. nothing) that ends with .exe
**.exe    //matches any string that contains *.exe (internally converted to *[*].exe*)
```

Further Remarks

Beginning with v26.20, live filtering is done in memory without accessing the file system, resulting in a much faster and smoother experience.

There is a function to focus the LFB via keyboard: Miscellaneous | Focus Functions | Focus Live Filter Box. The factory default is Ctrl+Alt+X. If the LFB is not visible the function will show it before focusing it.

Tip: Assign "Numpad Subtract" as additional keyboard shortcut to this (via Tools | Customize Keyboard Shortcuts...). A fast handy key which even makes sense here since you use the LFB to subtract from the list.

4.5 Color Filters

Index

Color Filters Overview
Control Selector
Scope Suffix
Switches
Style Switches
Exact Matching
Boolean Logic
The Types of Color Filters
Color-coding based on file or folder names (name:)
Color-coding based on folder names (dir:)
Color-coding based on file attributes (attr:)
Color-coding based on a list of file attributes (attrlist:)
Color-coding based on file size (size:)
Color-coding based on file date (date:)
Color-coding based on file age (age:)
Color-coding based on name length (len: and lenT:)
Color-coding based on properties (prop:)
Color-coding based on tag (lbl:, tags:, cmt:)
Instant Color Filters

Color Filters Overview

Color filters allow you to color-code the files and folders currently visible in the List pane based on various criteria like name, attributes, size, date, age, name length, and properties. Filters based on name, age, len, lent, and attributes optionally also work in the Tree pane. Color-coding items helps to easily spot and distinguish particular items.

Configuration of Color Filter

See Configuration | Color Filters.

General Color Filter Syntax

Color Filters are defined by a selector and a pattern in the general form `selector: pattern`. The following selectors (= filter types) are available:

Filter Type	Selector	2nd Selector

```
-----------------------------------------------
Name            [name:]
Folder Name     dir:
Attributes      attr:
Size            size:
Date Created    dateC:
Date Modified   dateM:
Date Accessed   dateA:
Age Created     ageC:
Age Modified    ageM:
Age Accessed    ageA:
Name Length     len:
Title Length    lenT:
Property        prop:      property selector:
```

Examples:

```
name: *.txt               = named *.txt
dir: *.txt                = folder named *.txt
attr: system              = SYSTEM attribute set
size: >= 1.5 GB           = 1.5 GB or bigger
dateM: 2010.01.01 -       = modified 2010 or later
dateC: dw 6-7             = created on a weekend
ageA: < 5 d               = accessed less than 5 days ago
len: > 260                = path/name longer than 260 characters
prop:dimensions: 1500 x *  = 1500 pixels wide
```

Notes:

- The Name Selector is optional: if no other selector is found then Name is assumed by default.
- The space between selector and pattern is optional.
- Case does not matter in selectors.
- Property filters are special in that they need an additional property selector to specify the sort of property. This can be a verbal identifier (e.g. `prop:dimensions:`) or a numeric identifier (e.g. `prop:#31:`).

Control Selector

An optional control selector lets you define whether a filter is applied to the Tree, or to the List, or to both, for each filter individually.

```
Control Selector        Function
-----------------------------------------------------
T: or t:                Limit color filter to the Tree
```

```
  L: or l:                Limit color filter to the List
  B: or b:                Apply color filter to both Tree and List (useful for filters that
are by default applied only to the List)
  else                    No limits (use defaults)
```

The control selectors are prefixed to the filter definition (but come after any caption).

Examples:

```
T:ageM: <= 24 h
L:ageM: <= 24 h
"Modified Today" T:ageM: d
```

Remarks:

- Control selectors also allow you to color code the same criteria differently in Tree and List (whatever that would be good for).
- In case of normal (non-instant) Color Filters the control selectors are limited by the global settings "Apply color filters to the List" and "Apply color filters to the Tree": If any of them are disabled, then a Control Selector cannot overwrite this.

Scope Suffix

An optional scope suffix can be used to limit color filters to directories or to files.

Scope suffixes are appended to the primary selector separated by a space. "d" stands for "directories", "f" stands for "files", "df" (or "fd") for both (i.e. unlimited scope). A missing scope suffix defaults to "df" (unlimited scope) but see some obvious exceptions below.

Usage: For example, you want to color folders by age created in Tree and List but don't want to color the files.

Examples:

```
ageC d:d //folders created today
ageM f:d //files modified today
prop d:#HardLinks: > 1 //folders with more than one hardlink
```

Exceptions:

- "dir:" is always directory-only by definition
- [no selector] is equivalent to "name f:" (name files-only), for example: `*.txt`

Switches

The switch `/r` can be used to assign a color filter to a folder and all of its subfolders. It has to be appended to the selector (including any scope suffix) separated by a single space.

Examples, using a quoted pattern for exact matching:

```
Pattern             Auto-adjusted to     Match           No Match
-------------------------------------------------------------------------
dir:"code"          *\code\              C:\code\        C:\code\sub\
dir /r:"code"       *\code\*             C:\code\
                                         C:\code\sub\
name d /r:"code"    *\code\*             C:\code\
                                         C:\code\sub\
-------------------------------------------------------------------------
```

Of course, you could also use non-exact patterns (`dir /r:code`) for broader matching.

Style Switches

Optionally append style switches to the end of the pattern (but before any comment), separated by |. Switches can control the look of the color-coding on a per-filter level. Each switch is represented by a single letter (case-sensitive). The sequence of the letters has no significance.

Available switches:

```
b = Draw background colors as bold frame (2 pixels wide unfilled rectangle).
f = Draw background colors as frame (1 pixel wide unfilled rectangle).
p = Number plate style (frame with filled background); has to be combined with switch f (frame) or b (bold frame).
r = Draw background colors with rounded corners.
s = Draw background colors in distinctive shapes.
w = Draw background colors as wide as the column.
```

Available reversed switches, prefixed with "-" (minus):

```
-r = Don't draw background colors with rounded corners (even if the general setting is enabled).
-s = Don't draw background colors in distinctive shapes (even if the general setting is enabled).
```

Notes:

- With "f" it is recommendable to leave the text color undefined (click the "Clear" button) when you like to switch between dark and light mode.

Examples for Configuration | Colors and Styles | Color Filters:

```
B:prop:#empty:2|swf     //shape, wide, frame
len:>=260|-r            //sharp corners
```

Examples for Tools | List Management | Color Filters...:

```
len:>=260|f //overlong items>,DF2076    //frame, text color undefined
*.|r //no extension>,D8F0A6             //round corners, text color undefined
```

Exact Matching

You can force an exact match by wrapping your pattern in quotes. Otherwise the pattern is automatically wrapped into asterisks (*), unless it already contains any wildcards.

Examples:

```
Pattern             Matches
-------------------------------------------------------------------
File-31.jpg         File-31.jpg, File-31.jpg.webp, MyFile-31.jpg
"File-31.jpg"       File-31.jpg
dir:"This Folder"   This Folder
dir:This Folder     This Folder, This Folder 2, Whatthis Folder
-------------------------------------------------------------------
```

Boolean Logic

OR: In each line of the Color Filter List you can state an unlimited number of filters separated by ; (semi-colon). These will be logically combined with OR: The colors are applied if any of the filters in the line matches.

You can also OR-connect Color Filters by OR or | (both surrounded by spaces; OR is not case-sensitive, so or works as well).

AND: Filters in each line can also be connected by logical AND. The operator is AND (surrounded by spaces!); it's not case-sensitive, so and works as well.

You can also AND-connect Color Filters by AND or & (both surrounded by spaces; AND is not case-sensitive, so and works as well).

NOT: The prefixed unary NOT operator "!" is supported. The operator can also be used for individual operands in a Boolean expression. Note that this works only for non-first operands, since the NOT operator in front of the first operand is interpreted as NOT operator for the whole expression (see examples below). Also the verbal NOT-operator "NOT " (or "not ", or "Not ", it's case-insensitive) is supported.

Notes:

- The semi-colon (which works as Boolean OR) has a lower precedence than AND.
- Color Filters which contain the strings " AND " or ";" can/should be quoted.

Examples:

- TXT files modified in any of the last 5 days at the 11th hour:
  `ageM: <= 5 d AND dateC: h 11 AND *.txt`
- Files named "A AND B*" and larger than 1MB:
  `"A AND B*" AND size: > 1MB`
- Files modified Mondays at 11 OR Tuesdays at 12:
  `dateM: dw 1 AND dateM: h 11; dateM: dw 2 AND dateM: h 12;`

- Items not beginning with a:

  ```
  !a*
  NOT a*
  ```

- Items not beginning with a or b (NOT operator refers to whole expression):

  ```
  !a*;b*
  NOT a*;b*
  ```

- Items beginning with a or not with b (NOT operator refers only to 2nd operand):

  ```
  a*;!b*
  a*;NOT b*
  ```

Captions

Color Filters support captions prefixed in quotes to the filter definition. Currently only Instant Color Filters actually use this caption, where it is displayed as the menu item caption in the Instant Color Filters popup menu. For example:

```
"Created or Modified Today" ageC: d;ageM: d>FFFFFF,70B926
```

Comments (Advanced Feature only for Geeks)

You can add a free form user comment to the end of each line of the Color Filter List, separated from the rest of the line by "//", e.g.:

```
ageM: d AND *.txt //text files modified today
```

Note that the general form of the comments section is (see Switches and Pre-Filters in the following paragraphs):

```
//[switches]|[filter:pre-filters]|free form user comment
```

Switches (Advanced Feature only for Geeks)

There are two switches that can be placed in the comments section of a filter right after "//" and followed by a "|" (pipe) , m (merge) and n (no more). If switches are used, the user comments have to be placed after the "|" (see example below).

m-switch: merge

This switch can be used to merge the back color of the first matching filter with the text color of the next matching filter.

In the following example (copied from List Management | Color Filters to show the color codes), any TXT file that was modified today will now be displayed with colors 38A050,6798E0. What the m-switch actually does is: Instead of taking this color filter's text color, look for the next matching color filter that has NO backcolor defined and take its text color.

```
+ageM: d //m|modified today>FFFFFF,6798E0
+name:*.txt>38A050,
```

This example will set a light yellow background color to all items with SYSTEM attribute, and use the text color of the next matching color filter that has only a text color defined (note that the "|" after "m"

is obligatory!):

```
+attr:s//m|>,FFFF80
```

n-switch: no more

In Details mode, the first matching filter *with* backcolor (if any) is displayed combined with the first matching filter *without* backcolor (if any). The filter *with* backcolor must be the first match. The name column will show Text/Back color of the first match, the other columns will show the Text color of the second match.

This behavior can be suppressed using the n-switch. For example:

```
*.bak //n|Only color the name column
```

Pre-Filters (Advanced Feature only for Geeks)

There is a way to pre-filter the files that are checked for matching a particular Color Filter. This is very welcome for slow Color Filters such as the "prop:" filter because it can considerably speed up the processing.

All pre-filters are case-insensitive.

Filter

The pre-filter is located in the optional comments section of a filter, i.e. after "//". As the first part of the comments section is reserved for certain switches ("m" and "n", see above), the pre-filter is to be placed in the second part, after the first "|"; it has to be prefixed by "filter:". Any number of filters can be stated, separated by ";" which stands for logical OR. Pre-filters can include whole paths, actually anything that will work in a simple pattern matching against the full path/file name of the item in question. So you can limit the job to certain folders or branches if you like.

Examples:

These pre-filters limit the items that are checked for the CameraModel property to CRW and JPG files, or to the folders E:\photos and F:\photos:

```
prop:CameraModel:*Canon* //|filter:*.crw;*.jpg
prop:CameraModel:*Canon* //|filter:E:\photos\;F:\photos\
```

Filternot

There is also a logical inversion of the above. The "filternot:" prefix specifies a list of patterns that must not be matched.

Note that "filternot:" has to be in the 3rd section (separated by |) of the comment part.

Examples:

Match all files created today:

```
ageC f: d
```

Match all files created today, located in branch E:\Test\:

```
ageC f: d //|filter:E:\Test\*
```

Match all files created today, located in branch E:\Test, but not JPGs or PNGs:

```
ageC f: d //|filter:E:\Test\*|filternot:*jpg;*.png
```

Match all files created today, but not JPGs or PNGs:

```
ageC f: d //||filternot:*jpg;*.png
```

Content-based Color Filters like these make listing huge folders quite slow. The "filternot:" prefix allows you to exclude those folder:

```
B:prop:#empty:2|fl //||filternot:T:\Test ManyFiles\*
L:prop:#nosubs:2 //||filternot:T:\Test ManyFiles\*
```

The Types of Color Filters

Color-coding based on file or folder names (name:)

For example to color all TXT files blue, add the pattern name:*.txt to the list and then choose a nice blue. Or give an extraterrestrial ultra-violet to all files containing a space in their name by adding the pattern name:* *.

Note that the selector name: is optional (in other words, this is the default color filter type). Simply *.txt will do as well.

- Color Filter pattern matching is not case-sensitive: a=A.
- Only * and ? count as wildcards. There is no extended pattern matching for Color Filters, so #, [, and] are treated as literal characters.

Loose matches: Name and Dir patterns support Loose Match, i.e. wildcards (*) are internally auto-added left and right of a pattern if no wildcards (* or ?) are contained in the pattern. So "cat" will match "wildcat.txt".

Advanced filename patterns

You can as well color certain List items in specific directories by stating a pattern that will be matched against the full path of each item.

Rules:

If the pattern contains the character "\" then

it's treated against the full path (relative/portable syntax supported)

Else

it's treated against the file name (without path)

Examples:

```
?:\*.png            -> matches only those PNG files that are located on XY's home drive.
C:\Windows\*.dll    -> matches only those DLL files that are located in C:\Windows\.
```

```
C:\Windows\*        -> matches all items that are located in C:\Windows\.
```

You can define patterns with flanking spaces. Simply quote the whole pattern. For example, this will match all filenames ending with a space (invalid under Windows but possible anyway):

```
name:"* "
```

Note that quoted patterns have to start with "name:" or "dir:" to avoid a parsing ambiguity with pattern captions (which are also quoted). Alternatively you can supply a caption:

```
"Invalid Names" "* ";*.;dir:"* ";dir:*.
```

To match files without extension end the pattern with a dot:

```
*.
```

Color-coding based on folder names (dir:)

Works identical to the name: filter above, but is confined to folders. If "Apply color filters to the Tree" is ticked the coloring is also applied to the Tree.

For example to color all folders named "*_work" brown, add the pattern `dir:*_work` to the list and then choose a nice brown.

Tip: You can color code the folder of the inactive pane in the tree with this pattern: `T:dir:<otherpath>`

Color-coding based on file attributes (attr:)

You add an attribute pattern by simply adding the name of the attribute, preceded by `attr:`, to the color filters list (tip: stating the first letter is enough; case is ignored).

These are the available file attributes:

```
attr:readonly   (or attr:r) -> matches all items with READONLY set
attr:hidden     (or attr:h) -> matches all items with HIDDEN set
attr:system     (or attr:s) -> matches all items with SYSTEM set
attr:directory  (or attr:d) -> matches all items with DIRECTORY set
attr:archive    (or attr:a) -> matches all items with ARCHIVE set
attr:normal     (or attr:n) -> matches all items with NORMAL set
attr:temporary  (or attr:t) -> matches all items with TEMPORARY set
attr:junction   (or attr:l) -> matches all items with REPARSE POINT set (sic: l)
attr:compressed (or attr:c) -> matches all items with COMPRESSED set
attr:offline    (or attr:o) -> matches all items with OFFLINE set
attr:encrypted  (or attr:e) -> matches all items with ENCRYPTED set
attr:indexednot (or attr:i) -> matches all items with NOT_CONTENT_INDEXED set
attr:pinned     (or attr:p) -> matches all items with PINNED set
attr:unpinned   (or attr:u) -> matches all items with UNPINNED set
attr:j                      -> matches all items with RECALL_ON_DATA_ACCESS set
```

Attribute patterns can be freely OR-combined with each other and with name patterns into one filter

definition, e.g.:

```
attr:c; attr:r   -> matches all items that are compressed or readonly
attr:d; *.txt    -> matches all directories and all TXT files
```

Color-coding based on a list of file attributes (attrlist:)

Use `attrlist:` as selector for a combination of file attributes. To match the pattern an item has to have all of the listed attributes (it can have additional attributes but none of the listed ones must be missing). Each attribute is represented by its first letter (the sequence of the letters does not matter).

Examples:

```
attrlist: hs  = color all items HIDDEN AND SYSTEM
attrlist: hsr  = color all items HIDDEN AND SYSTEM AND READONLY
```

Color-coding based on file size (size:)

Using size-based color filters you can, for example, color all empty files in a certain way, or all files bigger than a certain size.

You add an size pattern according to the following syntax:

```
size: n        = exact size
size: == n     = exact (identical to the above)
size: < n      = smaller than n
size: > n      = bigger than n
size: <= n     = n or smaller
size: >= n     = n or bigger
size: min - max    = from min to max (both inclusive)
size: min -< max   = from min to max (max exclusive)
size: min >- max   = from min to max (min exclusive)
size: min >< max   = from min to max (both exclusive)
size: min-         = min or bigger  (same as >=n)
size: -max         = max or smaller  (same as <=n)
```

Here are some examples:

```
size: 0            = size 0 bytes (empty file)
size: 1000         = size 1000 bytes
size: < 10         = smaller than 10 bytes
size: > 1 MB       = bigger than 1 MB
size: >= 1.5 GB    = 1.5 GB or bigger
size: 1KB - 20KB   = from 1 KB to 20 KB (both incl.)
```

As you see, size-based filters support human-friendly size formats, recognized units are KB, MB, GB, TB, PB, and B (B is optional for bytes). Fractional values are okay, case does not matter, and you can throw in spaces as you like.

Color-coding based on file date (date:) - Part 1: Absolute Dates

All three file dates (Created, Modified, Accessed) are supported. So you can, for example, color all files that were modified on a certain day, or that were created before the year 2008.

General syntax:

```
dateC: = Date Created
dateM: = Date Modified
dateA: = Date Accessed


dateM: [date]        = exact date (down to the second)
dateM: == [date]     = exact date (down to the second)
dateM: < [date]      = earlier than
dateM: > [date]      = later than
dateM: <= [date]     = same or earlier
dateM: >= [date]     = same or later
dateM: [date] - [date] = from earliest to latest (both inclusive)
dateM: [date] -< [date] = from earliest to latest (latest exclusive)
dateM: [date] >- [date] = from earliest to latest (earliest exclusive)
dateM: [date] >< [date] = from earliest to latest (both exclusive)
dateM: [date] -      = same or later  (same as >= )
dateM: - [date]      = same or earlier  (same as <= )
```

Examples with time part:

```
dateM: == 2010.11.16 10:30:12    (all items modified then)
dateA: >= 11.11.2010 11:11:11    (all items accessed in this Carnival and later)
```

Examples without time part:

```
dateC: < 2008.01.01              (all items created before 2008)
dateM: 2010.01.01 -              (all items modified 2010 and later)
dateM: 2010.11.14 - 2010.11.15   (all items modified 2010.11.14 or 2010.11.15)
dateC: 05/22/2011                (all items created 05/22/2011)
Modified: 2025                   (all items modified in 2025; using the English
column name as selector; needs ISO 8601 date format)
Created: 2025-07                 (all items created in July 2025; using the English
column name as selector; needs ISO 8601 date format)
```

Notes:

- The date has to be given in a form that is recognized as a valid date format on your locale.
- The time part can be skipped. In this case, when operators are used in the term, the time it is internally set to 00:00:00 or 23:59:59, depending on the semantics of the term. When no operators

are used (you just give a date without the time part) the term is interpreted as a time range covering that whole day (see examples above).

- The selector is not case-sensitive: DaTem: would work as well. date: = defaults to "Date Modified".
- A trailing ";" is optional. Like with all color filters, ";" can also be used to concatenate patterns assigned to the same colors.
- These filters work for files and folders but only in the List (not in the Tree).

Color-coding based on file date (date:) - Part 2: Subranges

You also can color items if their file times match certain subranges, e.g. created from 8:00-10:00 in the morning, last modified on a weekend, last accessed in an April (of any year). To achieve this a unit selector is prefixed to a number or range (using the usual set of operators).

Unit selectors:

```
y = year (1601 - 32767)
q = quarter of year (1 - 4)
m = month (1 - 12)
dw = day of week (Monday = 1 ... Sunday = 7)
dy = day of year (1st January = 1)
d = day (of month)
h = hour (0 - 23)
n = minute (0 - 59)
s = second (0 - 59)
ms = millisecond (0 - 999)
```

Examples:

```
dateC: h 8-10      = created from 8:00-10:00 (of any day)
dateM: m 4         = last modified in an April (of any year)
dateM: dw 6-7      = last modified on a weekend
dateM: dw 7-2      = last modified from sunday to tuesday
dateC: dy 1-100    = created in the first 100 days (of any year)
dateC: q 3         = created in the 3rd quarter (of any year)
```

Notes:

- The Unit selectors can (but don't have to) be separated from the unit number or range by at least one space.

Color-coding based on file age (age:)

All three file dates (Created, Modified, Accessed) are supported. You can, for example, color all files that were modified today or last week, or that were created more than 10 years ago. This filter also supports folders in the folder Tree.

General syntax:

```
ageC: = Date Created
ageM: = Date Modified
ageA: = Date Accessed
```

The usual operators apply; [n] is a number and [u] a unit selector. There MUST be a space between counter and unit! For example:

```
ageM: [n] [u]        = specific unit
ageM: == [n] [u]     = specific unit
ageM: < [n] [u]      = less than so many units
ageM: > [n] [u]      = more than so many units
ageM: <= [n] [u]     = so many units or less
ageM: >= [n] [u]     = so many units or more
ageM: [min] - [max] [u] = from min to max units (both included)
```

Unit selectors:

```
y = year
q = quarter of year
m = month
w = week (Monday - Sunday)
d = day [default unit!]
h = hour
n = minute
s = second
```

Examples:

```
ageM: < 5 n    = modified less than 5 minutes ago
ageM: >= 3 h   = modified 3 or more hours ago
ageM: 1 w      = modified last week (previous Monday - Sunday)
ageM: 0 d      = modified today (from 00:00:00 till 23:59:59)
age:           = modified today (same as above, all defaults used)
ageM: 1 d      = modified yesterday
ageM: < 0 d    = modified in the future (tomorrow or later)
ageM: -1 d     = modified tomorrow
```

Notes:

- The Unit selectors can (but don't have to) be separated from the unit number or range by at least one space.
- The Age selector is not case-sensitive: AgEm: would work as well.
- age: defaults to "Date Modified".

Color-coding based on name length (len: and lenT:)

Name length here refers filename length (with full path) (len:) and file title length (name without path) (lenT:). The syntax is identical to the one of the "size:" selector, so you can use all sorts of numerical comparison operators to define limits and ranges. If "Apply color filters to the Tree" is ticked the coloring is also applied to the Tree.

Examples:

```
len: > 260    = color all files with a path/name longer than 260 characters
lent: <= 12   = color all files with a name with 12 or fewer characters
```

Color-coding based on properties (prop:)

Properties here refers to shell properties, that's the properties you see in the File Info Tips when hovering a file. The Properties filter is an extremely powerful color filter with a vast number of options. Win7 and later offer over 250 shell properties (of course, not all are used by each file type).

There is a drawback though: Property filters are quite slow! This is because retrieving shell properties has to rely on a slow shell mechanism. Each active "prop:" color filter will notably slow down listing and scrolling a list and selecting items in the list, and the more items are displayed simultaneously in the visible part of the list the slower it will be. Nevertheless, the "prop:" color filter can be of significant value. There is also a way to reduce the loss of speed (see Pre-Filters above).

Property filters are special in that they need an additional property selector to specify the sort of property. This can be a verbal identifier (e.g. `prop:dimensions:`) or a numeric identifier (e.g. `prop:#31:`).

Examples:

```
prop:dimensions:1500 x *    = color all files 1500 pixels wide
prop:#26:1500 x *           = (same as above on WinXP)
```

Note that #26 is identical to "dimensions" on WinXP but not on Win7 where "dimensions" is #31. You can find the correct indices in the list under Configuration | File Info Tips & Hover Box | Show these fields.

More examples:

```
prop:fileversion:8.60       = color all files versioned 8.60
prop:#26:1024 x 768         = color all files sized 1024 x 768
prop:CameraModel:*Canon*    = color all files snapped with a Canon
```

Also XYplorer-specific named arguments (see Special Properties) are supported, e.g.:

```
prop:#tags:dogs             = color all files tagged "dogs" exactly
prop:#tags:*dogs*           = color all files tagged "dogs" (and other tags)
prop:#ShortcutTarget:*.jpg  = color all files linking to JPGs
prop:#JunctionTarget:*blah* = color all junctions to "*blah*"
prop:#HardLinks: > 1        = color all files with more than one hard link
prop:#AspectRatio: 16:9     = color all image files with a 16:9 aspect ratio
```

Also the #Hash property is supported, but it is not recommended because it would extremely slow down browsing. For example (don't do this at home), this color filter will color all files with a certain

MD5 value:

```
prop:#Hash.MD5:58172fd15526e1249ac8bfd690ced076
```

Note that "prop:" filters have their own distinctive background color shapes (in case you enabled Configuration | Color Filters | Draw background colors in distinctive shapes).

Numeric Properties

The "prop:" filter also supports numeric comparisons. For example to color all JPG and PNG images with a width of at least 1024 pixels use any of these (two ways to express the same):

```
prop:#162:1024 - //|filter:*.jpg;*.png
prop:#162:>=1024 //|filter:*.jpg;*.png
```

Note that #162 is the numeric property identifier for "Width" on one Win7 system. It might be different on your system. You can find the correct indices in the list under Configuration | File Info Tips & Hover Box | Show these fields.

Date Properties

The "prop:" filters also support dates. For example, to color all files with a specific shooting time (valid date/time syntax varies with your locale):

```
prop:#25:02.08.2011 13:52 //|filter:*.jpg
```

You can omit the time part to only search for JPG images shot on a certain day:

```
prop:#25:02.08.2011 //|filter:*.jpg
```

You can as well specify ranges using the complete operator syntax that's already supported by the Date Color Filters ("dateM:" etc.). For example, to color all JPG images shot between 01.01.2011 and 31.12.2011 (both dates inclusive):

```
prop:#25:01.01.2011 - 31.12.2011 //|filter:*.jpg
```

Note that the numeric property identifier "#25" is preferred to the verbal "WhenTaken" because the latter returns the time in UTC (Coordinated Universal Time) instead of local time -- a mysterious shell anomaly. Note further that #25 is only valid for Windows XP, whereas Win7 needs #11 here.

Using Property-based color-coding in the Tree

To apply a Color Filter of type "prop" to the Tree you have to prefix the control selector "T:" or "B:" to the pattern. With other Color Filters the control selector "B:" is not necessary since the absence of any control selector does the same: Both Tree and List. With Color Filters of type "prop", however, the absence of any control selector defaults to "List only".

Now, most Properties don't have so much use in the Tree, but here is an interesting one:

```
T:prop:#empty:2                    //matches all empty folders in tree
```

This lets you color-code empty folders in the tree which is quite a cool usability gain.

- Note that this option has to be ticked, of course, to color-code the tree:
  Configuration | Colors and Styles | Color Filters | Apply color filters to the Tree

- It's limited to normal folders at the moment. No drives, or other non-straightforward folders.
- Also no UNC paths (unavailable network paths could slow this down unbearably).
- If Auto-Refresh is enabled the folder coloring state should auto-update when the empty-state changes.

More examples:

```
prop:#empty:2                    //List only
T:prop:#empty:2                  //Tree only
L:prop:#empty:2                  //List only
B:prop:#empty:2                  //Both Tree and List
L:prop:#contains.*.jpg:0         //matches all folders in List NOT containing any JPG files
B:prop:#contains.desktop.ini:1   //matches all folders in both Tree and List that contain a file "desktop.ini"
B:prop:#contains.desktop.ini:1 AND B:prop:#itemcount:1 //folders containing only Desktop.ini
```

Color-coding based on tags (lbl:, tags:, cmt:)

You can as well color-code items by their tags. The syntax is identical to that of Visual Filters by Tags.

Partial matches are supported in the Comment field without the need to add wildcards. To force an exact match enclose the pattern in quotes. For example:

```
Comment:Linda          -> match items where the comment contains Linda
Comment:"Linda"        -> match items where the comment is Linda
```

Dirty Tags: You can define a filter that matches items where any of the tags (Label, Tags, Comment, Extra) are dirty (i.e. not yet saved to disk). The pseudo pattern used for this is `"*"`:

```
tags:"*" //dirty tags
```

Instant Color Filters

XYplorer comes with a predefined set of so-called Instant Color Filters that can be applied individually and independently of the normal Color Filters.

Usage

Instant Color Filters are applied via the toolbar button "Toggle Instant Color Filter". The button has an arrow-dropdown where the filters can be individually applied. The button itself toggles the last applied filter.

- If you activate a new filter, the old filter is replaced by it. So you cannot have two at the same time.
- If you activate the same filter again it is deactivated (toggle).
- The filters activated by this command are completely independent of the Color Filters (which are defined and enabled via Configuration | Color Filters). They even work if the normal Color Filters are disabled.

Tip: The button has a right-click menu with some useful options.

Advantages

Instant Color Filters have a couple of advantages over the normal color filters:

- You can quickly and individually turn them on/off without going through Configuration.
- This means you can spotlight the files you are interested in now, just in time. With normal Color Filters you are often bothered with colors all over that are distracting rather than helpful.
- They are always processed on top of all other color filters, so you always see all matches. Whereas in normal color filters it's common that the first match covers any possible other matches because usually only one color can be shown at the time for each item (exception: Details view allows two colors simultaneously per item under certain conditions).
- Because lists of filters are supported you can use this function to toggle whole color schemes which would be very cumbersome to do using the normal color filters.
- It's also easy to share color schemes between users via user button snippets or included catalogs.
- Just like normal Color Filters, Instant color filters are retained across sessions.

Tip: You can assign a keyboard shortcut to toggle the last applied filter via *Customize Keyboard Shortcuts* | *Miscellaneous* | *Various* | *Toggle Instant Color Filter*.

Customization

The predefined filters can be freely customized via List Management | Instant Color Filters. Of course, the customizations are retained between sessions.

The color filter syntax is identical to that of normal Color Filters. The color filter definition uses the full Color Filter syntax including colors in RRGGBB and leading "+" (the "+" is optional here, the filters will also be applied without the "+"). You can see this syntax in List Management | Color Filters, in Editor Mode (F6).

Multi-Filter Definitions: There is one thing where ICFs go beyond normal CFs. You can append several filter definitions in one line separated by "||". Example:

```
"Files Shaded By Size" size: >= 1 GB>000000,8A939F||size: >= 100 MB>003080,8EA4C4||size: >=
10 MB>004080,A9BAD3||size: >= 1 MB>0053A6,BCCCDE||size: >= 1 KB>0E80DC,D1DAE9||size: >
0>4F91D2,E2E3EB||size: 0>96ABB8,ECEDF2
```

Note that the coloring displayed in List Management only takes the left-most filter into account if there are many filters in a line.

Factory Defaults

For reference, the following Instant Color Filters are supplied by factory default.

```
"Created or Modified Today" ageC: d;ageM: d>FFFFFF,70B926
```

```
"Created or Modified This Week" ageC: w;ageM: w>FFFFFF,B97026
```

```
"Folders Created Recently" ageC d: <= 3 h //folders created last 3 hours>FFFF00,0073E6||ageC
d: d //folders created today>FFFF80,379BFF||ageC d: w //folders created this
week>FFFFFF,71B8FF
```

```
"Files Modified Recently" ageM f: <= 5 n //files modified last 5 min>FFFF00,4F831B||ageM f:
<= 1 h //files modified last hour>FFFF00,63A521||ageM f: <= 24 h //files modified last 24
hours>FFFFFF,70B926
```

```
-
"Empty Files" size: 0>96ABB8,ECEDF2
"Files Shaded By Size" size: >= 1 GB>000000,8A939F||size: >= 100 MB>003080,8EA4C4||size: >=
10 MB>004080,A9BAD3||size: >= 1 MB>0053A6,BCCCDE||size: >= 1 KB>0E80DC,D1DAE9||size: >
0>4F91D2,E2E3EB||size: 0>96ABB8,ECEDF2
-
"Overlong Filenames" len: > 260>FFFF80,FB4F04
-
"Read Only Files" attr f:readonly>47A905,FFFF80
"System Files" attr f:system>FF0080,FFFF00
-
"Common Executables" *.exe;*.bat;*.cmd;*.com;*.scr>804000,FFFFAA
"Common Image Files" *.gif;*.jpg;*.png;*.tif>36530F,E6F786
-
"Image Aspect Ratio 16:9" prop:#AspectRatio: 16:9>5A4F36,F7E686
-
"Black Out" name: *>222222,222222
```

Tip: You can easily reset your filter collection to factory defaults by deleting all filters in List Management | Instant Color Filters.

Scripting support

The scripting function colorfilter() can also be used to apply Instant Color Filters. For details see colorfilter().

4.6 Tags

Tags: Labels, Tags, Comments, and Extra Tags

XYplorer lets you assign labels, tags, and comments to individual files and folders. This can be done through various interfaces: The Tags Tab on the Info Panel, various Toolbar buttons, the Tags menu, the context menu of the tags-related columns in the Details view, and scripting.

Labels

Labels let you put files into a closed set of freely definable categories. You can have your "blue" files, your "ready" files, your "important" files, your "junk" files, etc.

Labels are visual helpers in that they give colors to file names in the file list. You can sort files by Labels. And you can search files by Labels.

The labeling is done very comfortably via keyboard shortcut, toolbar button, or context menu. Labels are retained across sessions.

There is a choice of seven predefined color schemes (background and text color), which can easily be exchanged for other colors, and extended to up to 31 color schemes. The names of the schemes can be customized in Configuration | Tags. Here you can also choose how and where the colors are displayed in the file list.

The Toolbar offers the handy buttons Labels and Find by Label. You can configure the latter button to Search Everywhere (whole computer) or Search This Branch (current location including subfolders) or Search Here (current location excluding subfolders). The options are found at the bottom of the button's dropdown menu. Of course, the buttons also have a right-click menu with various useful commands.

Tags

You can define any number of Tags for any file or folder, and show them in the column "Tags" in Details view. They can also be shown in the File Info Tips, and you can find files by them (with all Boolean splendor, and extremely fast because these searches are indexed).

- Tags are not case-sensitive (A==a).
- Tags can contain any characters apart from:
  `,` (comma) which is used as separator between the tags.
  `|` (pipe) which is used as separator between the fields.
  `"` (double quote).
  ` ` (space) at the left and right end of a tag. Spaces within a tag are okay.
- In the database and in the Tags column, tags are separated by comma and space (`, `) for better readability. However, the space is not necessary for parsing and can optionally be omitted wherever tags are fed into the app.

- Commands to add or remove tags are found in the Tags menu.
- A "Tag List" (similar to a tag cloud, but not weighted) is automatically maintained in the background and across sessions that lets you select tags rather than type them again and again. In the Add Tags and Edit Tags dialogs there is a checkbox "Add new tags to tag list" by which new tags are automatically added to the tag list (always to the top of the list).
  You can also manually manage the Tag List in List Management. This way you can easily load a new cloud to have a different set of tags ready for selection. Note that editing the Tag List does not change or remove any tags from your files or folders! The Tag List is only your interface to selecting/attaching tags.
- The Toolbar offers four handy buttons to Add Tags, Set Tags, Remove Tags, and Find by Tags. They pop a menu with the top 20 tags from the tag list. All buttons work on the selected items only and show a "Nothing Selected" pseudo item in the popup menu if there are no selections.
  Especially interesting is the Find by Tags button because the "Tag List..." item allows for easy multi-tag searches. Tick Match All if all ticked tags must match (Boolean AND). You can configure the button to Search Everywhere (whole computer) or Search This Branch (current location including subfolders) or Search Here (current location excluding subfolders). The options are found at the bottom of the button's dropdown menu.
  Of course, the buttons also have a right-click menu with various useful commands.
- The Toolbar buttons also have a right-click menu with various useful commands:
  Edit Tag List...: Manually edit the tag list.
  Update Tag List: Update the tag list to the tags currently used in tag.dat.
- The Catalog offers Click and Tag via its right-click menu: Automatically add all Tags (or Labels) used in the tags database to the Catalog and enjoy one-click tagging -- the most convenient way to tag files.
- The Catalog also offers Click and Search for Tags and Labels via its right-click menu. See Click and Search.
- See Usage here below for more ways to handle Tags.

Comments

You may as well attach individual comments of virtually any size to your files and folders.

Unlike labels, comments are not taken from a closed set. You can have thousands of different comments, and they can be as long as you like.

Usage

In the file list, in Details View, tags (Labels, Tags, Comments) are displayed in individual columns (show them using menu View | Columns), which can be sorted and thus used to group items according to their tags.

Note that you can add or edit tags directly in the file list by clicking into the respective column: A Left-click in the Label column will set/unset (toggle) the current Label. A Right-click in the Label, Tag, or Comment column will pop a context menu with several options. Note that the latter needs Configuration | Tags | Popup by tag columns right-click enabled, and Full Row Select (in menu Tools | Customize List) should be disabled for the list.

File Operations

During file operations within XYplorer the tags are preserved in the most natural ways possible. Note that any file operations occurring outside of XYplorer will obviously not have the tags preserved (even for items in the current location with auto-refresh enabled).

Moving Items

Moving items, the tags will move along with the items. When moving an item and replacing an already existing one, any existing tags from source items will be set to destination items, replacing any previously existing tags on the destination items. If a source item does not have a tag, but the previously existing destination item does, they will be preserved.

Copying Items

Copying items, by factory default the tags will not be copied along with the items, in an effort to keep your database tidy and avoid multiplying tags which could eventually slow things down. Any previously existing items will keep their original tags.

However, you can enable copying tags by ticking Configuration | Tags | Copy tags on copy operations or Configuration | Tags | Copy tags on backup operations. Tip: Tick Configuration | Tags | Confirm copying tags and you are prompted before any tags are copied from the source items to the target items.

Renaming Items

Renaming items, the tags are obviously preserved.

Deleting Items

Deleting items, the tags are removed from the database.

Find Files by Tags

You can search items by tags using the Tags tab in the Find Files tab in the Info Panel.

Configuration

See Configuration | Tags.

The Database: tag.dat

All tag data are retained across sessions and stored in one file, tag.dat, located in the application data folder. The file is kept in an open, simple, and human-friendly text format (UTF-16 Text File) so that you can easily edit it manually using an editor or programmatically using any scriptable software.

File Format of the XYplorer Tag Database (version 5 -- from 19.70 onwards)

Line 1: Header and version info: `XYplorer File Tags v5`

Line 2: (empty)

Line 3: `Labels:`

Line 4: Label schemes, separated by "`;`". Each scheme: `Name|TextColor|BackColor`. Colors are in RRGGBB.

Line 5: (empty)

Line 6: `Extra Tags:`

Line 7-22: Extra Tag Definitions: `Caption|TypeID|0|0|||||` . Some fields are reserved for future use.

Line 23: (empty)

Line 24: `Storage: ...`

Line 25: (empty)

Line 26: `Data:`

Line 27: Tagged item #1: `Full filename|LabelID|Tags|Extra1|Extra2|Extra3|Extra4|Extra5|Extra6|Extra7|Extra8|Extra9|Extra10|Extra11|Extra12|Extra13|Extra14|Extra15|Extra16|Comment`

Line 28: Tagged item #2: ... etc. ...

In case you edit tag.dat manually...

Sort Order. The tagged item lines must be in binary ascending sort order (A < B ... < a < b ...). If you manually edit tag.dat and are unsure about the correctness of the sort order, you can have the data sorted by using "Options | Database Check..." in Configuration | Tags.

Capitalization matters. You must care for the right capitalization of the file names. Reason: For better performance, the comparisons are done by byte, not by character.

Portable Tags

Used Configuration | Tags | Storage to store tags in a portable manner.

See here for the details.

Extra Tags

Additionally to the above Label, Tags, and Comment, you get 16 optional Extra Tags, that can be shown in the file list as Extra Columns. Via clicking into these extra columns (or via scripting) you can assign user data of various formats (free text, checkboxes, rating stars, date stamps, linked locations, and more; see Extra Column Types below) to any file and folder. By these data you can then sort the file list and find files across your system at lightning fast speed.

Usage:

- Show the columns just like the other regular columns: via menu View, or via right-clicking the currently shown column headers (not with Size and Date columns, which show their special context menu), or the zone right of the column headers.
- Edit the column type by right-clicking the header of the column you want to edit (see below).
- Edit the content of the columns by right-clicking the cell you want to edit.
- Checkbox columns can be directly toggled by left-click.
- There is scripting support: See the scripting commands ExtraTag and Tag.

Notes:

- The extra tags support all characters (even line feeds) with one exception: the pipe (|).

Behavior on clicking a cell in an extra column

Left click: Only apply to the clicked file.

Right click: If other files are selected than the clicked file then only apply to the clicked file, else apply to all selected files.

Extra Column Types

Each extra column can be set to a certain type. Do this via Configure Extra Column [#]..., available from the column headers right-click menu. Here you can set the caption and the type of the column. The extra column type concerns sorting, ways to input data, and the display of the data.

Note that changing a column type will not affect any data in that column or in tag.dat. It will just affect how the data is displayed and what happens if you click a cell of that column.

The following types are supported:

Text

Free text, multiline.

Number

Integer value, negative allowed. Sorted numerically.

Date

Displayed in the current Date column format. Stored in tag.dat in format `yyyy-mm-dd hh:nn:ss.fffffff`, local time (not converted to UTC; reasons: performance and readability). Kept in memory also in format `yyyy-mm-dd hh:nn:ss.fffffff` (nicely sortable). Predefined options to set Current Date, and Created Date, Modified Date, Accessed Date of the right-clicked item.

Checkbox

Two states, internally "1" and "" (empty), displayed as a checkbox. Left-click toggles the state.

Rating Stars

Ratings (one to five) are printed as stars (Unicode character U+2605). Predefined options in right-click menu.

Pop-up List

Closed set of string values (defined in tag.dat section Extra Tags). Selectable via right-click, menu options. The set of string values is editable via Configure Extra Column [#]..., available from the column headers right-click menu.

Location

Location (folder or file, or anything that can be fed into the Address Bar; one per line). You can add more than one location. On left-click XYplorer will go to the first location. On right-click all locations are

offered as menu options. Editable via right-click, multi-line edit box.

Optional captions (prefixed in quotes to the location) are supported:

```
"Test" E:\Test\
"A" E:\Test\A.txt
C:\
"Say Hi" echo "hi";
```

Location columns also support scripts as locations (just like the Address Bar). In a location column script you can refer to the clicked cell (item and column) by these temporary variables:

```
<taggeditem>    resolves to the item (full path) that is being tagged via clicking a column cell
<taggedcolumn>  resolves to the column caption that is being tagged via clicking a column cell
```

Example: `::echo <taggeditem>;`

Icon

This type lets you tag particular files with particular icons (shown in the Extra Column in the file list). A visually strong way to mark files. The syntax is identical to that in Custom Columns of format "Icon".

Image

If you specify an image (all image formats that can have thumbnails are now supported) as column data (via right-clicking the cell), those images are displayed right in the cell. This opens up a world of possibilities. For example, you can attach waveform images (created by some 3rd party app) to WAV files, and have them shown right in the list. And not only one. You can have a couple of such Image Columns, each with its own image.

- The images can be resized on-the-fly just by changing the column width.
- Of course, if you want to see more than very tiny miniatures, you need some row height. Currently the best way to achieve an acceptable row height is by using the "Details with Thumbnails" view.
- The image path is resolved relative to the item path. So, if you (can) state a relative path, items and images can be moved together without need for adjustments.
- There is no caching for those images. They are always live. So you can increase performance by providing not too large source images.
- These images support the Hover Box. Tick "Image Columns" in Configuration | Information | File Info Tips & Hover Box | Show Hover Box | Select Context...

Script

The script you specify here (via the Configure Extra Column dialog which is accessible by right-clicking the column header) is run whenever you right-click a cell in that column. If that script calls the command "return" the "data" argument is used to fill the cell. Such a Scripted Extra Column provides wonderful and countless new ways to easily generate or select data to be placed in a cell and thus to be assigned to a file.

- The script should not contain the | character unless it is quoted, because that character is already used to separate extra fields in the tags database.
- There are some special variables only available in this context that can be used in the script:
  `<extra_cell>`: Resolves to the contents of the right-clicked cell.
  `<extra_item>`: Resolves to the full path of the right-clicked item.
  `<extra_popuplist>`: Resolves to the contents of the "Pop-up List" field of that same column, CRLF line breaks being replaced by ";".

Examples:

- This script fills the cell with a super-exact timestamp: `return now("yyyy-mm-dd hh:nn:ss.fffffff");`
- This script pops an inputselect interface filled with some hard-coded data: `return inputselect ("Select Country", "Afghanistan;Albania;Algeria;Andorra;Angola", ";", 32, , 400, 600);`
- This script pops an inputselect interface filled with the data present in the "Pop-up List" field of this Extra column: `return inputselect("Select Country", "<extra_popuplist>", ";", 32, , 400, 600);`
  If the data is many this interface is far superior to the menu that is popped by a "Pop-up List" type Extra column. And using the "Pop-up List" field as data store via `<extra_popuplist>` makes the script much easier to handle, and makes it shareable between columns and users.
- This script adds some bling to the previous one. It displays the right-clicked item in the header section, and preselects the current selection in the list: `return inputselect("Select Country<crlf><extra_item>", <extra_popuplist>, ";", 32, 10:=<extra_cell>);`

Find Files by Extra Tags

Of course, you can find files by Extra Tags. The field prefixes to use are either "ex1" to "ex16" (the current column names are irrelevant), or the actual (and localized to your interface language) column captions. The matching is done as simple text matching with wildcard support (as e.g. in the Name or the Comment field). Of course, Multi Field Search is fully supported.

For example, to find all Text files associated with "Byron" in the "Author" column (Extra 1) you can use either of these search terms (the leading colon specifies the term as a Boolean search).

```
:*.txt AND ex1:Byron
:*.txt AND Author:Byron
```

Comparison operators for number and dates are supported. For example, if Extra 2 is of type Number then this would work as expected:

```
ex2: >= 5
```

Dates should be stated in yyyy-mm-dd hh:nn:ss format since they are also stored in that format. For the comparison you can skip any part of the right side, e.g. (if Extra 5 is of type Date):

```
ex5: > 2014
ex5: > 2014-02
ex5: > 2014-02-09 10
```

You can also search all extra fields at once by using the "ex" selector. For example:

`ex:data` (finds all items with any extra field set to "data")

See also Search only among Tagged Items under Find Files by Tags.

Reporting Extra Columns

All reporting functions in the Report tab (Info Panel) support the extra columns. Additional options are offered by scripting command report().

Multi-User Tagging

Your team mate comments a file in his file manager and you see that comment in your file manager right away? That's not a dream anymore. With the new XYplorer you enter the wonderful world of streamlined Multi-User Tagging. And to set it up will take you about 5 minutes.

The following 4 steps are all it takes to create a Multi-User Tagging (MUT) network:

Step 1: Understand the Structure. Each team member needs a copy of XYplorer installed on his local Windows. The tags database is a single file that will be kept in a central location accessible to each member of the network. That location would typically be an UNC path pointing to a shared folder, for example "\\North\Share\XY\SharedTags.dat". Nothing to do yet, just understand the structure.

Step 2: Connect to the Database. Now create the shared database using the command Tags | Load Tags Database.... Enter the full path, e.g. "\\North\Share\XY\SharedTags.dat" (use any name you like). If the database does not exist yet using that command will create it, else it will open it. Now that the database exists each member of the network can connect to it using that same command Tags | Load Tags Database... and same full path in his XYplorer instance.

Step 3: Show the Columns. To each member: To actually see the tags in the file list do the following: Go to folder containing tagged files or files to be tagged. Select Details View in the file list. Then use command View | Columns | Show Columns... to show the columns (tags) you are interested in. Note that you can drag the columns to new positions if you please.

Step 4: Automate the Synchronization. The finishing touch is to have any changes in the shared tags automagically appear in each member's XYplorer instance in real time. To enable this feature the checkbox Tools | Configuration | Information | Tags | Auto-refresh tags has to be ticked in the member's XYplorer instance.

If all is set up right, each member of the team will see the same tags on the same files, and if any member changes any of them, the others will see the change in real-time in their instance of XYplorer.

SafeSave

A safety mechanism called "SafeSave" is implemented where when saving tags, the database was first read, matched to the tags in memory, and only then saved. Can be useful in a multi-user tagging context if for some reason your current tags in memory do not match the status of the database on disk. "SafeSave" is only executed if the key `FileTagDat` is defined accordingly in Admin Settings.

4.7 Address Bar

Address Bar

The Address Bar is the control where you can enter (type, paste) locations directly. You can show/hide it using Show Address Bar (Ctrl+Shift+F12) in menu Window.

Any locations entered via the address bar are stored in the bar's dropdown list (up to 256 items; last on top), and saved between sessions. On top of this XYplorer's Address Bar comes with a number of important usability enhancements:

(1) Keys F4, Up, Down: all open the list; F4 closes again; next Up or Down select list items; selection wraps around ends.

(2) It optionally auto-completes path names (see below: 6 Key Navigation, 4 Key Navigation, Mouse Click Navigation).

(3) It optionally auto-completes recently used items: When you start typing a location into the address bar, it shows you a list of places (sorted alphabetically) you've gone to before that match what you've typed so far. To see the complete list sorted alphabetically, empty the edit field, hold CTRL, and press the down or up arrow key.

(4) It supports Tabbing through the items.

(5) It supports select-all-by-triple-click.

(6) It is highly configurable (Configuration | Controls & More).

(7) It shows icons for edit box and list items. Icons help you to easily distinguish the various types of items that are supported by the Address Bar: Folders, Files, Drives, Servers, Quick Searches, Visual Filters, URLs, Scripts.

(8) You can use the mouse wheel to scroll through the dropdown list items when it's collapsed. You can also expand the list by Shift+WheelDown and collapse the list by Shift+WheelUp.

(9) The droplist has a minimal width of 400.

(10) The droplist is higher than Windows standard: 12 instead of 8 items.

(11) Right-clicking the dropdown arrow pops a menu featuring the breadcrumb menu for the current path, the virtual folders, and the drives.

(12) The edit box ignores mouse wheel (that's good!).

(13) The edit box supports automatic word breaking (dbl-click selection, moving cursor with CTRL). Recognized word boundaries: :\?|

(14) Enhanced usability through a number useful key combinations and mouse tricks (see below).

(15) Relative paths are resolved relative to the current path.

You can use `..` to go upwards.
You can use `C:\Windows\..\` to go to `C:\`.

(16) Shift+Enter will open the location in a new tab.

(17) When you attempt to go to a non-existing location you are prompted to create it on the fly. Will also create whole new paths (as long as the drive exists and is writable). Note that the location must end with a backslash to enable this feature.

(18) Partial Matching: See below.

(19) When the Address Bar shows an existing file (not a folder), then the icon pops a right-click menu with some useful commands: Go To, Open, Copy Item, Copy Path, Shell Context Menu. The latter will pop the Shell Context Menu as if the file was right-clicked in the file list.

(20) When the Address Bar shows an existing folder, then the icon pops a right-click menu with a breadcrumb, and some useful commands (see also below Address Bar Icon Context Menu): Copy Item, Copy Path, Shell Context Menu. The latter will pop the Shell Context Menu as if the file was right-clicked in the folder tree.

(21) The right-click menu of the icon now offers the additional command "Copy Real Path" if the displayed path is a special path, e.g. `Desktop\Test`.

(22) You can switch the dropdown button position on the fly. Ctrl+Right-Click the dropdown button and toggle "Dropdown Button on the Left" in the popup menu.

(23) Double-quoted and single-quoted paths work. They are handled as if they were unquoted, e.g. `"C:\Windows\debug"` or `'C:\Windows\debug'`.

(24) Last not least, it fully supports Unicode.

For power users here's a number if useful Key Combinations and Mouse Tricks:

(1) If dropdown list is not visible:

Enter:	Shoot, i.e. trigger the Address Bar's current contents (e.g. go to that location; run that search; apply that filter; run that script...).
Shift+Enter:	Open the location in a new tab
Ctrl+Enter:	A file in Address Bar is opened right away instead of browsing to its location.
Esc:	Reset edit box to last shot item. If there is nothing to reset: Toggle selection state (all versus nothing) of the edit box.
Ctrl+A:	Select All.
F2:	If all is selected: unselect all, move caret to end of text. Else: select all.
F4:	Open MRU list.
Down:	Open MRU list.

Ctrl+Down:	Open the auto-complete match list. Any selection in the edit field is removed and the caret is placed at the right end of the contents. If the edit box is empty the whole MRU list is shown as Match List, i.e. sorted alphabetically.
Right-click the icon:	Auto-Complete Path Names is OFF: Pops the shell context menu for the item (folder or file) currently displayed in the Address Bar. This will always be the standard shell context menu, without any XYplorer-only custom items. Auto-Complete Path Names is ON: Pops a breadcrumb menu for the item (folder or file) currently displayed in the Address Bar.

(2) If dropdown list is visible:

Down/Up:	Select next/previous item in list.
Tab/Shift+Tab:	Select next/previous item in list.
Page Down/Up:	Select next/previous item in list page-wise.
Right:	When the caret in the Edit field is at the right-most position, then pressing the "Right" arrow key will open the match list. If there is no trailing backslash then it is auto-appended.
Enter:	Undrop the list, and shoot.
Ctrl+Enter:	Just undrop the list.
F4:	Just undrop the list.
Ctrl+Down/Up:	Just undrop the list.
Click on list:	Set the edit box to clicked item, undrop the list, and shoot.
Ctrl+Click on list:	Set the edit box to clicked item, and undrop the list.
Click in edit box:	Just undrop the list.
Esc:	Undrop the list, reset edit box to last shot item.

Note that Address Bar and the "Go to" dialog share the same item history.

Tip: The special alias "*" (asterisk) can be used for "Computer" as location.

Address Bar Icon Context Menu

Show Real Path

Tick it to always show the real path with drive letter in the Address Bar and in the Find Files Location box, even if the tree is located in a "virtual" (or "special") path like "Desktop".

Show Trailing Slash

Tick it to show the trailing backslash for paths in the Address Bar box and in the Find Files Location box.

Open Files from Address Bar

Tick it to have "Go Now" open a file right away instead of browsing to its location. The "Go Now" event is triggered by:

- Pressing ENTER in the Address Bar.
- Selecting menu Go | Go Now.
- Selecting an item from the Address Bar dropdown.
  Tip: Hold CTRL to suppress "Go Now" when selecting an item from the Address Bar dropdown.
- Clicking the "Go Now" toolbar button.

If unticked then "Go Now" will browse to the location of the file and select it. Applies to files only, not to folders. Files are opened as if they were double-clicked, i.e. Custom File Associations are honored.

If ticked and the Address Bar shows a file, the icon in the Address Bar shows a little overlay to signal that ENTER will open that file.

This setting can be temporarily inverted by holding CTRL: On Ctrl+Enter in the Address Bar, a file is handled as if the setting of "Open files from Address Bar" would be inverted: It is opened if "Open files from Address Bar" is OFF, and it is gone to if the setting is ON.

Selection by Path Components

Tick it to recognize only path component separators (\/?|) as word boundaries when double-clicking the Address Bar contents, when moving the cursor by Ctrl+Arrows, or when deleting words by Ctrl+Backspace.

6 Key Navigation

6 Key Navigation (6KN) means you can browse the whole computer by just 6 keys (Ctrl, Left, Right, Up, Down, Enter) in the Address Bar.

Note: Configuration | Controls & More | Auto-Complete Path Names | Address Bar has to be ticked for this, and Filter should be set to "Files and Folders" or "Folders only".

```
6 Key Navigation in the Address Bar


Always:
  Left:       Move caret to left.
  Right:      On right end (if folder): Opens match list (appends backslash if there is
none yet).
              Else: Move caret to right.
  Ctrl+Left:  Move caret to parent folder component.
  Ctrl+Right: Move caret to child folder component.
  Ctrl+Down:  Open/Update match list at caret position.
```

```
  Enter:      Go to item or open file (if "Open files from Address Bar" is ticked).
              Close match list if it is open.
  Ctrl+Enter: Like Enter, but Go to and Open inverted.

On the dropped match list:
  Up:         Move up in match list (to last item if none selected).
  Down:       Move down in match list (to first item if none selected).
  Ctrl+Up:    Close match list (no action).
```

The crucial part of 6KN is that you can open a match list at the caret position. Move the caret at to some position in the edit field of the Address Bar. Now Ctrl+Down will open the match list for the part left of the caret and trim the part right of the caret. You now can live-filter the match list by pressing letters or select an item from it using the Down and Up keys.

Example 1:

```
C:\Program Files\ XYplorer\
                        ^
Now press Ctrl+Down: The match list will open for C:\Program Files\*.
```

Example 2:

```
C:\Program Files\X Yplorer\
                         ^
Now press Ctrl+Down: The match list will open for C:\Program Files\X*.
```

The next cool part of 6KN is that when the match list is already open, you can renew its contents (= browse another folder/pattern) without first closing it. Simply press Ctrl+Down again and the list will update.

Tips:

- In Configuration | Controls & More | Auto-Complete Path Names | Filter you can control whether the match list should contain "Folders only", "Files only", or "Files and Folders".
- You can open the auto-complete match list (for the whole path, regardless of caret position) by MouseDown on the icon.
- Delete all contents of the Address Bar to pop the drives list (including the Network node).
- Typing a single drive letter into the Address bar will drop the match list for that drive.
- On Ctrl+Enter, a file is handled as if the setting of "Open files from Address Bar" would be inverted: It is opened if "Open files from Address Bar" is OFF, and it is gone to if the setting is ON.

4 Key Navigation

4 Key Navigation (4KN) means you can browse the whole computer by just 4 keys (Ctrl, Shift, Tab,

Enter) in the Address Bar.

Note: Configuration | Controls & More | Auto-Complete Path Names | Address Bar has to be ticked for this, and Filter should be set to "Files and Folders" or "Folders only".

4 Key Navigation in the Address Bar

```
On closed match list:
  Ctrl+Tab:   Open match list at cursor position.
  Ctrl+Shift+Tab: Open match list to parent of current Address Bar item.

On the dropped match list:
  Tab:        Move down in match list (to first item if none selected).
  Shift+Tab:  Move up in match list (to last item if none selected).
  Ctrl+Tab:   Update match list to current list item.
              If no list item selected: close match list.
  Ctrl+Shift+Tab: Update match list to parent of current list item.
```

Note: In XP and earlier Ctrl+Alt+Tab works as an alternative to Ctrl+Shift+Tab.

Mouse Click Navigation

Mouse Click Navigation (MCN) means you can browse the whole computer by using just the Left and Right mouse button in the Address Bar.

Mouse Click Navigation in the Address Bar

```
On closed match list:
  Left-Click on icon: Opens match list at caret position.
  Right-Click on icon: Pops a breadcrumb menu for the current Address Bar contents.

On the dropped match list:
  Left-Click on Address Bar icon: Close match list (no action).
  Left-Click on file: Performs the default action for that file (open or go to).
  Left-Click on folder icon: Browse that folder (right in the dropdown!).
  Left-Click on folder name: Go to that folder.
  Right-Click on file: Pops a context menu that lets you open or go to the file,
     or go up to the parent folder.
  Right-Click on folder: Pops a context menu that lets you browse (right in the
dropdown!)
     or go to the folder, or go up to the parent folder.
```

Tip: Ctrl+Click on the edit box icon will open/update the match list to the parent of the current Address Bar item.

Quick Search via Address Bar

You can enter search patterns directly into the Address Bar. Simply put a ? (question mark) between location and pattern.

There are interesting possibilities, for example:

```
C:\WINDOWS\?*.log   = Find all LOGs in C:\WINDOWS\
Desktop?*.bmp       = Find all BMPs in Desktop
Desktop?            = Find all items in Desktop
?                   = Find all items in current folder
?*.txt              = Find all TXT files in current folder
```

It also works with Boolean and RegExp patterns, for example:

```
Desktop?>.*(\.bmp|\.jpg)$ = Find all BMPs and JPGs in Desktop
Desktop?:*.jpg | *.bmp    = Find all BMPs and JPGs in Desktop
?:!#                      = Find all items with no number in the name in current folder
```

A Quick Search bypasses the settings in the Find Files tab: They are not modified and not applied. Note especially that a Quick Search by default Includes subfolders and Does not follow folder links (both can be overwritten by switches). So, the Quick Search is a way to quickly perform a search without caring about the current state of all the other Find Files settings, and without having to open the Find Files tab or the Info Panel.

Notes:

- Quick Search support a couple of search pattern switches.
- In a branch-filtered search, any Visual Filters apply only to files, not to folders. So you can first filter the branches using the search pattern, and then filter the files within the shown branches using the visual filter pattern. This gives you some interesting possibilities, like show all *.jpg in branches not named "*copy*".
- The switch overwrites "Let folders pass all filters" (sets it to False internally).
- There is an extra interface for Quick Searches accessible from the Edit menu (F3).

Multiple Locations

Multiple locations have to be separated by "|" or ";". E.g., you may paste a line like these into the Address Bar...

```
E:\XY\;C:\Temp\;D:\Backup\?*.gif   = Show all GIFs in these three folders
E:\a\;E:\b\? /n                    = Show these two folders in one list, non-recursive (i.e. excluding subfolder contents).
E:\a\|E:\b\? /n                    = Same as above.
```

- The ";" (or "|") may be surrounded by any number of blanks. Built-in smartness will recognize a ";" that's not a separator but part of a folder name.
- Multi Location Quick Searches are remembered in tabs and across sessions.
- You can save them as Favorites or assign Keyboard Shortcuts to them via User-Defined Commands.
- As always with Quick Searches, the settings in the Find Tab are not touched.

Multiple Locations Advanced

Note that also CRLF (Windows newline sequence Carriage Return + Line Feed) is supported as multi location separator. This allows for interesting things like running a search on the clipboard contents (which are likely to be in the form of a line-by-line list).

For example, paste one of these lines into the Address Bar, then copy some files or filenames (full path) to the clipboard, the press ENTER in the Address Bar:

```
<clipboard>?*.jpg           //of all files in clipboard, list the JPGs
<clipboard>?dateM: y 2014   //of all files in clipboard, list the ones modified 2014
```

Note that <clipboard> works for real files in clipboard (Ctrl+C) as well as for just the path/names (Ctrl+P)!

Note that <clipboard> also works in the Location field of the Find Files tab.

Quick Visual Filters

You can set a Visual Filter directly through the Address Bar, the Catalog, the Favorites etc. This means you can set a new path and a new filter at the same time! For example:

```
Desktop|*.jpg;*.png    = browse to Desktop and show only *.jpg
                           and *.png files
```

The crucial operator here is the "|"-char (pipe). These are the general syntax options (Path can be slashed or not):

```
Path|a*    = set filter "a*" to Path
Path|      = remove any filters from Path
|a*        = set filter "a*" to current Path
|          = remove any filters from current Path
```

The filter will be added to the top of the Visual Filter MRU (most recently used) list, so it will be available for toggling on/off.

These "Quick Visual Filters" can not be combined with "Quick Searches" (using the "?" operator), as generally Searches cannot be visually filtered (it would mean filtering a filter...).

See Visual Filters for syntax.

Partial Matching

The last component of a non-existing location is wrapped in wildcards (C:\path*component*) and the first matching item is used as location. This works as well for relative paths. Note that the location must NOT end with a backslash to enable this feature.

Matching is done in this order:

```
C:\sub\lastcomponent
```

If this folder or file is not found, then *folders* (not files) are looked for in this order:

```
C:\sub\lastcomponent*
C:\sub\*lastcomponent*
```

Executing DOS commands

You can launch DOS commands directly through the XYplorer UI by prefixing "!" to the location term. The default startpath for the DOS commands is the current List folder. The DOS box will stay alive when you use a single "!", and it will auto-vanish when you use a double "!!".

Examples:

!	Simply open DOS box without any action. Hold SHIFT to open an elevated command prompt.
!dir	Directory listing of current list folder; DOS box stays visible.
!!dir	Directory listing of current list folder; DOS box will vanish immediately (not very useful with dir).

DOS commands support XYplorer native variables and Environment variables. Examples:

!dir "%temp%" /p	Directory listing of the TEMP folder; DOS box stays visible.
!!regsvr32 "<curitem>"	Register the currently focused and selected file.
!!regsvr32 <selitems>	Register all currently selected files.

Aliases

Aliases are user-definable variables where you can freely choose the name of the variable and their value. When you enter such an alias into the Address Bar (or any other location port) it will be resolved to its value before further processing.

Format of aliases: Aliases are marked by a prefixed @ (at-sign). The alias name may contain any character apart from "=", "/", and space. It may even be completely empty, so you can have one minimal alias that's simply @! The alias value may contain any character. Note, however, that line breaks do not count as "characters" here and are not allowed in name or value.

Usage of aliases: Aliases can be added, edited, or removed directly through the Address Bar:

```
@AliasName=value    define AliasName as an alias to value
                    adds a new alias or overwrites any existing
                    alias of the same name
@AliasName          go to/execute value associated with AliasName
@AliasName=         undefine AliasName (i.e. remove it)
```

For example:

```
@home=C:\mystuff\blah\yadda\ [ENTER]
```

Now typing @home into the Address Bar will carry you to

```
C:\mystuff\blah\yadda\
```

Another way to edit aliases is through Tools | List Management | Aliases...

Aliases with arguments

Taking it to the next level, aliases support up to 9 arguments. In the alias definition the arguments are referred to by placeholders <@1>, <@2>, <@3> etc. On using the alias the argument values are passed as a comma-separated list, separated from the alias name by at least one space. Any surrounding spaces are trimmed. Then <@1> is replaced with the first argument, <@2> with the second, etc.

A special placeholder <@0> stands for the non-split input, i.e. it will be replaced with everything following the first space.

Example 1: How to use 2 placeholders.

```
Define alias:    @Greet=::echo "Hello, <@1>! It's <@2>!";
Use alias:       @Greet Don, Daisy
Resolved alias:  ::echo "Hello, Don! It's Daisy!";
```

Example 2: To use the separator (comma) within an argument or have surrounding spaces you need to quote the argument:

```
Define alias:    @Greet=::echo "Hello, <@1>! It's <@2>!";
Use alias:       @Greet Don, "Duck, Daisy Duck"
Resolved alias:  ::echo "Hello, Don! It's Duck, Daisy Duck!";
```

Example 3: How to use the <@0> placeholder:

```
Define alias:    @Greet=::echo "Hello, <@0>!";
Use alias:       @Greet Don, Daisy, Huey, Dewey, and Louie
Resolved alias:  ::echo "Hello, Don, Daisy, Huey, Dewey, and Louie!";
```

Defining default values: You can optionally define a default value for each placeholder. The default is used when the respective argument is missing or empty. The default value is optionally stated within

the placeholder, separated from the beginning by a space: <@1 DefaultValue>. The default value can have all characters apart from >.

Example for using default values (one definition, several uses):

```
Define alias:    @Greet=::echo "Hello, <@1>! It's <@2 Dewey>!";

Use alias:       @Greet Don, Daisy
Resolved alias:  ::echo "Hello, Don! It's Daisy!";

Use alias:       @Greet Don
Resolved alias:  ::echo "Hello, Don! It's Dewey!";

Use alias:       @Greet
Resolved alias:  ::echo "Hello, ! It's Dewey!";
```

Further remarks on aliases

- Aliases are resolved very early, before any other special syntaxes are resolved. This means you can have e.g. small scripts, DOS commands, Jump and Spot commands, Visual Filters etc. defined as value of an alias and they will be treated as expected. Of course, they must fit in one line -- line breaks are not allowed.
- If you enter a non-existing alias it is not resolved and treated as relative path (which is probably not existing, but who knows...).
- The Address Bar icon for aliases is a blue heart. It turns red when you add/edit an alias.
- Aliases are case-sensitive: @hi is not the same as @Hi.
- Of course, aliases are retained across sessions.
- The number of possible aliases is not limited by XYplorer.
- Also Quick Searches support aliases. For example, this works in the Address Bar if you have an alias "home": `@home?a*` It will find all items beginning with "a" in the path that the alias "home" points to.
- You can use aliases also in the Location field on the Find Files tab.
- If you define <@2> but pass only one argument, <@2> is removed (replaced by nothing).
- If you define only <@1> but pass two arguments, the 2nd argument is ignored.
- Quotes inside quoted arguments have to be doubled.

Supported protocols

Address Bar (and also the Catalog) support the http[s]:// protocol and the file:// protocol (URI scheme).

HTTP locations are executed by the Shell which usually means they are opened by your default

browser.

A file URI takes the form of file://host/path. URI locations are resolved like this:

```
file://VEGA/shared          -> \\VEGA\shared          (opens folder)
file://VEGA/shared/XY.txt   -> \\VEGA\shared\XY.txt   (opens folder, focuses/selects file)
file://E:/Test              -> E:\Test                (opens folder)
file://E:/Test/XY.txt       -> E:\Test\XY.txt         (opens folder, focuses/selects file)
```

Note that after `file:` any number of slashes will work.

Support of common Windows environment variables

You can enter common Windows environment variables, e.g. %appdata%, into the Address Bar (Catalog, Favorites, Go To), to have a soft-coded approach to certain useful system paths. They have to be marked by a leading "%" and a trailing "%".

Obviously, to serve as a location those variables should point to individual paths. So, XYplorer allows only a subset of the Windows environment variables, but adds some proprietary variables to the set. For a list of the supported environment variables see here.

Note that these variables work well in combination with the extended XYplorer location syntax. It's a simple string replacement. For example:

```
%tmp%?*.tmp
%desktop%;%personal%?*.txt
%desktop%\archive
%desktopreal%\archive
```

4.8 Toolbar

Index

- Customizing the Toolbar
- Multi-Row Toolbar
- Other Toolbar Tricks
- Menu Buttons
- Drive Bars and Drive Buttons
- Special Toolbar Buttons
  - Go Now
  - Nuke
  - Touchscreen Mode
  - Edit Clipboard and Paste
  - Column Layouts
- Bookmark Buttons
- Custom Toolbar Buttons
- Smart Dropdown Buttons
- Vertical Popup Toolbars
- Droppable User Buttons (DUB)
- User Buttons as Open-With Panels
- Button Sets

Toolbar

To show/hide the toolbar use menu Window | Show Toolbar.

Context Menus

Note that almost every toolbar button has a right-click context menu.

White Space Context Menu

The toolbar itself has a right-click menu in its white space (where there are no buttons), where you can toggle Show Button Captions, Allow Multiple Button Rows, and switch the button set.

You can display this menu also from right-clicking a separator.

Customizing the Toolbar

The default toolbar contains a couple of buttons you might find useful. But there's much more in the store...

Right-click any toolbar button to customize the toolbar via the command Tools | Customize Toolbar... (Ctrl+Shift+F9). Here you can add more useful buttons and as many separators as you like.

Separator

At the top of the Available Buttons list you find Separator. Use it to separate groups of buttons.

New Row

Right below Separator you find New Row. Use it to separate rows of buttons, i.e. to create a Multi-Row Toolbar. You can have as many button rows as you like.
Tip: In the main window, the toggle Window | Arrangement | Allow Multiple Button Rows let's you show and hide less frequently used extra rows of buttons.

End of List

At the end of the Current Buttons list there is a pseudo-item "End of List". Its sole purpose is to make it easier to add buttons at the end of the list.

Filter Boxes

Let you filter both lists individually. The global "Ignore Diacritics" setting (Configuration | Find and Filter | Filters & Type Ahead Find | Visual Filters and Live Filter Box | Ignore diacritics) can be changed via the right-click menu of the filter icon in the filter box.

Options Button

The Options button opens a small popup menu with a couple of settings:

Button Size

- Autosize Buttons (button size and toolbar zoom auto-adjust to the resolution of the screen)
- Small Buttons (16x16 icon)
- Large Buttons (24x24 icons)
- Extra Large Buttons (48x48 icons)

Toolbar Style

Affects how the buttons are drawn, including the hover effect.

- Windows Theme Style
- XYplorer Style (square corners)
- XYplorer Style (Rounded) (rounded corners)

Other Properties

- Button Push Effect
  If ticked the button image is shifted 1 pixel down and right on pushing the button.
- Show Button Captions

The captions are printed in default font size 8.25 (can be changed in Configuration | Fonts), and the buttons are widened (depending on the number of caption lines) to make space for the text. Still many longer captions won't fit completely and you will see ellipses.
In the submenu Number of Caption Lines you can choose from 1 to 4 lines. Some smartness is built in: The more lines you use the less horizontal space is taken by the button.

- Button Text Font
  Define the font of the button text (e.g. for menu buttons).
  Tip: Ctrl+Wheel over the Toolbar changes the button text font size (over the upper half of the toolbar if button captions are shown). Apart from being highly practical as compared to going via the Configuration dialog, this way also offers a higher resolution (namely fractional sizes) than going by the ridiculously outdated Windows Font dialog.
- Button Captions Font
  Define the font of the button captions (see Show Button Captions above).
  Tip: Ctrl+Wheel over the lower half of the Toolbar changes the button captions font size (if button captions are shown).
- Scrollable Toolbar
  To scroll the toolbar drag it with left or right mouse button, or wheel it. Drag fast and you get a bit of animation when you release the mouse button. When you OK the Customize Toolbar dialog the scroll position is reset to initial. The scroll position is not stored between sessions.
- Scrolled Toolbar Wraps
  Check it to have the scrolled toolbar wrap around edges.
- Allow Button Set Switching
  Tick it to enable button set switching.
- Number of Button Sets
  Choose how many button sets you want to switch between.

Zoom

Here you can choose a Zoom factor from 0.5 to 3 in steps of 0.25.

Tip: Ctrl+Shift+Wheel over the Toolbar is another way to change the toolbar zoom.

Reset Button

Reset the settings to the state they were in when you opened the dialog.

Other Toolbar Tricks

Overflow Dropdown

Buttons that do not fit in the toolbar are reachable via a so-called overflow dropdown at the right end of the toolbar. You can freely combine it with the toolbar scrolling feature.

Zooming the Toolbar

Ctrl+Shift+Wheel over the Toolbar lets you control the toolbar zoom. See above under Toolbar Zoom for a non-Wheel way.

Revealing Toolbar Image Keys

Hold CTRL while hovering a button and the tooltip will show the button key and the current image key or image path (if different from button key). If you are into patching toolbar images you will like this.

Menu Buttons

Menu Buttons can be used to emulate the main menubar in the toolbar: File, Edit, View, Go ... etc., each pops one of the main menus on mousedown. You find them in the Customize Toolbar dialog after the main buttons and before the Drive buttons. Right-clicking those buttons pops a menu with all top menu items.

There is also a group button "All Menus (Group)". It adds all main menus as buttons to the toolbar at once.

The menus popped by these buttons have no item icons when the main menu is hidden. An unfortunate and mysterious fact that simply has to be accepted at this time.

Drive Bars and Drive Buttons

Drive Bars

Under Tools | Customize Toolbar | Available Buttons you find couple of drive-oriented "button groups" which softly and automatically (when you plug in/out a removable drive) react to the current drives setup. The groups represent drive bars with different scopes that partly contain each other, so for your toolbar you would typically choose one of them:

- All Drives (Group) = all drives as listed under the top tree node (Computer)
- Available Drives (Group) = all drives apart from empty CD drives and unconnected mapped network drives
- Hard Drives (Group) = only hard drives
- Removable Drives (Group) = only available removable drives (e.g. a floppy drive, thumb drive, or flash card reader)
- Network Drives (Group) = mapped network drives (connected and unconnected)
- CD-ROM Drives (Group) = CD-ROM drives (empty or not)

Further remarks:

- The soft drive bars are sensitive to the following settings in Configuration | General | Tree and List | Items in Tree and List | Select Items...: Floppy drives, Hidden drives.
- If Floppy drives is ticked then floppies are displayed even if empty to spare the annoying hardware check.

- You can use the normal drive buttons (see below) parallely to the soft groups without problems.

Drive Buttons

Under Tools | Customize Toolbar | Available Buttons you find buttons for all drives A: to Z: which you can use to build your own custom Drives Bar section within the main toolbar.

Further remarks:

- If the toolbar has large icons the buttons show the specific drive icon plus the drive letter, else just the letter.
- The icons react on adding or removing of removable drives if Auto-Refresh is enabled.
- The buttons' tooltips show the drive display captions.
- Right-clicking the buttons pops a small context menu where you can customize the function of the button. Choose between two options:
  - Go to Drive Root [Factory Default]
  - Go to Recent Path on Drive
  The button tooltips show the location they will go to.
- Last not least, the buttons function as drop targets, i.e. you can drop files on them to copy/move them to the drive's root. Use the right mouse drop menu for further options.

Special Toolbar Buttons

Go Now

Middle-click to open the current address bar location in a new tab.

Nuke

The Nuke command is only available as toolbar button. It's an alternative Delete command that is highly configurable.

By factory default Nuke pops a confirmation prompt and on OK deletes the selected items to the Recycle Bin. Via the button's right-click menu you can freely configure its behavior.

- With Confirmation: Enable or disable confirmation prompt.
- To Recycler: Recycle or delete permanently.
- Skipping Locked File: Skip any locked files.
- Wiping Beyond Recovery: Wipe files beyond recovery. See also Wipe under menu File | File Special.

Touchscreen Mode

Toggle Touchscreen Mode. Via the button's right-click menu you can customize the mode:

- Large Icons: usually 32x32.

- Extra Large Icons: usually 48x48.
- Scale Font: If checked the font is up-scaled by 33.33% on turning into the mode (e.g. from 9 to 12), and down-scaled by 25% when turning out of the mode (e.g. from 12 to 9). Affected are all controls that are enabled in Configuration | Fonts | Main Contents | Apply to...
- Scale Toolbar: If checked the Toolbar is scaled as well. Small buttons become large, large buttons become extra large (by 2x zoom).

Edit Clipboard and Paste

For these buttons the button tooltip serves as a quick and basic Clipboard Viewer. It shows text and file items contained in the clipboard (cropped after 2048 characters, or after 32 lines, whatever comes sooner). For text it shows the line count and the character count.

Yes, it even shows images in the clipboard in a Hover Box. Note that the Hover Box keyboard shortcuts work here, of course, so you can e.g. use Numpad Add/Subtract to size the box.

The Edit Clipboard button also gives a real time feedback of the state of the clipboard. It knows 5 states: Empty, Text, Image, Files (cut), Files (copied).

You can peek into the previous clipboard contents by holding SHIFT while you hover the Edit Clipboard or Paste toolbar buttons.

Column Layouts

The button pops a menu that lets you load predefined and user-defined column layouts by just one click. A column layout defines the visibility, the position, and the width of the columns.

These are the factory defaults:

- Standard Column Layout: Name, Size, Type, Modified (= what File Explorer offers by default).
- Extended Column Layout: Name, Ext, Size, Type, Modified, Created (= what XYplorer offers by default).
- Photo Column Layout: Name, Ext, Size, Modified, Created, Date Taken, Dimensions, Aspect Ratio, Exposure Time, Exposure Bias, F-Stop, Focal Length, ISO Speed, Camera Model.
- Audio Column Layout: Name, Ext, Size, Modified, Created, Length, Sample Rate, Bit Depth, Bit Rate, Channels.
- Audio Tags Column Layout: Name, Ext, Size, Modified, Created, Tag Title, Tag Artist, Tag Album, Tag Track, Tag Year, Tag Genre, Tag Comments.
- User Tags Column Layout: Name, Ext, Size, Modified, Created, Label, Tags, Comment, Extra 1, Extra 2, Extra 3, Extra 4, Extra 5

The Update... command will set this column layout to the current list columns. The factory default is still available via the Default command.

The Default command will load the factory default into the current list.

The clones of two commands related to the column layout are at the bottom of the menu: [View | Columns |] Load Column Layout... and [View | Columns |] Save Column Layout As...

Finally the Reset to Previous command brings you back to the previous layout. Can be used to toggle the last 2 layouts. Also works for Load Column Layout.... Not stored across sessions.

Tip 1: Hold down the CTRL key while clicking any of the column layout menu commands to display the column layout definition as text (widths are shown in DPI-aware values = as they would be in 100% screen resolution). All six "Update ..." commands display the same: the current column layout definition. You can use these definitions with the scripting command columnlayout().

Tip 2: Ctrl+Right-click on the line numbers column header ("#") in the file list pane to open the Column Layouts menu.

Tip 3: You can also show the column widths in true pixels (as opposed to DPI-aware values) when you hold CTRL+SHIFT while clicking any of the "Update..." items in the button's menu. Additionally this mode will return only the visible columns, and the Size column will return its actual width without graphic awareness (/g switch).

Bookmark Buttons

You can have up to 64 easily customizable Bookmark Buttons. They look and work like bookmark buttons as you know them from web browsers.

- In addition to the location, they support a custom caption. If no caption is specified, one is automatically generated from the location.
- In addition to normal paths (to folders or to files), special things like XYplorer native variables, environment variables, paper folders and virtual folders are supported as locations, as well as URLs and scripts.
- Button locations can also be one-line scripts, and you can drop items on them. Basic proof-of-concept example of a bookmark location that is a drop-on script: `echo <drop>;`
- The icon is automatically generated from the location.
- In the toolbar, these buttons display an icon and a caption to the right of the icon.
- Left-click the buttons to go to the specified location, or if they point to a non-executable file, to open the file with the associated application, or if they point to an executable file, to open that application .
- The tooltips for buttons pointing to non-executable files now tell you which application will open the file when you click the button.
- Right-clicking the buttons brings up a small menu from which you can edit the caption and location, and choose between go to and open where applicable.
- The right-click menu also lets you remove a bookmark button from the toolbar and reset its data.
- You can drag and drop items onto the bookmark buttons (except for virtual folders), to copy / move them to the location or open them with the application.
- You can add a "New Bookmark" button to the toolbar. Click it to add the currently selected list item as new bookmark button to the toolbar. If no item is selected, the currently listed folder is bookmarked. The button's context menu has some further options, like adding a bookmark from the

current clipboard contents.

Adding bookmark buttons by drag and drop

- You can also drag and drop items onto the toolbar to insert them as new bookmark buttons.
- The insertion hot zone is where two buttons touch (and at the beginning and end of each button row).
- An insertion marker and drag status box appear where the new button will be placed.
- If you drop more than one item, only the first one is used as the bookmark location.
- When you drop an item into a group element, such as {menus_all}, the group is automatically converted to individual elements.
- You can even drag items from other applications to the XYplorer toolbar.
- You can even drag and drop text (path, script, URL, etc.) from other applications onto the toolbar to insert it as a new bookmark button. With multi-line texts only the first line is used.

Custom Toolbar Buttons

[This feature is only available if the Scripting feature is enabled.]

You can have up to 64 easily customizable User Buttons, where you define how the button looks and what it does when pressed.

How to add a User Button: To add a new user button to the toolbar you first open the Customize Toolbar dialog (menu Tools) and add a User Button (bottom of the list) to the current buttons. Then you right-click the button in the toolbar and select Edit... In the Edit User Button dialog you set the Name and Icon (both optional) for the button, and at least one Script (required) that is triggered when clicking the button. That's all. Of course, the buttons are stored between sessions and portable.

User Button Icons: Simply state any existing file or folder, and its icon will be used for the button. Relative and portable paths are supported, as well as environment variables and short forms for registered applications (e.g. "Photoshop" without the quotes). Also generic patterns are allowed, e.g. "*.jpg".

If the toolbar has large icons, stretching or shrinking to 24x24 pixels would be needed, but experience shows that the quality of scaled icons is not satisfying, so it works as follows:

- If you point to an ICO, ICL, EXE, or DLL file then the first contained icon resource will be used on the large toolbar. Optimal results are achieved if this is a 24x24 icon resource (note that in multi-icon resources always the first icon is taken and scaled if necessary!). Other sizes are scaled. For the small toolbar this ICO file's small system icon is used.
- If you point to a PNG, JPG, GIF, BMP, or TIF file then this image is used for the button.
  Note that you don't even have to care for the size, not even for the ratio (it will be auto-shrunk and centered as necessary)! You can define a 3000 x 2000 pixel PNG as button icon if you like... and if you don't care for the time it takes to load and shrink such a file on every mouse over. But,

of course, a 24x24 pixels image would be optimal for speed.

- If you point to any other file type or folder, then its small system icon (16x16) is used on the small and the large toolbar without any scaling.

In other words, if you know how to make icons you can create high quality custom icons for the large and the small toolbar.

User Button Backgrounds: You can define the color for a circle (or rectangle) drawn as background to your icon. This can be useful in Dark Mode where some dark icons otherwise tend to disappear in the darkness. You simply append the RRGGBB color spec to the icon spec, separated by "*#", for example (in the Edit User Button dialog):

```
Name: Dark Cloud
Icon: c4.ico*#F6F6F6
```

To draw rectangular shape append "r" separated by a comma, eg:

```
Name: Dark Cloud
Icon: c4.ico*#F6F6F6,r
```

You can limit the drawing of the background to the Dark Mode by appending "d" separated by a comma. For example, this defines a rectangular shape that's only drawn in Dark Mode ("rd" or "dr", the sequence does not matter):

```
Name: Dark Cloud
Icon: c4.ico*#F6F6F6,rd
```

Note that you can abuse this feature to use colored circles as icons. Simply drop the file name:

```
Name: Golden Circle
Icon: *#FEB914
```

Tip: XYplorer variables and environment variables are supported in the icon paths. The path defaults to the XYplorer icon path <xyicons>.

Tip: You may as well re-use XYplorer's internal Toolbar icons for your own button. Simply state the toolbar icon key preceded by a colon (":") in the Icon field, e.g. :hotlist. You can find the toolbar icon keys in the Customize Toolbar dialog either when you hold CTRL while calling that dialog, or when you call the dialog from the context menu of a user button. Holding CTRL also works in the Customize Toolbar dialog.

Tip: Holding CTRL will also show (parts of) the script in the tooltip. Also works in the Customize Toolbar dialog.

User Button Labels: You can as well define a short text to be displayed in the toolbar instead of the icon. Simply prefix the contents of the Icon field with "label:", for example: `label:Fun`. Now "Fun" is printed to the button in the toolbar. Optionally you can define text and back colors in format RRGGBB, for example:

```
label:Fun>FFFFFF,555555  //white on dark grey
label:Fun>FF0000         //red (no background)
```

Of course, this also works for Droppable User Buttons (which are just a subtype of User Buttons).

User Button Scripts: Multi-line scripts (and hence also multi-scripts) are supported and can be entered directly in the button definition using the Edit... button.

Besides scripts also the Hamburger syntax is supported.

On left-click: Here you can define the script to be run on left-click.

On right-click: Here you can define a separate script for a user button's right-click event. It is merged with the standard right-click menu for user buttons. This means you can have user buttons that do one thing on left-click and pop lots of additional options on right-click.

There's special support for the load command. Example (in field On right-click):

```
load "test", "foo" //load script labeled "foo" from "test.xys"
```

The file "test.xys" will be loaded and parsed, and then the context menu is shown, with the standard items (Click, Edit... etc) appended at the bottom. If the file contains only one script, it is not executed directly (as it would be in a "non-Right-Click" load statement) but shown in the menu.

Tip 1: Using the internal button keys you can create context menus featuring buttons that might be not visible on your toolbar:

```
"Recent Locations|:mru" button "mru"
"Hotlist|:hotlist" button "hotlist"
-
"CKS|:cks" button "cks"
"UDC|:udc" button "udc"
-
"Exit no save|:exitnosave" button "exitnosave"
```

Tip 2: You can skip stating the caption when you use an internal icon. The internal caption and keyboard shortcut will be used. For example:

```
"|:hotlist" button "hotlist"
```

will show "Hotlist Ctrl+H" in the popup menu.

Tip 3: It might happen that you construct an invalid script that will corrupt the context menu of a user button, in which case you'd not be able to reach the Edit dialog via the button's context menu anymore to correct the bad script. To escape from this deadlock you now can hold CTRL while right-clicking the user button, and you will get the default context menu for user buttons.

On middle-click: Here you can define the script to be run on middle-click.

Fire click on mousedown: Check it to have the User Button trigger at MouseDown. The button will be displayed with an arrow overlay in the toolbar.

Note that you should typically use this option only if your Click script creates a popup menu, in which case it will save you one click because MouseDown will open the menu and MouseUp will select the desired item.

Clear: Press this button to clear all fields.

Drop on Custom Toolbar Buttons

All scripted buttons are "droppable". You can drop files or text onto a button, and the scripts assigned to the button are triggered. If you drop by left mouse button, the on-left-click script is triggered. If you drop by right mouse button, the on-right-click script is triggered. The dropped files or text can be referred to in the dropped-on scripts by <get drop> (see Drop on a Script File).

Notes:

- At least the on-left-click script has to be defined, else the button is interpreted as a Droppable User Button (DUB) of the old (and still working) style where the Name field is used for the location.
- If no right-click script is defined, then a right mouse drop will not do anything.

Text Buttons

You can make your own text buttons. Simply prefix "text:" to the Name, e.g. "text:Beer & Chips". Sure, they usually take up more space than an icon, but if you don't have a suitable icon handy, you can type your button now.

Smart Dropdown Buttons

It must not be scripts. User Buttons also support passing plain path/file specs as a shorthand for a well-formed goto-scripts, or open-scripts. The type of the listed items determines the action: The button will drop a menu that opens documents, runs executables, and goes to folders. Note that at least two items are needed to form a valid Smart Dropdown Button.

Examples:

```
C:\                     = go to C:\
C:\Windows              = go to C:\Windows
%winsysdir%\Calc.exe    = run   %winsysdir%\Calc.exe
%winsysdir%\setup.bmp   = open  %winsysdir%\setup.bmp  with the default application
```

This little shorthand trick comes extremely handy in the left- and right-click events of User Buttons. Simply paste a plain list of documents, executables, and folders (in any mix and order), and enjoy the power of type-aware default actions right from the toolbar. Environment variables, XYplorer native variables, and Aliases are supported.

For example (the hyphen stands for a menu separator):

```
C:\
%windir%
<xypath>
-
%winsysdir%\Calc.exe
%winsysdir%\setup.bmp
<xydata>\XYplorer.ini
<xy>
```

Note that the menu captions are reduced to the item titles, and the full paths are shown in the Status Bar on hovering the popup menu items. Optionally you may pass a quoted caption in front of the path spec. This can be useful when the item titles don't distinguish the items, for example:

```
"XY code www" G:\www\xyplorer.com\code\
"XY code" H:\XYplorer.dev\code\
```

Note that all the Smart Dropdown Button stuff above is just a special case of the Hamburger syntax.

Vertical Popup Toolbars

Custom Toolbar Buttons also support a popup menu syntax that uses toolbar button keys as item definition. For example, simply paste this (via the Edit button) into the "On left-click" field of the "Edit User Button" dialog:

```
:dpmoveto
:dpcopyto
-
:newfolder
:copypath
:showfolders
-
:pp
```

And tick "Fire click on mousedown" for the smoothest experience. Now when you click the button the vertical toolbar will pop up below the button.

Remarks

- More complex definitions are possible as well, including scripts and nested submenus (see Hamburger).
- The beauty of this feature is that it adds a 2nd dimension to the toolbar. With very little work you can create buttons that pop vertical toolbars right from the main horizontal toolbar.

Droppable User Buttons (DUB)

There is a way to turn a normal Custom Toolbar Button into a Droppable User Button which is a toolbar button onto which you can drag-n-drop files: Enter a Location or Executable or File into the "Name" field and leave the "On Click" field empty. The button will now automatically show the matching icon and certain behaviors depending on what you entered:

```
         Location        Executable      File
         -------------   -------------   --------------------------
 Click:  Go to           Run EXE         Open with Associated (Custom File Associations)
  Drop:  Copy/Move to    Open with EXE   (Nothing)
```

This means when dropping files onto a DUB that points to a location the files are copied or moved to that location. When dropping files onto a DUB that points to an executable the files are opened with this executable. Clicking a button that points to a location will bring you to that location, clicking a button that points to an executable will run this executable, clicking a button that points to a file will open that file with the associated application.

Remarks

- All portability feats are fully supported (XYplorer native variables, environment variables, portable paths). Also basenames (e.g. `ACDSee32` can stand for the full path to the Executable) are supported for registered applications.
- Command line parameters work also when dropping onto the buttons. Fictive example for an executable identified by basename and with a command switch, in the "Name" field:
  `ACDSee32 /fullscreen`
- Right-mouse dropping is supported, offering the full power of the drag-n-drop context menu..
- You can customize the button icon by stating an icon resource (or any file or folder that has a system icon) in the "Icon" field.
- The button tooltip tells you that it's a DUB (in case you wonder).
- You may as well state a single file (full path), for example an executable to launch, in the "On Click" field, and the button will run/open that file.
- You can prepend a quoted button caption to the path in the Name field:
  `"Drop Here" paper:DropStack`
- The "Fire click on mousedown" option is functional when the button points to a Paper Folder. Mousedown on the button will then pop a menu listing all the existing items in the Paper Folder in the original order of the Paper Folder. So now you can have buttons that act as favorites hubs, where you just drop your favorites on the buttons.
  If you hold down CTRL when you click the button, you can open menu items instead of going to them.

This is only for files, it makes no difference for folders.
To remove items from the menu or change their order, go to the Paper Folder and make the changes there. To go to the Paper Folder, right-click the button and choose Click from the menu.

User Buttons as Open-With Panels

Here is a simple way to make a User Button pop a menu offering a list of items where the opening application AND the opened file is defined. In the "On right-click" field of a User Button pointing to an executable you can pass a path/file spec (all portability feats are fully supported) as a shorthand for a well-formed open-with script. Such a line will be internally auto-converted to a valid open-with statement including the appropriate icon in the generated popup menu, and in all those open-with statements the open-with application will be the one defined in the Name field of the button.

In other words, paste the list of files you want to open with the executable into the "On right-click" field and you are done!

Three examples for the contents of Name field:

```
ACDSee32
ACDSee32 /fullscreen
C:\Program Files (x86)\ACDSee32\ACDSee32.exe
```

One example for the contents of the "On right-click" field:

```
E:\TestFiles\images\_misc\08-lrg-16bit.tga
E:\TestFiles\images\_misc\aguilera-32bit.tga
Desktop\voice_dailer_256.png
-
"Favorite Folders|:favs" button "favs";
```

Now right-clicking the button will pop a menu containing three image files, one separator, and one little script that opens the Favorite Folders menu. Selecting any of the image files from the menu will open it using ACDSee (all items not pointing to an existing file are processed as scripts).

Tip: This feature also works in the "On click" (= left-click) event of the button (also with "Fire click on mousedown" enabled, which is quite cool). However, the button will not work as a DUB in this case (DUBs by definition need this field to be empty).

Button Sets

There are up to four separate button sets that you can switch between using Shift+Wheel (or just Wheel, see below) over the toolbar (or using the button Switch Toolbar Button Set, or the command Window | Arrangement | Switch Toolbar Button Set). The 2nd one factory-defaults to the menu buttons ("All Menus (Group)"), the other ones are almost empty by factory default. All button sets can be designed independently.

Notes

- You have to tick "Allow Button Set Switching" (see above) to enable button set switching.
- When "Scrollable Toolbar" (see above) is OFF, you don't need to hold down the SHIFT key to switch

button sets by the wheel .

4.9 Tabbed Browsing

Tabbed Browsing

On each pane you can have any number tabs, where each tab can point to a different location (a "Browsing Tab") or to a search results listing (a "Finding Tab"). The tabs remember their configuration (list view, sort order, column widths and positions, scroll position, focused item, etc.) individually and across sessions.

The tab-related commands are found in menu View | Tab and in the context menu of the tab headers (right-click any tab header). See Configuration | Tabs for extended options concerning tabs.

Open a New Tab

To open a new tab use the command New Tab at the top of menu View | Tab.

Shortcut: Ctrl+T

Browsing Tabs and Finding Tabs

Each tab remembers the mode it was in when left (browse mode or find mode).

When you re-select a Browsing Tab (a tab in browse mode):

(1) The tree will jump to the tab's location and the list will display the contained items.

(2) Find Files settings do not change.

(3) Multi-selections are retained. To save resources selections are only retained in lists of no more than 32,767 items, and no more than 4,096 individual selections are retained between tab switches.

When you re-select a Finding Tab (a tab in find mode, showing search results) there are some differences:

(1) The last find will run again exactly as before (or the cached search results are loaded) and the list will show the results.

(2) Find Files settings change according to the tab's previous settings.

(3) The tree will not change.

Finding Tabs provide a very simple one-click-access to all imaginable search routines. You can for example design yourself a "find suite", a row of tabs with your recurrent find tasks just one click away. Hey, with XYplorer's ability to load specific configurations (INI files) on start-up you can have thousands of find suites!

View Settings are remembered per-tab

All tabs remember the following settings individually:

(1) View mode (details, thumbnails...)

(2) Sort order (column, direction)

(3) Column layout (positions, widths, visibility)

(4) List style (line numbers, auto-size, grid...) (see menu Tools | Customize List)

Tip: To set any of these settings to all open tabs at once simply hold SHIFT while making your choices. This trick does not work with the Column layout (positions, widths, visibility), however.

Multi-selections in tabs

Multi-selections are retained between tab switches.

Note that selections are remembered by filename, so if another process renames a selected file it will not be recognized as selected. Again, this is irrelevant if *all* files are selected.

Note that multi-selections are not retained between sessions.

Search results to tab

In Configuration | Find Files, you can choose where the search results shall be listed: on the Current tab, a New tab, or a (locked or unlocked) tab called "Search results" (automatically created if not yet existing).

Tabs are drop targets

Note that you can drag and drop files onto the tab headers to copy or move them to the location the hovered tab is pointing to. If you wait for a configurable time (see Configuration | Tabs) the hovered tab will be auto-selected.

Left and right button dragging is supported. The latter pops the usual drag and drop context menu.

Renaming tabs

You can give the tabs any name of your choice. Once a tab is named that name will stay fixed when you change the current directory inside the tab. A tab name can have any characters, even those that are illegal for filenames. To un-name a tab rename it to nothing.

When a tab is locked to a home zone AND is renamed then its icon will always be the one of the tab's home folder. This behavior is very practical when you use iconized tabs because once you have defined a special icon (using common Windows means) for the home folder you always recognize that tab by its icon no matter at what subfolder it is currently pointing to.

Custom tab icons

You can customize the icon of each Tab. Simply append any file spec to the caption of the tab, i.e. to the name you give to a tab by the Rename Tab function (found in the tab headers context menu), separated by a | (pipe). It can be an image file (GIF, JPG, PNG, ICO) or any other file or folder with associated system icon. XYplorer native variables, environment variables, and portable paths are supported. Also internal toolbar icons (prefixed with ":") are supported, as well as icons extracted from icon resources (exe; dll; cpl; ocx; scr; icl; bpl; wlx; wfx; wcx; wdx; acm).. Examples:

`fav|E:\Kiss.ico`	Use that specific icon.
`Texts|*.txt`	Use the generic system icon for TXT files.
`xy|<xy>`	Use the icon of XYplorer.exe.

`Machine|This PC` Use the icon of "This PC".

`|:home` Use the "home" toolbar icon. No particular name specified.

`|fun.png` If the full path is not given the path resolved relative to the Icons Path (`<xyicons>`).

`Charlie|C:\WINDOWS\system32\shell32.dll /160` Use embedded icon #160 in shell32.dll.

Tip: Right-clicking the icon of a tab pops a breadcrumb menu for the tab's path.

Portable Tabs and Portable Paths

You can make your tabs fully portable by relocating them to a portable path, i.e. by pointing them to a term that resolves to a path depending on the environment. Here are some examples:

- Absolute Path: C:\Myfolder (the normal case: not portable)
- Relative Path: appdata (relative to the app folder)
- Portable Path: ?:\ , ?:\Stuff\ (? stands for the application drive, the drive on which the current instance of XYplorer is running)
- Named Drives: TheSystem:\Windows (TheSystem stands for a drive named "TheSystem", see Named Drives)
- Environment Variable: %temp% (depends on environment variables)
- Native Variable: <xydata> (depends on the variable and the context)

These terms are remembered as such (unresolved) between sessions. But how do you define such a location? You relocate them using the command menu View | Tab | Relocate Tab.

Tip: To increase the usability of such a portable tab it is recommended to set its home (Set Home) to the portable path and then apply Lock Home Zone. That way you can browse the whole portable branch without ever losing the portability of that tab.

Named Drives

You can specify paths using their name (Volume Label) instead of the drive letter. So, for example, if your drive C: is called "TheSystem", then these two paths will refer to the same location: C:\Windows and TheSystem:\Windows. This works wherever Portable Paths work. Of course, this way to specify drives is particularly interesting for removable drives where the host system assigns an unpredictable drive letter.

Note: Support for Named Drives needs to be explicitly enabled: Configuration | Controls & More | Miscellaneous | Support volume labels in paths.

Tips for naming drives

- '@', '!', and '%' are valid characters in a volume label. But they create possibilities for parsing mess:
- '@a:\' takes you to the volume named '@a' unless '@a:\' is defined as an alias.
- '!dir:\' runs an invalid DOS command instead of taking you to the volume named '!dir'.
- With environment variable 'MyDrive' set to 'Q', '%MyDrive%:\' takes the user to 'Q:\' instead of the volume named '%MyDrive%' (which is okay if the volume mounted at 'Q:' is named '%MyDrive%').

So: Don't use such crazy Volume Labels when you enable "Support volume labels in paths".

Tabs with a Home

Each tab can have its own "Home", defined by a browse/find location and mode, and optionally the file list layout (turn it on at *Configuration | Tabs | Going home also restores the list layout*). To set (or update) the current tab's home click "Set Home" (menu View | Tab). To go to the current tab's home click "Go Home" (menu View/Tabs) or simply press Alt+Home. You cannot unset a home because it's not necessary (if you don't want a home, don't go home). Homes are remembered between sessions, and can be set independently of the tab being named or not. A new tab starts homeless. Apart from the possibility to click "Go Home" home tabs behave 100% like homeless tabs.

Portable Homes: Homes also support portable paths like %temp% or ?:\Stuff\, and variables like <xypath>.

See Lock Home Zone for a way to lock tabs to a branch.

Locked tabs

A locked tab will never leave its current location but instead open a new tab automatically. Of course, you can as well lock the Search results tab.

A locked tab is marked by an underlined caption.

Default tab

It's a means to reduce the number of opened tabs: Any passively (implicitly) opened new tab will open in this tab instead. There's a visual indicator (a green icon overlay) to show which tab is the default tab (if any). There can be only one default tab.

Note: the "Default Tab" will only be used if no other tab pointing to the desired location does already exist in which case this tab will be selected.

Tab switching by mouse wheel

By default you can use the mouse wheel over the tabs to switch tabs.

To toggle this behavior right-click the empty part of the Tab Bar, or the "New Tab" or "Tab List" button, and tick/untick the menu item "Tab Switching by Mouse Wheel".

Moving the tabs

Need a new order? You can easily drag-shift the tabs using the mouse.

Closing the tabs

In Configuration | Tabs you find an option to show X close buttons on each tab, either permanently or only when hovering the tabs.

Modifying the height of the tab bars

You can modify the height of the Tab Bars on-the-fly by Ctrl+Shift+Wheel over any of them.

4.10 Tabsets

Tabsets

You can save and load the tabset of a pane, that's all tabs including their layout (tab settings and tabwise list settings) and contents (cached search data, find settings, homes, history, etc.). This concept is also known as Workspaces or Environments.

While each pane always has exactly one tabset loaded, you can have any number of tabsets available on disk to choose from. The definition of each tabset is held in a folder (also referred to as "pane data path") containing all necessary information within a couple of files. The name of the folder is identical to the name of the tabset. Thus tabsets are perfectly portable and easily maintainable.

Interface

The commands related to tabsets are found in the main menu "Tabsets", in the toolbar button "Tabsets", and can also be accessed via the scripting command tabset().

1. Main Menu Commands

See Tabsets Menu.

2. Toolbar Button "Tabsets"

The names of the current tabsets are now shown in the button's tooltip.

MRU list: The button's left-click dropdown contains a list of the most recently used tabsets (MRU list) of the active pane. Click an item to load that tabset. The MRU tabsets lists hold 32 items per pane. The lists are stored between sessions in a portable manner

Tip: Hold CTRL when selecting a tabset from the MRU list to load it as a clone.

The button's right-click menu mirrors the main menu.

3. Scripting Command tabset()

See tabset().

Further Remarks

- On loading a new tabset the previous tabset will only be auto-saved if Configuration | Tabs | Auto-save tabsets on switch is checked. As long as no explicit Save or Auto-save is performed, you can always revert to the stored state of a tabset using the Revert to Saved command.
- You are not allowed to load the same tabset into both panes, i.e. to point both panes to the same pane data path, because this would lead to sure trouble.

- The current pane data paths are shown in Various Information window (menu Help). They are also available in the new variables <xypane1> and <xypane2>.
- The pane data paths are typically located under <xydata>\Panes, and they can be referenced relative to this location. It is recommended that you store all your tabsets here; makes it easier for you to handle.
  However, tabsets can be stored anywhere in the system, even in a network path. This means users can share tabsets over the network. Note, however, that currently no measures are implemented to handle concurrent save operations of such a shared tabset.
- The pane data paths are stored in the INI file in a portable way relative to <xydata>\Panes.
- The name of the current tabset can be shown in the title bar using the variable <tabset>. The variables <tabset1> and <tabset2> can be used to show the tabset names of both panes.

4.11 Breadcrumb Bars

Breadcrumb Bars

Both panes can feature a Breadcrumb Bar (toggled via menu Window | Show Breadcrumb Bar). It shows the full path of the current tab and lets you efficiently navigate the whole file system. Its layout is highly configurable.

Usage

Click on any of the components to go there.

Click on any of triangles (separators between the components) to pop a dropdown menu with all subfolders of the component left of the triangle, and select one of them to go there. Or, if Drop Menu on Hover is enabled (see below), hover them to pop the dropdown menu.

The setting of "Configuration | General | Tree and List | Items in Tree and List | Select Items... | Hidden files and folders" and "Configuration | General | Tree and List | Items in Tree and List | Select Items... | System files and folders" is honored by the dropdown menus.

You can right-click items in the dropdown menus to pop a little context menu with copy functions (only if the menu type is set to Custom Menu or Colored Menu, see below).

You can drop items onto the components in the Breadcrumb Bars.

Right-click anywhere in the Breadcrumb Bar to pop the right-click menu (see here below).

You can see the full real path of each component in a tooltip when you hold down CTRL while hovering over it.

Right-Click Menu Options

At the top of the right-click menu you find some useful commands:

Copy Path: Copy the right-clicked path to the clipboard. The text to be copied is shown in the Status Bar.

Copy Real Path (only visible when the current path is a special path) and Copy Special Path (only visible when the current path is a real path that has a special path counterpart). These commands can be used to copy the real/special counterpart of the currently shown path.

Copy Name: Copy the right-clicked component to the clipboard. The text to be copied is shown in the Status Bar.

Paste and Go: Goes to the current clipboard contents (can be a folder or file, or folder or file name). When hovering the command, the Status Bar displays the text found in the clipboard. Note that the clipboard can also hold relative paths. They will be resolved relative to the current list path.

Paste and Search: Searches the right-clicked component for the current clipboard contents. When hovering the command, the Status Bar displays the text found in the clipboard, prefixed with the right-clicked component and "?".

In the lower part of the right-click menu you find a couple of settings to control the looks of the Breadcrumb Bars.

Triangles: Show triangles as separators between the components.

Slashes: Show slashes as separators between the components.

Chevrons: Show chevrons as separators between the components. When Chevrons is selected, the four navigation buttons on the left side also change their style to something chevronesque.

Bold: Tick it to show the text in bold.

Show Navigation Buttons: Show a couple of standard navigation buttons (back, forward, up, down). Each of them has its own right-click menu.

Show Menu Button: Show the menu button, aka Hamburger. It pops a small menu with these default commands:

Move to Other Pane
Copy to Other Pane
New Folder
Copy Path/Name
Preview Pane

The idea is to bring often used commands closer to the mouse.

This menu follows the Hamburger syntax and is fully customizable.

Note that both Breadcrumb Bars share the same Hamburger menu.

Show Icon: Tick it to show the icon of the current path at the left end of the bar.

Show Drive Labels: Tick it to show the display name for the drive, not just the letter. Triangles mode only.

Show Downward Paths: If ticked then the most recently used downward path is shown in a somewhat lighter color than the current path. This allows you to quickly go up and down a path in the most easy manner.

Check for Subfolders: If ticked then the final triangle/slash is only shown when the last shown node has subfolders. If unticked the final triangle/slash is always shown. Why is this optional? Well, the check can take a while when a folder is very large. Note: In network locations the check is always skipped to avoid trouble with unavailable locations.

The dropdown menu popped on clicking a triangle can have three different formats:

Standard Menu: The menu is a standard Windows menu. It has no scrollbars, so if there are a lot of

subfolders the menu will extend to the whole screen height which makes it difficult to handle.

Custom Menu: The menu is a proprietary control featuring a vertical scrollbar if necessary.

Colored Menu: Same as Custom Menu, but inheriting the colors of its Breadcrumb Bar.

Drop Menu on Hover: Tick it to auto-drop the menu on hovering a component separator (typically a triangle symbol), and auto-undrop when moving away. No more clicking.

Note: Needs either Custom Menu or Colored Menu selected above.

Other Configuration Options

Adjustable font size: You can modify the font size of the Breadcrumbs on-the-fly by Ctrl+Wheel over any of the Breadcrumb Bars. Alternatively you can customize font and font size via Configuration | Fonts | Main Contents. (To apply the font size tick "Breadcrumb Bars" in the list popped by the Apply To... button).

Colors: All colors of the Breadcrumb Bars are configurable in Configuration | Colors.

4.12 Tree

A New Kind of Tree

XYplorer's Tree has some features that might be new to you when you are coming from Windows Explorer.

Showing the Tree

Use menu Window | Show Tree (Shift+F8) or Window | Show Navigation Panels (F8) to show/hide the Tree.

Desktop is located *under* Computer

Other than in Windows Explorer, the Desktop node is located under Computer instead of vice versa. This seems more logical and spares 16 pixels of horizontal space in the Tree.

Tree nodes unfold on hover

There's extended drag and drop functionality to the Tree: When dragging an object over a folded node, the node will automatically expand after 0.8 seconds. When the object leaves the tree area or Drag&Drop is cancelled by ESC, the previous state of the tree is restored.

Note that this auto expansion only happens when you hover the expansion icon, the folder icon, or the folder caption, but not on any other part of the node row. This gives you a little more control.

Highlighting features

Various highlighting features add more visual grip to the tree. Color Filters, Highlight Folder, Boxed Branch, Favorite Folder Bold, Tree Path Tracing... all colors are fully customizable.

Mini Tree

The tree can be turned into a Mini Tree, that's a tree that only shows the paths that you have actually visited.

Bypassed Tree

A Tree bypassed by browsing (i.e. the location shown in the file list is not shown, let alone selected, in the Tree) is displayed with a grey background, or with the color you have defined for a Locked Tree (Tools | Customize Tree | Lock Tree). This makes it immediately clear that Tree and List are not synced at the moment and protects you from false assumptions and potentially fatal decisions.

Context menus

A right-mouse-click on Tree will popup various context menus depending on mouse position and held control keys:

Folder Caption	Right-Click: Pops the Windows shell context menu for folders, plus some custom commands (if Configuration | Menus, Mouse, Usability | Context Menus | Custom items in the context menu is ticked). Customize the context menu in Configuration | Menus, Mouse, Usability | Custom items in the context menu | Folder Tree... Ctrl+Right-Click: Pops a super-fast internal context menu instead of the slower shell context menu. Gives you quick access to XYplorer's native menu commands.
White Space	Right-Click: Pops the Favorites context menu.
Expansion Icon	Right-Click: Pops a list of all child folders of that folder, aka Tree Node Crumbs. That way you can select any child folder of a visible folder without opening the whole node. This is especially useful in the Mini Tree. - Child folders that are not yet present in the Mini Tree are colored with "Marked Text 2" color in the popup. - Right-click the same Expansion Icon again to hide the popup. Ctrl+Right-Click: Pops a menu that lets you choose another Expansion Icon.

4.13 Mini Tree

A Tree-Shaped History

The Tree is a great user interface for file management... but it quickly becomes a problem when it's too large. And isn't it always too large? Much too large?? How many of the displayed folders do you really need to see? 20%? 10%? Maybe just 5 out of 400?

This is where Mini Tree comes to the rescue. It's an extremely simple idea, but it might change your way of browsing the file system forever: The Mini Tree displays only the paths you have actually used. This makes browsing blindingly fast (it's instantaneous even with deeply nested subfolders), it makes the tree ridiculously small, and it makes you feel like being back in control (because you look at just the folders you are actually using, instead of being avalanched by all the folders that have accumulated on your drives over the years).

Usage

You turn it on/off in menu View | Mini Tree. Immediately the tree is reduced to the current folder. You now typically use Favorites, History, Recent Locations, the Hotlist, the Catalog, the Address Bar, or the file list to jump to another location. It will be added to the Mini Tree automatically. Soon, the Mini Tree will show the locations you actually need. It has learned.

Features of the Mini Tree

(1) The Mini Tree is retained across sessions: When you start XYplorer on Monday you'll look at exactly the same tree you left Friday afternoon.

(2) You can easily hide folders from the Mini Tree via the context menu command "Hide Folder from Tree".

(3) There's the scripting command loadtree by which you can load a previously stored Mini Tree, add folders to it, or remove folders from it.

(4) The Catalog supports storing and restoring Mini Trees via its context menu.

(5) There's a toolbar button "Mini Tree" which works as a one-click toggle between Mini Tree and normal tree (Maxi Tree). The button also features a context menu with a couple of related commands.

(6) The extreme speed and smallness of the Mini Tree makes it especially valuable for network browsing. If you work in an environment with hundreds or thousands of servers, you might welcome the idea of seeing just the two servers you actually need...

(7) If you fully collapse a tree folder (Numpad Divide, or Ctrl+Click the expansion icon of the folder) it is not only collapsed but reset to initial state: When you expand it in the tree it will do a fresh browse and show all existing subfolders.

If you regularly collapse a tree folder (click the expansion icon, or press ENTER) its internal state is remembered (also across sessions) and restored when you expand it

the next time.

(8) The Mini Tree is distinguished from the normal tree by inverted Expansion Icon (white-on-blue).

(9) Pressing Shift+F4 (Refresh Current Folder) will list all existing subfolders of the current folder. So you can quickly inject some reality into your Mini Tree.

(10) If the current node is Computer, then pressing Shift+F4 (Refresh Current Folder) will reset the Tree and list all child folders of Computer in collapsed state while keeping the tree in "Mini" mode.

(11) Pressing F4 (Refresh Tree) will it will take care that all available drives are shown, and it removes all non-existing children (unless Configuration | Controls & More | Miscellaneous | Allow zombies in the Mini Tree is ticked).

(12) Via menu Tools | List Management you can directly edit the Mini Tree.

(13) To keep a good performance expanded nodes with more than 5000 visible children will be stored in collapsed state.

Tree Node Crumbs

Right-click the expansion icon of a folder to pop a list of all child folders of that folder. That way you can select any child folder of a visible folder without opening the whole node. This is especially useful in the Mini Tree. See also here.

Toolbar button "Mini Tree"

The button has a context menu with some Mini Tree related commands.

4.14 Glider

Glider

You move the mouse over a specific area of the tree, and for each node, the glider, a floating mini toolbar, magically appears, offering the most basic file management buttons: Copy, Move, and Paste (and optionally one other customizable button). Clicking them will copy or move whatever is selected in the list (or paste whatever is on the clipboard) to the folder the glider points to. It's a one-click wonder that you'll get right away because it's so natural. A game changer when it comes to manually organizing files into folders.

Enabling the Glider

By default, the glider is turned off. To turn it on, click Tools | Customize Tree | Show Glider. A quick way to access this menu is to Shift-Right-click anywhere in the tree's white space.

Where is it?

The Glider appears when you stop the mouse in the so-called hover zone. It will never appear while the mouse is moving.

The hover zone defaults the right edge of the tree, with a width of 4 times the scrollbar width. The settings can be easily changed using the Glider's right-click menu.

Appearance of the Glider

When the glider appears, the tree node to which it belongs is surrounded by a rectangle in the color of the glider. And if the glider is not very close to the node's caption, an arrow points from the glider to the node.

Buttons that would do nothing when pressed are disabled.

Customizing the Glider

Virtually everything about the glider is customizable via its right-click menu.

Hover Zone

The hover zone is the area where the glider appears when you hover it. In this submenu you can set the following properties:

The location of the hover zone: Left Edge, Right Edge, Both Edges. Edge refers to the edge of the tree pane.

The size of the hover zone:

- Narrow (as wide as a scroll bar)
- Wide (4 times as wide as a scroll bar)
- All White Space (everything except the icons and captions).

Where the glider appears:

- Snap Next to Mouse: Next to but not right under.
- Snap to Edge: Left or right tree pane edge.
- Snap to First Button: Right under the mouse, ready to click.
  NOTE: Use this setting carefully, as it can cause unintended clicks if the glider appears in the split second you click in the tree's white space.

When the glider will appear: Initial Delay... Specify a value in ms. The default is 200 ms (valid values are 1 to 1000). The glider will not appear as long as the mouse is moving, it will only appear when and where the mouse rests for the number of ms you define here. This allows you to move the mouse in the hover zone without having the glider appear all the time.

Button Size

Choose a size: Small, Large, Extra Large.

Buttons per Row

Select how many buttons to display per row. Tip: Select "1" to have a vertical glider, "4" to have a horizontal glider.

Select Buttons...

Choose which buttons to show, and in which order (drag them, or move them by Ctrl+Up/Down). Currently four buttons are available:

- Copy selected items here: This button is disabled when nothing is selected in the file list pane.
- Move selected items here: This button is disabled when nothing is selected in the file list pane.
- Paste here: This button is disabled if there is nothing pastable in the clipboard.
- Run script here: See below for how to define the script.

Select Color...

Select a color for the glider.

Edit Script...

In the script, the glider path (i.e. the folder the glider points to) can be referenced with the `<g_path>` variable (returns the path without the trailing slash).

Tip: You can use multi-scripts to pop a menu, for example:

```
"Copy Path" copytext <g_path>;
"Open in New Tab" tab("new", <g_path>);
```

See Scripting for how to script, and the Scripting Commands Reference for all the available commands.

Using the Glider

If you're about to use the Glider to move a bunch of files into different tree folders, your task will be streamlined if you show just that one Glider button (Move selected items here) and make it appear right under the mouse pointer (Snap to first button).

4.15 List

A New Kind of List

XYplorer's List has some features that might be new to you.

Columns headers have a context menu

Here the columns can be individually shown/hidden, and many other things, depending on the type of the right-clicked column.

Columns can be reordered by dragging

List columns can be mouse-dragged to the position you prefer: hold the mouse down on a column header and drag to the new position.

Show line numbers

List row numbers are displayed when you check the corresponding entry in Tools | Customize List.

Displaying thumbnails

There are various "views" (i.e. List display styles) featuring thumbnails. See menu View | Views, or the Views button on the Toolbar.

In any of those views the thumbnails have the "Mouse Down Blow Up" functionality you already know and love from the Image Preview:

Long* Left-click on thumb	Mouse Down Blow Up, while mouse button is down.
Right-click on thumb	Mouse Down Blow Up, stay up until you hit any key or click it again.

* Note: hold the mouse down longer than 150ms. Quick clicks will simply select the item.

Further details see Configuration | Thumbnails.

"desktop.ini" and other invisible files

What you see in XYplorer is the raw files that are actually there. So, in some folders XYplorer shows files that are not visible in the standard Explorer, for example files called desktop.ini: DO NOT DELETE THEM! You could destroy the functionality of a special folder. Remember: not knowing what a file is good for is the worst reason for destroying it.

Shortcut files

Shortcut files are displayed with their extensions (*.lnk, *.pif, *.url), whereas Explorer does not show those extensions.

Tip: A middle-click on LNK files behaves like Shift+Dbl-Click on LNK files: The link target is opened in a new tab.

Drag and Drop within file list

You can easily make a safety copy of a file (or more files) by drag-and-dropping it from the file list into

the file list: select the file(s) and then hold down the Ctrl-key and drag. The copy will be automatically renamed, eg.: the copy of Myfile.txt is named Copy of Myfile.txt.

You can as well drag and drop items onto a folder displayed within the same list. XYplorer also supports dropping items on executables or ZIP-archives.

Type Ahead Find

The list supports Type Ahead Find (aka "Find as you type"): When the focus is on the list, you can simply type a letter, say E, and the next file starting with letter "e" will be selected. You can also type a sequence of letters (quickly one after the other as in normal typing), e.g. V-A-C, and the next file starting with "vac" will be selected.

Pressing ESC will reset the internal keystroke store. So quickly typing K-E-Esc-Y will match "Y", not "KEY". Also pressing any arrow keys, or changing the listed folder will reset the internal keystroke store.

Type-ahead find delays the actual find while the list is still loading, and jumps to the matching item once the load is complete. This means you can happily type in your pattern while the desired item is not yet contained in the list and it still will be found.

Matching

"Match at beginning" of filenames is the factory default (and the only thing Windows Explorer can do), but XYplorer gives you four different matching options, selectable in *Configuration | Filters & Type Ahead Find | Type Ahead Find*.

Advanced Usage

There are two keyboard-only commands to jump to the next/previous match: *Tools* | *Customize Keyboard Shortcuts* | *Miscellaneous* | *Focus Functions*: "Next Type Ahead Match" and "Previous Type Ahead Match". They repeat the last used Type Ahead Find pattern in forward or backward direction. Usage: Serves nicely to quickly repeat a multi-character Type Ahead Find pattern.

The last used Type Ahead Find pattern is now stored between sessions. It's one global last pattern for all panes, tabs, folders, lists, etc. Usage: In connection with the Next/Previous commands mentioned above this can be extremely useful when you need quick jumping to files of a certain name in different locations.

Use Sorted Column

XYplorer also supports Type Ahead Find for other columns than the Name column. Check Configuration | Filters & Type Ahead Find | Type Ahead Find | Use sorted column and Type Ahead Find will use the columns Name, Ext, Type, or Path if they are the sorted columns.

Live Filtering

Tick Configuration | Filters & Type Ahead Find | Type Ahead Find | Redirect typing to Live Filter Box to enable Filter-As-You-Type right in the file list.

Empty List Message

You get a message when the list is empty. There is some additional information if items are present in the folder but currently not shown.

Cell Context Menu

In the Details view (including "Details with Thumbnails"), if you hold down Ctrl and right-click a non-blank cell, you will get a small pop-up menu of commands useful for that particular cell, the "Cell Context Menu".

Tip 1: Instead of holding CTRL you can hold the left mouse button down and then right-click simultaneously, aka rocker-click gesture.

Tip 2: If you untick Configuration | General | Menus, Mouse, Usability | Context Menus | Hold Ctrl to show cell context menu a right-click alone (without Ctrl) will be sufficient. In that case you need to hold Ctrl to pop the default context menu of the list.

Live Filter via Cell Context Menu

The Live Filter command allows you to filter the list by the contents of the cell.

Tip: You can modify by keyboard what happens when you click "Live Filter".

```
          Click => Filter list by the selected property (replace any current filter).
     Ctrl+Click => Boolean AND the selected property to the current filter (if any).
   Shift+Click => Boolean OR the selected property to the current filter (if any).
```

No smartness built in. It just appends " & [selected property]" or " | [selected property]" to whatever is in the box.

Quick Search via Cell Context Menu

The Quick Search command lets you search the current location for the contents of the cell.

Selection Filter via Cell Context Menu

The Selection Filter command allows you to select all items with the same value in this column.

Tip: Hold down CTRL (SHIFT) when you click Selection Filter to add to (remove from) the current selections.

Pro tip: Hold down CTRL+SHIFT to toggle the selection state of matching items.

Copy Data via Cell Context Menu

The command Copy Data lets you copy the contents of the cell to the clipboard. The feature also works on the text area in Tiles views.

Tip: You can hold down the Shift key while clicking Copy Data to copy the data of all selected items. The return format is one cell per row (CRLF).

Copy File Name: If the clicked cell is in the Name column the popup menu shows various components of the file name: full path, special path (if there is any), title, base, and quoted variants:

```
Full Path     C:\Users\Donald\Desktop\Thumbs.db
Special Path  Desktop\Thumbs.db
File Title    Thumbs.db
Base Name     Thumbs
-
Full Path Quoted     "C:\Users\Donald\Desktop\Thumbs.db"
File Title Quoted    "Thumbs.db"
```

Copy File Size: If the clicked cell is in the Size column the popup menu shows a number of common size display formats (RAW, Bytes, KB, MB, GB) of the file's size, ready to be copied to the clipboard.

Copy File Dates: If the clicked cell is in one of the Date columns the popup menu shows the date in various formats:

```
System Long Date  + Long Time
System Short Date + Long Time
System Short Date + Short Time
ISO 8601
-
Age
```

Tip: Note that you can copy the data without even selecting the items.

Time-Stamping via Cell Context Menu
Time-Stamping can be done right in the list, through the Cell Context Menu of any cell in any date column.

Touch: Find the Touch command in the bottom section of the menu: Touch Created Date, Touch Modified Date, Touch Accessed Date. Use it to set the file's timestamp to now.

Set: If a valid date string happens to be in the clipboard then you can paste-stamp that date onto any file date in any date column. The menu will feature the command Set [Created/Modified/Accessed] Date to [Date].

Edit: Find the Edit command at the bottom of the menu: Edit Created Date..., Edit Modified Date..., Edit Accessed Date....

Further Remarks:

- The Time-Stamping commands apply to all selected items if the right-clicked item is one of the selected items. Otherwise they apply only to the right-clicked item.
- Time-stamping a file ensures that the archive bit is set in that file.
- Time-Stamping commands support fractions of a second, and times in UTC (date with suffixed "Z").
- Set the date to nothing to set a file time to now.
- You can use the pseudo date 0 (zero) to set a file time to the lowest possible value.

Display Modes

The List supports 5 different modes: Browse, Find, Drives, Network, and Recycler. Each mode has its own column headers and type of entries. Each mode can be configured individually regarding column size, column position, background color, sort order and sorted column.

Browse

The standard browse mode (as known from Explorer). The List now displays all items contained in the currently selected folder.

Find

Invoked by a File Find and lists the search results of a find. By default the background color of this mode is green to make it easily distinguishable from the other modes.

Once you selected any folders in Tree, the mode is set back to Browse (or Drives, when you select the Computer Icon in Tree).

Drives

When you select the Computer Icon in Tree, the Drives mode is invoked. The List now displays various information about your local and mapped logical hard drives.

Network (My Network Places)

This mode with only 2 columns (Name and Comments) is shown when you selected Network Neighborhood or any Server (network machine).

Network Places On Demand. You can browse to servers outside the local/primary workgroup/domain. Simply enter them (or items located on them) into the Address Bar, or browse to them via Catalog or Favorites. If reachable they will be automatically/on-the-fly added to the Network (My Network Places) node in the tree.

Network servers are remembered between sessions. This means: (a) Network servers load at the speed of light, and (b) Servers outside the primary workgroup that have been entered via Address Bar are now remembered and don't have to be entered again.

To refresh this server cache select My Network Places in the tree and press F4 or F5. This will re-browse the network and list all servers in the primary workgroup that are currently reachable.

Recycler

The Recycle Bin with some special columns like Deleted (Date), Original Location, and Recycled Name .

Columns in the File List

The following columns can be displayed in the list in Browse and Find mode (the last one, Path, only in Find mode because in Browse mode it would be redundant):

#: Line number.

Index: This column displays the original index of each item as it was added to the list. So sorting the list by this column will recreate the original order of the items (you can achieve the same by clicking

View | Sort By | Unsorted).

Tip: This column is especially useful in Paper Folders which have been manually sorted. It offers a quick and transparent way to recreate the custom sort order.

Path: Path of the file (by default only shown in Search Results and Branch View).

Tip 1: Double-click a cell in this column to go to that path (and auto-select the item).

Tip 2: In column header special context menu (hold down the CTRL while right-clicking on the Path column header) you find the toggle Truncate Paths from Beginning. Useful since, typically, a path specification has a redundancy gradient from beginning to end.

Name: File or folder name.

Ext: File extension (txt, doc, jpg...).

Size: File size (and folder size, if Configuration | General | Tree and List | List | Always show folder sizes or Show Folder Sizes is enabled). Can show graphical representations of the items size (circles, bars). The Size column always shows a tooltip (even if the text is not cropped) with the exact byte count of the hovered item.

Type: File type (Application, File Folder, Shortcut...).

Modified: Last modified date. When you hover any cell in any date column the age of that item is displayed in a tooltip.

Created: Created date.

Accessed: Last accessed date.

Attr: File attributes, in the following order:

R = READONLY

H = HIDDEN

S = SYSTEM

D = DIRECTORY

A = ARCHIVE

N = NORMAL

T = TEMPORARY

L = REPARSE_POINT (=Junction)

C = COMPRESSED

O = OFFLINE

I = NOT_CONTENT_INDEXED

E = ENCRYPTED

P = PINNED

U = UNPINNED

J = RECALL (RECALL_ON_DATA_ACCESS)

M = INTEGRITY_STREAM

V = VIRTUAL

X = NO_SCRUB_DATA

K = RECALL_ON_OPEN

Tip: You can toggle the display style (go back to the old DOS Attributes style) by Ctrl+Right-clicking any item in the Attr column. Then click "Show Attributes in DOS Style".

Status: Displays icons that indicate the availability of your cloud-based (OneDrive) items. Supported on Windows 10 and later. The icons have tooltips that tell you what they mean.

Icon	Meaning	Internal Number
Blue cloud	Available when online	1
Green on white check	Available on this device	2
White on green check	Always available on this device	3
Blue arrows	Sync pending (files)	4
Blue arrows	Sync pending (directories)	10
Blue arrows	Syncing	6
White X on red	Error	7
Grey nope sign	Excluded (not synched)	9

The internal number is used when you search or filter by availability status, e.g. `Status:2` to match all items that are "Available on this device".

Note: The Status column only shows data in locations that are connected to the cloud (OneDrive).

Tip: Ctrl+Right-Click the Status column header to get a Refresh Column command (rocker-click supported). Of course, a simple F5 will also refresh the column (and the whole list).

Len: Length of the file name (including the full path, but see tip below) in characters. Note that for performance reasons the "Len" column will only be filled with data when visible.

Tip: In column header special context menu (hold down the CTRL while right-clicking on the Len column header) you find the toggle Show Length without Path. Tick it to display the length of the file name alone, without the path. The setting works globally for all Len columns in all tabs and panes. It does NOT affect Live Filters, Visual Filters, Color Filters, Quick Search and Find Files. They all have a dedicated selector "LenT" for the length of the file name alone.

Label: Label of the item (only if item has been tagged).

Tags: Tags of the item (only if item has been tagged).

Comment: Comment of the item (only if item has been tagged).

Extra 1 - 16: Extra tags. You can define the columns as well as their contents. See Extra Columns.

Custom 1 - 10: Custom columns. See Custom Columns.

By the way, if you show Special Property | Length in a Custom Column: The context menu of the Length column header (column shows media length, aka duration) offers the toggles Show Hours and Show Milliseconds. Lets you format the length display the way you like it.

Hint: When the sorted column is "dirty" i.e. the sort order is probably not perfect (e.g. because a new folder has been added to the bottom of the list), the little sorting symbol changes its color (to color "Marked Text 1") to show the dirtiness.

Secondary Sorting

There's secondary sorting, aka multi-column sort. Simply hold SHIFT while you click on another column header to secondary-sort by this column. Of course, secondary sorting makes only sense where the primary sorting results in groups. This typically is the case with the following columns: Path, Ext, Type, and Attr; less so with Size and Dates. The column that's sorted secondarily is marked by a smaller sorting icon.

Day Groups: When the primary sorting is by date (e.g. Modified), the secondary sorting will ignore the time parts of the file dates when looking for groups of files that should be secondary-sorted. In other words you can sort the files by the day they have been modified, and then sub sort each day by Name, Type, or Size.

Column Width by Keyboard

When the mouse pointer is over a column header you can resize that column by Ctrl+Left and Ctrl+Right. The exact new width is shown in the status bar. Works also in non-Details views if column headers are visible.

Context menus

A right-mouse-click on List will popup one of three context menus depending on the mouse position:

On filename	File context menu (incl. Explorer's standard context menu for folders) Customize the context menu in Configuration | Menus, Mouse, Usability | Custom items in the context menu | File List...
On column headers	Show Columns context menu
On Label column	(If showing tags is enabled) Label selection context menu
On Tags column	(If showing tags is enabled) Tags selection context menu
On Comment column	(If showing tags is enabled) Comment context menu
Elsewhere	Edit context menu

The context menu of Shortcuts has a special submenu Shortcut Target with these extra commands:

Go to Shortcut Target	
Copy Shortcut Target Item	(applies to all selected shortcuts)
Copy Shortcut Target Name	(applies to all selected shortcuts)

The context menu of Junctions has a special submenu Junction with these extra commands:

Go to Junction Target
Copy Junction Target Name
Delete Junction

Jump and Spot

Jump and Spot

A highlighting feature inspired by the Google Toolbar and the Firefox Find Bar. It has no separate GUI but can be invoked via the Address Bar (or any of the other location interfaces, e.g. Favorites, Catalog, Goto, etc.). Simply type in a string prefixed with ">" press Enter. The string will be highlighted in the file list wherever it's found (case-insensitive: a=A) and focus will jump to the next matching name.

For example, to highlight all strings "xy" and jump to the next item containing "xy":

```
>xy    [Enter]
```

To turn it off again, enter ">" without any pattern (or simply go to another folder):

```
>     [Enter]
```

If you are not interested in jumping and just want the colors, put ">>" at the beginning. If you use more than one pattern at a time (separated by |) no jumping takes place:

```
>>xy        = color all occurrences of "xy"
>>xy|2009   = color all occurrences of "xy" and "2009"
>xy|2009    = (same as above)
```

To turn it off again, either repeat the same pattern, or enter ">" or ">>" without any pattern:

```
>>xy     [Enter]
>>     [Enter]
>     [Enter]
```

Spot supports an unlimited number of strings, separated by | (pipe). Four predefined colors are rotated. For example (the forth item is a space):

```
>7.|xy|2008| |-|~
```

For easier typing also spaces work as alternative pattern separators:

```
>chuck berry    = spot all "chuck" and all "berry"
>chuck|berry    = (same)
```

The | separator only has to be used when spaces are part of a pattern:

```
>chuck berry|   = spot all "chuck berry"
```

The highlighting is not retained across different listings: refreshing the list or browsing to the next location will remove the colors.

Advanced

Wildcards are fully supported, and you can store an unlimited number of jump links in your Catalog or Favorites. Next time you want to listen to your Chuck Berry collection buried in a list of 20,000 mp3s -- just define a jump link once and the rest is a single click that takes you from one Chuck Berry to the next.

Examples:

```
>chuck   = jump to the next List item containing "chuck"
           anywhere in the name
>????    = jump to the next List item that's 4 chars long
>*##*    = jump to the next List item with at least
           2 consecutive numbers in the name
>*.txt   = jump to the next item ending with ".txt"
>        = remove any coloring now
```

Columns

You can optionally prefix a column caption to the pattern. The column prefix can be partial. If no column is identical to the prefix, then the first column with a partial match (from beginning: "Modified" matches "mod*") is used.

Examples:

```
>>mod: 2015            spot (highlight) all items where the Modified column matches
"*2015*"
>size: 0*              jump to the next item where the Size column starts with "0"
>size: *KB             jump to the next item where the Size column ends with "KB"
```

Tip: Jump links are easily added to the Catalog and thus turned into a one-click affair.

Tip: You can jump upwards by holding Shift.

Tip: Check Configuration | Filters & Type Ahead Find | Type Ahead Find | Ignore diacritics to ignore diacritics (o==ö) when matching the patterns.

Quick Select

Syntactically similar to Jump and Spot, Quick Select is a fast and simple way to select items directly through the Address Bar. So the patterns can be easily shared, are stored in the Address Bar MRU, and are accessible by scripting.

The magic prefix here is ">>>" (without the quotes). Multiple patterns separated by | are allowed.

Examples:

```
>>>Beyoncè      = select all items matching *Beyoncè*
>>>ko|cha       = select all items matching *ko* or *cha*
>>>ko*|*.txt    = select all items matching ko* or *.txt
>>>             = unselect all
```

You can optionally prefix a column caption to the pattern.

Examples:

```
>>>modified: 2015       matches all items where the Modified column matches "*2015*"
>>>SIZE: 0*             matches all items where the Size column starts with "0"
>>>size: 0*|1*          matches all items where the Size column starts with "0" or "1"
```

The column prefix can be partial. If no column is identical to the prefix, then the first column with a partial match (from beginning: "Modified" matches "mod*") is used.

Examples:

```
>>>MOD: 2015            matches all items where the Modified column matches "*2015*"
>>>siz: 0*              matches all items where the Size column starts with "0"
>>>siz: 0*|1*           matches all items where the Size column starts with "0" or "1"
```

Remarks:

- This setting is honored by Quick Select: "Configuration | Filters & Type Ahead Find | Type Ahead Find | Ignore diacritics".
- The other settings under "Configuration | Filters & Type Ahead Find | Type Ahead Find" are ignored.

4.16 Hover Box

Hover Box

The Hover Box is a pop-up window that appears when you hover over a file name in various parts of the interface. It shows a small preview of the file or folder and some basic item information.

Conditions for where and when the hover box appears can be set in Configuration | Information | File Info Tips & Hover Box.

How to open the Hover Box

- Move mouse over the file or folder (icon or caption, depending on settings and list view).

How to close the Hover Box

- Move mouse out of the file.
- Move mouse over another file (pops a new box).
- Press ESC.

Scrolling the Hover Box

- A subtle visual scrollbar at the right margin of the box tells you whether you can scroll, where about in the file you currently are, and how large the visible part is relative to the whole contents.
- You can keyboard-scroll text or folder contents by Down, Up, PageDown, PageUp, End, Home.
- You can wheel-scroll text or folder contents.
- If the text contents are larger than 64KB you can scroll through the first and the last 32KB of the contents.

Scaling the Hover Box

- The mouse wheel can scale images and PDFs. Up = bigger, Down = smaller (8 pixels per notch). Same functionality as Numpad Add and Subtract.
- Hold SHIFT to increase the effect by 8 times: 64 pixels per notch.
- Hold CTRL to decrease the effect by 8 times: 1 pixel per notch.

Moving the Hover Box (aka Mobile Hover Box)

Once the Hover Box is showing you can move it to another item using the navigation keys (Left, Right,

Up, Down, PageUp, PageDown, Home, End).

The box is only updated on key up. This way you can use fast key repetition to move through a list without being hindered by preview work.

Tip 1: You can press SPACE while the Hover Box is showing to select the currently hovered item, Ctrl+SPACE to toggle the current item's selection state.

Tip 2: The keyboard-shortcut-only command Miscellaneous | Focus Functions | Trigger Mouse Move triggers a fake mouse move at the current mouse position. Allows you, for example, to open a Hover Box or File Info Tip using the keyboard if the current mouse position is suitable.

Select Item Types...: Click the button to select the file types the Hover Box shall be shown for. These file types are supported (in parentheses the type of preview):

- Text Files (Text)
- Document Files (extracted Text or Thumbnail (PDF))
- Web Files (Text)
- Font Files (Thumbnail)
- Image Files (Thumbnail)
- Video Files (Thumbnail)
- Archive Files (see below Archive Contents Preview)
- Folders (see below Folder Contents Preview)

Note: What works and what not depends on the Preview Handlers and IFilters that are installed on your system. Exceptions: Archive Files and Folders don't need any Preview Handlers or IFilters to be previewed.

Archive Contents Preview

The Archive Contents Preview lists the top 20 items of the hovered archive file.

- Items are sorted by name ascending, folders on top.
- Of each item you see the icon (CFI honored) and the name. Note that of file inside a Zip archive only the generic icons can be shown, so e.g. all EXE files will have the same icon (unless overwritten by CFI rules).
- In the bottom area of the Hover Box you get the ZIP archive name and basic stats (item count, folder count, file count, total bytes of files).
- The stats don't include the contents of any subfolders, but only the top level of that ZIP archive, i.e. what you would see in the preview of that ZIP archive.
- Shortcuts (LNK) to ZIP archives are resolved and you get the preview of the target ZIP archive.
- Out of the box only the ZIP format is supported. But see here below for other archive formats.

Beyond ZIP: For archive formats other than ZIP the help of other applications is needed.

- UnRAR.exe: Will provide previews of the RAR format.
  XYplorer tries to find the needed helper file "UnRAR.exe" automatically in the usual locations (C:\Program Files\WinRAR\UnRAR.exe or C:\Program Files (x86)\WinRAR\UnRAR.exe). So if you are reasonably lucky it should just work. If the preview fails you can tweak the location by editing the XYplorer.ini file: `ZipPathUnRAR=C:\My Crazy Path To WinRAR\UnRAR.exe`
- 7z.exe: Will provide previews of the 7z format (and many other formats: *.001; *.7z; *.arj; *.bz; *.bz2; *.bzip2; *.cab; *.deb; *.gz; *.gzip; *.img; *.iso; *.lha; *.lzh; *.ova; *.rar; *.rpm; *.tar; *.taz; *.tbz; *.tbz2; *.tgz; *.tz; *.vhd; *.wim; *.wmz; *.wsz; *.xz; *.yfs; *.z; *.zip; *.zipx)
  XYplorer tries to find the needed helper file "7z.exe" automatically in the usual locations (C:\Program Files\7-Zip\7z.exe or C:\Program Files (x86)\7-Zip\7z.exe). So if you are reasonably lucky it should just work. If the preview fails you can tweak the location by editing the XYplorer.ini file: `ZipPath7zip=C:\My Crazy Path To 7-Zip\7z.exe`

Folder Contents Preview

The Folder Contents Preview lists the top 20 items of the hovered folder. In the bottom area of the Hover Box you get the folder name and basic stats (item count, folder count, file count, total bytes of files). Shortcuts (LNK) to folders are resolved and you get the preview of the target folder. It does*honor the general visibility settings that also control the file list. So, when an item is hidden from the list then it's also hidden from the Folder Contents Preview. In the unlikely case that all items are hidden, you won't see "Folder is empty." but "[n] items are hidden." If a folder has more than fits in the Hover Box you get an arrow at the bottom right showing that there's more.

Select Context...: Here you can select the context where the Hover Box shall be shown. Currently available:

- Tree
- List
- Tiles and Thumbnails (untick it if you don't want to see the Hover Box in those list views)
- Folder Contents Preview
- Clipboard (the Hover Box is shown over the "Edit Clipboard" toolbar button when there is an image in the Clipboard)
- Tabs (the Folder Contents Preview is shown for the tab's location by hovering the tab header icon)
- Portable Devices (the preview often doesn't work on portable devices and the unsuccessful attempt takes a long time; untick this to save time)
- Breadcrumb Bar Drop-Down Menus (the Folder Contents Preview is shown for each item in the down-down menus when the icon is hovered). Works only for "Custom Menu" and "Colored Menu". If unchecked you still can force the Hover Box by holding CTRL.
- Image Columns (the Hover Box is shown over images in Extra Columns of type "Image", or Custom Columns of format "Image")

Hover Box Properties and Keyboard Tricks

Hover Box Size: You can scale the size of the image shown in the Hover Box. While the Hover Box is visible showing an image press these keys:

Numpad Add	= Increase size by 8 pixels
Numpad Subtract	= Decrease size by 8 pixels
Ctrl+Numpad Add	= Increase size by 1 pixel
Ctrl+Numpad Subtract	= Decrease size by 1 pixel
Shift+Numpad Add	= Increase size by 64 pixels
Shift+Numpad Subtract	= Decrease size by 64 pixels

- An alternative to Numpad Add is Wheel Up, an alternative to Numpad Subtract is Wheel Down.
- The size defines both sides of the bounding rectangle, so it's a square.
- The initial factory default is 512 x 512 pixels (on 100% screen resolution).
- The minimum allowed size is 64 x 64 pixels.
- The maximum allowed size is 80% of the workspace height.
- The Status Bar informs you about the size you have set.
- The size (and all other Hover Box properties here below) is remembered across sessions.

Hover Box Hotkeys

F1 - Show Hover Box Help: While the Hover Box is showing you can press the "F1" key to show a list of all the available special keys listed here above.

B - Toggle Background Color: You can toggle the background color between dark grey and white: Simply press key "B" while the Hover Box is visible and showing an image.

C - Copy Contents: You can copy an image of the current Hover Box contents to the clipboard by pressing the "C" key while the Hover Box is visible. Note that not only images, but also text, PDF previews, archive listings, folder listings, etc. can be copied to the clipboard this way.

D - Reset to Default Size: While the Hover Box is showing an image you can reset the scale to factory default (512 x 512 pixels bounding box) by pressing the "D" key.

F - Cycle Frame Width: You can cycle the width of the image frame between three values: 0, 12, 24 pixels. Simply press key "F" while the Hover Box is visible.

G - Toggle Grayscale: You can have a grayscale version of the hovered images in the Hover Box by pressing the "G" key while the Hover Box is showing an image.

H - Toggle Path Display in Status Area: If the full path is shown in the status area, you can choose whether to show the real path (e.g. "C:\Users\Donald\Desktop") or the special path if there is any (e.g. "Desktop"). Press the "H" key to toggle the setting.

I - Toggle Icon in Status Area: You can toggle whether the icon (Custom File Icons are honored) of the hovered item is shown in the status section of the Hover Box: Simply press key "I" while the Hover Box is visible.

K - Toggle Key Scrolling: While the Hover Box is showing text you can toggle the scrollability with keys by pressing the "K".

L - Toggle Scrolling: While the Hover Box is showing text you can toggle the scrollability by pressing the "L". Hover Boxes can be scrolled by wheel and (unless Hover Box Key Scrolling is disabled) by the navigation keys (Up, Down, PageUp, PageDown, Home, End). Scrollable Hover Boxes show either the textual contents of files, or the items contained in folders.

O - Cycle Sort Order: For a folder contents preview you can cycle the sort order by pressing key "O " (letter o) while the Hover Box is visible.

P - Toggle Photo Data Display: While the Hover Box is showing an image you now can toggle whether photo data are shown in the Hover Box status by pressing the "P" key.

R - Toggle Resizing: While the Hover Box is showing an image you can toggle the scalability by pressing the "R". Hover Boxes can be resized by wheel and by Numpad Add and Numpad Subtract.

S - Toggle Status Area: You can cycle the amount of information shown in the status section of the Hover Box (Full Status with Full Path, Full Status with Name Only, Name Only, No Status): Simply press key "S" while the Hover Box is visible.

T - Cycle Transparency Background: Cycle the background of images with transparency: White, Black, Neutral, Transparent.

U - Toggle sharp/rounded corners: While the hover box is displaying an image, you can press "U" to toggle between rounded and sharp corners. If necessary, the frame will be adjusted internally to make it look good.

W - Toggle Word Wrap: While the Hover Box is showing text you can toggle word wrap by pressing the "W" key. When viewing text content, the status area now shows a "Leftwards Arrow with Hook" (U+21A9) in the right bottom corner when word wrap is enabled.

X - Trigger hotkeys with/without CTRL: Force holding down CTRL to trigger one of the one-letter hotkeys while the Hover Box is displayed. Note that when CTRL is forced, the letters A-Z just go through the hover box to wherever the focus is.

Z - Toggle Zoom To Fit: While the Hover Box is showing an image you can toggle Zoom to Fit (enlarged display of smaller images) by pressing the "Z" key.

Selecting Items while the Hover Box is showing: Press SPACE to select the item, Ctrl+SPACE to toggle-select the item, Shift+SPACE to pivot-select the item.

More about the Hover Box

- Only files that are physically present on the local machine will display a hover box. No cloud files where the hover box would trigger an on-demand download.

4.17 Custom Columns

Custom Columns

With Custom Columns it's you who decides which information about each file is displayed in the column.

You show the columns just like the other regular columns: Via menu View, or via right-clicking the currently shown column headers (hold CTRL with Size and Date columns, else they show their special context menu), or the zone right of the column headers.

Tip: A couple of Custom Columns are predefined when you first start XYplorer. This is just for illustration and convenience. You can freely modify them.

Configuration

Custom Columns can be configured in *Configuration* | *Custom Columns* | *Custom column definitions*. You may define up to 64 Custom Columns. The first 10 definitions are "global" in the sense that they are available as columns "Custom 1" to "Custom 10" in each file list on any tab and pane. The other definitions can be used in so-called "Soft Columns" that can be added on-the-fly to any particular file list.

Tip: Any Custom Columns that is currently shown in the list can be configured via right-clicking the column header.

The "Configure Custom Column" dialog

Caption

User-defined caption of the column. If you leave it empty a default caption will be shown depending on the type of the column.

Type

Type of the column. Controls how the definition (next field) is interpreted. The following Types are available:

- Property: Any of the so-called extended file properties that are made available by Windows for every file. Most of them only return data for certain file type.
- Special Property: Any of XYplorer's Special Properties, that's stuff not included in the Windows extended file properties. E.g. Aspect Ratio, MD5 hash (and many other hash functions), Number of Hard Links, Junction Target, various audio tags (MP3, FLAC, OGG), and more.
  Note that a column showing the Aspect Ratio will be sorted by the actual quotient (eg 0.5), not by the term of the quotient (eg "1:2").

Note that a column showing the Dimensions will be sorted by the area (eg 300 x 200 = 60000), not by the term of the Dimensions (eg "300 x 200").

- Template: The template is a text with variables.
  - Using the `<prop ...>` variable you can add more than one property to a column.
  - The template input (and hence output) may be multi-line.
  - Examples:
    `Owner: <prop #10>, Size: <prop #1>`
    `<prop #aspectratio>`
- Script: The definition is one script (single- or multi-line). Details see Scripted Columns below.
- Mixed: The column is defined as a set of other Custom Columns. If those other Custom Columns have non-overlapping Filters defined you can make a context-sensitive column this way which will display data in dependence of the file type. You could e.g. show pixel dimensions for images, play time for audio files, page count for documents, and the version number for executables -- all in ONE column! Details see Mixed Columns below.

Format

The format controls display and sorting of the column. In the following three of the available formats are discussed: Text, Icon, and Image.

- Text: This will be the best setting in almost all cases.
- Icon: If this format is selected then the column contents are taken to specify an icon to be drawn.
  - Icons sources can be ICO files, other files (their system icon is taken), or XYplorer native toolbar graphics.
  - Relative file paths are resolved to the "<xydata>\Icons" folder.
  - Toolbar icons are marked by a prefixed colon, e.g. ":go" (as elsewhere in the application).
  - Extracted icons from icon resources (exe; dll; cpl; ocx; scr; icl; bpl; wlx; wfx; wcx; wdx; acm). For example: "%winsysdir%\shell32.dll /14"
  Example for a script (Column type: Script), where different icons are shown depending on the length of the filename base (nutty idea, just for demo purposes). The icon specification is done using the "return" command:

```
$base = getpathcomponent(<cc_name>, "base");
if (strlen($base) <= 8) {
  return ":go";
} elseif (strlen($base) <= 12) {
  return "%winsysdir%\shell32.dll /14
} elseif (strlen($base) <= 16) {
  return "SmileyHearts.ico";
} elseif (strlen($base) <= 32) {
  return "D:\Temp\Acorn.ico";
} else {
  return <xy>;
```

```
}
```

This script would show a Nikon icon for all photos that were shot with a Nikon:

```
$camera = property("#image.cameramodel", <cc_name>);
if (strpos($camera, "Nikon") != -1) {
  return "Nikon.ico";
}
```

Lists of Icons: You can as well specify a list of icons that are shown side by side. You can mix system icons with toolbar graphics. Icons can be separated by ; or | or CRLF. The distance between the icons is hard-coded to 2 pixels. In Extra Tags of type "Icon", since | is not allowed in tags, you have to use ";" as separator.

Drawing Circles: Apart from showing icons the Icon format also lets you draw circles of your own design. The column has to be of type "Script" and format "Icon", and instead of the icon specification you return a draw command with the following syntax:

```
>draw.circle [diameter=smalliconsize-2], [color=textcolor], [opacity=255], [borderwidth=0], [offsetx=0]
```

So there is a command name "draw.circle" prefixed by ">" (to avoid any ambiguities with possible filenames), then there are 5 arguments, all optional with default values. If you omit all arguments, the result will typically be a black circle of 14 pixels diameter.

- The Diameter is limited to the current row height of the list.
- Color is given in RRGGBB format.
- Opacity is from 1 (almost totally transparent) to 255 (fully opaque).
- If Borderwidth is > 0 then the circle is not filled but outlined.
- With OffsetX you can add an horizontal offset to the circle.
- More than one definition can be passed, separated by ; (semicolon) or CRLF (line feed).
- Works also with Extra Tags of type "Icon".

Example for a column script drawing various circles:

```
$base = getpathcomponent(<cc_name>, "base");
if (strlen($base) <= 8) {
  return ">draw.circle 12, 004444, 127, 4";
} elseif (strpos($base, "t") == 0) {
  return ">draw.circle 8, ff0000, 127, 3";
} elseif (strpos($base, "v") != -1) {
  return ">draw.circle; , 66cc55, 200, , 7";
} elseif (strpos($base, "ph") == 0) {
  return ">draw.circle 8, ff0000, 127; 8, ff0000, 127,, 10; 8, ff0000, 127,, 20";
```

```
} else {
  return ">draw.circle 18, C70102, 192; 12, FFFFFF; 6, 0380B8, 192";
}
```

So here we have a pretty nerdy life time killer with a creative touch.

Macro: If you have less time to kill you can as well use the macro `>draw.circle.size` which will draw blue-colored (khaki for folders) size-related circles in a hard-coded way and with maximum performance. It's totally easy to do, no scripting involved. Just add a Custom Column of type "Template", format "Icon", and enter ">draw.circle.size" in the template field. It's very fast since it does not run the scripting engine for each row.

Disadvantage: Since each cell contains the same data you cannot sort by this column. There is a trick though: Append the size right to the template like this:

`>draw.circle.size <cc_size>`

Now you can sort by the column. But ensure that "Configuration | Sort and Rename | Sort | Sort method" is set to "Natural", else sorting by numbers won't make you happy.

Drawing Bars: The Icon format also lets you draw column-wide 2-colored percentage bars of your own design. The column has to be of type "Script" and format "Icon", and instead of the icon specification you return a draw command with the following syntax:

`>draw.bar [percentage=50], [colorleft], [colorright], [opacity=224], [borderwidth=0]`

- The Percentage controls the relative size of left and right part.
- Color for left and right part is given in RRGGBB format.
- If a color argument is omitted that part is not drawn at all.
- If both color arguments are omitted then hard-coded default blues take over.
- Opacity is from 1 (almost totally transparent) to 255 (fully opaque).
- If Borderwidth is > 0 then the bar is not filled but outlined.
- Works also with Extra Tags of type "Icon".

Here is an example for a column script drawing various bars.

Macro: Just like with circles above there is also a macro for drawing bars: `>draw.bar.size`. It will draw a column-wide colored size-related bar according to a hard-coded color scheme:

grey: Empty
green: < 1KB
yellow: < 1MB
red: < 1GB
purple: 1GB or more

All notes at `>draw.circle.size` (see above) also apply here.

- Image: Via variables in the Template (see above under Type) you can define a column to show images in a systematic relation to each listed file. Perfect e.g. for waveform images whose filename matches the

wave filename (e.g. `<cc_path><cc_base>.png`), or album covers assigned to sound files. These images support the Hover Box. Tick "Image Columns" in Configuration | Information | File Info Tips & Hover Box | Show Hover Box | Select Context...

Trigger

Here you can select at which point the script or process that generates the cell data is actually triggered. You have three choices:

- Browse [Default]: Column is triggered when the list is filled with folder contents or search results.
- List: Column is triggered when the items are listed. Only the currently visible items are processed. This option will be much faster than "Browse" on long lists, but the scrolling will be not as smooth.
- Click: Column is triggered when the symbol that is shown as a placeholder is clicked (actually you can click anywhere in the cell). Maximum speed, also when scrolling. All data are only shown on demand.

Notes:

- When you right-click a column header and select Refresh Column then the whole column data will be generated/refreshed.
- Columns of type "Mixed" inherit the trigger per-cell from the chosen sub-column.
- When Custom Columns of type Script are triggered by Click and the script has no "return" command then the click symbol remains in the list and can be clicked again and again.

Item Type

Here you can limit the action to Files, Folders, or Files and Folders.

Item Filter

Here you can enter a semicolon-separated list of wildcard patterns. Only for items whose name (without path) matches at least one of the patterns the column data are retrieved. By setting those patterns wisely you can speed up browsing considerably.

Notes:

- Patterns without any wildcards are treated as "ending matches" ("*pattern"); that way you can simply enter a couple of extensions without caring about wildcards.
- To suppress auto-wildcarding you can quote the pattern.
- Matching is case-insensitive (A==a).
- You can logically invert filters by prefixing a "!".
- You may as well use generic file types, e.g. `{:Image}`. Also in combination with normal wildcard patterns.

Examples:

- `jpg` matches *.jpg
- `!jpg` matches all non-JPEGs
- `!*.*` matches all items without extension
- `jpg;png;gif` matches *.jpg and *.png and *.gif
- `"readme.txt"` matches only readme.txt (and Readme.txt)
- `{:Audio}` matches all audio formats
- `{:Audio};txt` matches all audio formats and TXT
- `!nef;{:Photo}` matches all photo formats but not NEF

Scripted Columns

Here you write a script (using XYplorer's scripting language) that generates the data displayed in each cell of the column. Note that such a column script is executed for each file in the file list! So you should take care that the script does not take too long.

The special command "return" is used to set the column data.

There are a couple of special variables just for this context by which you can refer to the item for which the data are being generated:

```
- <cc_item> = the item name with path; unslashed always
- <cc_item_ds> = the item name with path; directories slashed, files unslashed
- <cc_path> = the item's path
- <cc_name> = the item name (no path)
- <cc_base> = the item's base name (name without extension; for folders it's identical to
cc_name)
- <cc_ext> = the item's extension
- <cc_size> = the item's raw size (number of bytes); -1 when the size has not been determined
- <cc_isfolder> = the item's nature (1 = folder, 0 = file)
- <cc_isselected> = the selected state of the item: 1 = selected, 0 = not selected
- <cc_overlay> = the overlay index of the item's icon
- <cc_index> = the index (Index column) of the item
- <cc_cell> = the current textual content of the clicked cell; can be useful if "Refresh Cell
by Click" is enabled
- <cc_trigger> = when you click in a custom column cell, this tells you the type of click in
a bit field:
  1 = LeftClick, 4 = Shift, 8 = Ctrl, 32 = Alt; needs "Refresh Cell by Click" to be enabled
- <cc_caption> = the caption of the custom column in which the script is executed
```

Example 1:

```
// displays the first 12 characters of the content (!) of each file
return readfile(<cc_item> , , 12);
```

Example 2:

```
// fills the column with characters 4-8 of each item name
$a = substr(<cc_name>, 3, 5);
return $a;
```

Example 3:

```
// calculates the number of pixels in an image
$dims = property("#image.dimensions", <cc_item>);
$width = gettoken($dims, 1, " x ");
$height = gettoken($dims, 2, " x ");
return $width * $height;
```

Example 4:

```
// draws colored bars according to the file size
$size = "<cc_size>";
switch (true){
  case $size == 0:          return ''; // draw nothing
  case $size < 1024:        return '>draw.bar ' . 100 * $size \ 1024 . ', 66CC66,
BBEEBB';        //smaller 1 KB
  case $size < 1048576:     return '>draw.bar ' . 100 * $size \ 1048576 . ', FFBB00,
FFEE88';     //smaller 1 MB
  case $size < 1073741824:  return '>draw.bar ' . 100 * $size \ 1073741824 . ',
DD4444, FFAAAA';  //smaller 1 GB
  default:                  return '>draw.bar ' . 100 * $size \ 1099511627776 . ',
DD44AA, FFAADD';  //1 GB or more
}
```

NOTE: Integer division (\ operator) is used when calculating the percentage! This is important in locales like German where the decimal separator is a comma. Returning this comma would screw the parsing of the >draw.bar command. Integer division is a way to avoid this.

Example 5:

```
// shows the GPS data of a file and, when clicked, opens Google Maps at the location
// <cc_trigger> needs "Refresh Cell by Click" to be enabled
if (<cc_trigger>==1) {
```

```
  open("https://www.google.com/maps/place/<cc_cell>");
} else {
  return get("exif", "gps", <cc_item>);
}
```

Notes on Scripted Columns:

- These scripts cannot be stepped. You have to debug them in another context.
- You can as well define and call user functions right in the column script (don't expect speed wonders for such columns).
- Press ESC while a list is loading to stop processing any scripted columns (actually any Custom Columns).

Mixed Columns

A mixed column is sort of a meta column, defined by a list of other column definitions (which need to have been defined previously) which are simply referenced by their serial number. The order is relevant since the listed column definitions are worked from top to bottom and the first match wins (i.e. it will be used to fill the cell). The filename in question is matched against the filters defined in each column definition. If no filters are present the column will match any filename. So, to make sense, all column definitions (apart from the last one) joined in a Mixed Column should have filters defined.

Find Files by Custom Columns

Referring to the columns works just like with Find Files by Extra Tags. You can either state a canonic field prefix "cc1" to "cc64" which will work independently of the column captions. Or you state the column caption as prefix.

For example, find all items where the column called "Contents" returns "Hash":

```
contents:Hash
```

Or, find all items where the first Custom Column returns "100" and the second Custom Column returns "Blue".

```
:cc1:100 AND cc2:Blue
```

When columns use the localized default captions inherited from the Windows properties, also the search term must use the localized caption:

German Windows: `Abmessungen:500 x *`

English Windows: `Dimensions:500 x *`

This, as another example, will match all files with at most 4096 pixels if Custom Column 14 returns the pixels (width * height) of an image file:

```
cc14: <= 4096
```

If that column is called "Pixels" then this will achieve the same:

```
Pixels: <= 4096
```

Note that the column does not have to be currently visible for this to work.

Note: You can even search by columns that are not currently visible. The search results will not automatically show the column(s) by which you have searched.

Soft Columns

Additionally to the 10 global Custom Columns you can add any number of so-called "Soft Columns" to the file list. You add a Soft Column by using the command Add Column. You find it in menu *View | Columns*, and in the context menu of each column header.

Freshly added, a Soft Column is just named ""Right-click here..." and has no contents. Right-click the column header to pop a menu with configuration options:

- Select Property...: Select any of the Shell's properties for this column.
- Select Special Property...: Select any of XYplorer's native properties for this column. They are usually much faster than the Shell's properties.
- Select Custom Column...: Select any of the 64 Custom Column definitions for this column. This adds the ultimate column power to Soft Columns since, as you know, with Custom Columns anything is possible.
- Refresh Column: Refreshes the column contents. Can be useful for certain scripted columns which yield different returns on each run.
- Add Column: Insert a new soft column before the clicked column.
- Rename Column...: Here you can edit the name of the column. Leave it empty if you are happy with the default name, which is automatically given according to the selected property.
- Remove Column: Totally removes the column from the current list.
- Hide Column: Hides the column. You can show it again via View | Columns. It will survive across sessions even if hidden.
- Show Columns...: Pops a list of all column where you can show/hide them and modify their position.

The major differences to the hard Custom Columns:

- Soft Columns are stored with the tab, with the Home, with the Folder View Setting. They are local and self-sufficient, no external dependencies.
- You can change the contents of a Soft Column on-the-fly. The column will instantly update without reloading the whole list. Other tabs or other panes are not affected.
- Soft Columns are unlimited in number.
- Soft Columns can easily and totally be controlled (defined, loaded, unloaded) via scripting command SetColumns().

Find Files by Soft Columns

The columns are referenced by their caption as shown currently in the list.

```
Aspect Ratio:16:9 //find all images with Aspect Ratio 16:9
```

The column caption is not case-sensitive, this works as well:

```
aspect ratio:16:9 //find all images with Aspect Ratio 16:9
```

Wildcards are supported as well:

```
Asp*:16:9 //find all images with Aspect Ratio 16:9
*tio:16:9 //find all images with Aspect Ratio 16:9
```

Notes:

- First match wins, and, of course, there is an internal order of precedence: Only if no Custom Columns (definitions 1 to 64) are matched, Soft Columns are looked for. And they are scanned in their original order of being added (which is from left to right, unless you later repositioned them by dragging).
- If "Search results inherit current columns" is OFF and you search for Soft Columns, the settings will be internally and silently turned on for this particular search. Otherwise it won't work.

Custom Columns Tips

- You can easily store/load columns layouts using the Catalog. The Catalog's right-click menu has a command Insert as New Item(s) Here | Current List Columns.
- When you hold CTRL while clicking the headers you get a mini context menu with just basic Soft Column related commands. This trick enables you to pop "Add Column" for Size and Date columns (which have different popup menus when you don't hold CTRL).
- Custom Columns can be temporarily disabled by checking menu View | Columns | Skip Custom Columns. Note that you can still manually populate a custom column by clicking "Refresh Column" in the context menu of the column header.
  Tip: The toggle is also available in the context menu of the Custom Column headers.
- In the context menu of the Custom Column headers you find a toggle Refresh Cell by Click. If enabled, the cell content can be refreshed by left-clicking in the cell. It also sets the `<cc_trigger>` variable which can be used in the Custom Column scripts (see Example 5 above).

General Remarks on Custom Columns

- You pay a price for this feature: Retrieving and processing all this extra information takes time. The shell as used here is v-e-r-y slow compared to XYplorer's native functions, so browsing a list with Custom Columns using shell properties will be notably slower than browsing a list without such columns. Also running a script for each item in a list and for each column of type "Script" takes a lot of time.
- You can escape the processing of Custom Columns by pressing ESC. The cells where data retrieval was skipped will then show "[Skipped by ESC]". Can be useful when browsing large folders with complex Custom Columns on a slow machine.
- Only visible columns are filled with data.
- Defining columns with a script is geek stuff. Be careful! You can easily create nasty things here.

- Custom Column definitions 1 to 10 are global to all tabs on both panes.
- The column definitions are stored in the INI file.
- Find Files, Reporting, Folder View Settings, and Set/Go Home are fully supported.

4.18 Dual Pane

See Panes menu.

4.19 Sync Folders

Sync Folders

The command is found in menu Panes | Sync Folders. It opens a dialog that lets you synchronize the folder in the inactive pane (target folder) to the folder in the active pane (source folder). Afterwards the target folder will be identical to the source folder (depending on the configuration). The source folder (active pane) is not modified. In other words it's a one-way sync, aka mirror sync. This is achieved by the following sub operations:

- Items that are only in source are copied to target.
- Items that are in source and target are overwritten in target.
- Items that are only in target are deleted from target.

Configuration

Some properties can be configured (the Sync Folders configuration dialog is shown invariably before every sync operation), and are remembered between calls and between sessions:

Copy items from source to target: Tick it to have any items copied at all. Otherwise there will be only deletions in the target.

On name collisions: Decide how to handle name collisions.

- Ask
- Overwrite if newer
- Overwrite if different size or date
- Overwrite if different contents (The contents are compared by comparing the SHA-256 hash of each file.)
- Overwrite
- Skip

Exclude hidden files and folders: Does exactly what it says. "Hidden" here refers to the file attribute "HIDDEN". Note that items with the "HIDDEN" attribute that are among the selected items (or among the items explicitly specified as source items) are NOT excluded.

Delete items in target that have no matches in source: Tick it to get any deletions in target at all. Otherwise there will be only copying to the target.

Delete to recycle bin (if possible): Note that on UNC paths, on removable drives, and where the Recycle Bin is turned off by Windows settings all deletions will be permanent (not to Recycle Bin) even if

the checkmark is set!

Prompt before delete: Highly recommended.

Some properties are hard-coded internally

- Preserve all item dates: YES
- Sync caps: YES. All files and folders in the target are ensured to have the same capitalization as their matching source item names. If necessary items in the target are renamed without asking (e.g. TEST.TXT -> Test.txt).
- Skip junctions: YES
- Safe overwrite: YES
- Show progress dialog: YES
- Keep progress dialog open: YES. In the final progress dialog you can show a detailed report listing each file copied and each item deleted.

Preview Button

Use it to run the operation in preview mode. Nothing actually happens, you just get a detailed report of what would happen in real mode.

The Sync Folders dialog stays up when you run the Preview from there. No need to open it again to do the real job.

Note: The Preview Mode assumes that everything works as planned (items to be deleted *can* be deleted, items to be overwritten *can* be overwritten, items to be renamed *can* be renamed).

Further Remarks

- Sync Folders does not support Undo/Redo (just like Backup does not). Pretty obvious since overwriting is the core element of it and this cannot be undone.
- Sync Folders does not support Paper Folders.
- Sync Folders does not support Portable Devices.
- Read-only files to be deleted will be deleted without extra asking.
- Sync Folders can be backgrounded and queued (Configuration | File Operations | Background Processing | Enable background processing).
- BE CAREFUL! Sync Folders can really shake up the target folder.

4.20 Catalog

Showing/Hiding the Catalog

First Step: Turn it on! Right after a fresh install you won't see the Catalog. To show it, go to menu Window and select Show Catalog (or simply press Ctrl+F8). The (now still empty) Catalog will be show below the Tree. If you think there's not enough space for Tree and Catalog and Info Panel on top of each other then uncheck Wide Info Panel in menu Window | Arrangement: The Info Panel will move to the right to make space for Tree and Catalog.

What is it? A very cool alternative to the Tree!

The Tree is an image of your computer's file system. It shows you all what's there. But, most of the time all is just too much...

The Catalog is the answer: here you can grow your own personal tree. Your favorite locations are deep down in some heavily nested structures? Lift them to the surface! Side by side with locations from the other end of your hard disk. You can navigate by the Catalog (finally a one-click favorite solution!) and you can drop onto the Catalog's items.

Adding categories and items to the Catalog

In the beginning the Catalog is empty. Open the Catalog's context menu (right-click) and start adding categories and items.

Under Insert as New Category Here you find a number of commands:

Selected List Item(s). Will create a new catalog category auto-named to "List Items", and add to it all items currently listed/selected in the List. Note that the items are added in the order in which there are currently shown. This feature allows you, for example, to easily store search results for later reuse.

Current Mini Tree. Will create a Category containing all unexpanded folders of the currently selected Mini Tree branch (the command is not available if Mini Tree is off). The first folder is set to the current folder.

Favorites Folders and Favorite Files. a little service to give you quick start into the Catalog in case you are upgrading from a previous XYplorer version and already have a nice Favorites collection.

Tabs. Imports all open Tabs into a new Category called "Tabs". Search Results and locked tabs are skipped. It's the complementary function to "Open Folders in Tabs" (see below).

Click and Tag: Labels. Adds all currently defined Labels to the Catalog as a new "Click and Tag: Labels" Category. The items in this category can be used for tagging. Click them to assign the Label to

all selected files.

Click and Tag: Tags. Adds all Tags used in the tags database to the Catalog as a new "Click and Tag: Tags" Category. The items in this category can be used for tagging. Click them to assign the Tag to all selected files.

- Add Tag: Click and Tag to add the clicked item's current tags with the clicked tag.
- Set Tag: Hold CTRL when doing Click and Tag to replace the clicked item's current tags with the clicked tag.
- Remove Tag: Hold SHIFT when doing Click and Tag to remove the clicked tag from the clicked item.

Click and Search: Labels. Adds all currently defined Labels to the Catalog as a new "Click and Search" Category. See Click and Search.

Click and Search: Tags. Adds all Tags used in the tags database to the Catalog as a new "Click and Search" Category. See Click and Search.

Update Category: All tag-related categories (Click and Tag, Click and Search) and their items have a command "Update Category" in their context menu. Click it to update the items listed in the category to the latest state of the tags database.

Under Insert as New Item(s) Here you find the following commands:

Selected List Item(s). Will add all items currently listed/selected in the List to the current Category. Note that the items are added in the order in which there are currently shown. This feature allows you, for example, to easily store search results for later reuse.

Current Mini Tree. Will create a new Item consisting of an auto-generated loadtree() script line containing all unexpanded folders of the current Mini Tree (the command is not available if Mini Tree is off). The first folder is set to the current folder. This is a slick way to save a Mini Tree for later usage within a single Catalog Item. If you later click this item, the Mini Tree is loaded.

Current List Columns. Will create a new Catalog item consisting of an auto-generated setcolumns() script line containing the complete definition of the currently visible columns including their widths. If you later click this item the columns are loaded into the current list.

Current Tree Folder, Current Address Bar Text, Current Tab Location, Clipboard Items. Self-explaining.

Duplicate Item. Will insert a copy of the current item above the current item.

Buttons In Catalog

You can easily add toolbar buttons with their icons, captions, and functionality to the Catalog. Simply add the button key prefixed by ":" to the Location field of the Catalog item. Everything else can be left empty and at the default values.

Examples:

```
:back           = Back
```

```
:fore          = Forward
:dice          = Random Order (resorts the list)
:qns           = Quick Search (opens the dialog)
:rfo           = Recent File Operations (pops a menu)
:cofi          = Enable Color Filters (toggles the state)
```

With the exception of the pressed state effect these Catalog Buttons have all the properties of the real toolbar buttons including left- and right-click menus.

Note: To get the original right-click menu of Toolbar buttons you right-click the icon of a Button In Catalog. For the normal Catalog Item right-click menu you right-click the caption, or you hold CTRL while right-clicking the icon.

Tip: To find out the button keys hold CTRL and mouse-over the button in the toolbar. The key is shown in the tooltip. Or hold CTRL while you click Tools | Customize Toolbar...: The button keys are shown in the list along with the captions.

Using the Catalog

Going to folders and files using the Catalog: Simply click an item to browse to it. The Tree will stay in sync. If the item is a file, it will be selected in the file list.

Action On Click: decide what happens on a single left-click on a catalog item.

(1) Go to Location: go to the file or folder the catalog item is pointing to.

(2) Open Item: open the file the catalog item is pointing to. An EXE file will be launched; any other file type will be opened with the associated application (minding XY's own file associations). As visual feedback you'll see an overlay on the item's icon.

(3) Open selected List item(s) with Application: will open all currently selected List item(s) with the application the catalog item is pointing to. As visual feedback you'll see an overlay on the item's icon.

Launching Applications from Catalog

Simply enter the application into the application field! Short forms are allowed for registered applications, so these lines will work the same (if the path is okay, of course):

```
C:\Program Files\Adobe\Photoshop 7.0\Photoshop.exe
Photoshop
```

The short form is, of course, more portable.

Command Line Parameters are fully supported. If you use them, the application must be in quotes. As an example, the following term in the application field...

```
"winzip32" -a test.zip <items>
```

... will pack all selected files into an archive called "test.zip", when you choose the action "Open selected List item(s) with Application" (and when you have WinZip installed). See the chapter "User-Defined Commands" for further details of the syntax.

Opening folders and files using the Catalog: You'll find a command in the catalog items' context menu called "Open Selected List Item(s)". It is only enabled if the item is an application. The currently

selected list item(s) will be opened with that application item you right-clicked in the Catalog.

Alternatively Ctrl+Shift+Click an application item. Note: The application will open minimized or in the background. Or press Ctrl+Shift+Enter on the selected application item. This way, the application will open in the foreground.

Dropping on a category heading (cursor turns to a white-on-black cross) will add the dragged item(s) to that category (appended at the bottom).

Dropping onto the Catalog: Works just like dropping on Tree and List. When drag-hovering over unexpanded category it is auto-expanded; the original expansion state is restored afterwards. What happens depends on the drop target:

Drop on items pointing to a folder: will move or copy the dropped items into that folder.

Drop on items pointing to an application (i.e. an EXE file): will open the dropped items with that application.

Drop on items pointing to an archive (i.e. an ZIP file): will add the dropped items to the archive.

On right-mouse-dropping a context menu with advanced options is popped up.

You can as well use Ctrl+V to paste items from the clipboard into the current item of the Catalog.

Dropping onto the Scripts: You can as well drop items onto scripts in the Catalog. The variable `<get drop [separator=CRLF]>` returns the dropped items and can be used in the script to process the dropped items. Here are two sample scripts to illustrate the usage of <get drop>:

```
text <get drop>;
```

```
text <get drop |>;
```

Notes:

- The <get drop> variable is cleared after the script is processed, so it cannot be used after the drop event is completed.
- If the script contains no <get drop> variable it is run nevertheless just as if you clicked the Catalog item.
- <drop> is a synonym for <get drop>.
- See scripting command self for an example.

Copying from the Catalog: So you think to copy an item to the clipboard you have to go and fetch it where it lives? Not anymore! Now you can order it from the Catalog: Simply right-click it (left-click would go to it) in the Catalog and choose the new "Copy" (Ctrl+C) command in the context menu. If the destination property of that item (file or folder) is copyable (and not a Quick Search, URL, Computer etc.) it will be copied to the clipboard and ready for pasting. But it gets even better...

Copy All Items in Category: You can copy all items contained in a Catalog category at once! Simply select a category and press Ctrl+C, or choose the "Copy All Items in Category" command from the category's context menu. Note that, of course, only files and folders are copyable, but not drives, Quick Searches, or web-URLs.

This little feature is a revolutionary new way to quickly and repeatedly copy distributed items onto the clipboard by a single click. You just have to collect them one single time under a catalog category, and they are ready for repeated joint copying. Note: of course, if you happen to have same-named items

(from different locations, naturally) on the clipboard you'll run into a collision-situation when pasting them.

Open Folders in Tabs: This feature allows you to easily add a sets of tabs. Selecting the command from the Context Menu will create a new background tab for each folder contained within the Category. Items pointing to anything but straight folders (like non-existing-folders, files, searches, scripts, filters, etc.) are ignored.

Make Mini Tree From Folders: Will create a Mini Tree from all valid folders found within the Category.

(1) If Mini Tree is not active it will be activated by this command (if any folders are found within the Category).

(2) The current tree folder will be set to the first (from top) folder found within the Category.

(3) Portable Paths, XYplorer Native Variables and Environment Variables are supported.

Configuring the Catalog

Positions: Need a new order? Freely move catalog items up and down (also between categories!) by dragging them to their new position. Hold Ctrl while dragging items to duplicate them. When you duplicate a category the new copy comes without child items. Or sort a whole Category by selecting Sort Category from the context menu.

Colors: For visual grip you can totally control the Catalog's colors per item. Select "Properties" in the context menu of an item.

Icons: You can define the icon for each Catalog item. Simply append any file spec to the Caption in Properties, separated by a | (pipe). It can be an icon file or any other file or folder with associated system icon. XYplorer native variables, environment variables, and portable paths are supported. For example: `Love|E:\Test\Kiss.ico`. You can as well reference internal icons (toolbar icons) using ":" as prefix, for example: `Add Tags|:tagsadd`.

You can also state an image file (GIF, JPG, PNG) for the custom item icon (defined in the Caption field, appended to the caption separated by |). It can be a full path, or just a filename (auto-resolved to the default icons path `<xyicons>`). Note that there is no caching. So if you use a large file for this image, there will be delays even when you hover the mouse over the item.

You can also assign icons extracted from icon resources (exe; dll; cpl; ocx; scr; icl; bpl; wlx; wfx; wcx; wdx; acm). For example: `Desk|C:\Windows\System32\imageres.dll /222` or `Home|C:\WINDOWS\system32\shell32.dll /163`

Tip: Keys to toolbar icons, e.g. [tagsadd], can be seen in square brackets when choosing Customize Toolbar from the right-click menu of a user button. Paths are resolved relative to the Icons Path (default: <xydata>\Icons).

Tip: You can define the color for a circle drawn as background to your icon. This can be useful in Dark Mode where some dark icons otherwise tend to disappear in the darkness. See here for the syntax.

Memory: Of course, the Catalog is saved between sessions (if you enable "Save settings on exit", and unless you "Exit without Saving"). The data are stored in a file called "Catalog.dat" located in the "Catalogs" subfolder in the application data path.

Tip: Dbl-click on left 9 pixels opens Item Properties.

Working with many Catalogs

Part 1: One Catalog at a time

You can switch catalogs on the fly and manage any number of Catalog resources. You find the usual commands in the right-click menu of any Catalog Category and of an empty Catalog in the submenu Catalog.

- New: Create a new empty Catalog. The initial filename defaults to "<xydata>\Catalogs\CatalogNew [##].dat", where [##] is a number suffix that makes it a unique name. You are prompted to save any changes if the current Catalog is dirty.
- Open...: Open an existing Catalog file, after saving any changes in the current Catalog. If you choose a filename that does not exist, a new empty Catalog of that name is created. You are prompted to save any changes if the current Catalog is dirty.
- Revert to Saved: Reload the current Catalog from file without saving any changes first.
- Save: Save the current Catalog.
- Save As...: Save the current Catalog under a new name.
- Save Copy As...: Save a copy of the current Catalog under a new name, without making this the current Catalog. Useful for making quick backups.
- Import Catalog...: Here you can import other catalogs into the current catalog at the position right before the currently selected category.
- Export Category...: Here you can export the current category as a self-contained catalog. This one-category-catalog can later be imported, but also be loaded stand-alone.
- Show Icons: Here you can have the Catalog without icons if you prefer.
- Refresh All Icons: Refresh all Catalog Icons.

Recently Used Catalogs: At the bottom of the Catalog submenu you find the most recently used Catalogs. They can be managed via menu Tools | List Management | Recent Catalogs. Note that the most recently used Catalogs are also found in the context menu of the Catalog toolbar button.

Part 2: Many Catalogs simultaneously - Including Catalogs

You can include any number of catalogs in the current catalog. Contrary to imported catalogs (using the Import command) included Catalogs are read-only and kind of temporary: Their contents are not saved with the parent catalog, and you can remove them from the master catalog at any time. On the other hand, as long you don't explicitly remove them they are retained across sessions (i.e. they are auto-included in the current catalog on start up).

Interface

- The category context menu features a new submenu "Include" where you find commands to open a catalog to include, and to Reload or Unload all included catalogs (Open Catalog to Include, Reload All Included Catalogs, Unload All Included Catalogs).

- At the bottom of the menu you find a MRU list of recently included catalogs. A click on any of them will include it with the current catalog. If it is already included, it will be reloaded (updated).
  Tip: You can edit this list in Tools | List Management | Recently Included Catalogs.
- Above that MRU list, catalogs that are currently included are shown with a checkmark. To unload any of them simply click on their name (as if to remove the checkmark).
- Both lists are limited to 8 items, although the number of included catalogs is not limited.
- In the Catalog, included items are marked by a colored square symbol at the right end of the row. 6 colors are predefined and cycled if more are needed, i.e. items from the 7th include source will get the same color as items from the 1st source.
- The items' tooltips show the included source if any.

The Mechanics of Merging

Included Catalogs are not simply appended or inserted, but merged into the master catalog, so that items from different catalogs are combined within the same categories.

- The current and the included catalog are compared category by category. If a category with the same caption (case-sensitive) already exists the items inside the included category are appended to the existing category. If a category does not exist yet, it is appended to the existing categories with all its items.
- After merging you can move the position of the included items just like normal items. But note that the new positions won't be retained across sessions as the catalogs are freshly included at next start up.
- You may also alter the Properties of a included item. However, also these edits are not retained across sessions! You are warned about this in the Edit dialog.
- When XYplorer is closed with included catalogs, their names are stored in the INI file (but their contents are not saved since they are read-only), and they are freshly reloaded (included) on next start up.

Remarks

- On exporting included categories (or categories containing included items), they will be locally saved as normal catalogs.
- Also when duplicating included categories or items, the "included nature" is not inherited.
- The MRU list of included catalogs is stored in a portable manner, so you can carry included catalogs around on a USB stick without problem (the drive letter is not hard-coded).

Usage

- This is mainly a company-oriented feature. Employees can access read-only catalogs that are centrally stored on a server and thus include extended functionality such as scripts, links, drop targets, text data, etc. right into their XYplorer interface.
- However, the feature also provides an elegant way to temporarily load read-only functionality into the interface.

Supported protocols

The Catalog (just like the Address Bar!) supports the http[s]:// protocol and the file:/// protocol. These files are executed by the Shell, so http[s]:// URLs are opened by your default browser whereas file:/// URLs are opened by the associated application. Which means:

You can add URLs to the Catalog and they'll work just like web favorites. One-click surfing! And you can add files to the Catalog and they'll work just like links. One-click-opening!

Click and Search

See Click and Search.

4.21 Click and Search

Click and Search

Catalog categories can be turned into something that will make searching for files your favorite pastime.

Defining a Click and Search category

Add a category to the Catalog and open the category's Properties dialog via its right-click menu. Now in the Advanced frame:

Use category for searching: Tick this to enable the feature, i.e. to turn the category into a Click and Search category.

Operator: Here you can define the Boolean operator to be used in a Combined Search (see below): AND or OR.

Location: Here you can state the location to be searched. Leave the field empty to always search the current location. Enter * (asterisk) to search the whole PC.

In the Location field you can also enter search switches (after the path if there is one). So here is the place to control whether the search is recursive (/r) (i.e. including subfolders) or not (/n), or whether only the tags database should be searched (/t). Examples for Location:#

```
  E:\Test       = search E:\Test (Include Subfolders is taken from the setting in the Find
Files tab)
  E:\Test /r    = search E:\Test (Include Subfolders)
  E:\Test /n    = search E:\Test (Do Not Include Subfolders)
  E:\Test;E:\Pics = search E:\Test and E:\Pics
  /r            = search the current location (Include Subfolders)
  /n            = search the current location (Do Not Include Subfolders)
  *             = search the whole PC
  /t            = search the tags database only (for items in the current location)
  * /t          = search the tags database only (for items anywhere)
  * /T          = search the tags database only (for items anywhere), the patterns are Tags
  * /L          = search the tags database only (for items anywhere), the patterns are Labels
```

Defining the Items in a Click and Search category

All items in a Click and Search category are expected to contain search patterns. You can add the items manually (Tip: Press INS to quickly add an item), or automatically (see below).

In each item you enter a search pattern into the field "Pattern". Here are some examples for search patterns:

```
  *.png
  !*.png                    (NOT *.png)
```

```
prop:#AspectRatio: 16:9
lbl:red
tags:jazz
tags:rock
tags:pop
jazz                        seen as tag if Location has switch /T
red                         seen as label if Location has switch /L
a* OR b* /n                 name is a* or b* (a Boolean term, with switch "don't include
subfolders")
dateM: !dw 6-7              NOT modified on a weekend
ageM: 1 d                   modified yesterday
ageM: 1 w                   modified last week
```

Clicking an item will run the search (functionally it's identical to a Quick Search) for location defined in the Location field, or, if none is defined, for the current location.

Combined Searches

Additionally to running the category items individually you can also combine the items in one search.

- The items can be individually enabled via checkboxes. Clicking the category will run a combined search of all ticked items.
- A combined search can be either AND- or OR-combined. This can be configured in the category's Properties dialog (see above).
- Any switches are removed from the combined terms.
- Toggling the state of any item's checkbox will also run a combined search. Hold CTRL to suppress this trigger.

This feature becomes particularly interesting in combination with tags. See next paragraph.

Tag Clouds

Now for something completely wonderful. The Catalog right-click menu has a command Insert as New Category Here | Tags by which you can add all Tags used in the tags database to the Catalog as a new "Click and Search" Category. The Location defaults to whole PC, tags database only, and the operator to "AND".

Voilà, now you can list your tagged files in a most natural way, just like you know it from tag clouds.

Label Clouds

The same is available for Labels. The Catalog right-click menu has a command Insert as New Category Here | Labels by which you can add all Labels defined in the tags database to the Catalog as a new "Click and Search" Category. The Location defaults to whole PC, tags database only, and the operator to "OR" (no file can have more than one label at the same time).

Note that, contrary to the "Tags" category, also labels are listed that are currently not assigned to any item. This is done because getting an empty list when searching "Urgent" is an agreeable experience you should be allowed to have.

Comments as Captions

When clicking a category item the caption of that item (if there is any) will be passed as comment (if there is none already). That way you get the caption of the CAS item in the Quick Search bar which looks kind of natural and slick.

4.22 Status Bar

Sections of the Status Bar

Objects/Files Found

Displays the number of objects currently listed in List. In Browse mode this is the number of objects within the current folder, in Find mode the number of objects found.

Bytes

Displays the bytes sum of all currently selected items in List. If nothing is selected, the bytes sum of all items is shown.

Name/Progress

Displays the icon and name of the currently focused item (file or folder).

Version in the Status Bar: Tick Configuration | Controls & More | Miscellaneous | Show version information in the Status Bar to show the embedded version information of executables, DLLs, DRVs, etc. (if they have any) right in the Status Bar on selecting such a file.

If nothing is selected, the used and free bytes of the current drive are shown.

During File Find the same section gives you feedback on the progress of the search by displaying the name of the currently searched location. When the find is finished, the used time is shown, and the number of files scanned (this is the number of all files that match at least the Name filter of the search).

Clicking the Status Bar

Right-click: Pops a menu with a couple of related/useful commands.

- Live Filter Box in Status Bar: Here you can choose the position of the Live Filter Box. Either it is shown right of the Address Bar, or in the left end of the Status Bar.
- Show Exact Bytes in Status Bar: Here you can toggle whether the exact byte count is shown in parentheses additionally to the rounded byte count.
- Smart Section Sizing: Here you can toggle whether the middle section smartly negotiates the available space with the right section. In other words, the right-most section of the Status Bar auto-adjusts its left position to show as much as possible of its contents. Useful if you have a smaller screen.
- Use Status Bar Template: Tick it to actually use the template defined in "Configuration | Colors and Styles | Templates | Status Bar".

Ctrl+Right-click: When you hold CTRL and right-click a section you get a mini menu with one command "Copy Status to Clipboard". Lets you collect this data.

Swiping the Status Bar

The Status Bar right-click menu features two toggles:

- Horizontal Swipe Toggles the Preview Pane: If ticked you can left-button-drag the Status Bar left/right to open/close the Preview Pane. The action is fired when you drag more than 8 pixels horizontally.
- Vertical Swipe Toggles the Info Panel: If ticked you can left-button-drag the Status Bar up/down to open/close the Info Panel. The action is fired when you drag more than 8 pixels vertically.

Notes:

- If "Window | Arrangement | Preview Pane to the Left" is ticked then of course you drag to the right to open the pane, and to the left to close it.
- You can undo/redo the last action while the mouse is held down, so you can quickly open and close the Preview Pane by a slick mouse move (or finger move if you have a touch screen).

Status Log

The Status Log logs all changes in the right-most Status Bar section. The command Show Status Log in the right-click menu of the Status Bar will show the log.

- The current location is shown at the beginning of each entry.
- All three status bar sections are shown.
- Duplicate messages are suppressed if they appear within 10 seconds.
- The Status Log does not grow forever but auto-cycles at 1024 entries (= when a new entry is added the oldest one is dumped).

See also slog.

Font Size of the Status Bar

You can independently customize the font size of the Status Bar by Ctrl+Wheel over the Status Bar.

4.23 Status Bar Buttons

Status Bar Buttons

There's an optional mini toolbar at the right end of the Status Bar containing two buttons. You can show/hide it using Show Status Bar Buttons in menu Window.

1. The Background Jobs Button

This button tells you about the status of background jobs (if any) by showing an animated icon. If the animation is active the button's tooltip tells you how many jobs are not yet completed (either in progress or pending).

Clicking the button will open the Background Jobs dialog (see below). The button also has a right-click menu with some handy commands.

The "Background Jobs" button is only shown when background processing is enabled (Configuration | File Operations | File Operations | Background Processing | Enable background processing).

Background Jobs Dialog

The right-click menu of the Background Jobs button and the "Queue file operations" toolbar button has an item Background Jobs... where you can display the background file operations that are completed, in progress, or pending within the current session, as list in a non-modal window. You will also see which jobs have been skipped, cancelled, or only partially completed. The position and size of the window is stored between calls and sessions.

The status is auto-updated each second. You can force an immediate update by pressing F5.

Hide completed and skipped

Check to show only pending and in-progress jobs in the list.

Pause queue

You can completely pause the queue using the checkbox "Pause queue". Jobs that are already in progress are not affected by pausing. On unpause the processing starts with the next pending queued job.

The setting is also available in the context menu of the Background Jobs button.

When you close XYplorer all pending jobs in a paused queue are forgotten. You are prompted whether you really want to purge the jobs.

Note: Pausing only affects *queued* (pending and new) jobs. If Queue file operations is off then new

background jobs will be started immediately regardless of the state of "Pause queue".

Tip: While the queue is paused you can still add new jobs to the queue. Once the queue is to your liking you can release the Pause and go have a coffee while your machine is working the queue.

Background Jobs Context Menu

Each job has a context menu with a number of commands.

Details...

Here, you can view the list of source files for each job, as well as the error message in case of a failed job.

Cancel Job

Here you can cancel the currently running job. The next job in the queue is automatically started.

Skip Job

Here you can temporarily or permanently skip pending jobs. A job will not be started as long as it is skipped.

You can unskip a job at any point in time. If other jobs are in progress while you unskip a job then it will be started whenever its turn comes.

Start Job Now

This command is only present in the context menu of paused pending jobs. It allows you to single-click-start a particular job from a paused queue, regardless of the paused state and regardless of the original queue order.

The command is also available on the context menu of skipped jobs, so if you change your mind and want to do a skipped job now you don't need to unskip it first. Simply click "Start Job Now".

2. The Info Panel Toggle

This button toggles the Info Panel. The button's right-click menu has a couple of related commands.

4.24 Info Panel

Showing/Hiding the Info Panel

The Info Panel is located at the bottom of the main window. You can toggle its visibility by pressing F12.

Tabs on the Info Panel

The Info Panel contains the following tabs:

Properties Tab

Version Tab

Meta Tab

Preview Tab

Raw View Tab

Tags Tab

Find Files Tab

Report Tab

4.24.1 Properties Tab

Properties Tab

Gives you various information on the selected file.

Tip You can copy virtually any information on this tab by double-clicking it. Double-clicking the key copies key and value, double-clicking the value copies just the value.

Tip To read a value that is too long to be fully displayed, position the mouse over this values to see the complete string in a tooltip.

Various File Information

Pretty self-explaining: you know this stuff from the Explorer Properties dialog.

Filename: Right-clicking the filename (i.e. the name of the selected item) pops a small menu with the following items:

- Copy Name: Copies this filename to the clipboard.
- Copy Name with Path: Copies this filename with path to the clipboard.
- Show Character Table: Click it to get a vertical list of the characters and their Unicode ordinals in decimal and hexadecimal, and their utf8 sequences. It also shows short descriptions for the Directional Formatting Codes, e.g. "Right-to-Left Override" for U+202E.
- Convert to ASCII: See Rename Special | Convert to ASCII. It is only enabled if the filename contains extended ASCII (ANSI) or Unicode characters.

Size on Disk: Files use up disk space by clusters, not by file bytes, taking into account compressed and sparse files.

Contents: If a folder is selected Size shows the total bytes of all contained files (subfolders included), and Contents tells you the number of files and folders found below this folder. This feature has to be explicitly activated in configuration since for large deeply nested folders it takes some time to gather this info.

Embedded Icons

Check Configuration | General | Refresh, Icons, History | Icons | Show embedded icons on Properties tab to show any embedded icons extracted from the selected file. Embedded icons are typically found in EXE and DLL files (other formats include CPL; OCX; SCR; ICL; BPL; WLX; WFX; WCX; WDX; ACM).

- Double-click the currently displayed icon to copy it to the clipboard (in BITMAP format, no transparency).
- Double-click the embedded icons count label (only visible when you select a file that contains embedded icons) to open a list of all embedded icons in the selected file.
- Wheel over the embedded icon or the embedded icons scrollbar to quickly scroll through the embedded icons in the selected file. Hold SHIFT to wheel 6 times as fast.

Time-stamping

Clicking any of the three file times on Properties tab will activate the interface to actually set those times to the currently selected file(s). Note that you can call the Date Picker by pressing F1 inside those text boxes.

Press ENTER or click the green icon to set the new time-stamp (the changed time-stamps are applied to all currently selected files), or press ESC or click the red icon to cancel and reset the display to the original state.

- Time-stamping a file ensures that the archive bit is set in that file.
- Under NTFS the milliseconds in file times will be set to 0 (zero) when you touch a file date and do not pass milliseconds (see below).
- Under VFAT (Win95a), FAT, and FAT32, dates before 1980 and after 2107 cannot be set, and dates 28.02.2106 - 31.12.2107 set the file time to "unknown".

- Set the date to nothing to set a file time to now.
- You can use the pseudo date 0 (zero) to set a file time to the lowest possible value.

Milliseconds Support

Time-stamping fractions of a second (up to 7 decimal digits) is supported. The fraction should be appended to the time expression separated by a dot. The following example stamps the selected files with milliseconds precision:

```
2013-05-28 14:13:23.888
```

If "Show Milliseconds" is enabled (Tools | Customize List | Date Column Format) then the file dates on the Properties Tab always show the full precision of seven decimal places, regardless of the precision shown in the file list.

UTC Support (Universal Time Stamp)

By appending "Z" (or "z") you can mark the time expression as UTC (Coordinated Universal Time). This way you can modify the UTC date of a file directly (as it is internally used by NTFS). For example, this will set the modified date to UTC 12:00 today (displayed as 14:00 in Germany with Daylight Saving Time in effect):

```
timestamp "m", "<date yyyy-mm-dd> 12:00Z";
```

That way you can guarantee the same universal timestamp regardless of any DST (Daylight Saving Time) offset in effect. Very nice for software developers with a global market.

The ISO 8601 formats for UTC and time offsets from UTC are supported. The following time expressions all refer to the same point in time:

```
2013-05-14 10:00:00+02:00
2013-05-14 08:00:00+00:00
2013-05-14 08:00:00Z
2013-05-14 03:00:00-05
2013-05-13 23:00:00-0900
2013-05-13 20:00:00-12:00
```

Changing file attributes

The file attributes of the currently focused file are displayed as (un)checked boxes. You can change the settings of 7 attributes (Read-Only, System, Temporary, Hidden, Archive, Offline, and Not Indexed). Temporary only works for files, not or folder.

Click the green icon to set the new attributes (applied to all currently selected files), or click the red icon to cancel and reset the display to the original state.

4.24.2 Version Tab

Version Tab

Displays any version information contained in the selected file.

Pretty self-explaining: you know this stuff from the Explorer Properties dialog.

Tip You can copy virtually any information on this tab by double-clicking it. Double-clicking the key copies key and value, double-clicking the value copies just the value.

Note If the version tab is empty, the selected file has no version info. Version information is typically contained in EXE, DLL, DRV, OCX etc. files.

Fixed File Version
Here's some advanced version information that is not all revealed to you by Explorer.

4.24.3 Meta Tab

Meta Tab

Here some basic file-specific metadata provided by the shell are displayed. The default metadata layouts differ by file type (images, audio, and so on...).

A shortcut file (LNK) is resolved and the meta data of the target is displayed.

Right-click offers "Show in Popup" (handy if you want to select and copy particular metadata) and "Copy to Clipboard".

4.24.4 Preview Tab

Preview Tab

Here you can preview files of the following types: Text Files, Document Files, Web Files, Font Files, Icon Files, Image Files, Audio Files, and Video Files. All previews are invoked by selecting a file in the

List by left-mouse click or keyboard. No preview is shown when you select a file with the right mouse button.

The order of precedence of the various previews is:

Text > Document > Web > Font > Icon > Image > Audio > Video

So, to preview HTML as Web File (like in a browser), go to Configuration | Previewed Formats and uncheck HTML under "Text Files" and check it under "Web Files". If you want to preview HTML as source code leave it checked under "Text Files".

Note that also shortcuts (LNK files) are resolved and then previewed where applicable.

Invoking a preview needs the Preview tab selected and visible. Press F12 to show the Info Panel containing that tab.

Tip To enlarge or shrink the display area, use the horizontal splitter, which is the 3 pixel high line located just above the Info Panel tabs, and below the status area for the list. When positioned on the splitter, your cursor will become double headed arrows.

Text preview

Preview more than 50 different plain text formats. For huge text files only the first 5 MB are shown for performance reasons.

Dropdown Menu (Orange Button)

In the dropdown menu on the Preview tab, you can manually select a different encoding for text file to preview. Choosing System Default enables auto-detection for UTF-16LE, UTF-16BE, and UTF-8. In parentheses "System Default" shows the Default ANSI Code Page, e.g. "System Default (1252)". Choose UTF-16LE, UTF-16BE, UTF-8, and UTF-7 to force decoding the current file in any of these formats.

A couple of Western code pages and East Asian languages using DBCS encoding are also supported. Click any of the code pages to reload the current file with that specific encoding. When reloading a previewed text the scroll position is preserved as good as possible.

Word wrap: Wrap lines at window border. Also controls word wrapping of text previews in in the Floating Preview and in the Preview Pane.

Edit User Code Pages: Here you can add any number of user-defined code pages. You can optionally also state the line length, aka record size.

General Format:

```
CodepageName=CodepageValue[,RecordSize]
```

Examples:

```
IBM 437 (DOS-US)=437
IBM 850 (DOS-Latin-1)=850
IBM EBCDIC International=500,82
```

Remember Selected Code Page: If ticked then you can, for example, set the codepage to Arabic (1256) and all text previews will use this codepage to interpret the bytes in the file. Advantage: You don't have to set the system-wide default codepage to Arabic. If unticked then the codepage will automatically switch back to the system default code page for the next preview.

Document preview

Preview for document files of the following types: *.cdr, *.csv, *.doc, *.docm, *.docx, *.dot, *.dotx, *.dwfx, *.dwg, *.dxf, *.easmx, *.edrwx, *.eml, *.eprtx, *.eps, *.jtx, *.msg, *.nws, *.odp, *.ods, *.odt, *.one, *.oxps, *.pdf, *.pot, *.potx, *.pps, *.ppsx, *.ppt, *.pptx, *.rtf, *.sldasm, *.slddrw, *.sldprt, *.wpd, *.wpf, *.xls, *.xlsm, *.xlsx, *.xlt, *.xltx, *.xps

More types can be added via Configuration | Previewed Formats | Category.

Web preview

Preview for web files of the following types: *.asp, *.aspx, *.cfg, *.htaccess, *.htm, *.html, *.inc, *.mht, *.mhtm, *.mhtml, *.php, *.php3, *.php4, *.php5, *.shtml, *.svg, *.svgz, *.swf, *.url, *.xml

More types can be added via Configuration | Previewed Formats | Category.

From the web preview you can follow links to the internet. It's nothing less than a fully capable browser based on the Internet Explorer rendering machine.

You can drag pictures from the previewed page to file list: drag with right-mouse or hold CTRL when start dragging to force a copy operation.

Server Mappings

Configurable server mappings allow you to pass local files through a local web server (eg your local Apache installation; this server must be running, of course) before displaying them in the preview. In Configuration | Preview | Web Preview:

Check the "Enable server mappings" option.

In the 1st field enter the local file path where documents are stored, for example: D:\wwwroot\.

In the 2nd field enter, for example, http://localhost/ if using a web server installed on the local system that's configured to serve files from the path above. Or enter a remote web server address to preview files that are stored on a remote server. The mapping is a simple string replacement.

Font preview

Preview for installed or uninstalled Windows font files. The following types are supported:

*.ttf = TrueType fonts

*.otf = OpenType fonts

*.pfb; *.pfm = Type-1 fonts (both files must be present in the same path)

*.fon = Bitmap/raster fonts

Additionally you get some version and copyright information.

With currently installed fonts, the displayed font name is blue.

Font size selection: Right of the sample text box you find a scrollbar where you can set the font size of the first preview line (1-128). Set size to 0 (zero) to hide the first line.

Activate/deactivate fonts on the fly

You can activate/deactivate fonts on the fly by simply clicking on the bold font name shown in the font preview's info frame. Very comfortable for screen and print designers who often need a number of fonts for a quick one-time job, and do not wish to go through the awkward font installation procedure provided by Windows while cluttering up their fonts folder with hundreds of never-again-used fonts.

The font file to be activated can sit anywhere. Take care however, that you do not move/delete it while it is activated, else you can neither use it nor deactivate it.

An activated font file will NOT be copied to the Windows font folder.

Deactivation does NOT delete the font file.

The activation/deactivation lasts only for the current Windows session -- no files are moved, the registry is not touched.

Image preview

Shows a thumbnail for the following image file types:

*.arw, Sony Raw Image
*.bmp, ACDSee BMP Image
*.cr2, Canon Raw Image
*.crw, Canon Raw Image
*.cur, Cursor
*.dcr, Kodak Raw Image
*.dib, ACDSee DIB Image
*.dng, Adobe DNG Image
*.emf, EMF-Bild
*.erf, Epson Raw Image
*.fff, Hasselblad Raw Image
*.gif, ACDSee GIF Image
*.ico, Icon
*.jpe, JPEG-Bild
*.jpeg, JPEG-Bild
*.jpg, JPEG-Bild
*.mef, Mamiya Raw Image
*.mrw, Minolta Raw Image
*.nef, Nikon Raw Image
*.nrw, Nikon Raw Image
*.orf, Olympus Raw Image

*.pef, Pentax Raw Image

*.png, ACDSee PNG Image

*.psb, Adobe Photoshop Large Document Format

*.psd, Adobe Photoshop Image

*.raf, Fuji Raw Image

*.raw, Leica Raw Image

*.rle, ACDSee RLE Image

*.rw2, Panasonic Raw Image

*.rwl, Leica Raw Image

*.sr2, Sony Raw Image

*.srf, Sony Raw Image

*.srw, Samsung Raw Image

*.tga, ACDSee TGA Image

*.tif, ACDSee TIF Image

*.tiff, ACDSee TIFF Image

*.wmf, ACDSee WMF Image

*.x3f, Sigma X3F Image

Note that Codecs for RAW image formats need to be installed on the system (they are not included in XYplorer).

Previewing Animated GIFs is supported since version 12.90 in the Preview tab, in Mouse Down Blow Up, and in the Floating Preview. The number of frames is displayed in the info section.

Depending on the image format you get various information:

Size: the original size of the image in pixels, and the size of the thumbnail.

Colors: maximal number of different colors used in the picture (aka color depth).

Area: space used by the image display in square pixels. Video bytes: number of bytes used up by your video memory when displaying the image in full size. This number is calculated by multiplying the area of the image with the current color depth of your monitor. For example: if your system is set to 32-bit (True color), 4 bytes are used for the display of every pixel.

File size: number of bytes used by the image file.

Compression ratio: tells you the degree of compression of the image file. Percentages below 100 mean that the file is compressed. This number is calculated like this: FileSize / (Width * Height * (BitsPerColor / 8)), where BitsPerColor is the color depth of the image, not of your display.

Hotspot at: (cursor files only) the position (x, y) of the hotspot, that's the pixel of the cursor image that corresponds to the exact mouse position.

Icons contained: (icon files only) number of icons contained in the file. Icon and Cursor files can contain more than one image. (This information is only given when more than image is contained and only for icons, since it's rare to find multiple images in cursors and there the space is used up already by the hotspot information.)

Dropdown Menu (Orange Button)

Copy Compact Info: copy a one-liner describing the currently previewed file to the clipboard.

Copy Extended Info: copy all information available on the currently previewed file to the clipboard.

The following for images only:

Copy Original: copy the image (original size) to the clipboard.

Copy Preview: copy the image (preview size) to the clipboard.

Full Screen: enter full screen mode. Note that you can full screen preview the currently selected image file even when the Info Panel is hidden by simply pressing Shift+F11!

The Full Screen Preview (Shift+F11) also works for HTML, MHT, URL, NWS, EML, and MSG files in perfect original size quality. Note that -- just like with thumbnails for non-picture files -- this service depends on the capabilities of your operating system and the installed software.

Shortcut: Shift+F11

Full screen preview: Press Shift+F11 to see the image in full screen mode. If the image is larger than the screen it is shrunk to fit. In Configuration | Preview you find options regarding background color, zooming, and more.

While in full screen mode, you can use the these keys to browse through the images found in the current list.

Other keyboard tricks see here.

Mouse Down Blow Up

Showing the preview image in original size: When you move the mouse over the preview image and hold down the left mouse button the original image pops up. If the image is larger than XYplorer's window, you can scroll about the image by moving the mouse while holding it down. This feature is called "Mouse Down Blow Up".

Note that the blow up exactly zooms into the point where you downed the mouse. Of course, this positioning only applies if "Configuration | Preview | Shrink to fit" is OFF.

Note that also the Thumbnails, the Floating Preview, and the Full Screen Preview support Mouse Down Blow Up.

Audio/Video preview

Plays an audio/video file, and shows various information depending on the format of the file:

Size: (for videos only) the original size of the video in pixels, and the size of the thumbnail.

Length: [hours:]minutes:seconds milliseconds

Format: [depends on the type of file] for example: sample rate (Hz), bits per sample, and number of channels (Mono/Stereo), number of tracks (MIDI).

Track: (CD-Audio only) choose the track you want to play ...

Progress Bar

The progress bar indicates the relative position of the audio/video currently playing. It also allows for some mouse interaction:

Left-Click	Play from here.
Right-Click	Toggle pause/play (also by key Space).
Shift+Left-Click	Restart the sound from beginning.
Shift+Right-Click	Go to beginning and stop (also by key Shift+Space).

The progress bar's tooltip shows the timeline position of the cursor.

Wind Backward and Forward

There's keyboard-driven backward and forward. Works in paused and running mode, on audio and video.

Left Arrow	5 sec backward
Right Arrow	5 sec forward
Shift+Left Arrow	30 sec backward
Shift+Right Arrow	30 sec forward
Ctrl+Left Arrow	1 sec backward (Video: Frame step backward)
Ctrl+Right Arrow	1 sec forward (Video: Frame step forward)

A-B repeat (Sub-Loop)

To repeat any particular section of the medium (audio or video), aka "A-B repeat", looping must be enabled (tick "Play Again"). The section is defined by clicking on the progress bar while the medium is previewed (no matter whether running or paused):

Ctrl+Left-Click	Set start of section to this point
Ctrl+Right-Click	Set end of section to this point
Ctrl+Alt+Left-Click	Unset any section

If a section start is defined then "Go To Beginning" (Shift+Space, or Shift+Click) will go to the beginning of the section. There is no visual marking of the section.

The section is auto-unset when you preview the next file.

Media Control Buttons

- Both "Play" and "Pause" have identical functionality, namely toggle Play / Pause, with inverse pressed-states.
- "Stop" rewinds to beginning and stops there.

- "Autoplay" toggles Autoplay.

Snapshot (Video only)

Copies a bitmap image (preview size) of the current scene to the clipboard (shoot a film still). Note that this works while the video is running or stopped.

Important: You cannot snapshot non-AVI videos when you have OS hardware acceleration on! If you want you have to Adjust Hardware Acceleration. Here's what to do in Win2K, XP, and higher:

1. Right-click on your desktop

2. Select Properties->Settings->Advanced->Troubleshoot

3. Then set the hardware acceleration slider to the third notch (from left) and apply the changes

4. Press Ok (Computer Restart maybe required)

Note: You cannot rename, copy, move, or delete an audio/video file while it is playing. The preview is automatically stopped when you start dragging a file.

Supported Audio/Video file formats

Among the supported files types are: *.aif; *.aifc; *.aiff; *.ape; *.asf; *.au; *.avi; *.cda; *.dsd; *.dsf; *.flac; *.m1v; *.m4a; *.m4r; *.mid; *.mov; *.mp+; *.mp2; *.mp3; *.mp4; *.mpa; *.mpc; *.mpe; *.mpeg; *.mpg; *.mpp; *.mpv2; *.ogg; *.ogm; *.rmi; *.snd; *.ts; *.vob; *.wav; *.wma; *.wmv

More types can be added via Configuration | Preview | Previewed Formats | Categories.

What formats actually work is dependent on the configuration of your local system: the Windows version, the version of Media Player is installed, the version of Internet Explorer is installed. By rule of thumb: If WindowsMediaPlayer can play it, XYplorer can play it too.

ID3 tag editing

While previewing an MP3 file you can view and edit its ID3v1-tags or ID3v2 tags.

Dropdown Menu (Orange Button)

Copy Compact Info: Copy a one-liner describing the currently previewed file to the clipboard.

Copy Extended Info: Copy all information available on the currently previewed file to the clipboard.

Next, there are some settings for audio/video file preview only like Play/Pause, Stop, Autoplay, etc, which are just repetitions of the same settings in Configuration | Preview | Preview | Audio/Video preview.

The only option that only lives here is Playback Speed: You can change the playback speed. Choose from 0.5, 1, 1.5, 2, or 4.

- You can change the speed on the fly while the media is playing.

- Changing the speed also changes the pitch of the sound. It's not like YouTube, where the pitch is kept constant.

4.24.5 Raw View Tab

Raw View Tab

Raw View displays the contents of ASCII and binary files (the file type is detected automatically).

Display options

ASCII file options

Line #: Show line numbers.

Hex: Toggle Hex view and Byte (text) view.

Word wrap: Wrap lines at window border. Also applies to Extract text (see below).

Binary file options

Extract text: Extract text portions out of binary files. The minimal text length is 5 bytes (smaller text portions are not displayed because they are probably junk). The left column shows the offset (start position in file) for each text. Individual texts can be selected and copied (see below). In the header area, the ordinal of the first currently displayed text is shown.

International: Extracted text includes upper-ASCII characters as used in French, German, Italian, Portuguese, Scandinavian, and other locales. Useful to grab text from non-english binaries.

Context Menu

Small Font, Large Font, Heavy Font, [Edit Text Font]: Select the font used in Raw View.

Resolve Links: Tick it to raw view the targets of LNK files. Untick it to raw view the LNK files themselves.

Auto-Scroll to End: Check it to automatically position the viewer at the end of the Raw View. Find the option in the context menu of the Raw View, or in the orange button drop down menu.

Copying lines

Although Raw View is a viewer, not an editor, there is a handy way to select and copy parts of the

displayed files.

Selecting works line-wise: to un/select a line simply left-click it. Hold the Shift-key to multi-select larger zones.

While you cannot select individual strings as in an editor, you can select as many different lines as you like, and then copy them to the clipboard in one block.

Copy a single line by dbl-clicking it (no need to select it first).

The copy command is located in the context menu (right-click the view area).

Remarks

- To enlarge or shrink the display area, use the horizontal splitter (the 4 pixel high zone right above the Info Panel).

4.24.6 Tags Tab

Tags Tab

This is the main interface for adding and editing Tags (Labels, Tags, Comments) to/of the items currently selected in the file list.

Viewing and Editing Tags

When you select an item in the file list the three fields are set to the Label, Tags, and Comment of that item. If more than one item is selected the fields are set to the tags of the focused item.

You can edit the tags in the fields and then apply the edited tags using the Apply button. On Apply all ticked tags (Labels, Tags, Comments) shown in the Tags tab are applied to all currently selected items .

Tip: When you right-click the Apply button, a toggle Apply Changed Tags Automatically appears. Tick it to have changes to tags applied automatically when another file is focused, without the "Apply Changed Tags?" prompt.

The Reset button will reset the three fields to the current item's tags.

Tagging Mode

The Tags field has a small dropdown where you can select the tagging mode:

- Add Add given tags to the existing tags of all selected items.
- Set Replace any existing tags in all selected items with the given tags.

- Remove Remove given tags from the existing tags of all selected items.

Keyboard Tricks

You can trigger Apply by pressing <enter> right in the edit fields (<ctrl+enter> for Comments). In that case the tag of this type is applied even if the checkbox is not ticked.

- Label: Press <Enter> to apply only the Label.
- Tags: Press <Enter> to apply only the Tags.
- Comment: Press <Ctrl+Enter> to apply only the Comment.

4.24.7 Find Files Tab

Index

Buttons "Find Now" and "Reset Filters"

Checkboxes "Applied Filters"

Tab "Name & Location"

- Name
- Mode (Standard, Boolean, RegExp)
- Checkboxes modifying the pattern matching (Match Case, Whole Words, Invert, Ignore Diacritics, Path, Find Hidden)
- Type
- Location
- Multiple Location Search
- Search in List
- Comparison Operators
- Find Files by Size
- Find Files by Date and Age
- Find Files by Tags (Labels, Tags, Comments)
- Find Files by Columns
- Find Files by the Size Column Contents
- Find Files by Properties
- Find Files by Contents
- Find Files by Contained Characters
- Find Files by Multi Field Search

Find Files by Meta Properties
Find Files by the Length of their Name
Tab "Size"
Tab "Date"
Tab "Attributes
Tab "Tags"
Tab "Contents"
Tab "Dupes" (Duplicate File Finder)
Tab "Excluded"
Search Results Context Menu
General File Find Tips

Find Files Tab

The Find Files Tab is found on the Info Panel (F12) at the bottom of the application. The Find Files Tab itself has a couple of subtabs, the so-called filter tabs. Let's start with the main buttons and options right of the filter tabs.

Find Now / Stop (ESC)
Start a file find by clicking the Find Now button, or by pressing Ctrl+Alt+F. You can abort a running find process by clicking the Stop button or by pressing Esc (Escape Key).

Note that the button has a right-click menu which allows you to restrict the scope of the search:

- Search in Location: Same as clicking the button right-away.
- Search in Listed Items: Same as entering # into Location (see Search in List). This allows you to refine a previous search by searching the search results.
- Search in Selected Items: Same as entering + into Location (see Search in List).

Reset Filters
Resets all settings on the filter tabs to factory defaults and deactivates all filters.

Untouched remain only the Location settings (Include Subfolders etc.) and the "Search Results to tab" setting.

Tip: There is an optional confirmation prompt. Right-click the button to pop the toggle "Show Confirmation Prompt" and make your choice.

Applied filters
On the Find Files tab there are a number of sub-tabs. All but the last represent different search parameters/filters. These filters can be individually activated (applied by the current search process) by checking the corresponding checkbox below Applied filters. The tab of an active filter has a colored caption (the color "Marked Text 1" can be defined in Configuration).

A file must pass each active filter to be found. In other words, the logical relationship between the filters is AND.

Note On the first tab (Name & Location), the Location parameter is always

active. Thus, deactivating this tab means you don't care how the file is named (equivalent to leaving the Name field empty or to entering just an asterisk).

There's a bit of "smart" automation built in: If you change any setting on a given tab, it is assumed that you're interested in that filter, and it will be automatically activated.

Tab "Name & Location"

The Name field provides a space where you can state the name or a pattern that the file to be found shall match. You may use various Wildcards, Boolean Logic Operators, and Regular Expressions.

The Name field also supports Single File Search.

Tip: You can use the mouse wheel to scroll through the dropdown list items when it's collapsed. You can also expand the list by Shift+WheelDown and collapse the list by Shift+WheelUp.

XYplorer native variables are resolved in the name pattern. So you can search for things like `<U+202E>` or `<clipboard>`.

Mode

There are different Name Pattern Modes:

(1) Standard

(2) Boolean

(3) RegExp

You can easily switch between modes via the Mode dropdown list, located directly below the Name dropdown list. "Mode" refers to the way the search pattern is interpreted. For example, in "Boolean" mode the sequence `" & "` is interpreted as Boolean AND, whereas in Standard mode it would simply be interpreted as those three characters.

Note that you can alternatively select the Boolean and RegExp modes by prefixing the name pattern with : or > and thereby *overwrite* the setting of the Mode dropdown! If you do, the dropdown will be replaced by a static label reflecting the current mode.

Wildcards (Pattern Matching Special Characters)

Characters in pattern	Matches in string
?	Any single character
*	Zero or more characters
#	Any single digit (0-9)
[charlist]	Any single character in charlist
[!charlist]	Any single character not in charlist
[[]	[
[#]	#

Examples:

```
a*          Finds all items beginning with "a"
*.jpg       Finds all JPG files
*.          Finds all files without extension
a*.jpg      Finds all JPG files beginning with "a"
a*.         Finds all files without extension beginning with "a"
```

Loose Match and Exact Match

Simple patterns are interpreted as in Windows Explorer: If you state a pattern without wildcards, all files are found whose name contains this pattern as a substring (equivalent to "*pattern*"). This is called Loose Match.

Additionally XYplorer also supports Exact Match: If the pattern is quoted, only files are found, whose name exactly matches the pattern.

If any wildcards are contained in a pattern, the pattern is matched as is.

Simple multiple pattern search

You can enter multiple wildcard patterns separated by ";" into the Name field.

Examples:

```
*.jpg;*.gif;*.png         Finds all *.jpg and *.gif and *.png
*.jpg; *.gif  ; *.png     same (the blanks are cropped)
```

Loose Boolean Match (LBM)

Mimics Windows Explorer behavior, where you can enter two or more words separated by blanks and find all items that match ALL of the strings (i.e. they are AND-combined logically).

For example, "work 2006" (without the quotes) will find all items that have "work" and "2006" somewhere in their name (regardless of the order). In other words, "work 2006" is internally-smartly-automatically-conveniently transmutated into the Boolean expression "*work* AND *2006*".

- For LBM to work you need this setting ticked: Configuration | Find and Filter | Find Files & Branch View | Find Files | Enable smart Boolean query parsing
- LBM does only work in Standard Mode (see above)
- Note that OR-combining patterns (using ";", see Simple multiple pattern search above) has precedence over LBM, so the following search term

  ```
  a;b c
  ```

  is interpreted as

  ```
  "*a*" OR "*b c*"
  ```

  and not as

  ```
  "*a;b*" AND "*c*"
  ```
- Note that parentheses have to be either in quotes or escaped by a prefixed \. To match a file called

"Black & White & Red (122).txt" you can use either of these patterns:

```
"Black & White & Red (122)*"  (Parsed: name:"black & white & red (122)*")
Black & White & Red \(122\)   (Parsed: name:"*black*" AND name:"*white*" AND name:"*red (122)*")
```

- To avoid LBM and find all items containing "work 2006" somewhere in their name enclose it in asterisks: "*work 2006*".
- To avoid LBM and find all items matching "work 2006" exactly put it in quotes (a leading quote is enough).

Boolean Logic Operators and Syntax

Logical function	Symbols (alternatives)	Comments
OR	| (or, OR)	Must have a space on both ends.
AND	& (and, AND)	Must have a space on both ends.
NOT	! (not, NOT)	Prefixed to the negated term, no space after "!" one space after "not/NOT".

Default operator precedence

```
NOT > AND > OR
```

Nesting terms with parentheses

You can overwrite the default operator precedence by using parentheses (up to 256 nesting levels). Also complex terms can be negated, for example "!(cat | dog)" is equivalent to "!cat & !dog".

Examples: Sequence of operands can make a difference. Using parentheses can protect you from surprises:

```
!:a OR b      is parsed as      NOT ("*a*" OR "*b*")
:!a OR b      is parsed as     (NOT "*a*") OR "*b*"
!:(a OR b)    is parsed as      NOT ("*a*" OR "*b*")
:!(a OR b)    is parsed as      NOT ("*a*" OR "*b*")
```

How to Escape Boolean Parsing

All these examples are with Mode = Boolean.

Not escaped:

```
!*cat* = NOT cat
cat & dog = cat AND dog
cat and dog = cat AND dog
!((1) or 2) = NOT (1 OR 2)
```

Use the backslash \ to escape critical sequences:

```
\!*cat* = !cat
```

```
cat \& dog = "cat & dog"
cat \and dog = "cat and dog"
!(\(1\) or 2) = NOT ("(1)" OR "2")
```

Alternatively you may quote your patterns to protect them from parsing. You must, however, supply all needed wildcards in quoted patterns, else they are taken as "exact matches".

```
"!*cat*" = !cat
"*cat & dog*" = "cat & dog"
"*cat and dog*" = "cat and dog"
!("*(1)*" or "*2*") = NOT ("(1)" OR "2")
```

Boolean Logic Examples

Find all files that have cat or dog in the name:

```
:cat | dog
:cat OR dog
```

Find all files that have cat and dog in the name:

```
:cat & dog
:cat AND dog
```

Find all files that have cat but not dog in the name:

```
:cat & !dog
:cat AND NOT dog
```

Find all files that have neither cat nor dog in the name:

```
:!(cat | dog)
:NOT (cat OR dog)
```

Note that you can as well use RegExp patterns in Boolean expressions. The RegExps must be preceded by ">" in that case. Find all files that have ####-#[#]-#[#] in the name and have the extension *.doc:

```
:>"\d{4}-\d{1,2}-\d{1,2}" AND "*.doc"
:>"\d{4}-\d{1,2}-\d{1,2}" AND >"\.doc$"
```

Regular Expressions

XYplorer's Find Files gives you the power of Regular Expressions (RegExp).

Prefix > for RegExp To activate the regular expressions mode simply prefix the search term with the character ">". If Regular Expressions mode is active, all rules described here above do not apply. Everything after ">" is treated as RegExp. You can still put a "!" before the ">" to invert the results (logical NOT).

XYplorer supports Boolean RegExp, i.e. you may combine any number of Regular Expressions with Boolean operators. The Regular Expressions must be quoted in that case and each RegExp needs to be prefixed with ">", e.g.:

```
:>"\d{4}-\d{1,2}-\d{1,2}" AND "*.doc"
:>"\d{4}-\d{1,2}-\d{1,2}" AND >"\.doc$"
```

XYplorer uses the VBScript RegExp Object (library: C:\WINDOWS\system32\vbscript.dll) which is similar to JavaScript's regular expression flavor. For details on XYplorer's variety of regular expressions, search the web for: vbscript regular expressions.

Generic File Types

You can as well specify Generic File Types as name pattern:

```
{:Video}
NOT {:Video}
:{:Image} AND prop:#AspectRatio: 16:9
```

Checkboxes modifying the pattern matching

Match Case: Check to do a case-sensitive search (where "A" is not the same as "a"). Note that Windows Explorer can do only case-insensitive searches ("A"="a").

Whole Words: Check to do a whole word only search. Word delimiters are

()[]{}\,.;=+~-_'$%&#@, space, string start, string end

Whole word searches can be freely combined with Boolean Logic, Match Case, Check Full Path, Exact Match (implied in Whole Words anyway), and Invert.

Note that Whole Words cannot be combined with RegExp, and you cannot use wildcards: if the search pattern contains wildcards Whole Words is internally set to False.

Invert: If checked the search will return everything that does not match your search term.

Ignore Diacritics: Tick it to match a pattern "Koln" with files "Koln.txt" and "Köln.txt" and "Kóln.txt" etc. Note that all accented characters (i.e. characters with diacritics) in the name pattern AND in the tested filenames are reduced to their base characters before the matching is performed. So an accented pattern "Köln" would also match files "Koln.txt" and "Köln.txt" and "Kóln.txt" if "Ignore diacritics" is ticked.

Path: If checked, the search term is compared with the whole path/name instead of the name only.

The term "!\temp\" will exclude all items inside/below a "temp" folder from the results.

The term ":\keyboard or \mouse" will find all items inside/below a "keyboard*" or "mouse*" folder.

The term ":*\Inbox*.msg and not *\Inbox**.msg" will find all *.msg files that are located directly inside all folders named "Inbox".

Note: When the pattern contains backslashes ("\") the Path checkbox does not have to be explicitly checked to trigger path matching. See Examples for implicit path matching.

Find Hidden: Tick it to find all items regardless of any settings that may currently hide them.

This option is a GUI-based alternative to appending the /a switch to the search pattern (see details here). So, with "Find hidden" checked, none of the following settings will hide anything from the search results listing:

- Configuration | General | Tree and List | Items in Tree and List | Select Items... | Hidden files and folders
- Configuration | General | Tree and List | Items in Tree and List | Select Items... | System files and folders
- Configuration | General | Tree and List | Items in Tree and List | Select Items... | Protected operating system files
- Configuration | General | Tree and List | Items in Tree and List | Select Items... | Junctions
- View | Show Items | Show Folders in List

Switches

You can append various switches to the Name pattern, for example to limit the search depth. See details here.

Type

Right of the Name field there's the Type Filter, a dropdown where you can select a general file type to narrow down the search results to only that type of files. Some of the available file types exactly correspond to the lists under Configuration | Previewed Formats and even can be extended there.

```
All types (= no type filtering at all)
Text files
Image files
Photo files
Audio files
Video files
Media files (= Audio files + Video files)
```

Font files

Vector files

Web files

Document files

Archive files

Executable files

Notes

Any extensions excluded from Preview in Configuration | Previewed Formats are nevertheless used by the find files type filter.

Note that the types partly overlap, e.g. *.txt is in "Text files" and in "Web files".

Apart from "All types" all type filters will return only files, not directories.

Tips

To quickly list e.g. all Images simply select type "Image files" and leave the Name field empty.

Focus the Type Filter field and press F1 for a listing of the contained file types.

The Type Filter field and dropdown support a simple Type Ahead Find. Simply type "t" to select "Text files", "v" to select "Vector files" or "Video files", etc.

Location

The Location field is automatically set to the current location in Tree (if Auto-Sync is checked, see below). You may also enter a path manually or select a path from the dropdown list. When you do a search, the search location is stored in the dropdown list and preserved between sessions. Up to 64 locations are remembered.

Include subfolders

Check this box to include in the search all folders below the Location location.

Maximum depth

Check it to limit the recursion depth of the search. The adjacent dropdown offers 16 depth levels (plus level "0" which is identical to "do NOT include any subfolders). E.g. maximum depth 2 means: Search the current location and 2 levels deeper.

Note: The same functionality is also available as a Search Pattern Switch: `/maxdepth` (or short form `/md`). Now you have it in the GUI as well. If present, the `/maxdepth` switch overrides the GUI setting.

Follow folder links

Similar to "Include subfolders", the search will treat folder links (aka "Shell Links to Folders") as if they were subfolders and search the links' target directory as well (again including subfolders if necessary). This is a very nice feature. For example, you can cultivate folder link farms, i.e. collections of links to

various folders in the wildest locations, which you then can search all at once by just running a search on their parent folder.

How to create a folder link: have the folder in the file list, then choose the Create Shortcut command from the folder's context menu, then move the newly created *.lnk file where you want to have it.

Notes

"Follow folder links" is independent of "Include subfolders" and both can be freely combined.

Note that there's a simple recursion blocker to avoid severe trouble: If a folder link points to a location that again has a folder link, that 2nd folder link is not followed anymore.

Selected Locations

Check this box to enable Multiple Location Search (via selection in the List) If any folders (or folder links!) are selected in List, the find operation will search these folders (thus ignoring what's currently entered into the Location dropdown). If no folders are selected, things work as before: Find searches the currently selected folder in Tree, or the selection in the Location dropdown (if different from Tree selection).

Note that multiple location search also works for find results. Here it is possible to select folders for a further search where one is a subfolder of the other. In this case, of course, you will probably get duplicates in the result list.

Multiple location search works also for drives, so if you want to search for example drives C:, E:, and Z:, simply browse to Computer, select these drives in the list, and click Find (F3).

Auto-Sync

Check this box to always (and now) automatically set the location field to the current location.

Multiple Location Search

All Drives Search: To perform a search that covers multiple drives at once, just type "This PC" (or whatever you've named it on your system) in the Location drop-down, or even easier: select the top node of the tree and you're done. All drives with a letter will be searched. To exclude specific drives add their name to the Exclude Folders list. For example, to exclude drive D, add D:\ to the list.

Location field: Another way to do a search over a bunch of different directories is to simply type a list of directories, separated by "|" (pipe) or " | " (space-pipe-space; the spaces are optional), into the location dropdown, for instance: C:\John\Letters\ | C:\Contracts\ | D:\Work\. Alternatively to "|" you can use ";" (in that case paths can and have to be quoted if they contain ";").

Edit Locations: An easier way to achieve the same is using the Edit Locations dialog that is reached via the "..." button's right-click menu. Here you can edit the search location(s) like in an editor. The dialog has its own Browse button. Selected locations are appended to the ones already present in the exit box. The Browse dialog is preselected to the last location in the box.

Tip: Multi Location Searches also support CRLF (Windows newline sequence Carriage Return + Line Feed) as an alternative separator. This allows for human-friendlier lists. To enter such list you need to use a variable, of course, e.g. `<clipboard>`.

To activate the Name parameter the corresponding checkbox below the Find Now button must be checked. Location and Include subfolders are, of course, always active.

Search In List

A search mode where you don't search in one or more locations but in a list of items (;-separated) passed in lieu of the search locations.

This list can contain files (they are directly checked against the search filters) and folders (subfolders are included like in a normal (multi-)location search).

The Search In List (SIL) mode is invoked automatically if any files are found in the list of search locations. No switch, no setting. It just works.

So, in other much simpler words: You can pass files as search locations and they are directly checked against the search filters. It's a simple idea but what you can do with it is pretty awesome.

Here are some examples:

- Some Quick Search lines to paste into the Address Bar (`<selitems ;>` returns a list of all selected items, separated by ";"):

```
E:\Test.jpg;E:\Test2.txt?ageC: > 1 y  //of these, list those older than 1 year
<selitems ;>?a*       //among the selected items, find all starting with "a"
<selitems ;>?* /e     //among the selected items, find all that are empty
<selitems ;>?cmt:"Done" //among the selected items, find all with comment "Done"
```

- You can also run a search and then another search over the search results of the first search, using scripting. For example, list all JPGs, then scan the listed JPGs for being last modified in 2014 (`<allitems ;>` returns a list of all items, separated by ";"). Again, paste this into the Address Bar:

```
goto "?*.jpg"; goto "<allitems ;>?dateM: y 2014";
```

  Note that the `<allitems ;>` separator ";" should not be present in any of the item names for the example to work correctly. To be on the safe side use `<allitems |>` or # (see below) instead.

- Or you can run a search over a hard set of files stored in some file, using a small script, fast non-GUI operation. ListToSearch.txt contains a list of ;-separated files with full path:

```
// return files modified in the last 7 days:
  $list = readfile("<xyscripts>\ListToSearch.txt");
  text quicksearch("ageM: <= 7 d", $list);
```

  One fascinating aspect of the above example is that you can super-quickly check out multiple items in the wildest locations all over your system without the need to browse the whole locations as in a normal find operation.

- Store a search result in a permanent variable and list it anytime later:

```
perm $results = "<allitems ;>"; //store today
goto "$results?*"; //restore tomorrow
```

  Or you could just make up $results manually by whatever criteria and then list it in the file list using:

```
goto "$results?*";
```

Paper Folder to go.

- The character # works as a shorthand for `<allitems |>`, and the character + works as a shorthand for `<selitems |>`. These characters are supported in the Location field of Find Files.

```
#     = Search all items in the current list
+     = Search the selected items in the current list
```

Some examples:

To search the current list (which can be a search results list) for all items last modified in 2014 you can do this through the Address Bar:

```
#?dateM: y 2014
```

To search the selected items in the current list (which can be a search results list) for all items last modified in 2014 you can do this through the Address Bar:

```
+?dateM: y 2014
```

Further Remarks on Search in List

- To make the search repeatable the searched list has to be stored in tabs and in certain setting files on disk, just as if it was a simple search location. However, such a list can easily get *very* long. It's your responsibility to use this feature with prudence.
- If a location term is longer than 32767 characters the Location field cannot display the string anymore and will appear empty. This is a Windows display limitation. It does not affect the functionality.
- Searching a list of 100 items will take notably longer than searching a location with 100 items. This is technically unavoidable and the price for the power of this feature.
- The switch /l (lower case L) is necessary when your passed list of search locations contains only folders, no files, AND you want to check these folders against the search criteria. For example, if ListToSearch.txt contains only folders, this will return all of them starting with "m" and nothing else:

```
text quicksearch("m* /nl", readfile("C:\Temp\ListToSearch.txt"));
```

Note that you get *implicit* Search In List whenever the list of passed search locations contains at least one file. In that case the switch is without any effect.

Comparison Operators

The Name field supports comparison operators. This means you can e.g. list all files from A to M.

`<`	Less than
`>`	Greater than
`<=`	Less than or equal to
`>=`	Greater than or equal to
`!=`	Not Equal
`==`	Equal (only there for completeness; is always optional)

The operators must be surrounded by spaces and are prefixed to the patterns.

Examples

When strings are compared it's about alphabetical order. Find all files before "m":

```
 < m
```

Find all items from "d.txt" through "m.txt". Watch the double space after AND: Boolean " AND " must be surrounded by spaces as well! The leading ":" is the inline Boolean search mode marker.

```
: >= d.txt AND  < m.txt
```

When numbers are compared it's about numerical values. Find all items whose full name (incl. path) is longer than 100 characters:

```
len: > 100
```

Notes on using Comparison Operators

- Patterns with comparison operators should not contain wildcards; they make no sense here anyway and will just be treated as characters "*" and "?" in the comparison.
- The setting "Whole words" is ignored with patterns using comparison operators. But, of course, the settings are applied with non-comparison-patterns. E.g., with "Whole words" enabled, the following term will match "a d.txt" but not "ad.txt", and not "d a.txt" (because it's > c.txt):
  `:a AND  < c.txt`
  In words: Find all items that contain "a" as a whole word, and are alphabetically sorted before "c. txt".
- It does not make sense to combine comparison operators with RegExp patterns; the RegExp will not be analyzed but simply be used as a string in the comparison.

Find Files by Size

You can find files by size using the field prefix (selector) "size:". Sure, there is also the "Size" tab (see below) but this enhancement opens find-by-size to the Quick Search and hence also to Scripting.

Examples

- Find all files exactly 999 bytes long: `size: 999` or `size: == 999`
- Find all files of 1 MB or larger: `size: >= 1MB`
- Find all folders (here folders have no size): `size:`
- Find all empty files: `size: 0`

The Size field is particularly useful for reducing the work when searching for files with a certain hash value. When you happen to know the exact size or at least the approximate size of the files you are looking for you can save a lot of time. For example:

- Find all files between 5MB and 6MB AND with a certain hash value (the first colon marks Boolean mode):

```
:size: > 5MB AND size: < 6MB AND prop:#Hash.MD5:db24bcdf4098bda916a949f5554d2110
```

Find Files by Date and Age

You can find files by date and age using the field prefixes (selectors) "dateC:", "dateM:", "dateA:" and "ageC:", "ageM:", "ageA:" (CMA are for Created, Modified, Accessed). Sure, there is also the "Date" tab (see below) but this enhancement opens finding files by date to Quick Search, to Scripting, and to Click and Search.

Examples

These examples are ready to paste into the Quick Search box, but beware: valid date syntax depends on your system locale.

```
dateM: 20.05.2014 16:16:40       = exact timestamp
dateM: == 20.05.2014 16:16:40    = same
dateM: >= 20.05.2014 16:16:40    = then or later
dateM: 22.05.2014                = covers the whole day
dateM: 05/22/2014                = same
dateM: 19.05.2014 - 20.05.2014   = covers two whole days
dateM: 05/19/2014 - 05/20/2014   = same
dateM: 2014-05-19 - 2014-05-20   = same
dateM: dw 6-7                    = on a weekend (Saturday and Sunday)
dateM: h 11                      = at eleven o'clock
ageM: d         = modified today
ageM: w         = modified this week
ageM: m         = modified this month
ageM: 1 d       = modified yesterday
ageM: 1 w       = modified last week
ageM: <= 30 n   = modified in the last 30 mins
ageM: <= 3 h    = modified in the last 3 hours
ageM: <= 7 d    = modified last 7 days
```

Notes

- Prefix "date:" works as an alias for "dateM:".
- Prefix "age:" works as an alias for "ageM:".

Find Files by Attributes

You can find files by attributes using the field prefixes (selectors) "attr" for single attributes and "attrlist" for a combination of attributes (represented by single letters).

So, for example, now you can search for read-only items like this:

```
attr:readonly
attr:r
```

If you look for a combination of attributes, e.g. readonly AND directory, you do it like this:

```
attrlist:rd
```

Find Files by Tags (Labels, Tags, Comments)

You can directly search for Labels, Tags, and Comments. Simply prefix the selector "lbl:", "tags:", or "cmt:" (case does not matter) to the search term in the name field, and the whole term is matched against the label, tags, or comment of each file instead of against the name.

All other things work as usual, including Boolean and RegExp search.

Examples

- `lbl:#1`
  find all tagged items with label #1 (whatever it is named)
- `lbl:"red"`
  find all tagged items with label "red"
- `lbl:!"red"`
  find all tagged items with no label "red"
- `lbl::"red" or "green"`
  find all tagged items with label "red" or "green"; the second ":" after "lbl:" is the inline Boolean search marker
- `lbl:"red";"green"`
  same using "Loose Boolean Match" OR syntax
- `lbl:r`
  find all tagged items with "r" in the label
- `lbl:r a p`
  find all tagged items with "r" and "a" and "p" in the label, employing the "Loose Boolean Match" syntax with " " separator as AND
- `tags:cats`
  find all tagged items with a tag "cats"
- `tags:cats,dogs`
  find all tagged items with a tag "cats" AND a tag "dogs" (see more examples here below under Examples for searching tags)
- `tags:""`
  find all tagged items with no tags
- `tags:""quotedtag""`
  find all items with tag "quotedtag" (including the quotes)
- `cmt:2008;2009;2010`
  find all tagged items with 2008 or 2009 or 2010 in the comment field, employing the "Loose Boolean Match" syntax with ";" separator as OR

Examples for Searching Files by Tags

Loose and Full Boolean: Of course, Tags call for Boolean search terms. This is very easy to do:

```
Name Pattern          Find all items with
-------------------------------------------------------------
tags:cats             tag "cats"
tags:cats;dogs        tag "cats" OR "dogs"  (loose Boolean syntax)
tags:cats dogs        tag "cats" AND "dogs" (same)
tags:"cats dogs"      tag "cats dogs"
tags:"cats dogs";ants tag "cats dogs" OR "ants"

tags::cat* OR dog*    tag "cat*" OR "dog*"  (full Boolean syntax, with Boolean marker
: prefixed)
tags::cat* AND dog*   tag "cat*" AND "dog*" (same)

tags:?*               any tag
```

Comma as AND, Pipe as OR: There is a special shorthand syntax that's only applicable for the tags selector: You can use the comma as an AND operator, and the pipe as an OR operator. Here are some examples, including possible combinations with Loose Boolean operators ; (OR) and [space] (AND):

```
tags:cats,dogs,ants   tag "cats" AND "dogs" AND "ants"
tags:cats,dogs;ants   tag ("cats" AND "dogs") OR "ants"
tags:cats|dogs|ants   tag "cats" OR "dogs" OR "ants"
tags:cats|dogs ants   tag ("cats" OR "dogs") AND "ants"
tags:"birds,m m"      tag "birds" AND "m m"
```

The quotes in the last example are necessary because of the space in "m m".

Note that you cannot mix commas and pipes.

Wildcards in Tags

Tags can have wildcards ("*" or "?") and still be searched for in an easy manner. To enable searching for wildcards you use the switch `/v` ("verbatim") which tells the interpreter that "*" and "?" in the search term shall *not* be taken as wildcards but as normal characters. The verbatim switch is exclusively used for label, tags, and comments search. Here are some examples for Quick Searches passed through the Address Bar and searching the current location:

```
Address Bar      Finds (including subfolders) all items with
-----------------------------------------------------------
?tags:a* /rv     a tag "a*"
?tags:a* /r      a tag matching pattern "a*"
?cmt:"*" /rv     a comment "*" (note that "*" must be quoted in the term;
                 a single non-quoted "*" means "ignore comments")
?a /v            all items named exactly "a"
```

Note that, of course, even without the verbatim switch you can search for wildcard characters by enclosing them in square brackets. For some jobs there is no other way, for example to find all items with a "*" anywhere in the tags:

```
?tags:*[*]* /r
```

Search only among Tagged Items

Append the switch `/t` to the search term to confine the search to all items present in the tags database (tag.dat). Examples (ready for pasting into the Address Bar):

List all items (excluding any orphans) in the database:

```
*?* /t
```

List all TXT items in the database:

```
*?*.txt /t
```

List all tagged items in the current location with no comment:

```
?cmt:"" /t
```

List all tagged items in the current location without comment AND number 5 in ex2:

```
?:cmt:"" and ex2:5 /t
----------------------------------------
? = search
 : = Boolean term
  cmt: = comment
      "" = empty
         and = Boolean AND
             ex2: = extra tag 2
                 5 = value 5
                   /t = in tags database
----------------------------------------
```

Invert the previous results: List all tagged items in the current location NOT without comment AND number 5 in ex2:

```
?!:cmt:"" and ex2:5 /t
```

Notes

- Quick Search supports this syntax, for example:
  `E:\Test\?lbl:"blue";"red"`
- The selector "comment:" is a valid alternative to "cmt:".
  Patterns after the "comment:" selector are automatically wildcarded (internally surrounded by asterisks to allow partial matches) if no wildcards (* or ?) are passed. To force an exact match (no wildcards), enclose the pattern in quotes.
- The selector "tag:" is a working (but deprecated!) alternative to "lbl:" (for historical reasons).
- Tip: The asterisk "*" is an alias for "Computer" as location. For example, to find all items on all local

drives with a comment containing "India" you simply run this through the Address Bar:
`*?cmt:India /r`

- Tip: To find all items on your computer with any tags defined run this through the Address Bar:
  `*?tags:?* /r`

Find Files by Columns

You can prefix any column header (canonic name or current caption) to the search term (separated by :), including Custom Columns.

Examples:

```
Ext: jpg
Ext: j*g
Size: >= 500000
Size: 529 KB
Modified: 2014-09-21 12:55:09
```

Find Files by the Size Column Contents

You can pass file sizes with units in the search term. Local thousand and decimal separators are handled.

Examples:

```
Search Pattern          Notes
-------------------------------------------------------------------------
Size: 529 KB            assumes "KB (rounded up)" as unit
Size: 528.49 KB         assumes "KB" (= KB rounded to 2 decimal places) as unit
Size: <= 528.49 KB      assumes "KB" as unit
Size: >= 500,000        assumes "B" (bytes) as unit
Size: >= 500000         ~ same as above ~
```

Works independently of the current Size Column Format in the list.

Find Files by Properties

You can also find files by Shell Properties and Special Properties. Works independently of any currently visible columns. Simply use the property name as selector, for example:

```
Item type: TXT File
MD5: d41d8cd98f00b204e9800998ecf8427e
Aspect Ratio: 2:3
Aspect Ratio: v2
```

Alternatively you can use the internal selector (if you know it ... see Tip below). It is locale

independent and pretty safe against ambiguities (there could be one or more Custom Columns named "MD5" which would make the actual outcome of the simple forms above unpredictable):

```
hash.md5: d41d8cd98f00b204e9800998ecf8427e
aspectratio: 2:3
aspectratio: v2
```

Tip: You can show the internal Special Property selectors (e.g. "hash.md5") in the "Select Special Property" dialog if you hold CTRL down while clicking "Select Special Property..." in the right-click menu of a user column header (user columns can be added to the native columns via *View* | *Columns* | *Add Column*). They are canonic, i.e. locale independent.

Another way is using the prop: selector (see Find Files by Meta Properties). It is more clumsy and harder to remember but even more safe against ambiguities:

```
prop:#hash.md5: d41d8cd98f00b204e9800998ecf8427e
prop:#aspectratio: 2:3
prop:#aspectratio: v2
```

Find Files by Contents

You can find files by contents using the field prefix (selector) "cont:" or "content:". Sure, there is also the "Contents" tab (see below) but this enhancement opens finding files by contents to Quick Search, to Scripting, and to Click and Search.

Use the cont: or content: selector to compare the pattern with the textual contents of the scanned files (folders are ignored). It's a simple vanilla text comparison supporting wildcards * and ?, equivalent to the following settings on the Contents tab:

Mode: Wildcards
Type: Text
All else unchecked.

So the search is case-insensitive (A==a).

Examples

```
                          //matches all files...
cont:chicken              //containing "chicken" (or "CHICKEN") anywhere
cont:chicken*             //containing "chicken" at the beginning
:cont:chicken or dog      //containing "chicken" or "dog" anywhere
:chicken or cont:chicken  //containing "chicken" in the name or the contents
cont:"chicken"            //with content == "chicken"
cont:*                    //that are content searched (e.g. PNG files are not)
cont:?*                   //that are content searched and not empty
cont:""                   //that are content searched and empty
```

Of course, it also works in Quick Search:

```
?cont:chicken             //containing "chicken" (or "CHICKEN") anywhere
```

Remarks

- Remember that content search can take time.
- This feature opens content search to Boolean logic in a Multi Field Search.

Find Files by Contained Characters

You can find files by contained characters using the field prefix (selector) "contchar:". Use it to find files that contain certain characters or character ranges within their textual contents (non-textual files and folders are ignored).

Syntax

```
contchar: min - max    = from min to max (both inclusive)
contchar: min -< max   = from min to max (max exclusive)
contchar: min >- max   = from min to max (min exclusive)
contchar: min >< max   = from min to max (both exclusive)
contchar: > n          = bigger than
contchar: >= n         = bigger than or equal
contchar: < n          = smaller than
contchar: <= n         = smaller than or equal
contchar: == n         = exact (identical to the next example)
contchar: n            = exact
```

Notation

- `min` = lower Unicode value
- `max` = upper Unicode value
- `n` = Unicode value
- Unicode values can be stated as decimal or hex numbers (eg `0xFF`).
- Instead of a number representing a Unicode value you can also pass a single character. Both ways can be mixed.
- Spaces are optional.

Remarks

- Only Text files and Document files are scanned. Where possible the text is extracted via IFilters, otherwise code pages are applied just as in other places of XY.
- On a logically inverted search (eg "`!contchar:>=256`" = find all files NOT containing characters beyond ANSI) only files are returned that actually have any textual contents. Not folders or images etc.
- The function is fast, but a lot of work has to be done. So when huge files are scanned it will take a while.
- Files are scanned from beginning to end. If the first character is found that matches the criteria the scanning stops and the file is counted as a match.

- Only double byte UTF-16 Unicode values are supported. The valid range is from 0 to 65535 (0x0 to 0xFFFF).

Examples

```
                        // matches all text/office files containing any...
  contchar:9            // TAB characters
  contchar:A-Z          // characters from A to Z
  contchar:65-90        // (same as above)
  contchar:A-90         // (same as above)
  contchar:>=128        // characters beyond ASCII
  contchar:>=256        // characters beyond ANSI
  contchar:0x1800-0x18AF  // Mongolian characters (U+1800-U+18AF)
```

If "Enable extended pattern matching" is ON then logical inversion like this works:

```
  !contchar:>=256         // all files not containing any characters beyond ANSI
  !contchar:0x1800-0x18AF // all files not containing any Mongolian characters
```

Find Files by Multi Field Search

In a Boolean search you can also mix different fields, e.g. find all images with 1024 x 768 pixels AND labeled "Red" AND having "2012" in the name.

The available fields are:

```
  name: Name (default field)
  path: Path (the full path excluding the item itself)
  size: Size
  lbl:  Label
  tags: Tags
  cmt:  Comment
  cont: Contents (or content:)
  prop: Property
  len:  Length of path and file
  lent: Length of file title (no path) only
```

Examples:

```
   Name Pattern               Find all items with
   ------------------------------------------------------------------
1) tags:cats,dogs OR name:a*   tags ("cats" AND "dogs") OR name "a*"
2) tags:cats|dogs AND name:a*  tags ("cats" OR "dogs") AND name "a*"
3) tags:cats|dogs AND ants     tags ("cats" OR "dogs") AND tag "ants"
```

```
4) xy*.exe AND prop:fileversion:10.80*    (self-explaining)
5) *.jpg AND lent: > 12        JPGs with long filenames
6) *.jpg AND size: > 10MB      JPGs larger than 10 MB
7) path: 2014                  all items with *2014* anywhere in the path
8) path: mars OR saturn        all items with *mars* or *saturn* anywhere in the path
9) *.jpg AND path: != \\saturn\\   all JPGs with folder "saturn" NOT anywhere in the path
```

Field Type Inheritance

The 3rd example above shows that the field type is inherited from left to right. This is done for backward compatibility and also for comfort. The following pairs are equivalent:

```
lent: > 12 AND  < 20
lent: > 12 AND lent: < 20

tags:cats|dogs AND ants OR name:a* AND *.png
tags:cats|dogs AND tags:ants OR name:a* AND name:*.png

a* AND (*.png OR lbl:green OR yellow OR red)
name:a* AND (name:*.png OR lbl:green OR lbl:yellow OR lbl:red)
```

Boolean RegExp

Multi Field Search is also supported in Boolean RegExp, for example:

```
:>.*test.* AND NOT >.*not.*test.* OR lbl:>^Gre.*
```

Further Remarks on Multi Field Search (MFS)

- Of course, MFS also works in Quick Searches which makes it available for Address Bar, Catalog, Favorites, User Buttons, Scripting etc. Remember that the Boolean marker ":" has to be prefixed, e.g.:
  `?:tags:dogs and name:t*`
- The Match Case checkbox applies to all operands of the MFS term.
- The Whole Words checkbox applies to all operands of the MFS term, with the exception of tags (tags:), which are always treated as whole words unless they are stated with wildcards.
- The size: selector will only match files, not folders. E.g., this pattern returns all small files:
  `size: < 1000`

Find Files by Meta Properties

You can directly search for the extended properties of file items, also known as Meta Properties. Their names and indices vary strongly between different Windows versions. A list is available in Configuration | File Info Tips & Hover Box (those properties are also featured in the File Info Tips). The syntax is simple: Prefix "prop:[propertyname]:" to the search term. To identify properties by their index use "#[index]" as propertyname.

Examples

```
prop:fileversion:8.60       // find all items with version *8.60*
prop:fileversion:"8.60"     // with version 8.60 (exactly)
prop:#8:Donald              // with owner *Donald*
prop:#26:"1024 x 768"       // with image dimensions "1024 x 768"
prop:#AspectRatio: 16:9     // with aspect ratio 16:9, see also Special Properties
prop:#Hash.MD5:58172fd15526e1249ac8bfd690ced076    // with a certain MD5 hash
prop:#empty:5         //finds all non-empty files
prop:#nosubs:2        //finds all folders without subfolders
prop:#nosubs:6        //finds all folders with subfolders
prop:#itemcount: >= 11      //finds all folders containing at least 11 items
prop:#itemcount: != 5       //finds all folders not containing exactly 5 items
prop:#tag.composer:Haydn    //finds all audio files with Composer tag containing "Haydn"
```

Tip: The latter example should be logically combined with other criteria to increase the speed (by avoiding unnecessary lengthy calculations), e.g.:

```
:*.png AND prop:#Hash.MD5:58172fd15526e1249ac8bfd690ced076
:size: > 5MB AND size: < 6MB AND prop:#Hash.MD5:db24bcdf4098bda916a949f5554d2110
```

Tip: prop:, also supports the locale-independent Windows canonical properties as listed here:

https://learn.microsoft.com/en-us/windows/win32/properties/props

For example:

```
prop:System.Image.HorizontalSize: > 300
```

Note that these canonical properties are case-sensitive.

Find Files by the Length of their Name

Simply prefix "len:" to the search term in the name field. For example, to find all items whose full name is longer than 260 characters:

`len: > 260`

Find all items whose title (name excl. path) is:

`lent: < 12` shorter than 12 characters

`lent: 12` exactly 12 characters

Tab "Size"

Search files of a certain size, defined by a lower and an upper limit (both limits are inclusive). Leave field empty to ignore a limit, e.g. to find all files bigger than 1 MB select the MB option, enter a 1 in the at least field and leave the at most field empty.

Note that you may also state fractions, e.g. 0.0001 GB (= 100 KB), or 0.000000000001 TB (= 1 byte).

FAT32: the highest allowed value is 2 GB (= 2,048MB = 2,097,152KB = 2,147,483,648 bytes) minus 1.

NTFS: you can search files of up to 1 TB (1024 GB, 2^40) by size filter.

Search for folders as well: If you use this setting then the search results will display the folder sizes, even if they are not shown by general user settings. BTW, finding emtpy folders is even easier now than before. Simply set "At most" to 0 (zero) and check "Search for folders as well". Found are all folders that are empty (displayed as "[Empty]") or contain a total of zero bytes (displayed as "0").

To apply this filter the corresponding checkbox below the Find Now button must be checked.

Tab "Date"

Search files of a certain date/time. There are three types of dates associated with Windows files and folders:

Created: The date when the file has been created at this location, where location is the path of the file (this date is only set once).

Modified: The date when the file has been last modified (set at each write operation).

Accessed: The date when the file has been last accessed (set at each open operation).

Created or Modified: Select this option to find all files that pass either of these filters.

In the last: Find file relative to Now.

From start of unit: When checked, the lower margin is set back to the start of the chosen unit (eg days start 00:00 midnight, weeks start Monday). Note that this setting does also apply to the "and/add"-field when filled with relative time units!

Between ... and/add: Here you can enter a date range manually. When you omit the time part of the date, 00:00:00 is assumed. Fractions of a second (up to 7 decimal digits) are supported.

Here are some keyboard tricks:

When the fields are empty:

```
key arrow up:   set to today
key arrow down: set to yesterday
```

Else:

```
key arrow up:   scroll days up
key arrow down: scroll days down
key arrow up + SHIFT:   scroll months up
key arrow down + SHIFT: scroll months down
```

Date Picker

To the right of each date field you find a button to pop the Date Picker. Here are some keyboard tricks:

- Left/Right: Previous/Next day.
- Up/Down: Previous/Next week.
- PageUp/PageDown: Previous/Next month.
- Ctrl+PageUp/Ctrl+PageDown: Previous/Next year.
- Ctrl+Left/Ctrl+Right: Cycle first day of the week. You may as well right-click the Day Bar to select the first day of the week by menu.

UTC Support (Universal Time Search)

By appending "Z" (or "z") you can mark the time expression as UTC (Coordinated Universal Time). This way you can find files by their UTC date (as it is internally used by NTFS) and use the same search pattern/template regardless of the time zone of the host system. Pretty cool, especially if you are a pilot (or love a pilot).

For example, this will match all files modified (or whatever date you are looking for) in Bamako between 08:20 and 08:40 Bamako time, no matter where on the planet you are now:

```
Between: 2013-05-14 08:20:00Z
And:     2013-05-14 08:40:00Z
```

The ISO 8601 formats for UTC and time offsets from UTC are supported. The following time expressions all refer to the same point in time:

```
2013-05-14 10:00:00+02:00
2013-05-14 08:00:00+00:00
2013-05-14 08:00:00Z
2013-05-14 03:00:00-05
2013-05-13 23:00:00-0900
2013-05-13 20:00:00-12:00
```

For example, to find all files modified in Cologne today between 16:00 and 17:00 Cologne local time (DST in effect), you can use these time strings, regardless of the time zone of your current host system:

```
Between: 2013-05-14 16:00:00+02
And:     2013-05-14 17:00:00+02
```

The +02 means: Cologne (DST in effect) is 2 hours later than UTC.

Date Range Popup Menu: For the lazy ones, there are some preset date ranges available to save you some typing. Click the menu button to enter a date range by selecting an option of the date range popup menu. The Unknown option allows you to search for files with the time-stamp "unknown" (that is, no date set on this file, which is sometimes the case with certain system files and folders). The Like option sets the date range to the date of the last selected file or folder plus/minus 10 minutes.

Check Not if you want to find files whose specified date is outside the given range.

To apply this filter the corresponding checkbox below the Find Now button must be checked.

Tab "Attributes"

Search files with or without certain attributes. For example, if you don't want to find any folders, check Directory and the corresponding Not. If you want to find only hidden files, check Hidden.

Note that Normal, Temporary, Compressed, Offline, Not Indexed, Encrypted, Pinned, Unpinned, and Recall are supported only by Windows NT/2K/XP, and higher.

Tip: For your nerdy pleasure, the attributes checkboxes show the value of each attribute in the tooltip.

To apply this filter the corresponding checkbox below the Find Now button must be checked.

Tab "Tags"

Find files with specific Labels, Tags, or Comments. Like all filters they are connected by logical AND, so e.g. if Labels and Tags are both ticked then both must match else the file in question will not be among the search results.

All three fields are treated case-insensitive (A==a) and support wildcards. Here are some field-specific details:

Filter "Labels"

Supports stating multiple labels combined by logical OR (|); surrounding spaces are ignored. Examples:

```
green
Green
green|red
green | red
g*|*e*
```

Logical AND is not supported, of course, because each file can only have one label by definition.

A "Select Labels..." button is added for comfort.

Filter "Tags"

Supports stating multiple tags, combined by either logical OR (|) or logical AND (,); mixing both operators is not supported; surrounding spaces are ignored. Examples:

```
dogs
Dogs
dogs|cats     dogs OR cats
dogs | cats   dogs OR cats
dogs,cats     dogs AND cats
dogs, cats    dogs AND cats
```

A "Select Tags..." button is added for comfort.

Filter "Comment"

Wildcards (*) are auto-added left and right if no wildcards (* or ?) are contained in the pattern (aka "loose match").

Search everywhere

Tick it to search the whole computer. It will overwrite what is stated in the Location field on the "Name & Location" tab.

Finding Empty Fields

Contrary to the Name field, leaving the fields on the Tags tab empty is NOT equivalent to "*" (match all) but means "empty". For example, to find all items that are tagged (are present in the tags database) but have no comment: Simply tick the Comment checkbox and leave the Comment field empty. Same way works for Labels and Tags.

Speed

All searches including the Tags filter are ultra-fast because they work directly on the tags database, i. e. they are indexed searches.

Tab "Contents"

Find files containing a specific text string, aka content search.

Mode. You can choose between various modes, which refers to the way the given search pattern is to be interpreted.

1. Normal: The pattern should match any part of the file contents.

2. Whole Words: Like Normal, but the matching string should be a whole word, not just a part of a word.

3. Wildcards: The pattern contains wildcards (* or ?). For example, the pattern drag*drop will find all files that contain the words "drag" and "drop" in that order and with any characters in between them.

Notes:

(1) Since larger files are read chunkwise (performance!) there are certain limits to the scope of your pattern: In files larger than 32,768 bytes, a pattern "a*b" is guaranteed to be found if "a" and "b" are within a range of 1,024 bytes! It also *might* be found if "a" and "b" are within a range of 1,025 to 32,768 bytes! It will certainly not be found if "a" and "b" are only within an even larger range.

(2) Just like with non-wildcard searches it is assumed that you look for a match anywhere within the file, so asterisks (*) are silently added to your pattern at both ends in case they are missing. This is just to spare you typing too many asterisks. So you may simply enter "D?g" to look for "D?g" anywhere in the file (match file contents with "*D?g*").

(3) You need to check Wildcards (see below) to enable pattern matching support.

4. RegExp: The pattern is a Regular Expression (aka "Grep"). Examples:

- Match 2011-10-20 or 2011-1-2 etc.: `\d{4}-\d{1,2}-\d{1,2}`
- Match lines that begin with "New": `^New` (but see Note (1) below)
- Match lines that end with "*/": `\*\/$`

Notes:

(1) If Type (see below) is set to Binary then RegExp will work in "single line mode" (otherwise it's in "multi-line mode"). This means a pattern like `^\xFF\xD8` will now match only the beginning of the file, not the beginning of any line in the file.

(2) Since larger files are read chunkwise (performance!) there are certain limits to the scope of your pattern: In files larger than 64 MB, a pattern is guaranteed to be found if it matches a range of 1,024 bytes! It also *might* be found if it matches a range of 1,025 to 64 MB bytes! It will certainly not be found if it matches only an even larger range.

Type. You can specify the nature of the searched content.

1. Text: Match textual data in textual documents (DOC, PDF, TXT etc), and (only if tweak FindContentMetadata=1) the meta data of media and image files. Textual/meta data from complex documents is only retrievable if supported by installed IFilters. Non-textual files are skipped. This is probably what most users expect to happen when they do a content search, and it's usually much faster than searching the binary content of every file (see next option).

2. Binary: Match the raw bytes of all sorts of files. Rather interesting for programmers.

3. Text and Binary: Combination of both. First textual data are checked for textual documents, if no match is found or it's a non-textual file then binary data are checked.

Note: Both "Text" and "Binary" include automatic UTF-8 decoding for files types where UTF-8 encoding is a possibility. Also BOM-less UTF-8 is supported if enabled by ticking Configuration | Preview | Text preview | UTF-8 auto-detection. UTF-16LE is supported if the file has a BOM. UTF-32 is not supported.

Match Case: If checked string search is case-sensitive, else case is ignored.

Invert: Check it to find all files NOT containing the given text string. Note that the Contents filter never returns folders (although they naturally do NOT contain any given text string).

It's a Hex String: Activates Hex mode. Check this option to look for any binary sequence contained in a file, including the null char. Binary sequences are to be represented in hexadecimal code. Hex strings must have 2 valid hex characters per byte and each pair may be separated from another by spaces or line feeds. Example: 00C0FF or 00 C0 FF (lower case letters are also ok: 00 c0 ff). In hex string mode the coloring and font of the text is changed to make it dramatically clear that you are in a special mode. See Advanced Hex Search below.

Metadata: Tick it to search any embedded metadata in image and media files. This can be e.g. EXIF data. Note that metadata extraction here depends on the applied IFilter. XYplorer does not natively extract metadata.

Advanced Hex Search: Position

You can prefix the exact position of the string you want to match. Only files with those bytes at that position will be returned.

Notes

- The first position in a file is 0 (not 1). This is common usage.
- The position can be given in 0x hexadecimal format, or in decimal format.
- As special service an 8-digit number starting with 0 (zero) is interpreted as hex.
- The highest position you can state in hex format is 0xFFFFFFFF (4294967295).
- Positions stated in decimal format can be higher.
- Remember that position means exact position of match, not start of search.
- You don't have to set Type to "Binary"; when a position is specified then binary search is implied.
- This position-bound contents search is *MUCH* faster than the normal one where the whole file has to be scanned for matching bytes.

Examples

```
0: FF                Find all files starting with FF
0x0: FF              Find all files starting with FF
0x00000000: FF       Find all files starting with FF
00000000: FF         Find all files starting with FF
0xFFFFFFFF: 00       Find all files with 00 at pos 4294967295.
2: 69 00 73 00 6F 00 Find all files with this byte sequence at pos 2.
4294967296: AB CD EF Find all files with this byte sequence at pos 4294967296.
```

Advanced Hex Search: Wildcards

If you prefix a position to your hex data you now get support for Hex Wildcards. Simply type "??" in place of a hey value and this byte is ignored in the matching.

Notes

- You don't have to set Mode to "Wildcards" to enable Hex Wildcards.
- NOTE: Wildcards are NOT supported in non-position-bound searches. If no position is prefixed and the hex data contain wildcards anyway then internally the position is set to 0 (beginning of file), and the missing position prefix is auto-added to the contents field in the UI.

Examples

```
0: FF ?? FF
0x10: 00 ?? ?? 00
FF ?? FF           Position is assumed to be 0, see note above.
```

Note In either mode, the text entered cannot be be longer than 32,533 chars; of multi-line texts only the first line and a maximum of 1,024 characters is remembered between sessions.

Note The string search routine takes care that the Accessed date of the scanned files is reset to its original value when scanning is done.

IFilters

Content search supports IFilters. IFilters are Windows plugins used to extract pure text from complex file types like DOC, DOCX, ODT, PDF.

- IFilters are not used when "It's a hex string" is ticked.
- It depends on your system which IFilters are available. In the web you can find IFilters for almost anything.
- Where IFilters fail for whatever reason the logic falls back to binary search and checks the raw bytes for contents.

Tip: You can trigger Find Files by Ctrl+Enter in the contents field.

To apply this filter the corresponding checkbox below the Find Now button must be checked.

Tab "Dupes" (Duplicate File Finder)

Find duplicate files by name, date, size and/or content in any location. Spotting duplicates, especially those by content, can be useful for freeing up storage space.

XYplorer's Duplicate File Finder is implemented as filter "Dupes" in the Find Files tab. If the filter is active then only duplicate files are listed in the search results, i.e. files that have one or more duplicates in the searched location(s).

What counts as a "duplicate" can be defined in the Dupes tab: It's either match by Name, or Date (modified), or Size, or Content, or Image, or any combination of these.

- Name: The name of the file.
- Date: The modified date and time of the file.
- Size: The exact size of the file. Note that Size is logically implied when Content is checked.
- Content: The content of the file.
- Image: The visual content of an image file. Non-image files or folders will not be listed. For details see Image Hash below. See also Find Similar Images below for a way to quickly find images similar to a reference image.

For Name you can further define the scope of comparison:

- Same Extension: Dupes need a matching file title (base plus extension).
- Different Extension: Dupes need a matching base, and a different extension.

- Ignore Extension: Dupes need a matching base only (the extension is ignored). This will return groups of files with the same base regardless of their extensions.

The option Ignore numbers is mainly meant to ignore serial numbers when searching for Dupes by Name, so "Fred (1).txt" and "Fred (2).txt" will be considered dupes.
Actually it ignores all numbers in the name, though not the amount of numbers. So "1979.png" and "1980.png" will be considered dupes, but not "1979.png" and "19791.png". It also ignores strings that are auto-attached by Windows to the name of a file copied in the same place, e.g. "Copy of " or " - Copy", and it also ignores any accompanying serial numbers, e.g. "Copy (3) of " or " - Copy (3)".

For Content you can further define the method of comparison: The choices go from MD5 (fastest but least reliable) via SHA-1, SHA-256, SHA-512, to Byte-to-byte (slow but most reliable). Generally, SHA-1 is thought to be reliable enough for real world usage and hence it is also the factory default.

For Image you can further define the Tolerance of comparison. Values are from 0 to 16, factory default is 7. For details see Image Hash below.

Tick the Invert option to find all singular/unique files; quite useful when comparing two folders that should be in sync.

The Dupes filter is fully integrated with the XYplorer Find Files module and can be combined with all other search filters. Note that the duplicate check is applied *after* the other filters so if you know what you are looking for you can save a lot of time by pre-filtering the checked files by Name, Size, Date, Tags etc.

The results list of a dupes search features a special column "Dupes" where each group of dupes is referred to by a unique ID starting from number 1 for the first group detected. Sorting the list by this column will bring the duplicate files that belong together into adjacent order. A dupes search results list is always initialized to be sorted by the Dupes column. If sorted by Dupes, the list is internally set to grid style "Highlighted Groups", and the Dupes column is additionally highlighted with a color depending on the "highest" match criterion:

```
    If by Image:    white-on-light blue
ElseIf by Content:  white-on-green
ElseIf by Size:     white-on-brown
ElseIf by Date:     white-on-purple
ElseIf by Name:     white-on-dark blue
```

Right-clicking the Dupes column when the list is sorted by this column pops a special context menu with two commands for selecting items based on the dupe groups:

- Select the First Item in Each Group
- Select the Non-First Items in Each Group

Duplicate Folders: Normally the dupe finder will only find files, no folders. However, you can also find

duplicate folders by Name and Date: Simply tick Directory in the Find Files | Attributes tab, and the dupe finder will look for folders instead of files. This opens a fast way to compare directory trees, e.g. supposedly mirrored folder structures on two different drives.

Further Notes:

- The Dupes filter finds only files, no folders. Equally the Invert option will return only singular files.
- Of course, checking for dupes by content can take a lot of time. Still XYplorer is pretty fast here due to some smart logic. The bottleneck, however, is the hard drive read speed.
- Also search results caching is supported by dupes searches. But always remember that caches can get stale: It might be that files marked as dupes are not dupes anymore.

Image Hash

XYplorer can generate an "image hash" (aka "perceptual image hash" or "fingerprint") for images by which you can find duplicate images, sort images by their visual similarity, measure the degree of similarity between two images, and search for images that are similar to a given one. Unlike the common data hashes (MD5, SHA-256, ...) the image hash has an iconic relation to the visible image (the pixels), which means similar images have similar hashes.

When looking for duplicate images the setting of Tolerance comes into play. When Tolerance = 0 then all the pixels of the original full size images are compared and only perfect duplicates are listed. When Tolerance > 0 then the comparison is much more tolerant: it's scale-invariant, brightness-invariant, contrast-invariant, saturation-invariant, and hue-invariant. The higher the value the larger the tolerance for other small differences in image details. The factory default is set to 7 which has turned out by experiment to be a good value yielding a minimum of false positives (found too many) and true negatives (missed out too many). Note that perfection is not realistic in this business but near-perfection is a reasonable goal. Even that will vastly speed up finding duplicate images, which by eye alone is near impossible if the number of images is higher than a few dozen.

Additionally there is a Special Property Column called "Image Hash" that can be used to sort images by similarity. An interesting feature and hard to find elsewhere. In the column you can actually see the hash for each image.

Find Similar Images

Using Quick Search

You can use `prop:#image.hash` in a search term to find images that are visually similar to a reference image. The reference image can be passed as a path (will be converted to an image hash) or directly as an image hash (retrieved e.g. by `<prop #image.hash>` or from the Image Hash special property column). You can define what's similar by setting a tolerance value (0 to 99), prefixed to the path/hash separated with `~`. Here are some Quick Search examples where the tolerance is set to 13:

```
prop:#image.hash:13~518ce71800d7
```

```
prop:#image.hash:13~E:\Test\ImageHash\FindSimilar\Reference.jpg
prop:#image.hash:13~<curitem>
```

Skip the tolerance value to compare the image hashes as plain strings (only identical hashes are a match):

```
prop:#image.hash:518ce71800d7
prop:#image.hash:E:\Test\ImageHash\FindSimilar\Reference.jpg
prop:#image.hash:<curitem>
```

Set the tolerance to 0 for full image identity (internally done by comparing the MD5 hashes of all pixels in the full size image):

```
prop:#image.hash:0~<curitem>
```

Don't expect miracles. There is no AI involved, and the dHash algorithm has its limitations. You will get some false positives and negatives.

Using the Toolbar Button

A quick and easy way to find images similar to a reference image is to use the "Find Similar Images" toolbar button. Click it to find images similar to the image currently focused in the file list (it does not have to be selected). The right-click menu offers an easy way to adjust the required degree of similarity.

There is also an interesting command "Find Images Similar to Clipboard Image" which supports image files in the clipboard (items/objects as well as paths/text to items), and also plain images (bitmaps), which is pretty cool. For example, you can copy an image from a website to the clipboard and then search for files that look similar.

As you will quickly see, the button is just a shortcut to the above described `prop:#image.hash:[tolerance]~[item]`.

Note that it can make a difference which image you compare the others to. You will get the most matches if your image is somewhere in the middle of the similarity distribution. See "Family resemblance" (Wittgenstein) for more insight.

Tab "Excluded"

Here you can exclude specific files, folders, or drives from file search. Any subfolders of excluded folders are automatically also excluded. The matching of the excluded items is case-insensitive (A==a).

Add Current: Will add the current path, or the currently focused list item.

Browse...: Browse for a folder to exclude.

Manage...: Manage the excluded items using the List Management interface.

File: This checkbox shows and controls whether the selected item in the excluded items list is a folder (or drive) or a file, in other words: whether it defines an excluded location or an excluded item. If the checkbox is ticked the pattern is marked by the prefix "file:". Note that the "file:" patterns are matched with the item name only, not with the whole path, so they should not contain any other path components or backslashes.

Note It's possible to use wildcards, eg to exclude the Recycle Bins on all drives, add ?:\Recycled\ (XP) or ?:\$Recycle.Bin\ (Vista/Win7). Or adding *~* would exclude all folders containing the "~"-character anywhere in their path specification.

Note The matching is loose, e.g. file:Lucky will match all items with "Lucky" somewhere in the name.

Note When you start your search from an excluded folder, this folder is not excluded.

To apply this filter the corresponding checkbox below the Find Now button must be checked.

Search Results Context Menu

The context menu of items in Search Results and Branch View contains a couple of useful extra commands.

Go to Focused Item

Go to the focused item in its containing folder.

Keyboard Shortcut: Ctrl+Alt+Left

Go to Focused Item in New Tab

Go to the focused item in its containing folder in a newly opened tab.

Open Containing Folder in New Background Tab

Open the focused item's containing folder in a newly opened background tab.

Copy Containing Folder(s)

Copies the containing folders of all selected items to the clipboard.

Usage: Use Find Files to identify folders by their contents, and then copy those folders using Copy Containing Folder(s) for further processing.

Notes:

- The containing folders are the direct parent folders.
- If the containing folder is a drive it will NOT be copied.
- If more than one item is selected the list of containing folders is silently cleared from duplicate and folders contained in other folders in the list before it is copied to the clipboard. So it will not be unusual that less folders are copied than items are selected.

General File Find Tips

Tip You can abort a running file find process by pressing Esc (Escape Key).

Tip Press Ctrl+Alt+Left to jump to the currently focused file in the folder

where it was found. This trick works for search results only, and obviously it does only make sense when file find included subfolders.

4.24.8 Report Tab

Report tab

You can create two sorts of reports: Current Folder reports are on folders and have basically the format known from the classic DOS directory dump, Current List is on the stuff currently listed in the List being either the contents of the current folder or the results of a previous file find.

All reports can be sent to a popup text box (To Popup), the clipboard (To Clipboard), or to an ASCII text file (To File).

The interface for reports is found on the Report Tab on the Info Panel. It's subdivided into two tabs, Current Folder and Current List:

Current Folder

You have the choice here between quite different things:

Classic directory dump

Mimics the classic DOS way of listing directory contents. Additionally includes file version information for those files that actually have a version info.Example

Basic info to CSV

Exports Path/Name, Size (Raw), and Modified Date for each file item in CSV-format.

CSV: CSV stands for "Comma-Separated-Values" (the actually used list separator is taken from the current locale, and is often not comma but semi-colon), a very simple database format which can be read and written by any ASCII editor. CSV-files can be easily imported for example to Excel tables or Access tables or probably any table format existing. This feature makes XYplorer a superb tool for exhaustive file system documentation. Will save you not hours but days!

Extended info to CSV

Exports extended file information of whole directories (or even directory trees) in CSV-format. The exported info includes all data shown on the Properties and Version tab in the Info Panel.

Example

Metadata to CSV

Exports the complete set of metadata (also known as "Extended Properties", and identical to the fields used in File Info Tips) in CSV format. The first line has the field names (column headers). The first field "Path" has been added to the standard set.

This is the "total report". Everything that the system knows about a file is exported in a well-defined, easily processable, and widely portable file format. It's the documentation junkie's wildest dream come true.

Tree structure

Creates a textual representation of the deep structure of the current folder (or branch, if Include subfolders it checked), with the usual connecting lines.

Files and basic item data can optionally be included, see Configuration | Information | Report & Data | Info Panel / Report | Tree structure.

Example

Include subfolders: Report includes all subfolders (and their subfolders, etc.) of the current folder. When you do a structure report you would normally check this box.

Include Version: Additionally you can have version information included.

Tip You can abort a running report process by pressing Esc (Escape Key).

Current List

Reports the List as is. That is, the current column order, visibility, and item sort order is exactly reproduced in the report. This report is your choice when you want to document find results or make a detailed snapshot of the current state of your system directory.

Selected files only: Kind of self-explaining. However, when there are no selections then the report is on all listed items.

Output header format

Current date Adds the current date to the top of the report.

Path (Find settings) Adds the current path to the top the report.

In Find mode: adds the find parameters of the last File Find to the top the report (does only make sense, when you just made a File Find, of course).

Column headers Adds the column headers to the report.

Subitem separator

Tab separate items by Tabs

Other separate items by any chars of your choice

Pad blanks separate items by right-padding blanks: the current column widths are reproduced (as good as it gets) in the report (example).

Tip Make sure the columns are adjusted to the lengths of their contents to get a straight result with this option (to do this, either dbl-click the line number column header, or choose the Autosize Columns Now (Ctrl + Numpad Add) command from the column headers context menu).

4.25 Preview

Preview

Files can be previewed in Floating Preview, Preview Pane, or Preview Tab. While doing basically the same thing the three preview areas differ in size, position, display options, and displayed meta information, so depending on your task at hand you might prefer one over the other.

Only one of the three previews is used at each time, in this order of precedence:

Floating Preview > Preview Pane > Preview Tab

In words: Preview Tab is only used when neither Preview Pane nor Floating Preview are visible. Preview Pane is only used when Floating Preview is not visible.

Previewed Formats

Detailed information on the preview of different file categories (Text, Image, Document...) is bundled under Preview Tab.

Freezing the Preview

Normally the preview is automatically updated with each newly selected file. However, there is a way to keep the current preview: Click the toggle Freeze Info and Preview, found in the context menu of the Preview Pane, and in the context menu of the Status Bar and of the "Toggle Info Panel" Status Bar button. There is also a default keyboard shortcut for it: Ctrl+Alt+F11.

Closing the Preview

The preview is automatically closed when you deselect the previewed file. Alternatively tick Allow ESC to Close Preview, found in the context menu of the Preview Tab and then Preview Pane.

4.26 Floating Preview

Floating Preview

Note: All features and settings of the Floating Preview are shared by the Full Screen Preview. Below, "Floating Preview" stands for both.

The Floating Preview previews the current file in a non-modal preview window whose size and absolute screen position is stored between sessions. The window supports many media types (i.e. images, including Animated GIFs, and thumbnail previews for HTML, PDF, MPEG, DWG, etc., everything your system can generate thumbnails for), including the transparency background for transparent images. A right-click menu, and a set of keyboard shortcuts shown in that menu, lets you control various display options. Here is an overview over the related keyboard shortcuts and mouse tricks.

The Floating Preview optionally (see below) supports Mouse Down Blow Up, for images as well as for PDF files and other shell generated previews.

The Right-Click Menu of Image Previews

Here are some of the right-click menu options:

Next (PageDown) and Previous (PageUp): Browse previewed files. Note that changing the current file in the main form will update the preview, and vice versa: Wheeling through the preview (or key PageUp/PageDown) will change the current file in the main form browsing only previewable files.

You can as well use the arrow keys to navigate to the Next (Right, Down) or the Previous (Left, Up) image.

Fit All (A) shrinks larger originals to fit the width and height of the preview window. Resizing the window resizes the preview in real time. Toggles with Original Size.

Fit Width (W) shrinks larger originals to fit the width of the preview window. This serves to see the images that have a portrait orientation in more detail. Use Mouse Down Blow Up to view any cropped top and bottom parts. Fit Width is only supported in the Floating Preview, not in any other preview. Toggles with Fit All.

Fit Height (H) does for height what Fit Width does for width. Toggles with Fit All.

Zoom to Fill (I) fills all the available space with the preview. Left/right parts or top/bottom parts may be cropped. Using this option makes better use of the available space at the expense of the cropped areas. Note that the cropped preview is centered, i.e. the cropping is done equally at both of the cropped sides.

Original Size (O, 1, Numpad Divide) shows the image in original size (100%), cropped if there is not enough space. You can pan the preview by holding down the right mouse (or CTRL+left mouse) button and moving the mouse, or by the arrow keys (SHIFT + Up, Down, Left, Right; CTRL+SHIFT to pan up to the end). Toggles with Fit All.

Double Size (D, 2, Numpad Multiply) shows the image in original size (200%). See Original Size for panning. Toggles with Fit All.

Zoom In (Numpad Add, +, CTRL+Wheel Up) zooms into the image.

Zoom Out (Numpad Subtract, -, CTRL+Wheel Down) zooms out of the image.

- Zooming uses a logarithmic scale (based on the 4th root of 2, which means you need 4 steps to get to the half or double size), and it starts flexibly at the current zoom whatever it is. The possible range is from 0.5% to 6400% of the original size.
- To ensure that zooming passes the 100% point it initially snaps to a normalized scale that contains the 100% point, so the first zooming step might not be exactly by 4th root of 2.
- There is an even finer zooming by the 12th root of 2, which matches the chromatic scale of Western music and e.g. echoes the fret pattern on a guitar neck. Hold SHIFT to zoom by the 12th root of 2 (1.0594630943593).
- On zoom 400% and higher you will see the pixels. This is intended because "High Quality" resampling just looks bad with such high zoom factors.
- The zooming is auto-reset when the next image is previewed, unless Lock Zoom (see below) is enabled.

Toggle Zoom (G): Use this command to toggle between the last zoom level and the default size of the image (Fit All, Original Size, etc.). The last zoom level is remembered across openings of the preview window (but not across application sessions).

Lock Zoom (K): If ticked then the current zoom level is kept when switching images as long as the preview window is open.

Lock Zoom Position (X): If "Lock Zoom Position" AND "Lock Zoom" are both ticked then the current zoom level AND zoomed position are kept when switching images. This is very useful when comparing details of different versions of the same image. The zoom position is also kept when toggling between a zoomed view and the "Fit All" view by "Toggle Zoom" (G). So now you can zoom back and forth from/ to a particular detail of the full image.

Zoom by Wheel (Y): If ticked then Wheel zooms in and out, and Ctrl+Wheel browses through the files. If unticked then Ctrl+Wheel zooms in and out, and Wheel browses through the files.

Zoom to Fit (Z): Zooms smaller originals to fit the preview window.

Top-align if Vertically Cropped (V): Tick it to always top-align images that don't fit vertically.

Rotate Left (L) and Rotate Right (R): Rotate the preview by 90° steps.

- Of course, the original image itself is not modified.
- The rotate state is remembered for a preview until the file is unselected. You can even switch between Floating and Full Screen Preview while keeping the rotation as long as the file is not unselected.
- Press F5 (Refresh) to show the preview in non-rotated state, or in auto-rotated state if that's

enabled.

- Pixel-exact Mouse Down Blow Up works just fine in rotated previews.
- The image preview in the Info Panel, if active at the same time as the Floating Preview, is rotated as well. This has technical reasons but might as well be useful.
- Works for all previewed file types, even for RAW images, PSD, TGA, PDF, and video thumbs.

Flipped (F): If ticked then the preview is flipped (mirror image) horizontally (relative to the preview, also if rotated).

- All notes given for rotated images also apply to flipped images.
- For a flipped image Rotate Left and Right are interpreted as being applied to the flipped image, not to the original. This seems to better match the user expectation. Nevertheless, the status info displays the correct rotation degrees with reference to the original.

Open With... (Ctrl+Alt+Enter): Open XYplorer's Portable Openwith Menu.

Delete File (Del): Delete the previewed file right from the preview by pressing DEL or Shift+DEL.

Rename File (F2): Rename the previewed file right from the preview.

Full Screen (F12): Toggle Full Screen and normal mode while the preview is on.

Snap to Main Window (Ctrl+Shift+Alt+F12): Turn it on to keep the Floating Preview snapped to the main window's right side: When you move the main window the Floating Preview is dragged along with it.

Advanced Options... (Ctrl+Right-Click): Pop a menu with some advanced options.

Advanced Options Menu (Ctrl+Right-Click)

If you hold CTRL and right-click anywhere in the Floating Preview you get a popup menu with some advanced options:

Allow Custom Keyboard Shortcuts in Preview: Tick it and you can use all Custom Keyboard Shortcuts in the Floating Preview. This means you can e.g. label/tag/comment the previewed file, or call User-Defined Command scripts, or whatever, right in the Floating Preview.

Note: The Floating Preview's own keyboard shortcuts have precedence over the Custom Keyboard Shortcuts.

Cyclic Navigation: Tick it to navigate images around the edges: "Next" on the last image jumps to the first image, "Previous" on the first image jumps to the last image.

Navigate by Click (N): Tick it to enable navigation by left-clicking anywhere on the preview window or

the previewed image itself.

- Click on left half of the preview: Go back to the previous image.
- Click on right half of the preview: Go forward to the next image.

If Navigate by Click is off then middle-click goes to the next, and shift+middle-click goes to the previous image.

In the following section you can define the group of files in which you navigate when going Next/Previous:

Navigate by Extension: Next file has same extension as the current.

Navigate by Category: Next file is in same category as the current.

Navigate All Files: Next file is the next in the list.

Note that "Navigate by Category" navigates within Image files, Text files, Document files, etc (see Configuration | Previewed Formats). It's the factory default.

Mouse Down Blow Up (M): If ticked then the preview supports Mouse Down Blow Up (MDBU). Note that here MDBU always uses the whole screen (ignoring the setting of Configuration | Mouse Down Blow Up | General | Use whole screen). Here you can also control panning: If unticked you can pan a cropped image (partially shown image) using the Left Mouse Button. Note that you can always pan a cropped image (partially shown image) using the Right Mouse Button.

Mouse Down Blow Up: Shrink to Fit (Ctrl+M): If ticked the blow up is shrunk to fit the screen.

High Quality Image Resampling (Q) : Here you can control the quality of the image resampling.

White Border (Ctrl+W): Tick it to view your photos with a nice white border.

Cycle Background Color (B): Black, Dark Grey, Button Face, Light Grey, White.

Cycle Transparency Background (Ctrl+T): Cycles through the four options "Neutral", "Grid", "White", and "Black" for the background of the transparent parts of the previewed image (PNG, GIF, and other formats supporting transparency).

Show Status Bar (S): Show various image information at the bottom of the preview.

Tip: Right-click the Status Bar to pop a menu with some further options.

Show Photo Data (P): Show basic photo data in the status bar.

Show Tag Bar (T): Show the Tag Bar, an additional row in the Floating Preview Status Bar, showing Label, Tags, and Comment (if any) of the previewed file. Click (Left or Right) on any of the sections of the Tag Bar to change Label, Tags, or Comment of the previewed file right from the Floating Preview.

Show Script Button: Show a script button in the Tag Bar (which hence has to be visible as well to see the script button).

- Left-click the button to run the Floating Preview Script (if any).
- Right-click the button to create/edit the Floating Preview Script.
- You can as well run the script by key F8 (only from Floating Preview). Note that the script is triggered on KeyUp, not KeyDown. This avoids potential trouble when the scripts itself receives the KeyUp event.
- The Floating Preview Script is remembered between sessions.
- Tip: F1 in the Edit Script dialog will show the Scripting Commands Reference.

Run Script (F8): Run the Floating Preview Script.

Show Histogram (Ctrl+H): Toggles a (color or luminance) histogram of the current preview in the lower right corner of the preview window.

- The histogram is always created on the currently visible preview pixels. So if you just view a part of the picture you will get a histogram just of that part.
- Luminance Histogram: The median is marked by coloring. Separates the dark half of the pixels from the bright half. The mean is marked by coloring. The Histogram bar at the mean is colored darker.
- Color Histogram: Shows the three color channels (RGB) individually. It has a slight brightness gradient from left to right.

Color Histogram (Ctrl+R): Toggles between Color Histogram and Luminance Histogram.

Invert (Ctrl+I): Inverts the preview of the image (like a negative). Of course, the original image is not touched.

Grayscale (Ctrl+G): Converts the preview of the image to grayscale. Of course, the original image is not touched.

Copy Original (C): Copy the original image (bitmap) to the clipboard.

Copy Preview (Ctrl+C): Copy the preview image (bitmap) to the clipboard.

The Right-Click Menu of the Status Bar

The status bar has its own context menu with a selection of the above commands. There is also one toggle command found only here:

Auto-Scroll to End (Ctrl+E): Enable it to automatically position the viewer at the end of a previewed text file. This can be useful e.g. when viewing log files.

Other Remarks

Note that the Quick File View is auto-opened on a file that's not supported by the Floating Preview.

There is a menu command (menu File | Floating Preview, Default Shortcut: F11) and a toolbar button to toggle the Floating Preview.

Zip Support: When you call the Floating Preview on a Zip file, the contents of the Zip are shown in a list. See details here:

4.27 Preview Pane

Preview Pane

The Preview Pane can be toggled by menu Window | Show Preview Pane (Ctrl+F11). It is designed to be a right-bound subsection of the list pane(s). Its width is adjustable by dragging a vertical splitter. Its height is implied with the pane(s) height.

There is also a Toolbar Button to toggle the Preview Pane.

Functionally the Preview Pane is quite similar to the preview pane in Windows Explorer, with some differences and some enhancements:

- There is a mouse over tooltip showing original and preview size.
- There is a context menu with a couple of options.
- Previews are horizontally and vertically centered in the pane. There is a 16 pixels padding on all sides. No zoom or so, you always see the whole image optimally scaled to the available space.
- Minimal pane width is 48 pixels, minimal preview width is 16 pixels.
- Mouse Down Blow Up is supported.
- Optional Transparency Background.
- Animated GIFs are previewed.
- Icons are previewed.
- You can right-click on the empty parts of the pane to open a small menu that allows you to close the pane.

Small Audio Preview

If you select an audio-only file nothing will be shown in the Preview Pane itself but you get a little status feedback, the "Small Audio Preview": Once the audio preview starts a basic progress bar is shown in the main Status Bar giving you an idea of the current position within the piece.

The progress bar is interactive:

- You can left-click on it to change the position of the playback.
- Right-click the progress bar to toggle Pause/Play.
- There is also a tiny button to toggle Pause/Play.

Various key combinations are supported while the audio preview is active.

Stop, Pause, Play:

- Space: Toggle Play/Pause
- Shift+Space: Stop (Pause and Rewind)
- ESC: End Preview

Winding backward and forward:

- Left: 5 sec backward
- Right: 5 sec forward
- Shift+Left: 30 sec backward
- Shift+Right: 30 sec forward
- Ctrl+Left: 1 sec backward
- Ctrl+Right: 1 sec forward

4.28 Search Templates

Search Templates

The command to open the Search Templates dialog is found in menu Edit. It is also found in the context menu of the Find Files button on Info Panel | Find Files.

You can save your current Find Files settings to template files, which are stored as simple INI files in a subdirectory of the application data path called "FindTemplates". For the template name you can use all characters including those that are invalid for filenames (<,>,?,",: etc.).

A template contains all settings found on the Find Files tab, with the exception of the Name and Location dropdowns, where only the current top entry is remembered.

Load Template

Load the selected template and close the Search Templates dialog. Alternatively double-click the template name in the list. There are some options:

Load saved results: Here you have a choice whether to load any cached results (see Caching below) or not. The enabled state of the checkbox tells you whether cached results are existing in the template or not. Any implied states of the other options are auto-handled for you.

Run search at once: If checked the search will start immediately after you clicked Load Template.

Load search location: When loading a (previously stored) template you have the option whether to use the current location or the location stored in the template for the search.

Load Excluded Folders: Check to overwrite the current excluded folders listing by the one stored in the template. The default is unchecked because it is assumed that the excluded folders are usually a global affair.

Save to Template...

Save current Find Files settings to template file. If you enter an already existing template name this template will be overwritten/updated.

Save search results: If checked then search results are stored with the template and can be reloaded at a later point. Of course, the option is only enabled when search results are present in the file list.

Note: You cannot store any Quick Searches, but only searches triggered through the Find Files tab.

Note: You cannot cache the results of any visually filtered searches anymore. Reason: The filter is not stored along with the template so when loading the Search Templates you have no way to find out by what the data had been filtered.

Caching

Beginning with version 8.40 Search Templates support caching. i.e. you can save search results in a template file (in <xydata>\FindTemplates), and load them back into the list at any point in future via the Search Templates dialog.

Writing the cache: You can control whether search results are stored on a per-template basis using

the new option "Save search results" below the "Save to Template..." button.

Reading the cache: Any previously cached results are read when you check the Load saved results option. If you do, the other three options are implied as well. Reason: Otherwise the displayed results would be out of sync with the Find Files panel.

4.29 Branch View

Branch View

Branch View (BV), also known as "Flat View", is a mode of the file list that lets you view the contents of all subfolders of a folder in one list.

To turn on Branch View for the current tab, use the toggle "Branch View" in menu View | Views (also available in the file list's white space context menu), or the "Branch View" toolbar button.

Implementation

Branch View is implemented in a flat, human-readable, and editor-editable way. It is triggered by a switch at the end of the location spec, e.g.:

```
E:\XYplorer\appdata\? /flat
```

The question mark is a part of it, no wonder since Branch View is technically nothing but a recursive search (i.e. including subfolders), and the normal Quick Search syntax applies, for example:

`? /f /flat` = Show only files in Branch View.

`?a*;b* /flat` = Show only items beginning with a* OR b* in Branch View.

This flat syntax enables you to trigger Branch View from the Address Bar, Catalog, Favorites, User-Defined Commands, Scripting, etc. Note that also Boolean Logic and Regular Expressions are supported.

Types of Branch View

There are four types of Branch View:

1. `/flat` = Show files and folders (= Mixed Branch View)
2. `/flatfiles` = Show files only
3. `/flatdirs` = Show folders only
4. `/flatnoempty` = Show files and non-empty folders only

By factory default the first one is used when Branch View is toggled via menu or toolbar. You can change this default in Configuration | Find and Filter | Find Files & Branch View | Branch View | Default branch view type.

Configuration

See Branch View Configuration. Note that the toolbar button has a context menu featuring some configuration options.

Filtered Branch

By ticking Let folders pass all filters you get the following behavior: On a mixed Branch View (i.e. /flat = showing files and folders), any Visual Filters or Quick Search patterns only apply to files, not to folders.

So the folders always stay visible while their contents are filtered.
This feature, combined with level-indent and tree-like sort order (see here below), enables you to view a Filtered Branch in the file list, for example a branch showing only images or only EXE files. A real time-saver when perusing the file system under certain aspects.

Tree-Like Sort Order

Sorting the list (in Branch View and also in general Search Results) by Path will create a perfect tree-like sort order, that is all folders sorted like in the tree and in each folder you get the files listed ahead of any subfolders. Sorting by Path again will reverse this order.

Note: To make this happen sort settings must be so that folders are NOT sorted apart, so either you turn off "Configuration | Sort and Rename | Sort folders apart" (which is unlikely) or you turn on "Configuration | Sort and Rename | Mixed sort on path columns" (which you will likely do).

Multi Branch View (MBV)

You can select multiple folders in the list view and then branch view them together in one list. If Configuration | Find Files & Branch View | Branch View | Multi branch view lists top folders is enabled then the containing/top level folders in a Multi Branch View are shown as well, not just the items contained in those folders.

To trigger an MBV from the toolbar or by keyboard, the list must have the focus and more than one folder (or LNK to a folder) must be selected. A MBV can also be triggered via the Address Bar, Catalog, Favorites, etc. using the following multi location syntax (paths separated by ;):

```
E:\Test\A\;E:\Test\B\? /flat
```

This syntax also allows MBV for folders in completely different locations, for example:

```
D:\Work\;E:\Test\B\;\\Server\Docs\? /flat
```

Tip 1: You can even trigger a Multi Branch View *from* a Branch View when two or more folders are selected in the focused list. The "Branch View" toggle silently turns into a "Branch View" *command* under these conditions.

Tip 2: You can also select files along with the folders you want to dive into and they will just be added to the Branch View.

Tip 3: Toggling back to normal view from a Multi Branch View will restore the multi-folder selection, the scroll position, and the focus position.

Note: Unless the tab captions are set to "Full path" in Configuration | Tabs, the tab headers show the suffix /M to make clear that the contents are from multiple locations.

General Properties of Branch View

- Like Search, Branch View is stored tab-wise and remembered between sessions.
- Unlike Search, Branch View is always done in the current tab (even if it is locked).
- A tab on Branch View is marked by a special icon.
- When toggling Branch View, the sort order (unless you enabled Default to tree-like sort order) and view mode remains the same. Other list settings (back color, column order, column visibility, column widths) are taken from the Find mode (aka Search Results).

- Refresh List will refresh Branch View, it will not re-browse the folder in normal view.
- Branch View never follows folder links or junctions.
- You can drag-n-drop items from one folder into another within Branch View.
- You can drag and drop items directly to the subfolders listed in the Path column.
- When you use Edit | New | New Folder (Ctrl+N) or Edit | New | New Text File (Ctrl+Shift+N) in a Branch View, the currently focused item defines the depth level on which the new item is created. The same is true for Edit | New Items.
- When you use Edit | Paste (Ctrl+V) and a folder is focused, the items are pasted into the focused folder.
- When you use Edit | Paste (Ctrl+V) and a file is focused, the items are pasted into the parent folder as siblings of that file.
- Branch View can be combined with Visual Filters.
- Branch View can be combined with Quick Searches (including Boolean Logic and Regular Expressions).
- Specific Branch View patterns can be stored in and triggered from Address Bar, Favorites, Catalog, User-Defined Commands, and Scripts.
- Press ESC to interrupt a running Branch View.

4.30 Dark Mode

Dark Mode

Dark Mode concerns the look of the application. Basically white and black are inverted, so you will see most texts white on black. This has several advantages:

- It looks nice.
- It's relaxing to your eyes.
- It saves energy, money, and the climate.

How to turn it on

You can toggle it via menu View | Views | Dark Mode.

Additionally there is a toolbar button "Dark Mode".

Properties

It's a portable Dark Mode. It's independent of the host system display settings.

It's a smart Dark Mode. You don't have to configure anything (and you can't). Your current color configuration is automatically converted into its dark sister. She's much darker and a bit desaturated. Brightness of foreground (text) and background is mostly inverted.

It's an instant Dark Mode. It can be toggled on-the-fly. No restart necessary.

Configuration

In Configuration | Colors and Styles | Highlights & Dark Mode | Dark mode you can control two parameters, the darkness and the text contrast.

Darkness: There are 51 levels of darkness, 0 - 50. The factory default is 25.

Text Contrast: There are 31 levels of contrast, 0 - 30. The factory default is 15.

Color Tint: There are 360 different color hues, 1 - 360, and the setting 0 for neutral. The factory default is 0.

Additional Remarks

Due to certain Windows shortcomings or lack of documentation the functionality of checkboxes and option buttons is slightly diminished in dark mode:

- You cannot click the caption to alter the state of the checkbox/option but must click the checkbox/

option itself.

- No accelerators.

Some elements are darkened only in Windows 10 or higher:

- Title Bar
- Buttons
- Collapsed dropdown lists
- Scrollbars

Some elements are darkened only in Windows 10 or higher AND if Windows is in Dark Mode:

- Popup Menus

Some elements could not be darkened:

- Main Menu Bar

In Dark Mode the setting of "Configuration | Colors and Styles | Highlights & Dark Mode | Selections" is ignored. Internally it's set to "XYplorer Style" as Windows Themes have to overwritten Dark Mode as far as possible.

Within the Configuration dialog any configurable colors, e.g. Color Filters, are not displayed darkened in the way they will be when actually applied to the main interface. Otherwise it would be confusing while adjusting the colors. So, what-you-see-is-what-you-set, but not what-you-see-is-what-you-get.

4.31 Ghost Filter

Ghost Filter

The Ghost Filter lets you globally hide particular files and folders from Tree and List (all tabs, both panes).

The items to be hidden are identified by wildcard patterns matching their name. To enter the desired patterns click View | Show Items | Edit Ghost Filter.... You can enter any number of patterns, one per line. Examples:

```
.*                              Hide all items beginning with a dot.
desktop.ini                     Hide all items called "desktop.ini".
```

You toggle the Ghost Filter either by clicking View | Show Items | Ghost Filter or by clicking the toolbar button "Ghost Filter" (it looks like a ghost).

Hiding will equally affect files and folders (unless you employ scope prefixes, see below).

Hiding will happen in Tree and List.

Hidden items will not appear in search results (unless Find Hidden is ticked). But hidden folders are searched for any non-hidden items they might contain.

If the current list is actually hiding items by the current Ghost Filter, a little ghost icon is shown in the first section of the status bar.

The number of "ghosted" items in the List is counted and shown in the status bar tooltip (left-most section).

The Mini Tree is not affected by this, just the Maxi Tree.

Named drives are supported if Configuration | Controls & More | Miscellaneous | Support volume labels in paths is ticked. For example, this Ghost Filter pattern will hide folder T:\DVD and all of its contents if named drive "Terra:\" maps to "T:\":

```
d: "Terra:\DVD"
```

Patterns

Patterns are not case-sensitive.

Name Match: Patterns are matched with the filename only (no path), unless they contain backslashes.

Path Match: If a pattern contains backslashes it is compared with the full path of the item:

```
"E:\Test\hidden\readme.txt"         Match just this item.
```

`E:\*.txt`	Match all TXT files on drive E:.
`*\junk\*`	Match all items under a folder "junk" (not necessarily the immediate parent!)

Scope Prefixes: You can define whether a pattern affects files (f:), directories (d:), or both (no scope prefix). Note that there must be at least one space after the ":".

`.*`	Affects files and directories.
`f: .*`	Affects files only.
`d: .*`	Affects directories only.
`f: "readme.txt"`	Affects files called readme.txt.

Auto-wildcards: Patterns without wildcards are auto-surrounded by wildcards, unless they are double-quoted (just like Visual Filters).

Exact Match: Double-quote patterns to force an exact match:

`readme.txt`	Matches "readme.txt" and "alsoreadme.txt".
`"readme.txt"`	Matches only "readme.txt".

No-Extension: Patterns ending with a dot match files that have no extension. Folders never match such a pattern.

`*.`	Match any files without extension.
`a*.`	Match all files beginning with "a" and without extension.

Warning

The Ghost Filter should be used with care: It might trick you into believing that files went AWOL!

4.32 Paper Folders

Index

Paper Folders
Permanent Custom Sort Order
Drop Stacks

Paper Folders

Paper folders are simple text files containing the paths to items (files and folders), one per line. XYplorer uses these text files as virtual folders.

The files are formatted as UTF-16LE and located by default under the application data folder in the subfolder "Paper" (<xydata>\Paper, or simply <xypaper>).

Format of a Paper Folder

A Paper Folder source file might look like this. It can contain full paths to files and folders, network or local, for example:

```
C:\Users\Donald\Pictures\Tower-The-Of-Babel-Brueghel-Elder.jpg
C:\Users\Donald\Desktop\
E:\Test\Charsets 日本人\北京体育大学
\\VEGA\Test\Admin.ini
```

You can also specify relative paths and they will be resolved relative to the path of that source file. This allows you to move or copy paper folders along with their contents. A major gain in portability! For example:

```
Alvaro Dominguez.jpg
Köln.jpg
NewYorkCity.jpg
Autumn de Wilde\A Very She and Him.jpg
Autumn de Wilde\Arrow (15).jpg
Autumn de Wilde\Chelsea Hotel.jpg
```

Referencing a Paper Folder

The syntax is simple, just prefix "paper:" to the base name of the text file:

```
Location:   paper:foo
Refers to:  <xydata>\Paper\foo.txt
```

Consequently the characters used for a Paper Folder name must be valid in a filename, and they are not case-sensitive.

Integration

The objective here was maximum integration of Paper Folders. Ideally they should work just like normal folders, and this has been largely achieved:

- History, Recent Locations, Hotlist, Favorites, Tabs, Catalog are supported.
- Visual Filters supported. Examples:
  `paper:Test|*.txt`
  `paper:Test|ageM: w //modified this week`
- However, Global Visual Filters and Ghost Filters are not applied to Paper Folders! After all Paper Folders are hand-picked collections of items that should really be spared from those global filters working in the background. If you wouldn't want those items in the Paper Folder you wouldn't have put them there in the first place, right?
- Live Filters supported.
- Quick Search supported. Examples:
  `paper:Test?b*`
  `paper:Test?tags:done`
  `paper:Test?1 w /fld=ageM //modified last week`
- Also Multi Location Quick Searches are supported. Examples:
  `paper:Test1;paper:Test2?b* /n`
  `E:\Test;paper:Test?b* /n`
- Branch View supported:
  `paper:Test? /flat`
- Find Files supported (Paper Folder as searched location).
- Breadcrumb Bar supported: Click "paper:" or the triangle right of it to pop a list of all stored Paper Folders.
- Custom File Icons supported. Example:
  `paper:Test>diamond.ico`
- Folder View Settings supported.
- The right-click menu of items in a Paper Folder offers to go to the original location (just like with search results). The menu also offers a command to remove items from the Paper Folder (they will not be deleted from the file system).
- Rename and Delete operations can performed from within Paper Folders. They affect the real files!
- All possible Copy/Move operations to and from Paper Folders are supported, including Paste, Copy To, Drag-and-Drop, etc.
  Note: Operations *to* Paper Folders just add pointers to the Paper Folder files: they never involve real file operations.
  The mouse pointer is a white-on-black "plus" icon.

- Operations *from* Paper Folders operate on the real locations of the files and affect the files as if they were performed in the real locations!
  The mouse pointer is the normal plus (copy) or arrow (move).
- You can also drag items from one Paper Folder to another.
- Go to Last Target works after adding items to a Paper Folder.
- Thumbnails can be cached for Paper Folders.
- You can tag items in Paper Folders.
- Custom columns work in Paper Folders.
- All reporting functions support Paper Folders.

How to Create a Paper Folder

See the functions described here.

There is also an implicit way: When you go to a Paper Folder that does not exist you are prompted whether you want to create it on the fly. That way you can create new Paper Folders right from the Address Bar very easily.

Remarks on Usage

- When you copy / move (this makes no difference here!) items to a Paper Folder they will be appended to the bottom of the list.
- You can drag-and-drop items to Paper Folders in the list, across Panes, onto foreground and background tabs, onto the Toolbar (droppable buttons), and onto the Catalog.
- Go Up from a Paper Folder will take you to the path of the Paper Folder source file.

What's the Use of Paper Folders

- You can use them as temporary or permanent file collections for viewing or further processing. Contrary to normal folders the files in a Paper Folder can be located in totally different real locations, even all over the network. And they take almost no additional space since internally it's just a list of pointers.
- You can use them to store search results. Going back to the Paper Folder will often be much faster and easier than repeating the search.
- You can use them as light table to manually arrange photos in a custom sort order that is permanent.
- In teams you can easily share documents via a commonly accessible Paper Folder without copying a single byte.
- You can use them as simple reports since the Paper Folder files themselves are simple text files that can be viewed in any editor.
- You can design your virtual folders in a text editor.
- You can share them, and carry them along on a stick.

Paper Folders Toolbar Button

There is a Toolbar button "Paper Folders" with a right-click menu featuring the following configuration options:

- Allow Zombies: If enabled, non-existent or currently unavailable items are also listed. And a number of special items are tolerated in addition to regular files and folders: Paper folders, virtual folders, servers, drives, special folders, even scripts and URLs:
  `paper:DropStack`
  `vi:<clp>`
  `echo "hi!";`
  `https://www.xyplorer.com/`
  `https://www.xyplorer.com/images/whatsnew/StatusTooltip_t.png`
  `Documents`
  `%rapidaccess%`
  `%user%`
  `E:\`
  `\\VEGA\`
- Always Show Path Column: When going back to a normal folder the Path column is hidden.
- On Delete Remove Items from Paper Folder: Saves you from accidentally deleting items in a Paper Folder. Factory default is ON. Note that "Delete" from the Shell Context menu will still delete the actual file!
- Explicit Save Only: Tick it to suppress auto-save of the current Paper Folder on tab switch and on app exit.
  If ticked you can still explicitly save a Paper Folder using menu "View | Paper Folders | Save" or the Save command in the Toolbar button's dropdown menu.
  Note that even with this setting enabled it's not possible to create a new Paper Folder without auto-creating a Paper Folder source file on disk. It can be empty, but it has to be there.
- Show Information Bar in the List: Toggles the Paper Folder Information Bar shown at the top of the file list. The bar has a right-click menu.

Final Remarks

- You can have as many Paper Folders as you like.
- Paper Folders can also be located in subfolders of <xydata>\Paper:
  `Location: paper:Sub\foo`
  `Refers to: <xydata>\Paper\Sub\foo.txt`
- Paper Folders can even be located in any location if you pass the full path. (Note that the TXT extension is obligatory.)
  `Location: paper:E:\Test\somewhere\foo.txt`
  `Refers to: E:\Test\somewhere\foo.txt`
  So you can also do this when the current file is a plain text file containing a list of items:
  `paper:<curitem>`
  BUT NOTE: Like all Paper Folder files this file will be overwritten whenever the Paper Folder is saved to disk (and often it is saved automatically, for example when you add something to it, or when you leave it, or when you save settings on exit). So: Do not open files as Paper Folders that you don't

want to be overwritten! There is a safety belt built-in to avoid undesired overwriting: If a source file doesn't look like a valid paper folder file you are prompted before it is overwritten.

- Paper Folders have a static nature. They are not sensitive to changes in the file system.
  Exception: File operations performed on items in a currently opened Paper Folder will be reflected by the folder as if it would be a normal folder.
- Paper Folders are self-cleaning: Any stale items in a Paper Folder are automatically removed from it when the folder is shown the next time.
  Exception: Allow Zombies (see above) is enabled.

Permanent Custom Sort Order

Paper Folders can be used to achieve a Permanent Custom Sort Order. *(Note: From v16.60 onwards Permanent Custom Sort Order is also supported in normal folder tabs.)*

- Open a Paper Folder and manually sort it (Tools | Customize List | Manual Sorting).
- Save the folder (View | Paper Folders | Save).
- Use the Index column to invoke the custom sort order.

Drop Stacks

[This feature is only available if the Scripting feature is enabled.]

Combine Paper Folders with Droppable User Buttons and you have Drop Stacks in the Toolbar.

- Create a new Paper Folder, just for fun call it "PhotoSelection".
- Add a new User Button to the Toolbar, and set its Name field to "paper:PhotoSelection" (w/o quotes). Leave all other fields empty (or set the Icon if you feel like it).
- Done! Now you can drag-drop items to that button and they will be added to "PhotoSelection". At any time click the button to go to "PhotoSelection".

Note that you can have as many such Drop Stacks as you want (so long as it is less than 65, since 64 is the maximum number of user buttons).

4.33 Virtual Folders

Index

Syntax
Switches
Properties
Integration
Using Variables in Virtual Folders
Virtual Folders as Filters
Listing URLs in Virtual Folders

Virtual Folders

Virtual Folders are folders where you define the content directly in the folder name. Think of Virtual Folders as Paper Folders without the paper. You can directly "create" such a folder by typing it into the Address Bar. Nothing is created on disk, it's all in the name.

Syntax

The path of a Virtual Folder consists of the prefix "vi:" followed by the folder definition. The folder definition controls which files are listed in the Virtual Folder.

Here are some examples. You can open these paths directly through the Address Bar:

```
vi:C:\Program Files\XYplorer\XYplorer.exe
vi:C:\Windows\explorer.exe
```

Multiple Terms: You can specify more than one term, separated by CRLF (which means you specify one per line) or by | (pipe). The following examples are functionally identical:

```
vi:<xypath>\Startup.ini|<xypath>\XYplorer.exe
```

```
vi:<xypath>\Startup.ini
<xypath>\XYplorer.exe
```

```
vi:
<xypath>\Startup.ini
<xypath>\XYplorer.exe
```

Variables: As already illustrated in the above examples, XYplorer native variables are resolved, all of them. Also all environment variables. For example:

```
vi:<xy>
vi:%temp%
```

This gives you interesting options, see below Using Variables in Virtual Folders.

Folder Contents: You can as well list the contents of real folders. Simply add a trailing * (asterisk) to the folder path (the final backslash in the path is optional). To limit the listing to just files or just folders append " /f" or " /d":

```
vi:C:\*                          //list all items in C:\
vi:<xypath>*                     //list all items in XYplorer's app path
vi:%temp%*                       //list all items in the temp folder
vi:%temp%* /d                    //list all directories in the temp folder
vi:%temp%* /f                    //list all files in the temp folder
vi:C:\* /d                       //list all directories in C:\
vi:C:* /f|D:* /f|E:* /f          //list all files in C:\ and D:\ and E:\
vi:\\VEGA\shared*|T:\shared*     //list all items in those two locations
```

Remarks:

- There is no recursion, just first level.
- This gives you fast treeless browsing of any location, and it gives you an easy and fast way to list the joined contents of several locations.
- For example, pointing a locked tab to such a VFO gives you quick and indestructive access to files you often need to work with.

Comments: You can append a final comment to your Virtual Folder, separated by " //":

```
vi:%tmp%* /f          //list all files in TEMP
vi:C:*|D:*|E:*        //list all items in C:\ and D:\ and E:\
```

Caption: You can use the comment to specify a caption for the Virtual Folder. It will be displayed in Title Bar, Info Bar, Status Bar, Tab Headers, Catalog and other places. The first piece of text in quotation marks in the comment is used as the caption:

```
vi:%tmp%* /f          //list all files in TEMP ("Tmp Files")
vi:C:*|D:*|E:*        //"C, D, E" (list all items in C:\ and D:\ and E:\)
```

Switches

Switches are letters surrounded by | and can be combined in any order.

Dupe Removal: In Virtual Folders duplicate items in one list are possible. You can have the list deduped by passing switch "d" surrounded by | like this:.

```
vi:|d|<get list_recentlyopenedfiles>|<xy>
```

Existence Check: You can enforce a check for existence on all listed items. This is useful for commands the return potentially stale data, for example:

```
vi:<get list_recentlyopenedfiles>
```

To enforce a check for existence prefix the new switch "e" surrounded by | like this:

```
vi:|e|<get list_recentlyopenedfiles>
```

Note that URLs and scripts will always pass the test.

Properties

- While the folder is virtual, its contents are not. They are real. To prevent a disaster, the deletion of items from Virtual Folders is not permitted.
- You can always drag and copy from a Virtual Folder.
- You paste and drag *into* a Virtual Folder only if the Virtual Folder has one common path (if all its items are in the same location).
- Note that you are dragging real files, not virtual files. Only the folder, the current collection in one place, is virtual.
- Listing Virtual Folders always leaves the folder tree alone, so it is very quick and does not change the structure or position of the folder tree.
- In Virtual Folders zombies (files that don't exist) are always welcome.
- You can even list URLs. See below Listing URLs in Virtual Folders.
- A good place to keep Virtual Folders handy is in the Catalog, in Locked Tabs, or in the Hamburger.
- Within a Virtual Folder just press F5 to refresh.
- A Visual Filter remains intact when you refresh a Virtual Folder tab (F5).
- The Turkish Rose colored Information Bar in the Virtual Folder file list shows the real path from which the currently listed items come (or the common branch when they come from several places).
- They optionally (*Configuration* | *General* | *Refresh, Icons, History* | *Auto-Refresh* | *Include virtual folders*) support Auto-Refresh, *if* the common path or branch of the contained items supports it. That path is displayed in the Information Bar; if no path is displayed then you get no Auto-Refresh.

Integration

The objective was maximum integration of Virtual Folders. Ideally they should work just like normal folders, and this has been largely achieved:

- History, Recent Locations, Hotlist, Favorite Folders, Tabs, and Catalog are supported.
- Visual Filters are supported.
- However, Global Visual Filters and Ghost Filters are not applied to Virtual Folders! After all Virtual Folders are hand-picked collections of items that should really be spared from those global filters

working in the background. If you wouldn't want those items in the Virtual Folder you wouldn't have put them there in the first place, right?

- Live Filters are supported.
- Quick Search is supported.
- Branch View is supported
- Find Files are supported (Virtual Folder as searched location).
- Breadcrumb Bar is supported.
- 6 Key Navigation in the Address Bar is supported
- Hover Box is supported (on Virtual Folder Tab Icons, also on background tabs and in the background pane). Handy for a quick look at the clipboard:
  `vi:<clp>`
- Custom File Icons are supported. Examples for CFI definitions:
  `vi:*pick*>cup_coffee.ico`
  `vi:*>omelet.ico`
- Folder View Settings are supported.
- Rename operations can performed from within Virtual Folders. Note that they affect the real files!
- Operations *from* Virtual Folders operate on the real locations of the files and affect the files as if they were performed in the real locations!
- Thumbnails can be cached for Virtual Folders.
- You can tag items in Virtual Folders.
- Custom columns work in Virtual Folders.
- All reporting functions support Virtual Folders, apart from "Tree Structure".
- You can convert a Virtual Folder to a Paper Folder using View | Paper Folders | Toggle Paper Folder.

Using Variables in Virtual Folders

The <get ...> variable is an interesting candidate for Virtual Folders:

```
vi:<get list_recentlocations>
vi:<get list_recentlyopenedfiles>
```

You can as well show the current clipboard clipboard contents, whether they are file items (objects), or just their textual representation (paths). Lets you list your clipboard as if it was a folder.

```
vi:<clp>
```

To pick all JPG, JPEG, and PNG files in the current folder:

```
vi:<get pick *.jpg;*.jpeg;*.png>
```

Generic file types are supported in <get pick ...>, e.g. {:Image}. This will list all image files present in the

current folder:

```
vi:<get pick {:Image}>
```

The following provides you with a snapshot of the currently selected items. Works great as a locked tab. You switch to it and see just the selections from the previous tab. Cool! This tab can also offer a different view on those selected files, eg Thumbnails or Tiles. And yes, the thumbnails cache also works for Virtual Folders!

```
vi:<get selecteditemspathnames>
```

You can also read contents of a file using the <file ...> variable:

```
vi:<file C:\Users\Donald\Desktop\Desk\files.txt>
```

Quite similar to a Paper Folder then, but read-only. That source file is never changed by the Virtual Folder.

Note that <get pick> even supports Paper Folders. Assuming you have a Paper Folder called "Pictures", you can do this:

```
vi:<get pick 8 paper:Pictures>
```

More useful examples with <get pick ...> (all pick from the current folder), including how to pick files by age:

```
vi:<get pick>               //pick all files
vi:<get pick 8>             //pick any 8 files
vi:<get pick 8.m>           //pick the latest 8 files
vi:<get pick *.txt>         //pick all TXT files
vi:<get pick 7/*.txt>       //pick any 7 TXT files
vi:<get pick 7.m/*.txt>     //pick the latest 7 TXT files
vi:<get pick {:Image}>      //pick all image files
vi:<get pick 3/{:Image}>    //pick any 3 image files
vi:<get pick 3.m/{:Image}>  //pick the latest 3 image files

vi:<get pick 3.m>                //pick the latest 3 files from the current folder
vi:<get pick 3.m %SystemRoot%>   //the latest 3 files (by modified date) from C:\Windows
```

You can reverse the pick order by adding "r" (reverse) to the sort field:

```
vi:<get pick 12.mr>          //pick the oldest (by modified) 12 files from here
```

Insider tip: You can use <pick [...]> as a short form for <get pick [...]>:

```
vi:<pick 8>                  //pick any 8 files from here
```

If you are combining multiple picks, you should separate them with <crlf> or | (pipe):

```
vi:<pick 8.m "Documents\Songs"><crlf><pick 8.m "Documents\Paintings">  //"8 latest songs and
paintings"
vi:<pick 8.m "Documents\Songs">|<pick 8.m "Documents\Paintings">  //"8 latest songs and
```

```
paintings"
```

Virtual Folders as Filters

If you set a tab to a virtual folder like this:

```
vi:|t|<pick 8.m>  //"Last 8"
```

... and then use "Set Home" and "Lock Home Zone" you can inject new locations into the tab simply by selecting folders in the tree. So you can have a tab that only shows you the last (newest) 8 files of any location you click on. Pretty cool.

By the way, the `|t|` switch in the definition above means: Show the virtual folder path in the tree if possible (by default virtual folders are not represented in the tree). Note that this only works if the items listed in the virtual folder have a common path. If they don't, or if there are no items at all, nothing will be selected in the tree.

Listing URLs in Virtual Folders

Just pack your URLs into the definition, for example:
`vi:https://www.xyplorer.com|https://www.xyplorer.com/xyfc/index.php|https://www.xyplorer.com/xyfc/viewtopic.php?p=206411#p206411`

- Double-click will open them in the default browser.
- They can be previewed in the Preview Tab, Preview Pane, and in the Floating Preview.
- The command File | Quick File View shows the HTML source code.
- The tab Info Panel | Raw View shows the HTML source code.

4.34 Folder View Settings

Introduction

Folder View Settings (FVS) is the commonly used label for a feature that allows you to associate certain folders with certain view settings. It should really be called "Save and Automatically Restore Folder-Specific View Settings" but that wouldn't look good in a menu.

You know this feature from Windows Explorer: You can save the view settings (view type, sort order, column layout...) of any particular folder and have them automatically restored whenever you come back to this folder. It's a great time saver when e.g. column widths, sort order, or thumbnails view are remembered and automagically adjusted for you.

Of course, the settings are retained between sessions. They are stored in a data file named fvs.dat which is located in your application data path.

Basic Usage

The usage could not be easier: You like the settings of the current folder and want to preserve them? Use the command Save Folder View in menu View | Folder View Settings.

Or even simpler: Use the toolbar button for Folder View Settings (shaped like letter F). And if you ever want to get back to default settings, simply press the button again. The toolbar button also shows you the state of affairs. If there is a Folder View assigned to the current folder, the button is in pressed state.

How it works

What is saved. The following view settings are saved by default with each folder view:

(1) View mode (details, thumbnails...)

(2) Sort order (column, direction). Note that a custom sort order (manual or random) is now also remembered, aka Permanent Custom Sort Order.

(3) Column layout (positions, widths, visibility)

(4) List style (line numbers, auto-size, grid...) (see menu Tools | Customize List)

Optionally a Visual Filter can be saved that will be applied to the folder.

What is restored. You will see later that you can further configure which of these settings are actually restored on a per-folder basis.

When it is restored. If FVS is enabled and you go to a folder for which you have previously saved the view settings, then those settings are restored.

Note that XYplorer's multi-tabs also store view settings, of course. When going to a folder via tab switch, the tab's memory is valued higher than any associated Folder View. This way it is ensured that you always find a tab in the state you left it. The same is true on application startup: The last state will be exactly restored regardless of any active Folder View.

Advanced Usage: Editing a Folder View

You can customize the active Folder View using the Edit Folder View dialog (menu View | Folder View Settings | Edit...). Here you can basically modify two things: which folder(s) are associated with the given Folder View, and which of the saved settings are restored.

How Folders Are Identified

Folder to apply the settings to. Folders are identified by their full name or by a wildcard pattern (* for any string including nothing, ? for a single character). By using a pattern e.g. like `*\Images*` you can define a common Folder View all folders named "Images". Note that it's also possible to state a list of patterns separated by | (pipe). An example for a list of patterns would be `*\Images*|*\Pics*`.

Tip: You can use volume labels instead of drive letters to specify the folder to apply the setting to. E.g.: `System:\` Of course, this general setting has to be ticked to make it work: Configuration | General | Controls & More | Miscellaneous | Support volume labels in paths.

If more than one folder view matches the current folder, then the Best Match Algorithm will decide (see below).

Match case. If checked a pattern `*\Images*` would not match a folder D:\images.

Tip: You don't need to add a trailing backslash "\" to your patterns. XYplorer knows that this is about folders and cares for it. So a pattern "C:" will match folder C:\.

Include subfolders. Check to apply the folder view settings also to all subfolders of the matching folder.

Visual Filter. Here you can define a Visual Filter that will be applied to the folder.

Tip: When you select the Save Folder View command from the main menu any active Visual Filter is automatically stored along with the other Folder View Settings.

Which Settings are Restored

Applied Settings. Here you can define per-folder (or per-folder-view in case of a wild-carded pattern which applies to many folders), which of the following settings are restored:

(1) View mode (details, thumbnails...)

(2) Sort order (column, direction)

(3) Column layout (positions, widths, visibility)

(4) List style (line numbers, auto-size, grid...) (see menu Tools | Customize List)

For the settings that are not restored, the Default Folder View is used (see below).

Auto-Save Changes. If checked then any manual changes to the current folder's view (e.g. you alter a

column width) are automatically stored to the active folder view and will be remembered the next time you come back to the folder.

Make Default. Make the current settings (all checkboxes in the dialog) the default for new Folder Views.

Best Match Algorithm

If more than one folder view matches the current folder then the best match is determined like this:

(1) Match ranking from best to worst in 4 groups: Full Match > Pattern Match > Full Match Including Subs > Pattern Match Including Subs.

(2) If more than one pattern matches within one group, then the longest pattern (character count) wins.

This way you don't have to care about manually sorting the Folder Views (which could be a lot of work if there are hundreds of them!).

Apply this Folder View Also To

The command Apply this Folder View Also To... will open the same dialog as Edit.... The only, but important, difference comes up when you press OK. When the dialog was invoked by Apply this Folder View Also To... any currently active folder view will not be overwritten when you have defined a different name/pattern.

The Default Folder View

By using the command Define this Folder View as Default you can define a Default Folder View (DFV) for all folders with no associated specific folder view.

Note that even if you do not define a DFV, an *internal* default folder view is always applied if no specific folder view is active! The advantage of saving an explicit DFV is that you can configure it using Edit Folder View dialog, i.e. you can determine which settings are applied and whether Auto-Save Changes is enabled. The *internal* default folder view always restores all settings and auto-saves all changes.

Further Remarks

Portability of Folder View Settings

Folder specifications support portability, but in a somewhat limited way, since wildcard patterns are also allowed, so ambiguity must be avoided.

If XYplorer is at C:\Program Files\XY\XYplorer.exe then

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  Pattern           is resolved to
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  MyFolder          C:\Program Files\XY\MyFolder\
```

```
.\MyFolder        C:\Program Files\XY\MyFolder\
..\MyFolder       C:\Program Files\MyFolder\
..\..\MyFolder    C:\MyFolder\
..\MyFolder*      C:\Program Files\MyFolder*\
?:\MyFolder       C:\MyFolder\
D: ...            [unchanged: not relative]
\\ ...            [unchanged: not relative]
*MyFolder*        [unchanged: is pattern]
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Note that "..\MyFolder*" is resolved because the initial "." (dot) is interpreted as "Please, resolve this relative path!".

Browse mode only: Note that FVS is only implemented for browsing folders. Not for the drives listing, and not for search results listings.

Menu Commands. See Folder View Settings (Submenu).

4.35 Custom File Icons

Introduction

Custom File Icons means you can define which icons are shown for files and folders within XYplorer.

The customization does not change the system at all (other applications will still show the standard icons). All information is stored within XYplorer.ini and thus is fully portable. This means you now can take your file icons with you and have the same icons on any host system. Hence this feature could also be called "Portable File Icons".

How to Customize Your File Icons

Open the interface to customize the file icons via menu Tools | Customize File Icons... Alternatively you can use Tools | List Management | Custom File Icons...

To (de)activate the feature (un)tick Configuration | Refresh, Icons, History | Icons | Show custom file icons.

Rules

- Custom icons can be assigned to generic file types (e.g. "*.png"), to specific file names (e.g. "Readme.txt"), and also to specific files ("C:\pagefile.sys"). Actually to any wildcard pattern your fantasy is able to create.
- Custom icons can also be assigned to folders, drives, servers, and shares. For this to work the patterns must end with a backslash.
- Custom icons can also be assigned to special folders. In this case the matching is case-sensitive and the patterns should NOT end with a backslash and have no wildcards. Actually it is recommended to use environments variables for the patterns since these names are system-specific. Supported are the following special folders: %rapidaccess%, %computer%, %desktop%, %personal%, %user%, %net%, %recycler%.
- General syntax for a CFI definition:
  `"Caption" Pattern(s)>Icon Source`
  The caption is optional and has no function anywhere in the GUI. You might use it as a place for personal comments.
- You can create as many CFI definitions as you like.
- You can assign one or more pattern to an icon source, separated by "|" or ";", e.g. `*.jpg;*.png`.
- Patterns support XY native and environment variables.
- Matching is case insensitive (A==a).
- The patterns are tested from left to right, top to bottom. The first match wins.

- The patterns are matched against the whole path of the item that wants an icon. Wildcards *, ?, and # are supported as well as character groups (e.g. [2-8] or [a-f]).
  A bit of built-in smartness auto-adds implied wildcards, so e.g. "test.txt" would match all files called "test.txt" in any location because the pattern in internally completed to "*\test.txt" to match the whole path of the items in question.
- You can as well state whole filenames or even full paths as pattern. If you don't provide any wildcards then by default an asterisk (*) is prefixed to the pattern internally (unless the pattern is a full path).
- A pattern "txt" is interpreted as "*.txt", so you can spare some typing here.
- Generic file types are supported, e.g. {:Image}. For example, this line would set icon "Look.ico" to all image files:
  `{:Image}>Look.ico`
  Note that generic file types can be combined with other patterns:
  `txt;{:Image};xys>Look.ico`
- Use "*." as pattern for files without any extension.
- You can also customize the generic icon for files of an unknown file type, i.e. those files that would normally show the default system icon for unknown file types. The pattern to be used for this is "*?".
- You can also customize the generic icon for folders, in normal and selected state (the later is only used in the Tree). So you can have Win8 folder icons in WinXP or vice versa, or whatever other icons you prefer to see. The pattern for the generic folder icon is "*\", the one for the selected state is "**\". If no generic icon for the selected tree folder (**\) is defined, then it defaults to the generic icon for all folders (*\) if one is defined.
- Portable syntax is supported: Use the "?:\" notation to refer to the drive of this XYplorer instance. Relative paths are resolved relative to the path <xyicons> (= the "Icons" subfolder of XYplorer's application data folder).
- Anything that has a system icon (the icon shown in the file list) can be stated as custom icon source, typically you would take either executables or *.ico files. The icon source can be stated as full path, portable path, or ProgID ("Notepad"). Even XY native and environment variables are supported.
- If an icon is not found, the definition is skipped (ignored).
- Changing icon definitions or enabling/disabling the feature all works smoothly, ultra-fast, and on the fly. No restart required.
- Customizing file icons is a one-time job. Once you have customized your icon associations there is nothing else to do or to learn. The feature is simply there for you and works for you. All you have to do is sit and watch. As simple as TV.
- You can append a comment to a definition, separated by //. Any spaces before the comment separator are trimmed off.
  `*.txt>Houellebecq.ico //comment`

Tips

In the Customize File Icons dialog you can dbl-click an item to trigger the Browse... button and select a new icon.

A Bunch Of Useful Patterns That Will Start You Off

```
%rapidaccess% matches the Rapid Access special folder
%computer%    matches the Computer special folder
%desktop%     matches the Desktop special folder
%personal%    matches the Personal special folder
%user%        matches the User special folder
%net%         matches the Network special folder
%recycler%    matches the Recycle Bin special folder
E:\           matches drive E:\
E:\*\         matches all folders in E:\
E:\ /r        matches all folders in E:\ and E: itself
E:\Jobs\*\    matches all folders in E:\Jobs\
[F-L]:\*\     matches all folders on drives F-L
*\            matches all folders that have no specific icon
**\           same as above for the current tree folder
pics\         matches all folders named "pics"
E:\*          matches all files in E:\
pics\*        matches all files in folders named "pics"
pics*\*       matches all files in folders beginning with "pics"
*.jpg;*.png   matches all JPG and PNG files
*.fnd         matches all FND files and is used for Search Results
*.            matches all files with no extension
*?            matches all files of unknown file type
*23*          matches all files containing "23" in the name
*2013*.txt    matches all TXT files containing "2013" in the name
readme.txt    matches all files named "readme.txt"
<xyicons>\*   matches all files in the default Icons folder
\\Wagner\     matches server "Wagner"
```

Switches

A couple of optional switches provide advanced possibilities. Switches have to be appended to the pattern list separated by a single space. Switches can be combined in any order (e.g. `/dr` or `/rd`) and apply to all patterns in the pattern list.

Switch /d

A switch to only customize default icons.

Use it, for example, to spare folder icons from customizing where the folders are already customized using desktop.ini. The switch is "/d" and it has to be appended to the pattern list separated by a single space. It applies to all patterns in the pattern list. Examples:

```
a*\>Red.ico         [assign red.ico to all folders beginning with "a"]
a*\ /d>Red.ico      [the same, but skip folders with a desktop.ini custom icon]
```

Or use it to define custom icons that are used only if there is no embedded specific icon, for example for executables without a specific icon:

```
*.exe /d>NiceGenericExe.ico //only for EXEs without embedded specific icon
```

Switch /r

A switch to assign a folder icon also to all subfolders of a folder. Examples:

```
code\>Red.ico       [assign red.ico to all folders named "code"]
code\ /r>Red.ico    [assign red.ico to all folders named "code" and their subfolders]
```

Content-Based Folder Icons: Switches /e /f /n /t /x

These switches can be used to assign a folder icon based on the content of the folder. /e is for empty folders, /f is for filled folders, /n is for branches that do not contain files, but may contain folders, /t is for branches that contain tagged items, /x is for inaccessible folders ("Access Denied"). Definitions can look like the following (using the icons that are part of the release package and placed in the executable path returned by the <xypath> variable):

```
*\ /e><xypath>\XYicon_FolderEmpty.ico //matches all empty folders
*\ /f><xypath>\XYicon_FolderGreen.ico //matches all filled folders
*\ /n><xypath>\XYicon_FolderGray.ico //matches all branches without files
*\ /t><xypath>\XYicon_FolderTagged.ico //matches all branches with tagged items
*\ /x><xypath>\XYicon_FolderDenied.ico //matches all inaccessible folders
```

Switches /e and /f (Empty/Filled): Note that determining whether a folder is empty or not is work and takes some time. So you pay a small price, but what you get might be worth it: everywhere, even in drop-down lists, menus, breadcrumbs, address bar, etc., you can immediately see whether a folder has contents or not. This includes the tree and all list views, and it works independently of displaying folder thumbnails.

Switch /t (Tagged): Note that the check is only performed against the tags database, the tagged files themselves are not accessed in any way. It's therefore very fast, but the results will include any orphans in your tags database (orphans can occur when other programs change the names or locations of tagged items).

It's all about the contents of the folder. Whether or not the folder itself is tagged is irrelevant.

If such a content-based folder icon is defined and enabled, the (un)tagging of items will be a bit slower than usual, because the folder icons in the whole interface have to be checked for necessary adjustments.

Network: By default, network locations (UNC or mapped) are not checked for contents (they can be quite slow). You can, however, include them by ticking "Check Network Locations for Content-Based Folder Icons" in the context menu of the "Show Custom File Icons" toolbar button.

Sleeping Drives: By default, folders on sleeping drives are not checked for contents (this would wake these drives). You can, however, include them by ticking "Check Sleeping Drives for Content-Based Folder Icons" in the context menu of the "Show Custom File Icons" toolbar button.

Factory defaults

The XYplorer download package includes a couple of folder icons in Win11 style for your pleasure, and three of them are predefined as Content-Based Folder Icons. You just have to enable the feature to see them in action. These icons are not located in the Icons folder, but in the folder where XYplorer.exe is located. These are the definitions:

```
*\><xypath>\XYicon_FolderGeneric.ico //matches all folders
*\ /e><xypath>\XYicon_FolderEmpty.ico //matches all empty folders
*\ /x><xypath>\XYicon_FolderDenied.ico //matches all accessed denied folders
```

You can, of course, use other icons of your own choice or creation.

Icons for the Selected Tree Folder

You can also define special icons for the currently opened/selected tree folder. So in the folder tree, each item can be assigned two icons, one for the normal state and one for the open state. In the CFI definition simply append the desired open state icon to the normal icon, separated by a "|" (pipe).

Some examples for defining the normal and the open state icon for all selected tree folders in drive E:\:

```
E:\*\>pot.ico            Show pot for normal and for open state
E:\*\>pot.ico|           Show pot for normal, system default for open state
E:\*\>|honey.ico         Show system default for normal, honey for open state
E:\*\>pot.ico|honey.ico  Show pot for normal, honey for open state
```

Notes:

- The Customize File Icons dialog does not fully support the double-icon syntax. The icon is not displayed in the list, and the "Browse..." button will only set the normal icon, so you will have to type or paste the open state icon name.
- If no open state icon is defined the normal icon is used by default.
- The open state icon is only shown in the Tree. Its typical use is to make the selected folder stand out from the crowd.
  In post-XP Windows Explorer this functionality has been completely dropped. Now you have it back in XYplorer.

Exclusions

You can exclude certain folders and files from the customization and show them with their standard icons. To exclude all items matching a pattern simply leave the icon source empty in the CFI definition.
Examples:

```
E:\*\         exclude all folders in E:\
E:\ /r        exclude all folders in E:\ and E: itself
pics\         exclude all folders named "pics"
E:\*          exclude all files in E:\
pics\*        exclude all files in folders named "pics"
pics*\*       exclude all files in folders named "pics*"
*23*          exclude all files containing "23" in the name
<xyicons>\*   exclude all files in the default Icons folder
```

Of course, the exclusions will only take effect when they match before any customization patterns, i.e. when they are positioned higher in the list of definitions.

By the way, adding the separator ">" after the pattern is optional and makes no difference.

Examples

For testing you can paste these examples into Tools | Customize File Icons... (toggle to Editor mode first!). The icons are given without path so they should be located in the Icons subfolder of the application data path.

Of course, for these examples to work you first need to get hold of ICO files with matching names!

```
+%computer%>cup_coffee.ico
+C:\WINDOWS\>machine.ico
+"XYplorer Configuration Files" <xypath>\*.ini;<xydata>\*.ini>rubberduck.ico
+"XYplorer Script Files" xys>xys.ico
+"XYplorer Icon Path" <xyicons>\>SmileyHearts.ico
+*\>Blue.ico
+**\>Blue_Open.ico
+code\>Green.ico
+docx>D:\Program Files (x86)\OpenOffice.org 3\program\swriter.exe
+xy*\>xy.ico
+"No Extension" *.>Notepad
```

Tip: Let all XYI files show the system icon of INC files:

```
+*.xyi>*.inc
```

4.36 Customize Keyboard Shortcuts

Introduction

Open the Customize Keyboard Shortcuts dialog by menu Tools | Customize Keyboard Shortcuts (Shift+F9). The dialog largely follows the de facto standard for such interfaces in Windows applications, so you'll immediately find your way.

Remarks

You can use any combination of Ctrl/Shift/Alt (incl. none) with any key on your board (with a few obvious exceptions, see next).

Some special keys like the Windows key, the Context Menu key, and the Numlock key are not available for customization. They are not used at all by the program.

Also the Tab key, the Space key, and the eight navigation keys are not customizable: they are internally hard-coded to their traditional tasks in navigating the GUI, and cannot be assigned nor removed by the user. Exceptions: Tab can be used in all combinations with Ctrl. The Space key and the navigation keys can be used in all combinations with Alt.

You can assign as many different shortcuts as you want to each of the functions.

You can also assign shortcuts to (non-clickable) menu items heading a submenu. On pressing the shortcut the submenu will pop-up at the current position of the mouse cursor.

You may also press new shortcuts while the focus is in the Commands list. Press <Enter> (or click Assign) to assign a new shortcut to the selected command. Press <Del> (or click Remove) to remove an existing shortcut from the selected command.

Click Default to add the factory default shortcut to a command.

Commands list context menu

The Commands list got a little context menu where you can copy the selected command to clipboard in various formats.

You can as well trigger any command right from here by chosing Close Dialog and Trigger Command. This is especially useful with the miscellaneous commands, which are not available anywhere else. You now can use them without having to assign a shortcut first.

Button "Options..."

The button pops a tiny menu with a couple of useful commands.

Copy Cheat Sheet. Create and copy a cheat sheet of all current (where "current" is the current state of the CKS dialog, even if it’s not OK-ed yet) shortcuts and commands in various layouts and sorting

orders.

Reset All Shortcuts To Defaults... Sets all shortcuts back to factory defaults.

Reset Unused Shortcuts To Defaults... Will assign those shortcuts to their factory default function that are currently unused. Might come handy when upgrading and new functions with default KS have been added.

Remove all Shortcuts... Well, removes all shortcuts.

Button "All Free Shortcuts..."

Pops a dialog presenting you all key combinations that are currently unused.

There's keyboard support. You press your desired key (including Ctrl, Shift, Alt combinations) and if it is in the list, it will be focused. Otherwise, if at least your key (without Ctrl, Shift, Alt) exists, its first occurrence will be focused. The Enter, ESC, and cursor keys are excluded from this service to keep the dialog efficiently usable for keyboard users.

Button "Jump..."

Allows you to directly to jump to a command via a filtered list, similar to the "Jump to Setting" button in Configuration.

Portable Storage

The shortcuts are saved in the application path in a binary file called "ks.dat". This allows you to carry your personal keyboard shortcut set with you.

Scope

Each function has its scope: only if the focus is within that scope pressing the shortcut will actually trigger the command. The scope can be Global (not restricted), or limited to certain parts of the interface like Tree, and/or List, etc.

Category “Miscellaneous”

This category contains commands that do not actually have any interface/menu (and, therefore, can only be triggered using the shortcuts assigned to them).

Last Size/Minimize Info Panel

Toggle the panel height between minimum height and last user selected height (by dragging the splitter). The same can be achieved by dbl-clicking the Info Panel splitter.

Maximize/Minimize Info Panel

Toggle the panel height between maximum and minimum.

4.37 Custom Context Menu Commands

Custom Context Menu Commands (Tree)

Configurable in Configuration | Menus, Mouse, Usability | Context Menus | Custom items in the context menu | Folder Tree.

Open in New Tab

Opens the right-clicked folder in a new tab.

You can as well "Open in New Tab" by simply holding Shift when clicking a folder in the tree.

Open in New Background Tab

Opens the right-clicked folder in a new tab without actually selecting the tab and listing the folder's contents.

You can as well "Open in New Background Tab" by simply holding Ctrl when clicking a folder in the tree.

Open in Other Pane

Will open the selected folder in the other pane (whether that pane is visible or not). Note that it also works when right-clicking a folder other than the current folder.

Shortcut: Alt+Click

Create New Subfolder Here

Creates a new subfolder to the right-clicked folder, selects it, and invokes rename mode.

Shortcut: Ctrl+N

Copy Path

Copies the full path of the right-clicked folder to the clipboard.

The "Copy Path" command is accompanied by a "Copy Special Path" command, if applicable. It will copy e.g. "Desktop" instead of "C:\Users\Donald\Desktop".

Shortcut: Ctrl+P

Empty Folder

Empties the right-clicked tree folder.

Flatten Folder

Flattens the right-clicked tree folder.

Note that this action supports Undo/Redo in two steps: 1) restore the deleted empty folders, 2) move the files back into them.

Hide Folder from Tree

Removes this folder from the Mini Tree.

Hide Siblings from Tree

Removes all siblings (folders under same parent) of the current folder from the Mini Tree.

Open Command Prompt Here

Opens a command prompt in the right-clicked folder.

Shortcut: Ctrl+Alt+P

Expansion (submenu)

Fully Expand: Fully expands the right-clicked folder.

Shortcut: Numpad Multiply, or Ctrl+Click the plus-icon of the folder

Fully Collapse: Fully collapses the right-clicked folder.

Shortcut: Numpad Divide, or Ctrl+Click the minus-icon of the folder

Fully Collapse Drive: Fully collapse current drive. The drive root will automatically be selected.

Shortcut: Ctrl+Numpad Divide

Optimize Tree: Only expand the current path, collapse all other paths.

Shortcut: Shift+Numpad Divide

Mark (submenu)

Contains various commands for highlighting folders.

Move/Copy/Backup To

Contains commands for file operations.

See menu File | Move To.

Add Folder to Toolbar

On selection a new Droppable User Button is appended to the right end of the Toolbar pointing to the folder. The command is only available if the Scripting feature is enabled.

Custom Context Menu Commands (List)

Configurable in Configuration | Menus, Mouse, Usability | Context Menus | Custom items in the context menu | File List.

Open in New Tab [folders only]

Opens the selected folder in a new foreground tab.

Open in New Background Tab [folders only]

Opens the selected folder in a new background tab.

Open in Other Pane [folders only]

Opens the selected folder in the other pane (whether that pane is visible or not).

Empty Folder [folders only]

Empties the right-clicked folder.

Flatten Folder [folders only]

Flattens the right-clicked folder.

To Clipboard

See menu File | To Clipboard.

Rename Special

See menu File | Rename Special.

Move/Copy/Backup To

See menu Move/Copy/Backup To (Submenu).

Copy/Move to Other Pane

The commands Panes | Move to Other Pane and Panes | Copy to Other Pane are show if Dual Pane is enabled.

Zip

Zip View: The contents of the selected archive are shown in a list.

- The items are sorted by path/name ascending.
- The items can be live-filtered to quickly find a particular file in an archive.
- Encrypted archives are marked as such in the top section of the dialog, and with a "+" behind the filenames.
- You cannot extract from or add to the zip via this interface, it's merely a view.
- The dialog shares size and position with the Quick File View.
- If you opened the preview by F11 you can also close it by F11 (or whatever other KS you have assigned to Floating Preview).
- The dialog is non-modal, so you can leave it open and preview Zip archives by simply selecting them in the file list.

Extract Here: Extracts the selected archive(s) here. Also extracts *.rar and other WinRAR formats if WinRAR is installed on the system, and *.7z and other 7-Zip formats if 7-Zip is installed on the system.

As a special service the command "Extract Here" also shows the top level folder, e.g. "Extract Here (Icons\)", if all contents of the archive are internally subsumed under one top level folder. This tells you that you can safely use "Extract Here" without making a mess of your current location.

Extract to {Folder}: {Folder} is auto-generated from the name of the archive, and auto-suffixed on collision.

Extract to Other Pane: Extracts the selected archive(s) to the other pane.

Paste into this Zip: Lets you paste the items in the clipboard into the right-clicked zip file.

Add to Zip Archive...: Lets you add items to an existing or new archive. You are prompted for a name for the archive.

Add to {Zipname Item}: {Zipname Item} is auto-generated from the item owning the context menu, and auto-suffixed on collision.

Add to {Zipname Parent}: {Zipname Parent} is auto-generated from the parent folder of the item owning the context menu, and auto-suffixed on collision.

Note that Zip Support needs Zip Folders enabled in Windows (which it is by default).

Move Up

Moves the selected items to the folder above their current folder.

Copy Path

Copies the full path of the right-clicked item(s) to the clipboard.

The "Copy Path" command is accompanied by a "Copy Special Path" command, if applicable. It will copy e.g. "Desktop\EnglishBritish.zip" instead of "C:\Users\Donald\Desktop\EnglishBritish.zip".

Open With...

Pops the Portable Open With menu.

Replace with File in Clipboard [files only]

Replaces the contents of the selected file with the contents of the file in clipboard. The selected file keeps its filename, just the contents are updated. Notes:

You are prompted before the replacement happens.

The action cannot be undone, always runs in the foreground, and is performed by kernel API (no Custom Copy or Shell Copy involved).

The file in clipboard is not deleted even if it had been "cut" to the clipboard.

4.38 Custom Drag'n'Drop Context Menu

Note

To show the Custom Drag and Drop Context Menu you have to check Configuration | General | Menus, Mouse, Usability | Context Menus | Native drag and drop context menu. Else the standard Shell Context Menu is shown.

Native Drag and Drop Context Menu Commands

Move Here

Move dragged stuff to drop location.

Copy Here

Copy dragged stuff to drop location.

Duplicate a file: If you want a copy/duplicate of any file, drag it with right-click just a little bit (you don't need to drop the file in another place). On mouse up, a popup menu will open with all sorts of copy, backup, create and move operations - even more than in the File | Duplicate submenu. Choose Copy Here to create a duplicate in the same location as the original.

Backup Here

Backup dragged stuff to drop location. See details.

Copy / Move Here As...

Allows you to create a renamed copy of the dragged file. The command is available only if exactly one file is drag-dropped; when triggered by keyboard it applies only to the focused file. If source and target path/file are identical, the copy is auto-renamed to "Copy of Original Name" (auto-rename scheme varies depending on Operating System locale).

Shortcut for Copy Here As: Ctrl+S (think "Save As...")

Copy / Move Here With Number

Creates auto-serial-numbered duplicates of the dragged file(s). Examples:

My.txt => My-01.txt

My.txt => My-02.txt (if My-01.txt already exists)

My.txt => My-03.txt (if My-01.txt and My-02.txt already exist)

etc.

Shortcut for Copy Here With Increment: Ctrl+D (think "Duplicate")

Copy / Move Here With Current / Last Modified Date

Makes a copy [or move] of the selected item(s) and automatically appends the last modified date [or current date] of the copied/moved item(s) in a format defined by a template. Date-stamped copies are very, v-e-r-y, useful when working with versions. If an item by the new name already exists, auto-incremented numbers are suffixed.

Shortcut for Copy Here With Last Modified Date: Ctrl+Shift+D (think "Duplicate Dated")

Copy / Move Here to New Subfolder...

Usage:

(1) select any number of files

(2) right-drag them somewhere

(3) click "Move (Copy) Here to New Subfolder..." in context menu

(4) dialog opens, enter new folder name, click OK

(5) this folder is now created under the dragged-to folder...

(6) ... and the selected files are moved (copied) into it

You are presented an array of 4 possible default names for the new subfolder about to be created. The first three are (can be) defined in the [NewTemplates] INI section. Here's an example:

```
[NewTemplates]

Folder0=\N\e\w \F\o\l\d\e\r

Folder1=yyyymmdd

Folder2=yyyy-mm-dd
```

The fourth is made from the first item to be moved [copied, pasted] into the new subfolder, i.e. the item that was focused in the moment of cutting, copying, dragging. The index of the last selection is remembered between sessions.

Tip: You can as well define a whole new subpath (e.g. download\2008\new) as target. It will be created on the fly (if necessary).

Tip: The forward slash "/" is supported as alternative folder separator. It will be auto-converted to "\".

Tip: The dialog suggests the last used Selection Filter as new subfolder name, if that filter is a valid filename, and if it was used less than 2 minutes ago.

Copy / Move Here with Path...

Copy or move items together with parts of their path. In other words, the copies will end up somewhere deeper below the actual drop target depending on your choice.

The dialog offers you a complete set of path choices auto-generated from the drop target and the source item(s). Any non-existing subpaths will be silently created in the target; non-existing paths are marked by the "New Folder" icon in the dialog (unless the target is a network location, where the standard icon is shown for better performance).

Note that in the dialog there is some smartness at work: The path which XYplorer thinks you are likely to choose is preselected. E.g., if you drag directly onto a drive root (or network share), the path that

will mirror your source path is preselected. Of course, you can change the preselection, and even type in a different path from the ones offered in the dropdown (it has to be located below the drop target though).

The same functionality is available for pasting from clipboard in menu Edit | Paste Special | Paste Here with Path....

These commands come in very handy when restoring or updating individual files from one deeply nested place to another one.

Rich Move Here

Move dragged stuff to drop location, recreating the source's folder structure in the destination folder. This command is only available in this context menu and only when dragging find results. See details.

Rich Copy Here

Copy dragged stuff to drop location, recreating the source's folder structure in the destination folder. This command is only available in this context menu and only when dragging find results. See details.

Create Folder(s) Here

Drag a folder (or several folders) onto a target location and create a copy of this folder without any of its contents in the target location. It's the flat version of "Create Branche(s) Here" (below). Very handy feature to quickly create some folders by simply drag-dropping them to their birth-place.

Create Branch(es) Here

Drag a folder (or several folders) onto a target location and create a copy of this folder and all nested folders below it in the target location, without copying any contained files. What you get is a perfect copy of a tree structure, completely empty.

Note This operation obviously will only work when you drag folders, not just files. If you forget, you will be kindly reminded.

Extract Here

Extracts the dragged archive(s) here. Also extracts *.rar and other WinRAR formats if WinRAR is installed on the system, and *.7z and other 7-Zip formats if 7-Zip is installed on the system.

Zip Here

Creates a Zip archive from the dragged items here.

Go to Dragged Item

Will go to the dragged item's containing folder and select the item, or simply open the folder if the item is a folder. If more than one item is dragged the first item is taken.

The command is only available for the List and the current Tab of the active pane. But, of course, each hovered tab becomes the current tab after a configurable while.

Main usage is probably when dragging items or paths from other apps into XYplorer. It's a kind of "Go to by Drag and Drop".

Shell Context Menu

Show the standard Shell Context Menu.

Drop on Zip

You can right-button-drop items into a zip file. A mini menu pops up with one command: Drop into Zip.

Drop-Text-To-File

Yet another revolutionary XYplorer-only feature. Now you can drag & drop a text chunk onto Tree, List, Catalog, or Tab header, where it will be automatically converted into a *.txt-file called something like "DroppedText-20061007.txt". The "-20061007" part is the date now (format defined in INI at key PostfixDate). After creation of the file, rename-mode is invoked. Try it, for example, with your web browser. Select a piece of text on a web page and drag it onto XYplorer...

Tips

Now you can do some pretty amazing things, for example:

(1) You can drag-drop selected text bits from XYplorer's own Web preview!

(2) You can drag-drop selected text bits from PDF documents and they will be saved as "Untitled[-##].rtf"!

(3) You can even drag-drop selected image bits (details of embedded images) from PDF documents and they will be saved as a BMP named to the full path/file of the source file!

(4) You can drag-drop images directly from web pages Browser-to-XYplorer.

4.39 Copying files with XYplorer

There are four different file copy commands in XYplorer, so here's a quick comparative overview.

Copy

The standard Windows file copy operation (the only copy Explorer can do).

Source: All currently selected files/folders in List (normal browse or find results), or the currently selected folder in Tree.

Target: All files contained in the source are copied to one and the same target folder. Any folders contained in the source are created below the target folder including all their contents.

Prompts: When a file or folder with the same name already exists in the target location, you are asked whether you want to overwrite the existing item. Your choices are Yes, Yes to All, No, or Cancel. If you answer Yes or No, you will be asked again every time this happens.

Issue 1: since there is no No to all button you'll have a hard time clicking Yes or No when doing a large synchronize job. This issue is one of the reasons why there's the Backup command (see below).

Issue 2: when using this command on find results of a recursive file find (subfolders included), the operation might lead to unwanted results: first, if the copy cargo contains files of the same name (eg. you might have found a lot of "index.htm" files in various locations), you'll face the overwrite prompt for every non-first file of the same name, since Copy puts all files to one and the same location; second, if the copy cargo contains folders, all their contents will be copied, too, although they probably do not all match your find criteria. This issue is one of the reasons why there's the Rich Copy command (see below).

Custom Copy

The details of this command are described here.

Source: Same as Copy above.

Target: All files contained in the source are copied to their relative path below the target folder (any needed subfolders are automatically created along the way).

Prompts: Can be configured.

Issues: This command avoids Issue 1 (above).

Backup

The details of this command are described here.

Source: Same as Copy above.

Target: All files contained in the source are copied to their relative path below the target folder (any needed subfolders are automatically created along the way).

Prompts: Can be configured.

Issues: This command avoids Issue 1 (above).

Rich Copy

A special command only available in XYplorer: Copy find results along with their original folder structure. Details are described here.

Source: All currently selected files/folders in List, but only when find results are displayed.

Target: All files contained in the source are copied to their relative path below the target folder (any needed subfolders are automatically created along the way).

Prompts: Same as Copy above. Or can be configured if Custom Copy is used.

Issues: This command avoids Issue 2 (above).

4.40 Rich Copy

Copy/Move items with their folder structure

Rich Copy (Rich Move) is a highly useful feature for archiving recursive search results or Branch Views: It enables you to copy/move files to a destination along with their folder structure, i.e. whatever directories are needed are automatically created under the destination path. You get a perfect image of the files as they were organized in their original location.

Note: You find the Rich Copy commands only in the drag and drop context menu. To do a Rich Copy, drag the selected items from the search results list to the destination folder (in XYplorer's folder Tree, in the file list, the Catalog, or a Tab header) using the right mouse button: On drop, a popup menu will give you the choice between normal and rich operations.

Note: Of course, Rich Copy commands will not appear when you drop outside XYplorer, e.g. to the Desktop! This is because Rich Copy is a feature of XYplorer, not of the Desktop.

Target folder structure

The target folder created by Rich Copy structure is defined by the path where you started the File Find and the target path of your copy/move operation. Example: let's say you do an Rich Copy of just one file, which you have found by doing a find starting in `J:\Test\MyStuff\`, and you drag and drop this file to `E:\MyStuffArchive\`, the new folder `Dir1\` is created under the destination folder and the file is copied there.

```
Source: J:\Test\MyStuff\Dir1\Bvb.ico
Target: E:\MyStuffArchive\Dir1\Bvb.ico
```

As you see, for every file in the find results, the File Find start path is replaced by the target path of the operation.

If the File Find started in Computer, the drive letter of the found files is converted into a folder, eg:

```
Source: [Computer]J:\Test\MyStuff\Dir1\Bvb.ico
Target: E:\MyStuffArchive\J\Test\MyStuff\Dir1\Bvb.ico
```

Automatic Redundancy Removal

Recursive Search Results and Branch Views: In such "deep data" (where containers can be listed side by side with their contents) you will run into name collisions or other unwanted situations when you select e.g. a folder AND any of its contents and then copy, move, or delete the selected items. To avoid this there is smartness built-in which will silently remove any redundant items from the job before processing continues.

Automatic Rich File Operations

Any copy/move operation (drop, paste, to new subfolder, etc.) will check whether its direct sources are "rich" (= come from more than one folder), and then offer to do a Rich Copy or Move instead of the standard flat operation (where things might collide/merge).

An example for a Rich Paste would be pasting two files D:\Test.txt and E:\Test.txt in one go to F:\Target:

Sources:

```
D:\Test.txt
E:\Test.txt
```

Targets:

```
F:\Target\D\Test.txt
F:\Target\E\Test.txt
```

Note that "rich sources" typically come from recursive search results or branch views (i.e. including subfolders).

The root path for such a rich operation is the greatest common parent folder of the sources.

Scripting and Rich Operations

Note that the scripting commands copyto and moveto support rich operations.

4.41 Backup

Backup

What is called "Backup" in XYplorer is a highly customizable file copy operation. You don't necessarily have to use it for backing up files. "Backup" is just a label for an alternative copy, alternative to Windows's shell copy (which has so many pitfalls).

Where to find the Backup command

You find the backup command in various menus:

Backup Here in the drag and drop context menu (popped when dropping items dragged with the right-mouse button).

Backup To in the Edit menu and in the file context menu (which is as well available in the main window menu when any items are selected).

Backup To Other Pane in the Panes menu.

All those commands offer exactly the same operation (as it is configured in Configuration | File Operations | Backup Operations) and differ only in the way you choose your target.

The laws of backup

1st, copy only those files that need to be copied.

If a file does not exist in the target location, it will be copied.

If a file does exist in the target location, the target file will be overwritten only if it is older than the source file.

Note: This behavior ("Overwrite if newer") can be modified in Configuration | File Operations | Backup Operations.

2nd, recreate the folder structure of the source.

The folder structure of the backup will be an exact image of the source's folder structure.

3rd, have a documentation of the backup.

A log-file is optionally created for every backup operation, where every single file operation is exactly documented.

A progress dialog tells you exactly what's happening and provides you with a final report. Alternatively

(progress dialog turned off) you get a status bar message after the operation is completed giving you a minimal summary report:

- new: files of that name did not exist in target
- over: files of the same name were overwritten
- skip: files were not copied, e.g. because they were older than files of same name in target
- fail: files that failed for some reason

Note: This behavior ("Show progress dialog", "Create log file") can be modified in Configuration | File Operations | Backup Operations.

4th, preserve all file dates.

The Backup file operation preserves all three dates (modified, created, accessed) in all copied files and folders.

Note: This behavior ("Preserve item dates") can be turned off in Configuration | File Operations | Backup Operations.

4.42 Undo/Redo

Multi-level Undo/Redo

XYplorer gives you multi-level undo/redo for the file operations Move, Copy, Rename, Delete, and New. Up to 100 user actions are logged and can be undone at any point. They are even remembered between sessions so you can fix today what you broke yesterday.

Usage

Undo is your friend when something went wrong. A typical use for Undo is when you drag-moved some items into the wrong folder, maybe by a slip of the finger, the famous accidental drop. You might not even know which folder it was. With Undo, just one click will bring everything back to normal. Even on the next morning after having closed and restarted XYplorer.

Or a batch rename of 250 files went wrong. No problem: Undo it with one click.

Optionally you can perform a cumulative undo, which means that you can undo a whole bunch of actions with a single click. Or you can perform a non-sequential undo, which means that you can undo any particular action from your action history without undoing the actions that came after.

The default keyboard shortcuts are Ctrl+Z for Undo and Ctrl+Shift+Z for Redo. However, the superior interface for these tasks is probably the toolbar. Because the Undo and Redo buttons inform you about what will be undone/redone, either by a tooltip when hovering the buttons, or by a dropdown list of recent actions when clicking the buttons' arrows.

Things to keep in mind

(1) Undo only knows item names and locations, not data. It will not restore lost data.

(2) Redo is not the Undo of Undo but it will redo the original action.

What is logged and what can be undone

The following operations are logged and can be undone/redone:

- Move: Move To, Cut & Paste, Move Here As (...), etc.
  Note that a command like "Move to New Subfolder" consists of two actions: (1) creating the subfolder, (2) moving the item. They are logged (and undone) separately.
- Copy: Copy To, Copy & Paste, Copy Here As (...), etc.
  Note that Undo will delete the copies for good. Redo will not recreate the deleted copies but copy the originals again.
- Rename: Inline Rename, Batch Rename.
  Note that a Batch Rename operation is logged as one action and can only be undone as a whole.
- New: Create New File/Folder, New Path, New Folders, New Items.
- Delete: Delete to Recycler.

Note that undoing a deletion of more than 5 selected items at the same time will take much longer (depends on the number of items in your Recycle Bin) than smaller numbers. This is caused by certain Windows behaviors (at least under XP).When you redo a deletion (i.e. delete again) there will be no confirmation prompt. When undoing a deletion results in a name collision you get an overwrite prompt.

- Create Folder(s)/Branch(es) Here.
  Note that folders that should be created but were already existing will be permanently removed on Undo.
- Create Shortcut(s): Paste Special, Drag Context Menu.
- Rich Copy.
  Note that Undo will remove the copied items, but it will not remove any created subfolders.

The following operations are logged but cannot be undone/redone:

- Delete: Delete (no Recycler), Wipe.
  No Undo because it's gone. Use recovery software to undo (unless Wipe which cannot be recovered).
- Backup: Backup, Backup To, Backup Here.
  Sync Folders.
  No Undo because Backup and Sync is about overwrite/no overwrite. Either it cannot be undone, or there is nothing to undo.

How to Undo

The recommendations for best Undo results are:

- Trigger Undo immediately after the logged action. The closer you are to the action, the better your chances that it can be perfectly undone.
- Start with the youngest logged action.
- Move through the action log in tight sequence (no skipping).

However, XYplorer's Undo grants you all kinds of liberties:

- You can pick out any logged action and (attempt to) undo it, also known as non-sequential undo.
- You don't have to care about the original sequence of actions. This does not mean, of course, that any sequence necessarily works, but it *might* work.
- The action log is saved between sessions, so theoretically you can undo actions that were done months ago.

Of course, since all logged actions are happening within one file system, it should be clear that they are interrelated. So undoing action #2 might be impossible before having undone action #1, or undoing action #2 might make impossible to undo action #1 later.

The Limits of Undo

"A man can undo what a man can undo."

Just keep in mind that XYplorer's Undo is not a forensic recovery tool for lost data. Not a single byte that was overwritten or permanently deleted will be brought back to life by Undo. Always remember:

(1) Undo only knows item names and locations, not data. It will not restore lost data.

(2) Redo is not the Undo of Undo but it will redo the original action as if you would redo it manually now.

Here are some examples that should be obvious by now:

(1) Create a New File; Undo will delete it; Redo will create a file of that name again. If you put any contents to that file then they will not magically come back when you Redo the creation of the file. Same with New Folder.

(2) Delete a file for good (no recycle bin): Cannot be undone using Undo.

(3) Move/Copy a file and overwrite same named items in the target folder: The overwritten data are lost and no Undo will bring them back.

(4) XYplorer disallows that any data are overwritten by an Undo or Redo operation. So e.g. if you try to undo a move, but in the meantime a file of the same name as the originally moved file came to existence, the Undo will not happen. In case of undoing a delete to recycler the shell will pop an overwrite prompt.

(5) If you undo the creation of a folder, any contents the folder might have gathered in the meantime will be lost and don't come back by redoing the creation.
There's a configuration option to have undo/redo delete to the recycle bin. This might save your neck in case of an erroneous undo.

(6) If you undo a Copy and then Redo it, the copy will be made from today's state of the original source and thus will differ from the original copy if the source has changed in the meantime.

Further Notes on Undo/Redo

You can stop an ongoing undo/redo by pressing ESC.

Action Log

The Action Log is the database for Undo/Redo. It tells you in chronological order what you have done, when you have done it, whether it can be undone, or whether it has already been undone and can be redone. It's optionally remembered across sessions so you can finally find out what you did yesterday.

(1) Note that also actions are logged that cannot be undone. E.g. no-recycle-deletions: you can still see what items you have permanently deleted and when; information that can be very useful when trying to recover them using a recovery application.

(2) The Action Log currently holds a maximum of 256 entries, older entries are forgotten. The actual size can be lowered in configuration in case you want to save a little memory. But don't worry, it

does not take much memory anyway, so it's recommended to leave it at the max.

(3) Note that you can undo actions right from the Action Log using the right-click menu.

(4) Backup and Sync operations are added to the Action Log (both as "Backup"). You cannot undo them (it would be meaningless) but you have them nicely logged so you can look up what you have done and when.

(5) The Action Log works as a history where you can go back and forward and be at a certain position between both ends. Now when you go back (Undo - Undo - Undo - ...) and then trigger some logged action (e.g. delete a file), then the previous future branch is dropped in favor of the new one.

(6) In the Action Log Dialog the green highlighted item is the current item in action history. It's the action that will be undone when you click Undo. Usually it will be the top item, but it moves downwards when you undo actions. When the complete action history is undone you are in "Prehistory". Take care.
You can set any item to be the current action using the right-click menu of the Actions list.

(7) The action is limited to save resources: Only actions with 32,767 or less items (that's the directly selected items, not any folder contents) involved are stored in the action log and can be undone.

(8) The Action Log is stored in the file "actions.dat".

Icons in the Action Log

The two lists in the lower part of the Action Log show the items involved in the selected action, before and after the action. Items that exist at this very moment show an icon, items that don't exist show no icon. So for example with a Move operation, the upper list will typically show no icons (the items have been moved away from that location), and the lower list will show icons (if the items are still where they have been moved to).

4.43 Recycle Bin

Recycle Bin Deluxe

XYplorer's Recycle Bin special folder surpasses Explorer's Recycle Bin in a number of ways:

1. The real path/name for each item is displayed in a separate column "Recycled Name". Ctrl+P will copy that info to the clipboard.
2. The context menu's custom command "Go to Real Location" will beam you to the real location of the item and select it in XYplorer's raw recycle bin view.
3. The context menu's custom command "Go to Original Location" will beam you to the original location of the item.
4. You can open (Enter, Dbl-Click) a deleted folder to browse its contents.
5. You can show the recursive folder sizes (i.e. including subfolders) of deleted folders right in the list (as usual in the Size column).
6. You can see the Space Used or Real Size of deleted items (Explorer shows only Space Used).
7. You get the exact time of deletion down to the second (Explorer only tells you the minute).
8. Apply Visual Filters to the Recycle Bin.
9. Sort folders separately from files.
10. Color code items in the Recycle Bin just like normal files.
11. Jump and Spot: Jump to items, and place a spotlight on them.
12. Compare deleted items.
13. Preview deleted items.
14. Open deleted items.
15. You can hide the node from the tree (Configuration | Items in Tree and List).
16. Images in the Recycle Bin can show thumbnails in Thumbnails and Tiles modes, the thumbnails can be cached, and they support Mouse Down Blow Up. In Tiles modes Photo Data are supported.
17. Full Meta Data (Shift+Enter) are shown also for items in the Recycle Bin.
18. And, last not least, it's faster.

Remarks

The Shell default action (Enter, Dbl-Click) for Recycle Bin items is to show the item's Properties dialog. This will happen in XY as well, unless you have a CFA defined for the file type in question.

The Recycle Bin is a pretty singular construct superimposed onto the regular file system and comes with a couple of pecularities:

- It is slower than "normal" folders.
- No drag from the Recycle Bin (but you can drop items onto the Recycle Bin to delete them).
- No renaming or cutting or moving or duplicating.
- No pasting.
- No custom File Info Tips, just the default ones.

The Len column refers to the original Path/Name, not to the Recycled Name.

The Info Panel refers to the Recycled Name because this is the file that exists, has properties, and can be previewed.

Raw Recycle Bins

The Recycle Bin special folder is a virtual folder that has its real base in a set of regular file system folders, one per volume (drive letter). Such a folder might be called "C:\RECYCLER\" under XP or "C:\$Recycle.Bin\" under Vista/Win7. This is where the deleted files continue their half-life until they get either restored or permanently deleted.

Contrary to Explorer, XYplorer allows you to browse these "Raw Recycle Bins" just like any normal folder. It is not recommended to delete any files out of a Raw Recycle Bin, unless you really know what you are doing! You risk corrupting the database underlying the virtual Recycle Bin.

4.44 Portable Devices

Portable Devices

From v15.00 onwards XYplorer supports so-called Portable Devices, i.e. devices like tablets, smart phones, or digital cameras, devices that are connected to the PC without being assigned a drive letter.

Portable Devices support is available from Windows Vista and later.

What works and what does not

File management on Portable Devices is quite limited compared to what you can do with XYplorer within the regular file system. Therefore there is an info bar in the list to clearly show that you are browsing a Portable Device (PD). The color is deep purple (DP).

Here is an overview of what can be done and what cannot be done:

This works:

- Browse the device, list files and folders with all data columns provided by the Windows shell.
- Breadcrumb Bar.
- Thumbnails (embedded thumbs only) with caching.
- Find Files (Name, Size, Modified, Attributes). No search results caching.
- Quick Search (Name, Size, Modified, Attributes).
- Visual Filters (Name, Size, Modified, Attributes).
- Color Filters (Name, Size, Modified, Attributes).
- Branch View.
- Custom File Icons.
- File operations to, from, and on the device: Copy, Move, Rename, Delete.
  Note while you can delete files from the device Undo is NOT possible here!
- You can create new files and folders on a device. Also the New Items menu works.
- Undo most of those file operations (Delete cannot be undone on a Portable Device).
  However, copy/cut from a Portable Device and paste to anywhere cannot be undone.
- Batch Rename.

This does not work:

- Custom Copy and Backup operations.
- Backgrounded file operations.
- Open files, or drag them into open applications.

- Show Folder Sizes.
- Tags.
- Folder View Settings.
- Custom File Associations.
- Custom Columns.
- Raw View.
- Paper Folders (you cannot list Portable Devices paths in Paper Folders, or convert Portable Device folders to Paper Folders).

Further Remarks:

- The first operation to or from a newly mounted device can take up to 30 seconds before it actually starts.

A Word of Warning...

There is a serious shell bug (a Windows bug, not an XYplorer bug; Explorer has it, too) in Windows 8.1 (and probably other Windows versions) that can lead to permanent data loss: When moving an item to a Portable Device (from the PC or from another location of the Portable Device) where an item of the same name already exists then on cancelling the overwrite prompt the move does not happen but the source item is deleted anyway!!! Also X-closing the overwrite prompt will lead to the destruction of the source item.

To see the bug in XYplorer use right-mouse drop, then Shell Context Menu... from the context menu, and then Move Here from the shell context menu.

...and a Tip

Some Portable Devices (typically digital cameras) do not allow (most, but not all!) operations that change the contents of the device. In other words they are (almost) read-only. However, those devices don't necessarily pop an error message if you try an invalid operation but simply do nothing. For example, and, just to make sure, this is also the case when working with Explorer: You can move a file from such a device to the PC, but you cannot move it back. One-way traffic!

To avoid surprises and frustration tick this setting: Configuration | General | Safety Belts, Network | Safety Belts | Treat portable devices as read-only. Now XYplorer will let you know beforehand that you attempt an operation that would change the content of the device and not continue with it. But note that even this safety belt cannot intercept all file operations triggered via the (drag and drop) shell context menu.

With the above setting ticked all you can do is copy items from the device to the PC. Given the limited file management you get on a Portable Device this is the recommend procedure anyway.

Part V

Advanced Topics

5 Advanced Topics

5.1 Admin Settings (Enterprise Edition Only)

Enterprise Edition Only

You need an Enterprise Edition license to use Admin Settings to enable Access Control and Multi-User Tagging, features that only make sense in a corporate network. All Professional Edition licenses purchased before 2023 will be internally upgraded to the Enterprise Edition at no extra charge.

Admin Settings

You use XYplorer in your company and don't want your employees to check for updates? You don't want them to edit the user commands? You don't want them to change the configuration? All this and more is possible through Admin Settings.

You (or your administrator) simply place a file "Admin.ini" in XYplorer's application path. The contents of this file will control the layout and functionality of the application in a way that a user without admin rights cannot influence. XYplorer will read this file at startup but never write to it. If you intend to use non-Latin characters in Admin.ini, it should be in UTF-16LE encoding.

[Settings] Section

Key: Profile

For example, to hide and disable the Check for Updates feature, you simply enter this into Admin.ini:

```
[Settings]
Profile=64
```

Here's an overview over the available values and their respective effects:

8	Disable menu User | Manage Commands...
16	Hide menu Tools | Configuration...
32	Hide menu Tools | Open Configuration File...
64	Hide menu Help | Online Support (includes Check for Updates and other online functions).
128	Hide menu Help | Update Registration Details.
256	Hide clickable URLs in Help | About XYplorer.
512	Disallow the user to modify tags (Labels, Tags, and Comments).

1024 Disallow the user to switch the language or edit the currently loaded language using the Interface Translation Tool.

2048 Disallow the user to load any language other than English into the interface.

4096 Hide the Tools | Tools Special menu and all of its functionality from the user.

8192 Enforce read-only mode (without using the /readonly switch). So the user cannot create or modify any app data files, and hence cannot save any changes in the user settings.

16384 Prevent dark menus in XYplorer when Windows is in Dark Mode. Useful when you prefer having XYplorer in Light Mode while Windows is in Dark Mode.

32768 Disallow updating (and checking for updates) to a new version from within the app.

65536 Prevent the Preview Now button from being displayed (the button is normally displayed for file types that are excluded from preview via Configuration | Preview | Previewed Formats).

Now you simply add the values of the desired effects. For example, if you want to hide Tools | Configuration... (16), Tools | Open Configuration File... (32) and Help | Online Support | Check for Updates (64), you add 16 + 32 + 64 = 112 and use this value in Admin.ini:

```
[Settings]
Profile=112
```

Key: Lic

You can pass the full path (or relative to application path, all variables and portability supported) to a file where the license data are stored. This is quite useful when installing XYplorer in companies for large numbers of users that are covered by a single Corporate License, or when you have many parallel XYplorer installations on your machine.

Example for the Lic key in "Admin.ini" pointing to a local license file with a name and location of your choice:

```
[Settings]
Lic="appdata\License.ini"
```

Another example, pointing to a remote license file:

```
[Settings]
Lic="\\CentralServer\XYdata\License.ini"
```

The contents of "License.ini" (or whatever you call it) should look like this (fill in your license data for <...>):

```
[Register]
Name=<Your Registration Name>
Code=<Your Site License Key>
```

If "License.ini" is not found, not accessible, or empty, then the procedure silently falls back to reading the license data from "XYplorer.ini".

Keys: Name, Code

You can also place your site license key directly in the file Admin.ini. From here the key will unlock all XYplorer instances in the network.

```
[Settings]
Name=<Your Registration Name>
Code=<Your Site License Key>
```

Special file "Lic.ini"

Instead of using "Admin.ini" with key Lic or keys Name and Code (see above) you can go yet another way to achieve the same: Place a file with the fixed name "Lic.ini" in the path of XYplorer.exe that contains the license data used to unlock the application. The contents of "Lic.ini" should look just as shown in the section here above (fill in your license data for <...>):

```
[Register]
Name=<Your Registration Name>
Code=<Your Site License Key>
```

[Paths] Section

The [Paths] section can be used to overwrite some otherwise hard-coded data paths. The following keys are supported (here with sample values):

```
[Paths]
Catalogs=C:\CommonXYCatalogs\
NewItems=\\North\Share\XY\NewItems\
FileTagDat=\\North\Share\XY\SharedTags.dat
Scripts=\\North\Share\XY\Scripts\
Paper=\\North\Share\XY\Paper\
```

Usage

These settings can be used for example to share Catalog or NewItems resources, or the Tags database (Labels, Tags, Comments, Extra Tags), or scripts across a network by different members of a team. Or, for a single user, to let different configurations (/ini=...) share the same resources or data.

The paths in the [Paths] section support environment variables, for example:

```
FileTagDat=%USERPROFILE%\OneDrive\DB.dat
Scripts=D:\XYusers\%USERNAME%\Scripts
```

The administrator is responsible for a sensible rights management. E.g. shared Catalogs or a shared Tags database should be read-only for the normal user to avoid write conflicts.

The Tags database can also be protected per-user via setting eAPDisallow_Write_Tags (512) in Admin.ini, which will deny the user to modify tags (Labels, Tags, and Comments):

```
[Settings]
Profile=512
```

Default Profile Path

Here is a way to define a default profile path in Admin.ini, eg (final backslash is optional):

```
[Paths]
DefaultProfilePath=C:\XYplorerVirginSetup\
```

Now, when XYplorer starts and does not find a file XYplorer.ini in its app data path, then the complete contents of C:\XYplorerVirginSetup\ are copied to XYplorer's app data path (it is debug-logged). No questions will be asked on collisions. Then, right after defloration, XYplorer starts with these prepared settings.

This function can be very useful in companies. It lets the administrator cleanly define how XYplorer is first launched, and this ability extends to ALL settings out there (even the thumbnail cache could be prefilled).

[Tags] Section

The [Tags] section offers some advanced configuration options for Multi-User Tagging.

```
[Tags]
TagsList=to do,done,forgotten
TagDatOpenTryMsecs=5000
TagDatSafeSave=1
TagFlags=1
TagDatModCheckMsecs=2000
```

TagsList: Limits the tags the user can apply to this comma-separated tags list.

TagDatOpenTryMsecs: Try so many milliseconds to open the tags database for read and write. Waiting for another user to finish a file operation can be necessary in a Multi-User context.

TagDatSafeSave: This key allows the admin to control whether "SafeSave" is used when saving your tags. Meaning of the values:

0 = Never.
1 = When necessary [Default].
2 = Always.

The key defaults to 1 (if Admin.ini exists but the key is missing). The default 1 ("when necessary") means that the database will only be read if it has been modified (by someone else) since this instance last read or wrote it.

TagFlags (bit field):

1 = Tags database headers are read-only.
2 = Tags database is read-only.
4 = Tags auto-refresh only when app is in the foreground.

When bit 1 is set, the user cannot modify the label colors or names, or the extra column definitions, or the storage method (Configuration | Information | Tags | Storage). Also the scripting command extratag () is disabled .

TagDatModCheckMsecs: Check every n milliseconds whether the tags database has been modified since this instance last read or wrote it. If it has been modified, it will be reloaded if the TagDatSafeSave and TagFlags keys allow it.

[AccessControl] Section

Access Control is a powerful corporate feature by which an admin can easily control which locations can be accessed by the user of XYplorer. If this feature is enabled then browsing and file operations are only possible in, from, and to the explicitly allowed paths. Apart from locations you can also limit the kind of allowed file operations, and disallow certain program features.

Access Control is invoked using Admin.ini, a plain text file located in the path of XYplorer.exe. If it is not there yet, simply create it. Create with admin rights so that the normal user cannot modify it. The section name for Access Control related keys in [AccessControl].

Allow only certain locations

The key AllowedDirs can be used to allow operations only in certain locations. In this example, access is allowed only for Desktop, D:\, and E:\Test. Note that allowance includes all subfolders.

```
[AccessControl]
AllowedDirs=Desktop|D:|E:\Test
```

Any number of paths can be concatenated, separated by a | (pipe). If the key AllowedDirs is not empty then the Mini Tree mode is forced on the user.

Note that environment variables and XYplorer native variables are supported. That way you can easily assign account-specific user rights. For example, allow only the "User" special path and drive "D:":

```
[AccessControl]
AllowedDirs=%user%|D:
```

Disallow certain locations

The key DisallowedDirs can be used to disallow operations in certain locations. In this example, the user is allowed to operate anywhere apart from E:\Test and E:\Secret. Note that disallowance includes all subfolders.

```
[AccessControl]
DisallowedDirs=E:\Test|E:\Secret
```

Any number of paths can be concatenated, separated by a | (pipe).

If both keys, AllowedDirs and DisallowedDirs are given then the logic is like this: First AllowedDirs is checked; if the path in question passes then it is checked against DisallowedDirs. For example:

```
[AccessControl]
AllowedDirs=E:|F:
DisallowedDirs=E:\Test|E:\Secret
```

Result: The user is allowed to operate only in drives E: and F:, but not in the folders E:\Test and E:\Secret.

The additional key DisallowedDirsExcept can be used to define exceptions from disallowed paths, i.e. to allow certain (or all) subpaths within a disallowed branch. For example, these entries will allow all subfolders of E:\Test, but not E:\Test itself:

```
DisallowedDirs=E:\Test
DisallowedDirsExcept=E:\Test\*\*
```

Contrary to the values (which are full paths) in DisallowedDirs the values in DisallowedDirsExcept are wildcard patterns. For example, these entries here will allow all subfolders called "a" or "b", in any disallowed branch:

```
DisallowedDirsExcept=*\a\|*\b\
```

Note that environment variables and XYplorer native variables are supported.

Disallow certain file operations

The key DisallowedOperations can be used to disallow certain file related operations. The value is a made by OR-ing (here equivalent to adding) the values of the following bit field:

```
eAACFO_Move = 1
eAACFO_Copy = 2
eAACFO_Delete = 4
eAACFO_Rename = 8
eAACFO_New = 16
eAACFO_Open = 32
eAACFO_Tag = 64
eAACFO_ShellCtxMenu = 128
```

For example, to disallow Move, Delete, and Rename, add 1 + 4 + 8 = 13:

```
[AccessControl]
DisallowedOperations=13
```

Disable certain features

The key DisabledFeatures can be used to disallow certain program features. The value is a made by OR-ing (here equivalent to adding) the values of the following bit field:

```
eMdlScripting = 1
eMdlUDC = 2                  ' user defined commands
```

```
eMdlCKS = 4                    ' custom keyboard shortcuts
eMdlDualPane = 8
eMdlTags = 16
eMdlCatalog = 32
eMdlTabsets = 64
eMdlAccessControl = 128
eMdlAdminSettings = 256
eMdlBackgroundFileOps = 512    ' includes queueing
eMdlCFI = 1024                 ' custom file icons
eMdlFVS = 2048                 ' folder view settings
eMdlGhostFilter = 4096
```

For example, to disable Scripting and Dual Pane (1+8=9):

```
[AccessControl]
DisabledFeatures=9
```

Hide certain items

The key HiddenItems can be used to state items that should be globally hidden from Tree and List just like with the Ghost Filter. Same syntax, the only difference being: You have only one line, so the wildcard patterns are to be concatenated separated by |.

For example, this would hide all items beginning with a dot and all EXE files from the user:

```
HiddenItems=.*|*.exe
```

- Of course, visibility cannot be user-toggled for items that are admin-hidden. They are hidden for good.
- This hiding is stealth: The count is not visible in the status bar.
- The pattern list is not shown in the tooltip of the "Ghost Filter" toolbar button. It's not shown anywhere (unless you look into Admin.ini).

Mark a shared tags database as read-only

The key ReadonlyTagsDB lets you specify a full filename or wildcard pattern to match a tags database that should be read-only for the user. That way the admin can protect one or more databases from the user while allowing him to write to others. Some examples:

```
[AccessControl]
ReadonlyTagsDB=\\North\Share\XY\SharedTags.dat       ;exclude this particular database
ReadonlyTagsDB=*\SharedTags.dat                      ;exclude all databases called
SharedTags.dat
ReadonlyTagsDB=*                                     ;exclude all databases
ReadonlyTagsDB=                                      ;exclude none (or just drop the key)
```

Fixed Mini Tree

There is a way to completely fix the last saved Mini Tree to its current state:

```
[AccessControl]
FixedMiniTree=1
```

These are the effects of this setting:

- The tree will be the last saved Mini Tree, and cannot be turned to a Maxi Tree.
- It is locked and the expansion state is locked, both cannot be unlocked.
- You cannot drag anything from the tree.
- You cannot rename anything in the tree.
- View | Mini Tree | - submenu - is gone.
- Tools | List Management | Mini Tree... is gone.
- Tree changing commands like "Hide Folder from Mini Tree" or "Flatten Folder" are not available anymore.
- The "Mini Tree" toolbar button is gone.
- SC button "minitree" does not work anymore.
- SC loadtree does not work anymore.

The Fixed Mini Tree is only useful for picking a location or dropping items in a location.

Managed Tree

If bits 1 and 2 of FixedMiniTree are set (`FixedMiniTree=3`), the "Fixed Mini Tree" turns into a so-called "Managed Tree" which is a Fixed Mini Tree with the following additional properties:

- It only shows the folders allowed by AllowedDirs, DisallowedDirs, and DisallowedDirsExcept.
- Within the AllowedDirs, you can freely expand and collapse nodes, add and remove folders, rename folders, drag and drop stuff. These areas are flexible.
- Above the AllowedDirs (incl. the AllowedDirs themselves), nothing is allowed. These areas are fixed.

Summary Example for Access Control

Here's an example for how to set up XYplorer to limit activity to Desktop and \\ServerB\Work, disallow Move, Delete, and Rename even in this allowed locations, and generally disable Scripting and User-Defined Commands (UDC). Create Admin.ini in the path of XYplorer.exe and paste this:

```
[AccessControl]
AllowedDirs=Desktop|\\ServerB\Work
DisallowedOperations=13
```

```
DisabledFeatures=3
```

Access Control Interface

When Access Control (AC) is active the title bar caption displays "[AC]" at its right end.

Also the status bar shows a button with the AC icon when AC is active. Click it to get some information about how you are limited by AC.

The menu popped by the AC status bar button has a toggle item "Use Status Bar for Messages". Check it if you need it. The setting is retained across sessions via the normal INI file, so this part of AC is customizable by the user.

Remarks

Access Control partly overlaps with the Profile key in the Settings section (see here above). A difference is that the latter has no interface or messages. It simply and silently removes features.

Redirecting Admin.ini

There is a way to read the admin settings from a file other than Admin.ini. Add this section to the beginning of Admin.ini (the path is an example):

```
[Redirect]
Path=\\ANDROMEDA_CORP\XYplorer\AdminCentral.ini
```

If this key is found, and if the file exists and can be read, the rest of this Admin.ini file is ignored and all admin settings are read from the new file. This allows managing a multi-user setup from one central location.

5.2 Command Line Switches

Command Line Switches

This is about the Windows command line, aka command prompt, as used e.g. in the DOS-Box or in the properties of shortcuts (LNK files).

The overall command line format is something like this (more switches exist, as shown below):

```
XYplorer.exe[ /win=min|max|normal][ /ini=myini][ startuppath]
```

The order of the switches is not relevant.

/exp
/feed
/flg
/fresh
/hwnd
/ini
/new
/path
/readonly
/restore
/script
/select
/title
/user
/win
/win (extended syntax)

/exp

Add this switch to ensure the startup folder is expanded in the tree.

```
XYplorer.exe /exp
```

```
XYplorer.exe /path=C:\ /exp
```

/feed

Lets you feed a script into an existing instance of XYplorer without changing the location or tabs of that

instance. A practical shorthand of using switches /script and /flg=2. The following lines are functionally identical:

```
XYplorer.exe /script="::msg 'Hi!';" /flg=2
XYplorer.exe /feed="::msg 'Hi!';"
```

If you use both /script and /feed then /feed will overwrite /script.

Tip: You can "pipe" a value instead of quoting it. This can be useful when unpredictable quotes (that might be returned from variables) make predictable parsing impossible, or simply to make a string more readable. For example, this is the same Command Line Switch, first quoted then piped:

```
/feed="::text '"R:\a b c"';"      (this would not work; parser cannot handle the ambiguous quotes)
/feed=|::text '"R:\a b c"';|      (this does work)
```

Of course, you must be sure that no pipes can come up inside the value.

When called from a dos prompt the pipes have to be escaped, e.g.:

```
XYplorer.exe /feed=^|::text '"R:\a b c"';^|
```

/flg

/flg=2: You can feed a script into an existing instance of XYplorer without changing the location or tabs of that instance. Simply pass the switch /flg=2 in the command line, for example:

```
XYplorer.exe /script="::msg 'Hi!';" /flg=2
```

/flg=8: No Custom Column scripts are executed. That way you can start XYplorer even when you messed up your scripts.

/fresh

Run the application without loading any settings apart from the license data. This can be useful for isolating any issues under factory defaults conditions and thus can help the analysis.

```
XYplorer.exe /fresh
```

Note that a new instance is opened regardless of any settings or existing instances (like with switch /new).

Tip 1: Open a fresh instance via Windows Run dialog:

```
"C:\Program Files\XYplorer\XYplorer.exe" /fresh
```

Tip 2: Or open a fresh instance using this mini script right from XYplorer's address bar:

```
fresh;
```

/hwnd

If feeding a script into XYplorer you can pass the window handle of the target XYplorer instance with this

switch if you know it. If you want to feed the script into this XYplorer, you can simply use the variable <hwnd>, for example:

```
run "<xy> /script="":msg 'Hi!';"" /flg=2 /hwnd=<hwnd>", , 0, 0;
```

/ini

You can specify which INI file to load by starting the application as follows:

```
/ini=myini    -> load [XYplorer application data path\] myini.ini
```

Notes:

- myini can be just a base name (myini) or include the extension .ini (myini.ini). If you pass no path with it then the file is looked for in the Application Data Path.
- myini must be quoted if the name contains any blanks.
- If you use the /ini switch then you will always get a new instance even if Configuration | General | Startup & Exit | Allow multiple instances is unchecked.

Date variables, e.g. <date yyyy>, are supported in the path. See examples below.

You can pass a non-existing path with the /ini switch, e.g.:

```
XYplorer.exe /ini=%temp%\XY\ /fresh
```

Only on saving any settings, the path will be created. Usage: By passing a non-existing path you can ensure that no current settings are overwritten, e.g. like this:

```
XYplorer.exe /ini="%temp%\XY_<date yyyymmddhhnnss>\"
```

Or from the Address Bar:

```
::run quote("<xy>") . ' /ini="%temp%\XY_<date yyyymmddhhnnss>\"';
```

Tip: The currently used INI file and Application Data Path is displayed in the title bar of the Configuration window

/new

Open a new instance regardless of any settings that would otherwise force any existing instance to be re-used.

```
XYplorer.exe /new
```

/path

Lets you control the startup path, optionally quoted (quotes are needed if the path contains spaces).

```
XYplorer.exe /path=C:\
```

```
XYplorer.exe /path="C:\New Folder"
```

Note that you can also pass the startup path as an unnamed switch, see below. However, the /path switch will overwrite any unnamed switch.

/readonly

Run the application in read-only mode, i.e. the settings are read from disk but never written to disk. So your configuration on disk will be untouched by this instance.

```
XYplorer.exe /readonly
```

Notes:

- [READONLY] in shown in the window title bar when in read-only mode.
- You get a Status Bar message when you actively attempt to save settings in read-only mode.

/restore

Can be used to specify an alternative temporary application data path from which the configuration is read on startup. After startup, this path isn't used anymore and all writing goes to the regular application data path (whether it's the default or one set with the /ini switch). The typical use of /restore is to restore the configuration from a backup path. Example:

```
/restore="E:\XYbackups\20240304\appdata\"
```

Another use could be to always start the application with a certain configuration, similar to a read-only mode, but here you can save data to the regular appdata path, so there is an option to use the saved data later. Example:

```
/restore="E:\XYconfigs\loveit\"
```

Note that you can combine /ini and /restore in one command line:

```
/ini="E:\XY\snapshot\<date yyyymmdd>\" /restore="E:\XYconfigs\loveit\"
```

/script

You can run a script at startup directly from the command line. The switch is /script=[script resource], where [script resource] can be:

1) The path to a script file (commonly called *.xys). The file spec is resolved as in the script command load, so you can skip the XYS extension. You can also skip the path if the XYS file is located in the default scripts folder. For example

```
XYplorer.exe /script=test
```

would run the file <xyscripts>\test.xys at startup. If the path contains spaces it must be quoted:

```
XYplorer.exe /script="C:\Zen\test one.xys"
```

2) A script. It must be preceded by ::, and it must not contain double quotes or other characters that might lead to problems when parsing the command line. For example

```
XYplorer.exe /script="::msg 'Welcome to XY!';"
```

Note the double quotes surrounding the whole argument which are needed because of the spaces in the script line.

If your script needs double quotes you should use the quote() function or runq with single quotes:

```
XYplorer.exe /script="::run quote('E:\Test\Has Space.txt');"
XYplorer.exe /script="::runq 'E:\Test\Has Space.txt';"
```

Tip: See the tip about "piping" as an alternative to quoting.

/select

XYplorer will go to the parent path in the folder tree and select the item in the file list. For example, this will go to C:\ and select Windows in the list:

```
XYplorer.exe /select=C:\Windows
```

/title

You can set a session title via the command line switch /title=[session title]. If the title contains any spaces it has to be quoted. Example:

```
/title="Mars Project"
```

A session title is useful when you need to distinguish several parallel sessions. See Title Bar Template in Configuration | Templates.

/user

Lets you pass a |-separated, optionally quoted, list of values to the application. The individual values can later be retrieved via scripting command get (with the named argument "CmdLineUser"). For example:

```
App Call:
  XYplorer.exe /user=a|b|c
  XYplorer.exe /user="a|b|c"    (with optional quoting)
Scripting:
  echo get("CmdLineUser"); //a|b|c
  echo get("CmdLineUser", 1); //a
  echo get("CmdLineUser", 2); //b
  echo get("CmdLineUser", 3); //c
```

/win

You can control the window state at startup:

```
/win=min      -> start minimized
/win=max      -> start maximized
/win=normal   -> start normal
/win=tray     -> start minimized to tray
```

/win (extended syntax)

The switch /win can also be used to specify the main window position and dimension at startup, overwriting the values in the INI file. Each value is optional and if missing the value found in the INI file is used. General format:

```
/win=[min|max|normal|tray],[x],[y],[width],[height]
```

Examples:

```
/win=normal,40,20,1200,800
/win=,20,20,1200            -> window state and height from INI
/win=,,,1000,800            -> window state and position from INI
/win=min                    -> start minimized (other values from INI)
/win=max,,,,400             -> start maximized, restore to height 400
/win=tray,40,20,1200,400    -> start minimized to tray
```

Setting the Application Data Path (ADP)

You can specify an alternate Application Data Path (ADP) directly from the command line. Just use the /ini switch and pass a full path.

For example:

```
XYplorer.exe /ini=C:\XYdata\XY.ini
```

Sets the ADP to C:\XYdata\, and loads XY.ini as main configuration file from there, as well as all other data files like cks.dat, udc.dat etc.

Note that portable syntax is supported:

```
XYplorer.exe /ini=?:\XYdata\XY.ini
```

? will be set to the drive XY is running from.

```
XYplorer.exe /ini=XYdata\XY.ini
```

Sets the ADP to [XY app path]\XYdata\

You can also pass the path alone without stating the INI file. Simply end the argument with a backslash:

```
XYplorer.exe /ini=C:\XYdata\
```

Sets the ADP to C:\XYdata\, and loads XYplorer.ini (the default INI file).

This new support for passing the ADP via command lines means that you now can run completely independent configurations simply using Windows shortcuts.

Tip: The currently used INI file and Application Data Path is displayed in the title bar of the Configuration window.

Setting the Startup Path

The INI contains all settings including the current startup path. You can however overwrite this value by stating the startup path in an (unnamed) switch, e.g. as the right-most argument, optionally quoted (quotes are needed if the path contains spaces). For example:

```
XYplorer.exe /ini=myini "C:\Windows"
```

You may also pass a startup path relative to the application path, e.g. "..\..\". Passing just a dot (".") will startup at application path.

Notes:

- Special paths, e.g. "Desktop\Blah\", are supported.
- Paths to files are supported. XYplorer will go to the parent path in the folder tree and select the file in the file list.
- Paper Folders are supported, e.g. "paper:Photos".
- You can inject a Dual Startup Path (one path for each pane) into the running or a new instance (depending on your settings) by passing two paths separated by a double-pipe (||). Example:
  `XYplorer.exe "C:\||D:\"`
  Note that Dual Startup Path forces startup in Dual Pane mode.
- See also the /path switch above for another way to pass a startup path.

5.3 Custom File Associations

Introduction

XYplorer can maintain private Custom File Associations independent of the global ones defined in the Windows registry. These associations are stored in the local INI file, which allows you to travel with your own file associations. So they could as well be called "Portable File Associations".

There are two to enable/disable the feature:

- In the Tools | List Management | Custom File Associations... dialog you find a toggle button in the bottom toolbar.
- Right-click the "Open With" button on the main window toolbar and click "Enable Custom File Associations".

How to define them

To define a custom file association simply open "Custom File Associations..." in menu Tools/List Management and add it to the list. The general syntax is: pattern>application, where pattern and application can have various formats. For example "extension>application (full path)":

```
png>C:\Program Files (x86)\Viewer\Viewer.exe
png>?:\Program Files (x86)\Viewer\Viewer.exe
```

The 2nd example is particularly interesting if you use XYplorer on portable drives: the little "?"-trick makes you independent of the local drive mappings! Double-clicking (or <enter>ing) a file in XYplorer will cause it to open in an application stored on your USB stick, independent (a) of the file associations maintained by the local copy of Windows, and (b) of the local drive mappings. What more portability do you want?

You can as well state several extensions in a row, separated by ";", for example:

```
png;jpg>C:\Program Files (x86)\Viewer\Viewer.exe
```

You may as well use generic file types, e.g. {:Photo}.

```
{:Photo}>C:\Program Files (x86)\Adobe\Photoshop 7.0\Photoshop.exe
"Text|Notepad" {:Text};csv>Notepad
```

You can as well state a path *relative* to this XYplorer's path, for example:

```
png;jpg>..\Viewer\Viewer.exe
png;jpg>..\..\Viewer\Viewer.exe
```

If this XYplorer is at C:\Program Files\XYplorer\ then the examples will be resolved to C:\Program Files\Viewer\Viewer.exe or C:\Viewer\Viewer.exe.

You may as well state the name of the executable (with or without ".exe") of registered applications. Both examples are functionally identical:

```
pdf>PDFXCview
```

```
pdf>PDFXCview.exe
```

See advanced possibilities here below under "The Patterns".

What they do for you

Custom File Associations decide which application is used to open your files on Dbl-Click, Enter and Ctrl+Enter (at least if "Open Items by First 'Open with...' Match" is ticked; see Configuration here below). The CFA definitions are processed from top to bottom. Hence their order is important because the first matching pattern will make it and select the opening application.

Note that the above is only true for Dbl-Click, Enter and Ctrl+Enter on a file, not on a folder. Folders are always simply browsed on those actions.

If none of the patterns matches the file to be opened then the global registry-based file association is used to open it. You still have direct access to the registry-based file associations via

(a) the files' context menu's "open" command,

(b) the Portable Openwith Menu (POM), or

(c) Custom Keyboard Shortcuts | Miscellaneous | File Operations | Open Selected Item(s) with OS Default [Ctrl+Shift+Enter].

The checkboxes in the list allow you to easily (de)activate particular file associations without the need to delete and retype them. As always in XYplorer's checkbox lists, a right-click menu offers (Un)Select All.

How to use them

Mouse users: Their recommended way would be to add the Open With button to you toolbar (it is by factory default), and pop the Open With menu from there.

Keyboard users: Their preferred way would be to press Ctrl+Alt+Enter which will pop the Open With menu for the currently selected file.

The Patterns

A file's extension is not the only possible pattern in XY's Custom File Associations. You can state whole file names or even full, relative, or portable paths; patterns containing wildcards are allowed.

Rules

If the pattern contains the character "\" then

  it's treated against the full path (relative/portable syntax supported)

Elseif the pattern contains wildcards (*, ?, #) or a dot (.) then

  it's treated against the file name

Else

  it's treated against the extension

Special patterns

To match all files use a single `*` (asterisk).

To match all folders use a single `\` (backslash).

To match all files and folders use `\;*` (backslash- semicolon- asterisk).

To match all files without extension use `*.` (asterisk-dot).

Variables

XYplorer native variables and environment variables are supported in the pattern list. E.g., the following pattern will only match TXT files in the Paper Folder folder:

```
<xypaper>\*.txt>::echo <focitem>;
```

Examples

```
?:\*.xys>::load <curitem>
```

Loads selected XYS file, but only if it's on app drive. This is very nice, since in a "portable situation" (e. g. you stick your USB-drive into another computer), you can have files on your drive opened by your apps, all other files opened by the system's default app, or by some other app you define; for example:

```
?:\*.dat>?:\app1.exe
```

```
*.dat>?:\app2.exe
```

The second definition will match all files that have not been matched by the first definition (= all DAT files that are not on app drive).

```
a*;b*>UEdit32
```

Opens a file with UltraEdit if its name starts with "a" or "b".

```
{:Text}>EmEditor
```

Opens all text files with EmEditor.

You can match all subfolders of a certain parent folder by appending *\ to the parent folder, for example:

```
E:\Test\*\>echo "It's a subfolder of E:\Test\";
```

Captions

A caption can be prefixed as quoted string. This caption is used in the Portable Openwith Menu (POM).

```
"No extension" *.>UEdit32
```

Open all files (not folders) without extension with UltraEdit. The item shows the caption "No extension" in the POM.

Icons

You can define the icon that's shown for each item in the Portable Openwith Menu (POM). Simply append any file spec to the Caption (see above), separated by a | (pipe). It can be an icon file or any other file or folder with associated system icon, toolbar icons (prefixed with :), and icons extracted from icon resources (exe; dll; cpl; ocx; scr; icl; bpl; wlx; wfx; wcx; wdx; acm). XYplorer native variables, environment variables, and portable paths are supported.

```
"No extension|E:\Test\Kiss.ico" *.>UEdit32
"PS|<xy>" jpg;png;gif>Photoshop
"Say hi!|E:\Test\Kiss.ico" jpg>::msg "hi";
"Say hi!|C:\WINDOWS\notepad.exe" jpg>::msg "hi";
|"Append date|E:\Test\Kiss.ico" \;*>::rename b, '*-<datem yyyymmdd>'
"No extension|:dice" *.>Notepad
"No extension|%winsysdir%\shell32.dll /160" *.>Notepad
```

The icon path defaults to <xyicons> (= [app data path]\Icons), so you don't need to state a path for icons located in that folder:

```
|"Append modified date|Kiss.ico" \;*>::rename , '*-<datem yyyymmdd>'
```

Menu Separators

You can add menu separators to your definitions, either global ones or context-specific ones. Lets you structure your open with popup menu. To do this use a hyphen (-):

```
-
txt>-
txt;ini>-
{:Image}>-
```

A logic has been implemented that will automatically reduce consecutive separators to one, and remove initial or final separators.

Submenus

The menu supports one level of nesting, i.e. you can optionally organize your items into submenus. It's done like elsewhere in XYplorer (e.g. in the Hamburger), just indent the items that should be in a submenu. The item right in front of the indented ones is the submenu title.

Example:

```
Text Editors
  txt>Notepad
  txt;rtf>Wordpad
  -
  txt;ini;dat;idl;xys>EmEditor
Scripts
  |"Copy file size in bytes" *>::e|copytext <cursize>
  |"Append Modified Date to Filename" \;*>::e|rename , '*-<datem yyyymmdd>'
  |"Append Image Dimensions to Filename" {:Image}>::e|rename , '*-<prop #image.dimensions>'
```

In the example above, the indent is 2 spaces. You can use a different number as long as you are consistent.

The Applications

There are two sorts of "applications" allowed in Custom File Associations: Executables and Scripts.

Executables

Using short forms for applications: If the EXE to run is registered (known by the registry) it is not necessary to state the full path. The base name of the EXE file is enough. Both ways will work the same:

```
png;jpg>C:\Program Files (x86)\Viewer\Viewer.exe
png;jpg>Viewer
```

Also environment variables are allowed:

```
txt;htm>%windir%\notepad.exe
```

Using command line parameters: Just as in User-Defined Commands of category "Open With", Custom File Associations accept command line parameters.

Scripts

You can also state a one-line script (see Scripting) as "application" in the definition of a Custom File Association.

Examples:

```
xys>::load <curitem>
```

Now dbl-clicking a *.xys file will actually load that file, and pop up the menu defined by it, or run the script contained in it if it's the only one.

```
readme.txt>::status "Who cares?!"
```

Any attempt to open a readme.txt file will just result in a status bar message "Who cares?!".

The special variable <pfaitem>

In CFA scripts the temporary variable <pfaitem> is supported. It resolves to the item (full path) the opening application or script in a CFA is associated with. Only in a script called via CFA <pfaitem> can be used. After the CFA is processed the variable is reset to nothing.

The special variable

You can use # to stand for the Windows associated application. Especially when using generic file types this little trick can be useful to exclude certain extensions from the broader definition. For example:

```
ini>#
{:Text}>EmEditor
```

URLs

You can also use CFA to define which browser should open a URL, for example:

```
https://*;http://*>C:\Program Files (x86)\Mozilla Firefox\firefox.exe
```

Or probably also (if Firefox is registered):

```
https://*;http://*>Firefox
```

Configuration

In the right-click menu of the Open With toolbar button you find some advanced options,

Open Items by First 'Open with...' Match: Tick it to use the first matching Custom File Association to open an item by Dbl-click or Enter. The factory default is ON. If set to OFF then the Windows associated application is used for opening an dbl-clicked item irrespective of any matching Custom File Associations .

Resolve Shortcuts before Matching: Tick it to resolve LNK-files before matching them to your Custom File Associations.

Show Full Paths in 'Open With...' Menu: Tick to show the full paths of the associated applications in the Open With... menu.

5.4 Portable Openwith Menu

Introduction

You probably know the situation: You need to open a file with a specific application that happens to be *not* the one associated to that file by Windows. What now? You may be lucky to find the application listed in the "Open With..." section of the file's Shell context menu -- but even then this is a clumsy way to work (at least one right-click, then expand a submenu, then a left-click), it lacks configurability (what if you need command line parameters?), and it's not portable (away from home, the Shell's "Open With..." is defined by the host system, not by your needs). All these issues are elegantly tackled by the Portable Openwith Menu (POM).

The Portable Openwith Menu (POM) is opened via menu File | Open With... (Ctrl+Alt+Enter), or via the "Open With" toolbar button. It is closely interrelated with your Custom File Associations (CFA) and will not show anything interesting before you have defined a couple of CFAs. Once done, a single click or keyboard shortcut will popup a context specific menu presenting an array of applications to open the currently selected file(s) with. The specific contents of this menu depend (A) on the currently selected files, and (B) on your Custom File Associations (CFA) setup, hence it is portable and utterly easy to manage.

With POM it makes sense to have more than one Custom File Association match the same pattern. As before, the standard "Open" command (Dbl- click/Enter) will only make use of the first matching CFA. However, the new "Open With" popup menu will list all matches, in the order given in the CFA customization dialog.

The POM has up to 3 sections (the first is only present, of course, when matching CFAs are found):

[Custom File Association(s)]

...

[OS File Association]

Customize File Associations...

Basic Usage

The following example assumes you have Photoshop and ACDSee32 installed.

Set up these CFAs:

```
jpg;png;gif>ACDSee32
gif>C:\Program Files (x86)\Adobe\Photoshop 7.0\ImageReady.exe
jpg;png;gif;tif>Photoshop
```

Now, when open the POM on a GIF file, it will offer the following apps to open it with:

```
ACDSee 32
ImageReady
Adobe Photoshop
```

When open the POM on a PNG file, it will offer the following apps to open it with:

```
ACDSee 32
Adobe Photoshop
```

And when open the POM on a TIF file, it will offer the following apps to open it with:

```
Adobe Photoshop
```

Advanced Usage

For advanced users the POM, in close collaboration with CFA, has a lot to offer:

(1) Rather than just extensions, you can define any wildcarded filename patterns, including paths and individual filenames. Thus you can have specific context menus for all "Readme.txt" files, or for all items in C:\Windows\, or for all items containing German umlauts (pattern: *[äöü]*).

(2) Rather than just files, you can also associate folders with applications. Tip: A single backslash (\) will match all folders.

(3) Rather than just executables, you can specify any kind and number of command line parameters in your Custom File Associations, which then will be used when triggering an Open operation from the POM.
The syntax is identical to all the other places where command line parameters are supported: Catalog (Open Selected Item(s) With), User-Defined Commands Open With, scripting "openwith".
It's basically one simple rule: The application must be quoted when command line parameters are used. For example:

```
jpg;png;gif;tif>"ACDSee32" /v
jpg;png;gif;tif>"C:\Program Files (x86)\ACDSee32\ACDSee32.exe" /v
```

By the way, quoting is also okay (but not necessary) when you have no parameters.

(4) Rather than just applications, you can also associate scripts with filename patterns, and thus directly run a specific script from the POM.

(5) Within the Custom File Associations dialog, you can define friendly captions for the POM items that make it easier for you to spot the right item in a POM. The syntax is identical to captions in

Favorites. Simply prefix the caption in quotes (optionally followed by any number of blanks). For example:

```
|"Edit with Notepad" txt;ini;xys>C:\WINDOWS\notepad.exe
|"Edit with UltraEdit" *>UEdit32
```

If a caption is defined for a CFA, then the caption will be shown in the menu instead of the app's friendly name (which is drawn from the registry if the app is registered).
You can define an accelerator by prefixing the "&" (ampersand) character to the desired character.

(6) The captions may contain XYplorer variables. They are resolved in the moment the POM is generated. For example:

```
|"Search IMDB for '<curbase>'" *>::Open("https://www.imdb.com/find?s=tt&q=<curbase>");
```

Will show the caption "Search IMDB for 'Mad Max'" when the current file is "Mad Max.jpg".

(7) In the POM, hold Ctrl when clicking an item to go to the application instead of opening it.

(8) Tip: Ctrl+Alt+Enter is identical to AltGr+Enter, so you can pop up the POM easily using the right hand only.

(9) You can define associations that should only be listed in the POM, but not be triggered on "open". It's done by simply prefixing the | (pipe) character to the CFA. The following example allows you to have this CFA enabled and listed in the POM:

```
|exe;dll>::copytext <curname> <curver>
```

while double-clicking an EXE file will still run the EXE, and not the CFA. In the CFA dialog, prefixed (= POM-only) items are shown in color "Marked Text 2" (see Configuration | Colors).

Examples

The following CFA will add a command to the POM that will open the selected folder(s) in ACDSee32:

```
\>ACDSee32
```

Open *.txt, *.ini, and *.xys files with Notepad (Editor), but only show this command in POM, do not trigger it on "open":

```
|"Edit with Notepad" txt;ini;xys>C:\WINDOWS\notepad.exe
```

Open any file (but not a folder) in UltraEdit:

```
"Edit with UltraEdit" *>UEdit32
```

Open any folder in UltraEdit:

```
"Edit with UltraEdit" \>UEdit32
```

Open any file or folder in UltraEdit:

```
"Edit with UltraEdit" *;\>UEdit32
```

Open PNG files in ACDSee, using comand line parameter /v

```
"ACDSee in View Mode" png>"ACDSee32" /v
```

5.5 User-Defined Commands

See User menu.

5.6 Hamburger

Hamburger

The Hamburger is a user-defined popup menu that can be used in various locations or contexts:

- The menu buttons of the Breadcrumb Bars.
- The click events of Custom User Buttons in the toolbar.
- The scripting commands popupmenu and popupnested.

You can pack all sorts of ingredients into a single Hamburger menu:

- Menu commands
- Toolbar Buttons
- Paths
- URLs
- Scripts

Menu Items Syntax

Let's go through it type by type:

Menu commands

Menu commands ("menu" here refers to the main menu of the XYplorer window) are referred to by their unique Command ID (CID), prefixed by #. You can display the Command IDs right in the menu by ticking this toggle: menu Help | Command IDs on Menu.

General form: `#CID[;caption;icon;state]`

For example:

```
#101
```

By default, the above definition would create this menu item: Item Path/Name(s) Ctrl+P

Clicking the item triggers the command just as if you clicked the original main menu item: File | To Clipboard | Item Path/Name(s)

Optionally you can also state a custom caption and a custom icon, separated by semi-colons, for example:

```
#101;Copy Stuff;copy.ico
```

State: Here and everywhere below, the optional `state` field is a bit field that lets you modify the state

of each menu item using these bits:

1 = default (bold)

2 = checked

4 = disabled

This e.g. would make the item bold and checked (1+2=3):

```
#101;Copy Stuff;copy.ico;3
```

Toolbar Buttons

Toolbar buttons are referred to by their unique button keys, prefixed by :. You can see the button keys by holding CTRL while you hover the button in the Toolbar. Alternatively hold CTRL while you click menu Tools | Customize Toolbar... (Ctrl+Shift+F9); the keys will appear next to each button caption.

General form: `:key[;caption;icon;state]`

For example:

```
:dice
```

By default, the above definition would create this menu item: Random Order Ctrl+Alt+R

Clicking the item triggers the same action as clicking the toolbar button itself.

Optionally you can also state a custom caption and a custom icon, separated by semi-colons, for example:

```
:dice;Random Sort;E:\pics\random.png
```

Toolbar buttons with a pressed state show "[On]" in the menu if they are currently pressed.

Paths

General form: `path[;caption;icon;state]`

Just paste the path as is. For example:

```
C:\Program Files\XYplorer
```

By default, the above definition would create this menu item: XYplorer

Clicking the item will:

- open a folder
- run an executable
- open any other file type with the associated executable

XYplorer native variables and environment variables are supported, for example:

`<xydata>` or `%user%`

Optionally you can also state a custom caption and a custom icon, separated by semi-colons, for example:

```
C:\;Home Drive;MiCasita.jpg
```

Also Paper Folders and Virtual Folders can be used as paths:

```
vi:<get pick *.jpg>
```

```
paper:Pictures
```

URLs

General form: `url[;caption;icon;state]`

Just paste the URL as is. For example:

```
https://www.xyplorer.com
```

By default, the above definition would create this menu item: https://www.xyplorer.com

Clicking the item will open the URL in the associated browser.

Optionally you can also state a custom caption and a custom icon, separated by semi-colons, for example:

```
https://www.xyplorer.com;XY;fullofbugs.ico
```

Scripts

General forms: `::script;` or `::caption;script;` or `::caption;/icon;script;`.

Scripts in Hamburgers have to be prefixed by ::. For example:

```
::echo "hi!";
```

By default, the above definition would create this menu item: echo "hi!";

Clicking the item will run the script (see Scripting for details of the syntax).

Optionally you can also state a custom caption, separated by a semi-colon, in front of the script, for example:

```
::Hi;echo "hi!";
```

Optionally you can also add a custom icon, after the custom caption and prefixed by /, for example:

```
::Hi;/CrazyNerd.ico;echo "hi!";
```

Alternative Syntax for Script Items

A alternative item syntax is supported, where you can define the item separator per item, and have script items with a custom caption, icon, and mouse-over-status.

Defining the separator: If the second character of the item definition is ">" then the first character is the separator.

The icon and status fields are optional.

These are the three general forms:

```
Definition                      Default values for non-defined fields
-------------------------------------------------------------------------
|>caption|::script              icon = script icon; status = the script
|>caption|::script|icon         status = the script
|>caption|::script|icon|status
```

Or using e.g. / as separator:

```
/>caption/::script
/>caption/::script/icon
/>caption/::script/icon/status
```

Example definitions as used in Hamburger:

```
|>Green|::tag 4
|>Blue|::tag 5|:labels
|>Purple|::tag 6|:labels|make it purple
```

Example for popupnested (here you can pass the separator, no need to prefix it):

```
$menu = <<<MENU
  Green|::tag 4|kiss.ico
  Blue|::tag 5|heart.ico
  MENU;
  $command = popupnested($menu,,,,,,,|);
```

Menu Syntax

Just define one menu item per line and you've defined the menu. You can freely mix all types of menu items. Optionally throw in some "-" as menu separators. For example:

```
#101
:dice
-
C:\Program Files\XYplorer
https://www.xyplorer.com
-
::Hi;echo "hi!";
```

When used in the scripting commands popupmenu and popupnested you can pack everything into one line, separated by a separator of your choice.

Nesting

The Hamburger supports nested submenus in totally intuitive way. Simply indent what should be in a submenu. Here, for example, is a Hamburger with three levels:

```
Mixed Stuff
  #101;
   :dice
   -
   <xypath>
   <xydata>
URLs
   https://www.xyplorer.com
   https://www.xyplorer.com/purchase.php
```

```
-
Buttons
  :copy
  :dice
  //:dice
-
Paths
  Desktop
  C:\
  D:\
  -
  Apps
    <xy>
    C:\Program Files\Notepad++\notepad++.exe
```

Comments

You can append comments to each menu item, separated by `  //` (space-slash-slash). For example:

```
<xy> //XYplorer.exe full path
```

You can as well hide items from the menu by placing `//` in front of the item definition:

```
//<xy>
```

Explicit Hamburger Marker

It may happen in rare cases that the parser doesn't recognize a definition as Hamburger. In that case you can explicitly mark a definition as Hamburger by putting `//H` as first line.

BTW, you can also explicitly mark a definition as non-Hamburger (but multi-script or multi-line-script) by putting `//S` as first line.

Further Notes

There is a status bar message about what will happen on click when hovering over Hamburger items.

5.7 Scripting

Index

- Arrays
- Binary Numbers
- Boolean Constants
- Boolean Operators
- Commands and Functions
- Comments
- Comparisons
- Compound Assignment Operators
- Control Structures
  - If/ElseIf/Else Blocks
  - While Loops
  - For Loops
  - Foreach Loops with Lists
  - Foreach Loops with Arrays
  - Switch Statements
- Dereference Operator
- General Command Syntax
- Heredoc and Nowdoc Syntax
- Hex Numbers
- Include Statement
- Include_once Statement
- Include Automatically
- Math
- Multi-line Scripts and Multi-Scripts
- Nested Expressions
- Operator Precedence
- Quick Scripting
- Remote Control
- Script Files
- Script Files for the Advanced
- Scripting and User-Defined Commands
- Scripting by Numbers
- Step Mode: Stepping Through Scripts
- Ternary Conditionals
- User-Defined Functions
- Using Quotes in Scripting
- Variables

Variables Scope and Lifetime: Local, Global, and Permanent Variables

Introduction

XYplorer Scripting, introduced with version 7.0, can truly be seen as the ultimate in file management efficiency. Roll your own custom commands, combine them to scripts, wrap them in an XYplorer Script file (XYS), or a User-Defined Command, and trigger them by just a click or a keystroke. You may share scripts with colleagues, or download them from the internet. Just drop a script file into your app folder and fresh plug-in commands are at your finger tips.

Without doubt, scripting is an advanced feature that only pays off if you take the time to dive into it and explore the ways and possibilities. However, you will eventually find out that it ain't rocket science at all.

More information, downloads, and a growing pool of examples is available here:

https://www.xyplorer.com/

Warming Up

To get you started let's try something easy:

(1) Select menu Scripting | Run Script...

(2) Paste `msg "Hello world!"` into the edit box.

(3) Click OK.

You should see a "Hello world!" message box now. Well done, you just wrote your first XYplorer script!

Okay, now for something a little bit more interesting:

(1) Try `msg %temp%`. You should see a message box that displays your TEMP path. %temp% is a standard Windows environment variable.

(2) Try `msg "XYplorer.exe runs from <xypath>"`. <xypath> is a native XYplorer variable that resolves to XYplorer's application path.

(3) Try `msg "Press OK to copy the time!"; copytext "<date hh:nn:ss>"`. When the message box is OKed, the second command copytext is immediately executed and the current time is copied to the clipboard.

(4) Try `set $a, "<curpath>"; msg $a`. You should see a message box displaying the current path. First the variable $a is set to the current path, then it is used in the msg command.

(5) Try `$a = "Year " . "<date yyyy>"; msg $a`. You should see a message box displaying "Year 2009" (or whatever the year might be when you try this). First the variable $a is set to two strings, one literal and one variable, concatenated by a dot, then it is used in the msg command.

General Command Syntax

(1) There are commands consisting of a function name like msg or copytext, and arguments that can contain literals like "Hello!", and variables like %temp% (environment), <curpath> (XYplorer), or $a (user-defined).

(2) An argument can have more than one part. The parts are concatenated by dots (.).

(3) A command can have more than one argument. The arguments are separated by commas (,).

(4) A script can have more than one command. The commands are separated by semi-colons (;).

(5) Arguments are identified either by position, or by a prefixed number denoting their position (see Numbered Arguments below).

Examples

This little script will rename all currently selected items by appending the current date:

```
rename b, "*-<datem yyyymmdd>"
```

This script goes to the Desktop folder (ensuring that the listing is unfiltered), sorts the list by date modified (descending), selects the first item, and moves the focus to the list:

```
goto "Desktop|"; sortby m, d; sel 1; focus "L";
```

Numbered Arguments

Instead of counting commas you now can prefix the position of the argument to the argument itself. The separator is ":=", the first position is 0 (zero).

For example, instead of this:

```
text popupmenu("A,B,C", , , , , , ",");
```

You can do this:

```
text popupmenu("A,B,C", 6:=",");
```

You can even do this (reversed or arbitrary order):

```
text popupmenu(6:=",", 0:="A,B,C");
```

Or this (processing is from left to right, the last mapping wins):

```
text popupmenu("a,b", 6:=";", 6:="|", 6:=",", 0:="A,B,C");
```

Concatenation

Strings and variables can be concatenated by dots. Any number of blanks surrounding the concatenator are ignored. The following scripts are identical:

```
msg "Hi there!"
```

```
msg "Hi "."there!"
```

```
msg "Hi " . "there!"
msg "Hi "   .  "there!"
msg "Hi"." "."there!"
```

Using Quotes in Scripting

It's strongly recommended that you (double- or single-) quote your strings! While the script engine is currently still permissive with unquoted strings (`msg Hi!` works) this might not be so in the future, so you better do `msg "Hi!"` right away!

Here's the basic laws of quoting:

(1) Everything is either in double-quotes, or in single-quotes, or outside of any quotes.

(2) To have double-quotes inside double-quotes they must be doubled.

(3) To have single-quotes inside single-quotes they must be doubled.

(4) Variables are resolved in double-quotes and outside of any quotes, but not in single-quotes.

Commands and Functions

There are two types of commands: commands (proper) and functions. Contrary to commands (e.g. echo), functions (e.g. quote()) return an in-place value.

In the first example, the quoting is achieved by doubled double-quotes. The only command in the statement is echo. In the second example, the quoting is achieved through the function quote():

```
echo """Hi!""";    //"Hi!"
echo quote("Hi!"); //"Hi!"
```

Remarks on Functions

(1) Function names are not case-sensitive: QUOTE() = quote().

(2) Even without any argument -- e.g. quote() -- the parentheses are mandatory, else a function is not recognized.

(3) You can call a function without caring for the return argument (but you need the parentheses!):
`renameitem("John", , , "-01");`
Optionally you can use the dummy command call:
`call renameitem("John", , , "-01");`

(4) Functions are not resolved (interpolated, see below) when inside double-quotes.

Command Prefixes

Now you can add a prefix to each command to modify its behavior. Currently one prefix is implemented:

```
e = skip the big error debugging dialog
```

The prefix is separated from the command by | (pipe). For example, when you run this command in the drives listing:

```
rename , '*-<datem yyyymmdd>';      //shows error dialog
e|rename , '*-<datem yyyymmdd>';    //skips error dialog
```

Quick Scripting

Quick Scripting means: Scripts, when prefixed with "::" (double-colon), can be executed directly (i.e. *quickly*) through any interface in XYplorer that can process locations: Address Bar, Go To, Favorites, Catalog etc. This gives you a lot of choices for usage and storage of scripts.

Let's try:

Paste `::msg "Hello world!"` into the Address Bar and press Enter. You should see a "Hello world!" message box.

Now, for mouse users, the Catalog is a very good place for scripts. Using the "::" prefix, scripts can be simply entered into the Location field of a catalog item. They are executed on a single click.

For example:

(1) Add a new category "Scripts" and add a new item to it.

(2) Set item's Location field to `::#1026` and save it. The icon turns into the script symbol.

(3) Now click the item. The Find Files tab on the Info Panel is opened, all search filters are reset, and the cursor is placed into the Name field. Nice!

But how did this work? And what is #1026? Read on...

Note: From version 9.70 onwards the "::" prefix is not necessary anymore when you end your script line with a trailing ";" (appended comments after the ";" are allowed). This behavior is enabled by the INI setting ScriptSmartDetect=1.

Scripting by Numbers

In XYplorer almost every function has a fixed number, the function ID, by which it can be referred to in a script. ID #1026 happens to refer to "Miscellaneous / Find Files / Open Find Files and Reset". Open the Customize Keyboard Shortcuts dialog (menu Tools) and find this function in category Miscellaneous. At the bottom of the dialog you'll see a button showing the function's ID. Clicking this button will copy the function's ID to the clipboard, making it easy to use it in a script.

You can execute almost any function in XY in a script by referring to its function ID. E.g. `#230` will pop up the submenu "New" of menu Edit.

Step Mode: Stepping Through Scripts

Suppose you have an older script `#230`, and you forgot what #230 refers to. Or you downloaded a script from the internet, and don't know exactly what it does. What now? Very simple, enter Step Mode:

(1) Open the Scripting menu and select Step Mode.

(2) Now execute your mysterious script.

A small window (resizable), the Step Dialog, will pop up and tell you what is about to happen. You can now decide whether to execute the command, or skip it, or cancel the whole script. In stepping mode you are on the safe side. It is highly recommended when writing or debugging scripts!

There's also a toolbar button for toggling the stepping mode. When pressed (stepping is ON) its color changes to red to make the current state very clear.

Error Messaging and Risk Classes

The Step Dialog also tells you what went wrong where, and it informs you about the potential risk associated with the command to be executed. There are the following risk classes:

#3 Potentially harmful because the function ID might refer to some risky process which is unknown at this point (either an internal command or a User-Defined Command): #[function ID]

#2 Potentially harmful because files or folders are affected: backupto, copyto, delete, download, moveto, new, open, openwith, rename, rotate, run, setkey, swapnames, timestamp

#1 Potentially harmful because Step Mode is terminated: unstep

#0 Harmless: (all others)

Class #3, #2, and #1 are marked with a yellow "warning" icon, class #0 with a blue marble (think "cool") icon.

Script Context

The first two lines in the Step Dialog display (1) the current script resource, and (2) the current script caption. Valuable information when writing and debugging scripts.

How Functions are Stepped

Functions are individually stepped. The current script line is marked green when a contained function is stepped as opposed to the main command of the line. You can even skip (button Skip) functions individually, in which case the function name is returned with the resolved arguments. For example:

```
msg quote(chr(64));
```

- If you continue both functions the result is: "@"
- If you skip chr() and continue quote() the result is: "chr(64)"
- If you continue chr() and skip quote() the result is: quote(@)
- If you skip both functions the result is: quote(chr(64))

When you "Continue without Stepping" on a function, the stepping is only suspended for all functions in the current script line / command.

Variables... (Button)

Click the Variables... button to display all currently assigned variables with their current values. Within the new "Variables on Stack" dialog you can double-click any listed variable to show its current value.

Right-click that button for further options:

Show User Functions will list all currently declared user functions. Within the new "User Functions" dialog you can double-click any listed function to show its code.

Scripting and User-Defined Commands

Now, for passionate keyboarders, there are User-Defined Commands (UDCs). Using the "::" prefix, scripts can be simply entered into the Location field of a UDC Go To item, which then can be assigned a keyboard shortcut to.

For example:

(1) Click menu User | Manage Commands...

(2) Select the category Go To, and click the New button.

(3) Enter `::text "<clipboard>"` into the Location field.

(4) Assign a keyboard shortcut, say Ctrl+Alt+3.

(5) Click OK.

Now press Ctrl+Alt+3. You should see a small window displaying the current textual contents of the clipboard.

Of course, you don't have to abuse the UDC Go To for scripts. There's also the UDC Run Script which accepts scripts without a prefixed "::", e.g. `text "<clipboard>"`.

Run Script can also handle more advanced stuff like multi-line scripts, and multi-scripts, the rules and possibilities of which will be further explained below under "XYplorer Script Files".

Multi-line Scripts and Multi-Scripts

A script can have more than one line (multi-line script), and a script resource can have more than one script (multi-script). There is one important formatting rule for multi-line scripts:

> In a multi-line script all lines apart from the first line have to be indented by at least one space.

Each non-indented line is interpreted as the first line of a separate script, and if there are more than one scripts in a loaded script resource (i.e. a multi-script) then a menu is popped where you can select the script to run.

For example, this is a multi-script containing two multi-line scripts. In the first script, the script caption (which is used as the menu caption) forms the first line:

```
"Go to C:\"
  goto "C:\";
goto "%winsysdir%";
  selectitems "calc.exe";
```

Comments are an exception: Any number of non-indented comments can be prefixed to a multi-line script. Actually non-indented comments can be inserted anywhere. This script is functionally identical to the above one:

```
// comment 1
// comment 2
"Go to C:\"
// comment 3
  goto "C:\";
// comment 4
goto "%winsysdir%";
  selectitems "calc.exe";
// comment 5
```

Initialize and Terminate

You can define two special-named scripts within multi-scripts, "_Initialize" and "_Terminate". They can be placed anywhere in the multi-script, and they are not shown in the popup menu. The script called "_Initialize" will be auto-processed before the menu is popped. The script called "_Terminate" will be auto-processed after the script selected from the menu has been processed (or after the menu has been canceled).

Note: If a script within a multi-script is called directly (SCs Sub or Load), then "_Initialize"/"_Terminate" are NOT called.

For example, this multi-script defines a permanent variable (which is also global by definition), then offers various scripts in a popup menu, then finally removes the variable from memory:

```
"_Initialize"
  // if variable already exists it is NOT reset here but keeps it current value
  perm $p_a;
  // explicitly initialize it to zero
  $p_a = 0;
"_Terminate"
  unset $p_a;

"Show Variable"
  echo $p_a;
"Plus One"
  $p_a = $p_a + 1;
  echo $p_a;
```

```
"Minus One"
  $p_a = $p_a - 1;
  echo $p_a;
"Load Plus One"
  // will NOT call "_Initialize"/"_Terminate"
  Load "*", "Plus One";
  // WILL call "_Initialize"/"_Terminate"
  Load "*", "*";
"Sub Plus One"
  // will NOT call "_Initialize"/"_Terminate"
  Sub "Plus One";
```

Script Files

Here we are talking popup menus that are user-defined by a text file. Let's make one:

Create a new text file in any editor.

Paste the following multi-line script (see above):

```
// some little test scripts
"Go to C:\"
  goto "C:\"
"Go to System Folder"
  goto "%winsysdir%"
"Go to XYplorer Folder"
  goto "<xypath>"
```

Save the file as "test.xys" (XYplorer Script File) in XYplorer's Scripts folder. In case you don't know that path, use menu Scripting | Go to Scripts Folder to go there.

Now, in XYplorer, click menu Scripting | Load Script Files... and open "test.xys". The following menu should pop up at your mouse cursor:

```
Go to C:\
Go to System Folder
Go to XYplorer Folder
```

Now you can choose where you want to go.

A script file is basically a library of scripts. It is nothing more than a simple text file, which can contain one or more scripts. You will be able to either call one of those scripts directly, or simply load the entire file. In such case, XY will create a menu based on the contained scripts in that file and pop it up, allowing you to choose which script to execute.

The syntax for the scripts themselves within script files is exactly the same as for scripts anywhere else

in XY since this is just another way to store and execute scripts.

Syntax rules for XYplorer Script Files

(1) Lines starting with // are ignored and can be used for comments.

(2) One script can run over multiple lines. Simply indent the lines after the first line using any number of space or tab characters.

(3) You can have more than one script inside a script file. In that case, loading the script file will pop up a menu presenting all scripts inside the script file by their captions.

(4) To set a script caption simply prefix the desired caption to the script, and wrap it in quotes.

(5) Within a caption you may define a label that allows you to execute a script from a file directly.

(6) Using the command sub in a script file you can execute other scripts inside the same file.

(7) You may hide scripts by prefixing an underscore (_) to their caption.

To Load a Script File do one of the following:

(1) Use menu Scripting | Load Script File...

(2) Use a User-Defined Command from category "Load Script File".

(3) Use the script command `load [scriptfile]`.

The existence of a command `load`, of course, means that one script file can load another. You will get an idea of the potential of scripting by now...

Drop on a Script File

You can drop files onto XYS-files. The dropped files are referred to in the dropped-on script by <get drop>.

This can get pretty cool in Dual Pane mode. You can have a folder with assorted script files in one pane, and your working folder in the other pane. Now you just drag-and-drop files onto the scripts for automated processing.

Primitive example script in a file "drop_on_me.xys":

```
text <get drop>;
```

Even cooler: You can also drop on multi-scripts. In that case you get the usual popup menu of choices, and the <get drop> will be available in each of the scripts.

Primitive example script in file "drop_on_me_2.xys":

```
"Text"
  text <get drop>;
"Echo"
  echo <get drop>;
```

Notes:

- If more than one file is dropped <get drop> returns one per line. Alternatively you can pass a separator like this:
  `text <get drop |>;`
- The <get drop> variable is cleared after the script is processed, so it cannot be used after the drop event is completed.
- <drop> is a synonym for <get drop>.
- If the script contains no <get drop> variable it is run nevertheless just as if you loaded the script file.
- You can as well drop files on shortcuts (LNK) to XYS-files.
- You can also drop text onto XYS-files (and LNKs to XYS-files), e.g. selected text from a webpage. Just like with dropped files, the dropped text can be referred to in the dropped-on script by <get drop>.

Script Files for the Advanced

Labels

By using labels you can execute a script inside a file directly, avoiding the popup menu. The label is attached to the caption, separated by " : " (space-colon-space). For example:

```
// some little test scripts, using labels
"Go to C:\ : croot"
  goto "C:\"
"Go to System Folder : system"
  goto "%winsysdir%"
"Go to XYplorer Folder : xy"
  goto "<xypath>"
```

If the above is saved to a file called "test.xys" in application data path then the following command will directly bring you to the System folder: `load "test.xys", "system"`.

You may as well specify a list of labels, by which you can easily control which scripts are displayed in the popup menu, and in which order. See load for the details.

Wildcard catch-all label: The label "*" matches all load calls if more than one label is stated. The third command will be shown on `load "test.xys", "croot;system"`:

```
// some little test scripts, using labels
"Go to C:\ : croot"
  goto "C:\"
"Go to System Folder : system"
  goto "%winsysdir%"
"Go to XYplorer Folder : *"
  goto "<xypath>"
```

Hiding scripts inside a script file

Hidden scripts can be executed but are not shown in the script file's popup menu. To hide a script simply prefix an underscore to the caption or label (a hidden script does not need a caption anyway). For example, create a script file "date.xys" in application data path with the following contents:

```
// this is in script file "date.xys"
"_date"
  msg "<date yyyy-mm-dd>"
"_time"
  msg "<date hh:nn:ss>"
"Show Date : date"
  sub "_date";
"Show Date && Time : datetime"
  sub "_date";
  sub "_time"
```

Now execute the script `load "date.xys"`. The popup menu will show only two of the four contained scripts. Select either and see what happens.

Now run the script `load "date.xys", "date"`. The script with the label "date" will be executed directly. It has only one command: `sub "_date";`. The sub command is a special command to call a script inside the same script file. In this case the hidden script with the label "_date" is called and executed. Its command `msg "<date yyyy-mm-dd>"` produces the message box showing the current date.

Variables in Captions

The captions of scripts in multi-script resources may contain XYplorer native variables. They are resolved in the moment the menu is popped up. For example, paste this into the Try Script box:

```
"Go to <xypath>"
  goto "<xypath>";
"Is it <date hh:nn:ss>?"
  msg "It's <date hh:nn:ss>!";
```

The 2nd script will show two different times if you wait longer than a second before you click the menu item.

Also Environment Variables and Permanent Variables are supported in captions.

Icons, States, and Levels

You can optionally define an icon and a state for each menu item.

Syntax:

```
"Caption|Icon|State|Level : Label" Script
```

```
  where
State Default = 1
State Checked = 2
State Disabled = 4
```

Examples for States

```
"Go C:|C:|1" goto "C:\";  //shown bold
"Go D:|D:|2" goto "D:\";  //shown checked
"Go E:|E:|3" goto "E:\";  //shown bold and checked
```

Icon can be any file or folder, and its small system icon (the one you see in Details View) will be used for the menu, or a PNG, JPG, GIF, BMP, or TIF file. XY variables and environment variables are supported. The path defaults to the XY icon path <xyicons>.

You can as well use XYplorer's internal toolbar icons using the button key, e.g.:

```
"Funny Script|Icons\fun.ico" echo "Ha!";
"Say Ho!|iexplore / 12" echo "Ho!";
"Recent Locations|:mru" button "mru"
"Hotlist|:hotlist" button "hotlist"
```

Note that the icon specification in a multi-script resource supports permanent global variables.

You can as well use generic file icons in the multi-script menus by passing the generic extension as "*.ext":

```
"Megan|*.png"  echo 'hello';
"Betty|*.jpg"  echo 'hello';
```

You can as well pass * as placeholder to use the caption as icon pointer:

```
"C:\|*" goto "C:\";
"C:\Windows|*" goto "C:\Windows";
```

Levels: Multi-Scripts support nesting. The level is simply stated by the number denoting its depth, first level is 0 (zero), which is also the default, of course. Up to 256 levels are possible. For example:

```
"C:|*"
"Go to C:\|||1"
  goto "C:\";
"|||1" goto "%winsysdir%";
  selectitems "calc.exe";
"D:|*"
"Go to D:\|||1"
  goto "D:\";
```

Relative Levels: Multi-script nesting can be defined with relative levels. This can be useful when building menus from local resources by use of the Include statement.

Syntax: A relative level is marked by a prefixed "+" character. The number following "+" is added to the last defined absolute level.

For example, the following two code samples create identical nested multi-scripts:

```
"A" echo "A";
"B|||1" echo "B";
"C|||2" echo "C";
"D|||1" echo "D";
"B|||2" echo "B";
"C|||3" echo "C";
```

```
"A" echo "A";
"B|||+1" echo "B";
"C|||+2" echo "C";
"D|||1" echo "D";
"B|||+1" echo "B";
"C|||+2" echo "C";
```

However, the second sample uses relative levels which allows it to re-use the same part of code twice:

```
"B|||+1" echo "B";
"C|||+2" echo "C";
```

So this part of code could be outsourced to a file say "IncludedMenuBC.xys" and then be included like this:

```
"A" echo "A";
include "IncludedMenuBC.xys"
"D|||1" echo "D";
include "IncludedMenuBC.xys"
```

The Goto-Shorthand

In a multi-script resource you can pass a plain path/file spec as a shorthand for a well-formed goto script. Such a line will be auto-converted to a valid goto command including the appropriate icon in the generated popup menu.

Long version using well-formed goto scripts with captions:

```
"C:\|C:\" goto "C:\";
"C:\Windows|C:\Windows" goto "C:\Windows";
"C:\WINDOWS\SoundMan.exe|*" goto "C:\WINDOWS\SoundMan.exe";
```

Equivalent shorthand version:

```
C:\
C:\Windows
C:\WINDOWS\SoundMan.exe
```

The goto-shorthand additionally supports environment and native variables, Quick Searches, and visual filters. For example:

```
Desktop
%tmp%
<xydata>
C:\WINDOWS\system.ini
C:\WINDOWS\system32
C:\WINDOWS\system32?a*
C:\WINDOWS\system32?:a* or b*
C:\WINDOWS\system32?:a* | b*
C:\WINDOWS\system32?lbl:blue
C:\WINDOWS\system32|a*
```

Comments

Two types of comments are supported, line end comments, and block comments.

Line End Comments

Line end comments begin with a double-forward slash (outside of any quotes) and end at the end of the line:

```
$a = "<xypath>"; assert $a=="<xypath>"; //should not fail
```

```
$a = "<xypath>";            //assign XY path
  assert $a=="<xypath>";  //should not fail
```

```
"get CountSelected" // comment
  $count = get("CountSelected");  // comment
  assert $count!=0,     // comment
  "You must select a file before running this script!"
```

Block Comments

Block comments (aka C-style comments) start with /* (outside of any quotes) and end with the next */ (outside of any quotes) and can span any number of lines:

```
/* This is
a multi-line
block comment
```

```
*/ msg "hi!";
```

```
msg /*they can be inserted anywhere!*/ "hi!" /* anytime */;
```

Remarks on Comments

(1) Line-end and block comments overwrite each other:

```
msg "Hi!"; // /*this is not a block comment starting...
msg /* //this is not a line end comment starting... */ "Hi!";
```

(2) Make sure you don't nest block comments. It is easy to make

this mistake if you are trying to comment out a large block of

code:

```
/*
msg 'This is a test'; /* Will cause a problem */
*/
```

This, however, will work since // overwrites the /* */ comment:

```
//msg 'This is a test'; /* This comment is NO problem */
```

Variables

Advancing in script writing, you will soon feel the need for variables. XYplorer allows you to define and use as many variables as you want, using a number of commands like set, input, replace, etc. The script `set $a, "Hi!"; msg $a;` will define a new variable $a and assign the string "Hi!" (without the quotes) to it; then a message box will display "Hi!". The same can be achieved using the assignment operator (=):

```
$a = "Hi!"; msg $a;
```

Interpolation

Variables are resolved wherever they are found in the arguments of all subsequent commands of the script, even if they are found inside double-quoted strings (see below, Interpolation); for example:

```
$name = "Ted"; msg "Hi, I'm uncle $name!";
```

will display the message "Hi, I'm uncle Ted!".

Format and Scope

Variables have to conform to the following rules:

(1) Variables are represented by a dollar sign ($) followed by the name of the variable.

(2) Names start with a letter or underscore, followed by any number of letters, numbers, or underscores.

(3) Letters are a-z and A-Z.

Good variables: $a, $ABC, $good_variable, $a1, $a_, $_a, $_

Bad variables: a, ABC, $1, a$, $ä, $., $

Variable names are case-sensitive.

The scope and lifetime of a variable begins and ends with the script where it has been defined.

Using the equal-operator (=)

To assign a value to a variable you can simply use the following syntax (as an alternative for the set command):

`$a = "b";` or `$a="b";` (spaces are ignored)

For the additional "reprocess" operand see description of the set command.

Increment Syntax (++/--)

The common increment syntax using the ++ (--) operator is supported.

```
$i=5; $i++; msg $i; //6
$i=5; $i--; msg $i; //4
$i++; msg $i; //1
```

You can also use $i++/$i-- as an argument. The value is incremented/decremented after it is passed to the function*:

```
$i = 1; echo $i++; echo $i;     //1; 2
$i = 1; echo 1 + $i++; echo $i; //2; 2
$i = 1; echo $i++ + 1; echo $i; //2; 2
$i = 1; echo $i++ . $i++ . $i;  //123
```

Note in the following that the left operand is incremented before the right operand is added. Finally, after the addition, the right operand is incremented:

```
$i = 1; echo $i++ + $i++; echo $i; //3; 3
```

* *Note that before v14.30.0100 the value was incremented/decremented before it was passed to the function!*

Interpolation

Interpolation means that variables that are embedded in double-quoted or unquoted strings are resolved (replaced with their corresponding value). Examples:

```
$name = "Ted"; msg "Hi, I'm uncle " . $name . "!";
```

Displays "Hi, I'm uncle Ted!". The variable is concatenated with literal strings.

```
$name = "Ted"; msg "Hi, I'm uncle $name!";
```

Displays "Hi, I'm uncle Ted". The variable is interpolated inside double-quoted literal strings.

```
$name = "Ted"; msg "Hi, I'm " . uncle $name!;
```

Displays "Hi, I'm uncle Ted". The variable is interpolated inside non-quoted literal string `uncle $name!`. Note that using non-quoted literal strings is not recommended and might even be deprecated in a future version!

You can block interpolation by using single-quotes:

```
$name = "Ted"; msg 'Hi, I''m uncle $name!';
```

Displays "Hi, I'm uncle $name!". The variable is not interpolated inside single-quoted literal strings. Note that single-quotes embedded in single-quotes have to be doubled (`I''m`).

```
msg '%TMP% = ' . %TMP%;
```

Displays "%TMP% = C:\Temp". The first `%TMP%` is blocked from interpolation by being embedded in single-quotes.

```
$date = "<date>"; msg '$date = ' . $date;
```

Displays "$date = 28.08.2008 12:23:12".

Variables Scope and Lifetime: Local, Global, and Permanent Variables

Local Variables

By default, all variables in XY scripting are local, i.e. they are not shared between called and calling scripts. In other words, whenever one script calls another script (e.g. using commands "sub" or "load"), a new local namespace is created by the called script and pushed on the stack.

Global Variables

Generally, global variables are shared between scripts. This can make scripts hard to maintain. However, the mechanism of "globalization" -- (almost) identical to the one used in PHP -- used by XY scripting gives you maximum control over what is shared and where. Global variables are implemented by means of the command global.

Permanent Variables

The lifetime of local and global variables ends with the current script or script stack. Permanent variables, however, stay alive in memory for the whole XYplorer session, or even across sessions if configured like this (Configuration | Refresh, Icons, History | Remember permanent variables). Hence permanent variables can be easily shared between scripts. Permanent variables are implemented by means of the command perm.

Permanent Variables created / modified in a called script are immediately visible / updated in the calling script when the called script returns.

Note the scripting commands writepv and readpv by which you can read and write permanent variables from/to file. This allows for some interesting use as portable data storage.

Permanent variables can be accessed from anywhere in the app using the `<perm ...>` meta variable. See Variables.

To release all permanent variables from memory use the command releaseglobals. You can as well unset them individually via menu Scripting | Permanent Variables directly from the right-click menu in the variable list.

To view and modify the current permanent variables use menu Scripting | Permanent Variables.

Initial Values

Local variables that have never been set to a value return their name as value (strictly speaking they aren't variables but normal strings). Global and permanent variables that have never been set to a value are initialized to "" when they are declared by global or perm:

```
msg $a;     // displays "$a" if $a is has not set to any value before, and has not been declared as global or permanent.
global $b;  // declare global variable
msg $b;     // displays "" if $b has not been set to any other value before.
perm $c;    // declare permanent variable
msg $c;     // displays "" if $c has not been set to any other value before.
```

Arrays

Two types of arrays are supported, indexed arrays and associative arrays. The syntax is similar to that used in eg C++, PHP or Java.

Here are some examples for indexed arrays:

```
$a[0]="pussy"; echo $a[0]; //pussy
$a[0]="pussy"; $a[1]="cat"; echo $a[0].$a[1]; //pussycat
$a[2]="cat"; $b=1; echo $a[$b+$b]; //cat
```

Array elements also work within quotes:

```
$a[0] ="cat"; echo "It is a $a[0]!";  //It is a cat!
$a[0]="pussy"; $a[1]="cat"; echo "$a[0]$a[1]"; //pussycat
```

Here are some examples for associative arrays (the named keys can be single- or double-quoted):

```
$a['pussy']="cat"; echo $a["pussy"]; //cat
$a["pussy"]="cat"; $b="pussy"; echo $a[$b]; //cat
```

If the index or key is invalid the variable is seen just as a bit of text:

```
$a[0] ="cat"; echo "It is a $a[1]!"; //It is a $a[1]!
$a["pussy"] ="cat"; echo "It is a $a['fussy']!"; //It is a $a['fussy']!
```

Also a missing key makes the variable invalid:

```
$a[0] ="cat"; echo "It is a $a[]!"; //It is a $a[]!
```

If you assign a non-first element in a new or smaller indexed array, all previous elements starting with [0] are automatically created (with value ""):

```
$a[1] ="cat"; echo $a[0]; //"" ($a[0] is implicitly created and set to "")
$a[0] ="cat"; echo $a[1]; //$a[1] ($a[1] does not exist as variable)
```

The index can be a complex expression:

```
$n = 4; $a[$n+4] = $n * 4; echo $a[$n+4]; //16
$b[0] = "pus"; $b[1] = "sy"; $a[$b[0] . $b[1]] = "cat"; echo $a[$b[0] . $b[1]]; //cat
```

You can copy one array to the other (appending [] to the variable name is optional):

```
$b[] = $a[];
```

```
$b = $a; //$a is an array variable; $b becomes an array variable
```

Some more properties of arrays:

- The global command is supported by arrays, eg: `global $a[];`
- The perm command is not supported by arrays but is simply ignored: Arrays cannot be permanent.
- Allowed range of elements in an indexed array: 0 to 32767.
- Maximum number of elements in and associative array: 32768.
- Various scripting functions help dealing with arrays: count() , explode(), implode().

Special Function array()

There is a special function array() to populate arrays. Non-existing arrays are created, dimensioned and populated, existing arrays are redimensioned and overwritten.

```
$a = array("cat", "dog"); echo $a[0]; $a = array("dog"); echo $a[0]; //cat, dog
```

Notes:

- You would usually pass literal strings as values, not variables. Values are separated by commas.
- You can also pass a single variable to the array() special function that resolves to a list of strings. For example:
  `$values = "vampire, cow"; $arr[] = array($values); echo $arr[0]; //vampire`
- The values can be in double quotes (which will be removed), or also without quotes (fine if you are not using any commas or flanking spaces within the values):
  `$a = array(cat, dog); echo $a[0]; //cat`
- The values can also be in single quotes but those will not be removed.
- If you like you can append [] to the variable, it makes no difference:
  `$a[] = array("cat", "dog"); echo $a[0]; //cat`
- The values are added to the array in the order they are listed, starting with element [0].
- So far the indexed arrays, but you can also populate associative arrays using array():
  `$name = array("cat" => "pussy", "dog" => "rex"); echo $name["cat"]; //pussy`
  General syntax:
  `... = array("key1" => "value1", "key2" => "value2")`
  Again, you can get away with stripping the quotes and the spaces:
  `$name = array(cat=>pussy,dog=>rex); echo $name["dog"]; //rex`
- You can use array() without any values to completely reset an array:
  `$a = array("cat", "dog"); $a = array(); echo $a[0]; //$a[0]`
- After "$a = array();" the variable $a is an array with zero elements:
  `$a = array("cat", "dog"); $a = array(); echo count($a); //0`

Foreach Loops with Arrays

The syntax here is a bit different from the Foreach Loops with Lists syntax.

General form:

```
foreach($array as $value, [flags]) {
  statement(s) using $value;
}
```

Example:

```
$a = array("cat", "dog", "bat");
  foreach($a as $value) {
    echo $value;
  }
```

Reversing the order (r flag) is supported:

```
$a = array("cat", "dog", "bat");
  foreach($a as $value, "r") {
    echo $value;
  }
```

Skipping empty items (e flag) is also supported:

```
$a = array("cat", "", "bat");
  foreach($a as $value, "re") {
    echo $value;
  }
```

The Foreach Loop also supports returning the keys in associative arrays. General form:

```
foreach($array as $key => $value, [flags]) {
  statement(s) using $key and $value;
}
```

Example:

```
// make associative array
  $freelancer = array(
    "name" => "Groot",
    "email" => "groot@gmail.com",
    "age" => 22,
    "gender" => "unknown"
  );
  // loop through array
  foreach($freelancer as $key => $value) {
    echo "$key: $value";
  }
```

Reversing the order (r flag) and skipping empty items (e flag) is supported.

The key variable is set to the numeric index if you do 'foreach($array as $key => $value)' on an non-associative array:

```
$key="FOO"; $a = array("cat", "dog", "bat");
  foreach($a as $key => $value) { //always overwrites $key
    $b[$key] = "$key=$value";
  }
  text implode($b);  //0=cat|1=dog|2=bat
```

Nested Expressions

You can nest expressions using parentheses, aka round brackets: ( ). There's no practical limit to nesting depth and superfluous parentheses are silently removed.

Examples where parentheses are merely decor:

```
msg "a" . "b";
msg ("a" . "b");
msg ("a") . ("b");
msg (("a") . ("b"));
msg ((("a") . ("b")));
msg "a" . "b" . (); //"ab"
msg "a" . ("b" . "c");
msg ("a" . "b") . "c"; //"abc"
msg "a" == "a" . "a" == "b";
msg ("a" == "a") . ("a" == "b"); //"10"
```

Examples where parentheses actually make a difference:

```
msg "a" == "a" . "a";
msg ("a" == "a") . "a"; //"1a"
msg "a" == ("a" . "a"); //"0"
```

Examples for nesting errors:

```
msg ("a" . "b"; // ')' missing! ->  ("a" . "b"
msg "a" . "b"); // '(' missing! -> = a"b")
```

Math

Scripting can do basic calculation using math operators +-*/, and also \ (integer division), % (modulo), and ^ (exponentiation). Fractions and parentheses are supported. Also unary operators + and - are

supported.

Examples

```
echo 1 + 1;
echo 1 - 2 - 3;   // -4
echo 1 - (2 - 3); //  2
echo 1/3 + 1/3;
echo 1 + 2 * 3;
echo (1 + 2) * 3;
echo 1/0;  // ERROR: division by zero
$a=3; $b=2; echo $a / $b; // 1.5
$fraction = 1.5; echo $fraction == 3/2; // true
echo 1.2 + 2.1; // 3.3
echo 1 + --1; //2
echo --(1 * ----2); //2
echo 5 \ 2; //2 (integer division)
echo 5 / 2; //2.5
echo 5 % 2; //1 (modulo)
echo 2 ^ 3;   //8
echo 4 ^ 0.5; //2
echo 4 ^ -1;  //0.25
```

Remarks

(1) Strings are converted to numbers as possible (just like in comparisons).

```
$a=""; $b="2"; echo $a + $b; // = 2
$a="1a"; $b="2b"; echo $a + $b; // = 3
```

(2) The decimal separator is NOT locale specific (e.g. dot in US, comma in Germany) but hard-coded to dot (period). This way scripts are interchangeable between regions.

(3) Calculation uses 8-byte floating point numbers with the following ranges:

```
negative: -1.79769313486232E308   to -4.94065645841247E-324
positive:  4.94065645841247E-324  to  1.79769313486232E308
```

Floating point arithmetic has its natural shortcomings so don't expect too much in terms of precise accuracy down to the 12th digit. A file manager is not a scientific calculator.

(4) The operator precedence is `^ > (*,/) > (\,%) > (+,-)` meaning that * and /, \ and %, + and - are of equal weight.

Processing of math terms is from left to right:

```
echo 36 / 4 * 3;  // 27!   (not 3)
```

(5) With integer division (the portion of the result after the decimal point is lost) and modulo (the remainder of integer division) the operands are rounded *before* the operation!

```
echo 5.9 \ 2.1; //3!
```

```
echo 7.9 % 2.5; //2! (8 % 3 = 2)
echo 7.9 \ 0.4; //ERROR: Division by zero.
echo 7.9 % 0.4; //ERROR: Division by zero.
```

(6) Fractional numbers should be stated with a dot as decimal separator, or alternatively in quotes with the local decimal separator:

```
echo 7.5 / 2.5;       //3 -- works under each locale
echo "7,5" / "2,5";   //3 -- works only where comma is the decimal separator
```

Internally decimal separator are stored and processed according to the system locale:

```
$n = 2.5; echo $n;    //$n is "2,5" internally where comma is the decimal separator
```

Hex Numbers

The parser recognizes the common prefix `0x` (zero-x) for hexadecimal values. Supported are up to 4 bytes per number, i.e. 8 hex digits. The hex digits are case-insensitive (the prefix is not).

Here for some examples; hex values are automatically resolved to decimal values:

```
echo 0xa; //10
echo 0xA; //10
echo 0xFFFFFF;   //16777215
echo 0xFFFFFFFF; //-1
echo 0x7FFFFFFF; // 2147483647
echo 0x80000000; //-2147483648
echo 0xA + 0xA; //20
```

The following are examples for invalid hex strings; they are returned unresolved:

```
echo 0xG;           //0xG (invalid value)
echo 0x;            //0x  (no value)
echo 0XA;           //0XA (wrong prefix)
echo 0x123456789;   //0x123456789 (too many characters)
echo "0x12345678";  //0x12345678 (quoted)
```

Tip: To convert HTML #RRGGBB colors to color decimals use this formula:

```
0xBBGGRR
```

Binary Numbers

The parser recognizes the common prefix `0b` (zero-b) for binary values. Supported are up to 4 bytes per number, i.e. 32 binary digits. Here for some examples:

```
echo 0b0; //0
echo 0b1; //1
```

```
echo 0b11; //3

echo 0b10000000; //128

echo 0b11111111; //255

echo 0b100000000; //256

echo 0b00000000000000000000000000000001; //1

echo 0b10000000000000000000000000000000; //-2147483648

echo 0b11111111111111111111111111111111; //-1
```

The following are examples for invalid binary strings; they are returned unresolved:

```
echo 0b2;           //0b2 (invalid value)

echo 0b;            //0b  (no value)

echo 0B1;           //0B1 (wrong prefix)

echo 0b11000000000000000000000000000000; //0b110000000000000000000000000000000 (too many
    characters)

echo "0b1";         //0b1 (quoted)
```

Comparisons

Comparisons of two values are evaluated in-place. The "comparison" has the general form

```
"a" operator "b"
```

where "a" and "b" are string expressions, and "operator" can be one of the following:

```
==  Equal
!=  Not Equal
<   Less than
>   Greater than
<=  Less than or Equal to
>=  Greater than or Equal to
 Like  Matches Pattern (case-sensitive)
LikeI  Matches Pattern (case-insensitive)
UnLike  Does not match Pattern (case-sensitive)
UnLikeI Does not match Pattern (case-insensitive)
```

If the comparison evaluates as True it is set to "1", else it is set to "0".

Examples

```
msg "a" == "a";
```

= shows '1'

```
msg ("a" == "a") . ("a" == "b");
```

= shows '10'

```
$r = ("a" == "a"); $r = ($r?"True":"False"); msg $r;
```

= shows 'True'

```
$comparison = "a == a"; assert $comparison == "a == a";
```

= no assertion error

```
$minver = "7.60.0009"; assert "<xyver>" >= $minver, "This script needs at least XY $minver!"
```

= no assertion error if XY version is >= 7.60.0009

Strings and Numbers

If you compare two numerical strings, they are compared as integers:

```
msg ("2" < "10"); // -> true! (both parts are numeric)
msg ("2" < "10a"); // -> false! (only one parts numeric)
msg ("2a" < "10a"); // -> false! (both parts are non-numeric)
```

The Like (LikeI) and UnLike (UnLikeI) operators

General form:

```
String Like Pattern   (case-sensitive)
String LikeI Pattern  (case-insensitive)
```

Where string is any string expression and Pattern may contain the usual wildcards and special chars used in XY patterns (*?#[]). Note that exact capitalization matters: "Unlike" or "unlike" won't work. These operators must be surrounded by spaces.

Examples:

```
echo "abc" Like "a*"; //1
echo "Abc" Like "a*"; //0!
echo "Abc" LikeI "a*"; //1!
echo "abc" UnLike "a*"; //0
echo "Abc" UnLike "a*"; //1!
echo "Abc" UnLikeI "a*"; //0!
echo "Abc" LikeI "abC"; //1 (no wildcard!)
echo "It's " . ("<date yyyy>" Like "20??"?:"not ") ."the 21st century";
```

Characters can be referred to by their ordinal value. Here, for example, a string is compared to a Unicode range covering most Hiragana and Katakana characters:

```
echo "kigi" Like "*[u+3041-u+3096u+30A0-u+30FF]*";  //0
echo "キギ" Like "*[u+3041-u+3096u+30A0-u+30FF]*";  //1
```

Boolean Operators

Scripting supports Boolean operators in the following order of precedence (see also Operator Precedence):

```
!     NOT    (unary operator)
&&    AND
||    OR
and   AND    (case-insensitive: AND, And...)
xor   XOR    (case-insensitive: xOR, Xor, XOR...)
or    OR     (case-insensitive: OR, Or...)
```

Examples

```
echo not 1; // 0
echo not 0; // 1
echo ! 1; // 0
echo !1; // 0
echo !!!1; // 0
echo !(1 and 0); // 1
```

```
// parsed as: (TRUE and FALSE) or (TRUE and TRUE);
  echo TRUE and FALSE or TRUE and TRUE; //1
```

```
// will show "1", then "done"
  $i = 1;
  while ($i < 2 && $i > 0) {
    echo $i;
    $i++;
  }
  echo "done";
```

Boolean Constants

The Boolean constants TRUE and FALSE (case is ignored) are recognized if they are used unquoted.

TRUE is resolved to 1.

FALSE is resolved to 0.

Examples

```
echo TRUE and TRUE;   // 1
echo TRUE and FALSE;  // 0
echo TRUE and false;  // 0 (constants are case-insensitive)
echo TRUE and "false";  // 1! (quoted "false" is NOT a constant)
echo (1==1) == TRUE;  // 1
echo (0==1) == FALSE; // 1
```

```
echo (1==1) != FALSE; // 1
```

Note that in a Boolean context the values "" and "0" evaluate to 0 (FALSE). All other values evaluates to 1 (TRUE):

```
echo "dog" and TRUE;  // 1  (TRUE and TRUE)(Boolean context)
echo "dog" == TRUE;   // 0  ("dog" == "1") (no Boolean context)
echo "0" == FALSE;    // 1  ("0" == "0")(same strings)
echo "" == FALSE;     // 0  ("" == "0") (no Boolean context)
echo "" XOR TRUE;     // 1  (FALSE XOR TRUE)
echo "0" XOR TRUE;    // 1  (FALSE XOR TRUE)
echo "dog" XOR TRUE;  // 0  (TRUE XOR TRUE)
```

Note that "0" and 0 ("1" and 1) are the same in XY scripting, so:

```
echo 0 == FALSE;      // 1  (0 == 0)
echo 0 XOR TRUE;      // 1  (FALSE XOR TRUE)
```

Ternary Conditionals

Scripting knows so-called "ternary conditionals" as used in many other programming languages. The logic is this:

```
if (condition) {
  variable = value-if-true;
} else {
  variable = value-if-false;
}
```

As ternary conditional the same can be written like follows:

```
variable = (condition) ? value-if-true : value-if-false;
```

The parentheses and the blanks are optional, so these are identical:

```
variable = (condition) ? value-if-true : value-if-false;
variable = (condition)? value-if-true: value-if-false;
variable = condition?value-if-true:value-if-false;
```

The part "condition" has the form

```
  "a" operator "b"
```

where "a" and "b" are string expressions and "operator" can be one of the following:

```
  ==  Equal
  !=  Not Equal
  <   Less than
  >   Greater than
```

```
<=  Less than or Equal to
>=  Greater than or Equal to
(none) True if expression is not 0 and not ""
```

The parts "value-if-true" and "value-if-false" are string expressions.

Examples

```
$a = "<date hh>" >= "12"? "afternoon": "morning"; msg "Good $a!";
$a = ("<date mm-dd>" == "12-24")? "": "not "; msg "It's $a"."X-mas!";
```

You can employ ternary conditionals in any argument or part of argument:

```
msg "Good " . ("<date hh>" >= "12"? "afternoon": "morning") . "!";
msg "It's " . ("<date mm-dd>" == "12-24"? "": "not") . " X-mas!";
```

Compound Assignment Operators

The Compound Assignment Operators .=, +=, -=, *=, /=, \= can be used to shortcut operations of two variables where the result is set to one of the variables. For example, both lines below are functionally identical; the latter one uses one of the Compound Assignment Operators:

```
$a = $a . "b";
$a .= "b";
```

More examples:

```
$a = "a"; $a .= "b"; echo $a;   //ab
$a = 1;   $a += 1; echo $a;     //2
$a = 1;   $a -= 1; echo $a;     //0
$a = 2;   $a *= 3; echo $a;     //6
$a = 5;   $a /= 2; echo $a;     //2.5
$a = 5;   $a \= 2; echo $a;     //2
```

Control Structures

Scripting offers various Control Structures by which you can control the order in which the individual statements are executed. Within these structures blocks of statements are grouped by encapsulating them with curly braces.

- If/ElseIf/Else Blocks
- While Loops
- Foreach Loops

If/ElseIf/Else Blocks

General syntax:

```
if (expression) {
  statement;
}
elseif (expression) {
  statement;
}
else {
  statement;
}
```

If expression evaluates to TRUE, the following statement block is executed and the other blocks are ignored. If expression evaluates to FALSE the following statement block is ignored and processing continues with the next Elseif or Else block.

Remarks

- Parentheses around the expression are mandatory.
- Curly braces around the statement block are mandatory (even if there is only one statement).
- The Elseif block(s) and Else block are optional. There can be only one Else block and it must be the last block in the whole control structure.

Examples

```
if (1 == 1) {echo "Hi!"}
```

```
// shows "Relax."
  if (1 == 2) {
    echo "Help!";
  }
  else {
    echo "Relax.";
  }
```

```
// shows "else", "elseif", "if"
  $i = 1;
  while ($i) {
    if ($i == 3) {
      echo "if";
      break 2;
    }
    elseif ($i == 2) {
      echo "elseif";
    }
    else {
      echo "else";
```

```
    }
    $i++;
  }
```

While Loops

General syntax:

```
while (expression) {
  statement;
}
```

The nested statement(s) are executed repeatedly, as long as the while expression evaluates to TRUE. The value of the expression is checked each time at the beginning of the loop. If the while expression evaluates to FALSE from the very beginning, the nested statement(s) won't even be run once.

Remarks

- Parentheses around the expression are mandatory.
- Curly braces around the statement block are mandatory (even if there is only one statement).

Examples

```
// will show 1, then 2, then terminate the script
  while ($i < 2) {$i++; echo $i;};
```

```
// will show 1, 2, 3, then terminate the script
  $x = 3;
  $i = 1;
  while ($i <= $x) {
    echo $i;
    $i++;
  }
```

```
// nested while blocks are okay
  $x = 3;
  $i = 1;
  while ($i <= $x) {
    $w = "Word";
    while ($w != "") {
      echo "$w No. $i";
      $w = "";
    }
    $i++;
  }
```

```
// expression is FALSE, message will never show
  while (1 == 2) {echo "Help!"};
```

For Loops

General syntax:

```
for (expression1; expression2; expression3) {
  statement(s);
}
```

The expression1 is executed once unconditionally at the beginning of the loop. In the beginning of each iteration, expression2 is evaluated. If it evaluates to true, the loop continues and the nested statement (s) are executed. If it evaluates to false, the execution of the loop ends. At the end of each iteration, expression3 is executed.

Remarks

- Parentheses around the expressions are mandatory.
- Curly braces around the statement block are mandatory (even if there is only one statement).

Internal Processing

For loops can be thought of as shorthand for While loops, and that's how they're supported now: For loops are internally converted to While loops. If you step through your scripts, you'll see that.

For example, this For loop:

```
for ($i = 1; $i <= 3; $i++) {
  echo $i;
}
```

... is internally converted to this While loop:

```
$i = 1;
while ($i <= 3) {
  echo $i;
  $i++;
}
```

Foreach Loops with Lists

General syntax:

```
foreach($variable, ListOfTokens, [separator="|"], [flags], [MsgOnEmpty]) {
  statement(s);
```

```
}
```

Remarks

- $variable is the variable which receives the value of the next token in each round of the loop. You can use a new variable or an already used one, it does not matter.
- The last value of $variable remains even after the foreach loop.
- ListOfTokens is a string of tokens, separated by a separator.
- The separator defaults to "|" (pipe), but can be set to anything, also multi-character strings. Passing an empty separator will tokenize the ListOfTokens by letter.
- Surrounding spaces are not trimmed off on each side of the tokens.
- flags can be either empty (default) or set to "r" for "reverse direction", or "e" to skip empty items in the list of tokens.
- In MsgOnEmpty you can state an error message for the case that ListOfTokens is empty. If MsgOnEmpty is given (even as empty string "") the loop will not be executed at all.
- SC break and SC continue are supported.

Examples

```
// returns 3 strings (moon, sun, venus):
  foreach($token, "moon,sun,venus", ",") {
    echo $token;
  }
```

```
// flag r: returns 3 strings in reversed order (venus, sun, moon):
  foreach($token, "moon,sun,venus", ",", "r") {
    echo $token;
  }
```

```
// flag e: returns 2 strings ("moon", "venus"); empty items are skipped:
  foreach($token, "moon,,venus,", ",", "e") {
    echo $token;
  }
```

```
// nested foreach loops
  foreach($token, "a|b|c") {
    foreach($token2, "1,2,3", ",") {
      echo $token . $token2;
    }
  }
```

```
// selected list items
  foreach($token, <get selecteditemspathnames |>) {
```

```
    echo $token;
  }
```

```
// shows "No files selected!" if no files are selected, and skips the loop
  foreach($item, <get selecteditemspathnames>, <crlf>, , "No files selected!") {
    echo $item;
  }
```

```
// passing an empty separator = tokenize by letter
  $string = 'string';
  foreach($letter, $string, '') {
    echo $letter;
  }
```

Switch Statements

The switch statement is similar to a series of IF statements on the same expression. In many occasions, you may want to compare the same variable (or expression) with many different values, and execute a different piece of code depending on which value it equals to. This is exactly what the switch statement is for.

In a switch statement, the condition is evaluated only once and the result is compared to each case statement. In an elseif statement, the condition is evaluated again. If your condition is more complicated than a simple compare and/or is in a tight loop, a switch may be faster.

General syntax:

```
switch (n) {
    case label1:
        code to be executed if n=label1;
        break;
    case label2:
        code to be executed if n=label2;
        break;
    case label3:
        code to be executed if n=label3;
        break;
    ...
    default:
        code to be executed if n is different from all labels;
}
```

Remarks

- The case statements are checked from top to bottom until a match is found.

- Each case should be closed by a break statement.
- It's possible to use a semicolon instead of a colon after a case.
- A special case is the default case. This case matches anything that wasn't matched by the other cases.

Examples

```
// switch
  $favcolor = "blue";
  switch ($favcolor) {
    case "red":
        echo "Your favorite color is red!";
        break;
    case "blue":
        echo "Your favorite color is blue!";
        break;
    default:
        echo "Your favorite color is neither red nor blue but $favcolor!";
  }
  echo "Switch done!";
```

The switch and case arguments can also be more complex expressions:

```
$a = 3; $b = 7; $c = 10.5; $d = 2;
  switch ($a * $b) {
    case $c * $d:
        echo "The result equals $c * $d = " . $c * $d;
        break;
    default:
        echo "The result is something else.";
  }
```

Usually each case should be closed by a break statement. Else processing continues with the next case (sometimes though this can be desired). Here is an example where omitting break statements makes sense. Also shows that the default case also accepts a break statement (although is totally superfluous here). Also shows that semicolons after the cases are an acceptable alternative to colons:

```
$beer = 'Kirin';
  switch($beer) {
      case 'Asahi';
      case 'Kirin';
```

```
      case 'Sapporo';
          echo 'Good choice';
          break;
      default;
          echo 'Please make a new selection...';
          break;
  }
```

The case statements don't have to be in their own line. This example also shows that using the return command within a function you can spare the break command:

```
function getChineseZodiac($year){
      switch ($year % 12) {
          case  0: return 'Monkey';  // Years 0, 12, 1200, 2004...
          case  1: return 'Rooster';
          case  2: return 'Dog';
          case  3: return 'Boar';
          case  4: return 'Rat';
          case  5: return 'Ox';
          case  6: return 'Tiger';
          case  7: return 'Rabbit';
          case  8: return 'Dragon';
          case  9: return 'Snake';
          case 10: return 'Horse';
          case 11: return 'Lamb';
      }
  }
  echo getChineseZodiac(2016);

  // BTW, user functions work in switch and in case:
  switch (getChineseZodiac(2016)) {
    case getChineseZodiac(2015): echo "2015"; break;
    case getChineseZodiac(2016): echo "2016"; break;
    case getChineseZodiac(2017): echo "2017"; break;
  }
```

Also note this somehow perverted use of a switch:

```
$a = "Kirin";
  switch (true){
    case $a=="Kirin":   echo "first choice"; break;
    case $a=="Sapporo": echo "second choice"; break;
```

```
  }
```

Heredoc and Nowdoc Syntax

Heredoc, using the <<< operator, is a way to define multi-line strings "as is" without the need to escape quotes etc.

The rules follow the PHP rules for Heredoc:

(1) After the <<< operator, an identifier (your free choice) is provided, then a new line. The string itself follows, and then the same identifier again to close the quotation. The closing identifier must begin in the *first* column of the line (no indent!). The line with the closing identifier must contain no other characters, except possibly a semicolon (;) directly after the identifier.

(2) Heredocs overwrite comments, and comments overwrite heredocs, depending on who comes first.

Within the Heredoc section:

(3) Line feeds, empty lines, and all kinds of comments survive.

(4) Lines are not trimmed (leading and trailing spaces are preserved).

(5) Quoting is handled automatically (no need to add outer quotes or to double inner quotes).

(6) Variables are resolved.

Example

A multiline script using Heredoc. Note that the Heredoc block is not indented, that comments are not removed, that quotes are not doubled, and that the variables are resolved (the only difference to Nowdoc), even within comments.

Basic example:

```
$str = <<<EOD
Example of string
spanning multiple lines
using heredoc syntax.
EOD;
  echo $str;
```

More complex example, with variables and comments:

```
$name = "Bond";
  text <<<FOO
My name is $name, "James $name". // line end comment: $name
```

```
Follow /* note: there's one space after me, */ me,
FOO
  ."please!";
```

Output:

```
My name is Bond, "James Bond". // line end comment: Bond

Follow /* note: there's one space after me, */ me, please!
```

Alternative Heredoc Syntax

A more radical parsing where the ending identifier can be anywhere is enabled when the identifier starts with a "#". These examples show two ways to code the same script.

```
$isOk = Confirm (<<<#FOO
Blah
BlahFOO#FOO); echo $isOk;
```

With the old-school parsing the ending identifier must be on its own line:

```
$isOk = Confirm (<<<FOO
Blah
BlahFOO
FOO
  ); echo $isOk;
```

Nowdoc Syntax

A special variety of Heredoc is the Nowdoc, that is a Heredoc without interpolation. Quoting from PHP Documentation:

> "Nowdocs are to single-quoted strings what heredocs are to double-quoted strings. A nowdoc is specified similarly to a heredoc, but no parsing is done inside a nowdoc. The construct is ideal for embedding PHP code or other large blocks of text without the need for escaping. It shares some features in common with the SGML <![CDATA[ ]]> construct, in that it declares a block of text which is not for parsing.
>
> A nowdoc is identified with the same <<< sequence used for heredocs, but the identifier which follows is enclosed in single quotes, e.g. <<<'EOT'. All the rules for heredoc identifiers also apply to nowdoc identifiers, especially those regarding the appearance of the closing identifier."

Example:

```
$var = "syntax";
  $str = <<<'EOD'
Example of string
```

```
spanning multiple lines
using nowdoc $var.
EOD;
  echo $str;
```

Output (note that $var has not been resolved):

```
Example of string
spanning multiple lines
using nowdoc $var.
```

Nowdoc also accepts the alternative radical option (see above) enabled by prefixing the identifier with '#' (within the quotes).

Dereference Operator

You can use the dereference operator * (asterisk) with all variables. Usage: Helpful when dynamically creating variables.

Examples

```
$var = '$a';        *$var = "TEST"; echo $a;  //TEST
$var = '$a';   perm *$var = "TEST"; echo $a;  //TEST
$var = '$a'; global *$var = "TEST"; echo $a;  //TEST
```

If the dereferenced variable is not defined, then the variable itself is used (as if there was no dereference operator):

```
*$undefined = "TEST"; echo $undefined; //TEST
```

Note that also unset and incr support the dereference operator:

```
$var = '$a';  *$var = "TEST"; unset *$var; echo $a; //$a
$var = '$a';  *$var = 1; incr *$var; echo $a; //2
$var = '$a';  *$var = 1; *$var++; echo $a; //2
$var = '$a';  *$var = 1; *$var--; echo $a; //0
$var = '$a'; *$var = 1; echo 1 + *$var++; echo *$var; //2; 2
```

Note that dereferencing is also supported in interpolation (including HEREDOC) if you explicitly allow it with the aid command.

```
aid 0; $var = '$a'; *$var = "TEST"; echo "*$var, $a!";  //*$a, TEST!
aid 1; $var = '$a'; *$var = "TEST"; echo "*$var, $a!";  //TEST, TEST!
```

Operator Precedence

The precedence of an operator specifies how "tightly" it is connected to its surrounding operands. For example, in the expression 1 + 2 * 3, the answer is 7 and not 9 because the multiplication ("*") operator has a higher precedence than the addition ("+") operator. Parentheses may be used to force precedence, if necessary. For instance: (1 + 2) * 3 evaluates to 9.

The following table lists the precedence of operators with the highest-precedence operators listed at the top of the table. Operators on the same line are evaluated from left to right (3 - 2 + 1 evaluates to 2, not 0):

```
Operators     Additional Information
-------------------------------------------
++ --         Math: Increment, Decrement
! Not         Boolean: NOT
+ -           Math: Unary Plus, Unary Minus (operator is prefixed to a number)
* / *= /=     Math: Multiply, Divide
%             Math: Modulo
\ \=          Math: Integer division
+ - += -=     Math: Add, Subtract
. .=          String concatenator
< <= > >= == !=             Comparison
Like LikeI UnLike UnLikeI   Comparison
&&            Boolean: AND
||            Boolean: OR
And           Boolean: AND
Xor           Boolean: XOR
Or            Boolean: OR
? :           Ternary
=             Set
```

Remote Control

*** For software developers only ***

You can run an XYplorer script from an external program using the WM_COPYDATA command with XYplorer's hWnd. This means if you are a programmer you can fully remote control XYplorer.

- cds.dwData: 4194305 (0x00400001)
- cds.lpData: The syntax is identical to the one of the command line switch /script=<script resource>, so you can either pass the path to a script file (commonly called *.xys), or pass the script directly (must

be preceded by ::).

User-Defined Functions

Like other aspects of XYplorer scripting also user-defined functions are closely modeled after PHP.

A user-defined function (or simply "user function") is a block of statements that can be used repeatedly in a program. It can return a value (using the command "return"). The function declaration starts with the word "function", then a space, then the name of the function:

General form:

```
function functionName() {
  code to be executed;
}
```

Alternative formatting:

```
function functionName() { code to be executed; }
```

Example with 2 arguments and a return value:

```
function sum($x, $y) {
  $z = $x + $y;
  return $z;
}
```

- User function names can only consist of letters (a-z, A-Z), numbers, and underscores, and must not start with a number.
- User function names are NOT case-sensitive.
- The first line of user function declarations can be left-bound or indented. It does not matter.
- User functions can be defined anywhere in a script resource, before or after the main script, and as many as you want.
- A user function will not execute immediately when a script is loaded. It has to be called just like native functions.
- All user functions have global scope.
- User functions overwrite native functions of the same name.
- If two or more user functions share the same name, the *first* one declared will be used, all other ones ignored.
- User functions do not support function overloading, nor is it possible to undefine or redefine previously-declared functions.
- User functions can call each other.
- It is possible to call recursive user functions, but be aware that too many recursions can smash the stack and cause a termination of the current script.

- Variables are by default passed by value, not by reference (so the function cannot modify the variables in the caller's scope).
- To pass variables by reference (so the function can modify the variables in the caller's scope) prefix them with & in the function declaration.
- All arguments are optional. Default values for missing arguments can be set. If no default value is set a missing argument is initialized to "".
- XYplorer native variables (e.g. <crlf>) and environment variables (e.g. %tmp%) are allowed in function argument default parameters. They are resolved if they are unquoted or double-quoted. They are NOT resolved if they are single-quoted.

Example 1: Arguments and Returns

```
echo multiply(3, 4) . <crlf> . sum(11,12);
function multiply($x, $y) { return $x * $y; }
function sum($x, $y) {
  $z = $x + $y;
  return $z;
}
```

Example 2: Argument by Value

```
$a = 1; add_one($a); echo $a; //1
function add_one($a) {
  $a++;
}
```

Example 3: Argument by Reference

```
$a = 1; add_one($a); echo $a; //2
function add_one(&$a) {
  $a++;
}
```

Example 4: Argument with Default Value

```
echo multiplyDefaults(2); //8 (2 * 4)
function multiplyDefaults($x = 3, $y = 4) {
  return $x * $y;
}
```

Include Statement

The "include" statement lets you include the content of a file at the place where the "include" statement was found. That way you can include e.g. function libraries into your script resources.

Note that the "include" statement is evaluated unconditionally when a script resource is loaded and before anything else is done.

Syntax

```
include file;
```

file: Path, if not fully stated, is resolved relative to <xyscripts>.

Extension defaults to ".xys".

Quotes are optional (they are totally function-less here, but might look better to some folks including me).

Note that you cannot use any script variables for file. However, XYplorer variables and environment variables are supported.

Remarks

You may end any include statement line with the usual instruction terminator ";". However, you must not append other instructions in the same line. The include statement needs to have its own line.

Include statements can be nested (included files can themselves include other files). The maximum nesting level for Include statements is 100 (one hundred). You will get an error if you go beyond. Beware of recursion: A file must not include itself or any file by which it has been included, else you will reach the maximum nesting level within the next millisecond...

Include statements can be indented. Included stuff will inherit the indent of the include statement.

Example (loaded script)

```
"Test 1"
  echo multiply(3, 4);
"Test 2"
  echo divide(10, 3);
include "math.inc"
```

Example (included library):

```
// math.inc
function sum($x, $y) { return $x + $y; }
function diff($x, $y) { return $x - $y; }
function multiply($x, $y) { return $x * $y; }
function divide($x, $y) { return $x / $y; }
```

Include_once Statement

The "include_once" statement is identical to the "include" statement, with the only difference being that if the code from a file has already been included, it will not be included again. There are no errors or messages. Trying to include_once an already included file will simply ignore the statement and continue.

Include Automatically

There is auto-include for scripts that are run directly from the address bar (and only from there). This way you can also use user functions from the address bar (which doesn't allow a proper include statement). It works like this:

- Create a file named "xy-autoinclude.xys" in path <xyscripts>.
- Fill it with the user functions you intend to use. For example:

```
function half($a) {return $a/2}
function sum($x, $y) {
  $z = $x + $y;
  return "$x + $y = $z";
}
```

- Now you can run these commands right from the address bar:

```
echo half(7); //returns 3.5
```

```
echo sum(172, 428); //returns "172 + 428 = 600"
```

Note: If "xy-autoinclude.xys" is not found when running a script from the address bar, XYplorer will not try again during that session. Saves speed, energy and material.

5.8 Scripting Commands Reference

A abs, age, ageclasses, aid, air, appicon, applyfolderview, asc, assert, attrstamp, automaxcolumn, autosizecolumns B backupto, base64decode, base64encode, beep, box, br, break, button, buttonset C catalogexecute, catalogload, cataloglocation, catalogreport, cea, ceil, ces, charview, chr, colorfilter, columnlayout, compare, conf, confirm, continue, controlatpos, controlposition, copier, copy, copyas, copydata, copyitem, copytext, copyto, count, ctbicon, ctbname, ctbstate D dark, datediff, datepicker, dectohex, delete, dlog, download E echo, editconf, end, eval, exists, exit, explode, extlist, extracttext, extratag F favs, filesequal, filesize, filetime, filetype, filter, flattenfolder, floor, focus, folderreport, foldersize, font, format, formatbytes, formatdate, formatlist, fresh, freshhere G get, getkey, getpathcomponent (gpc), getsectionlist, gettoken, gettokenindex, ghost, global, goto H hash, hashlist, hex, hexdump, hextodec, highlight, hotkeyshowapp, html, htmlencode I id3tag, id3v2tag, implode, incr, indexatpos, input, inputfile, inputfolder, inputselect, interfacecolors (ifc), internetflags, isset, isunicode, itematpos L lax, listfolder, listpane, llog, load, loadlayout, loadsearch, loadsettings, loadtree, logon M makecoffee, md5, moveas, moveto, mru, msg N new, newwindow, now O obfuscate, open, openwith, outputfile P paperfolder, pasteto, patchimage, pathreal, pathspecial, perm, popupcontextmenu, popupmainmenu, popupmenu, popupnativecontextmenu, popupnested, posatpos, preview, previewcheck, property Q quickfileview, quicksearch, quote R raf, rand, readfile, readonly, readonlyhere, readpv, readurl, readurlutf8, recase, refresh, refreshlist, regexmatches, regexreplace, releaseglobals, rename, renameitem, replace, replacelist, report, resolvepath, return, rotate, round, rtfm , run, runq, runret S savesettings, savethumb, searchtemplate, sel, selectitems, selectthumbs, self, selfilter, seltab, set, setcolumns, seticons, setkey, setlayout, setthumb, setting, settingp, shellopen, shellprops, showhash, showintree, showstatus, skipundo, slog, sortby, sortbylist, sound, status, statusbartemplate, step, strlen, strpos, strrepeat, strreverse, sub, substr, swapnames, sync, syncselect T tab, tabset, tag, tagcheck, tagexport, tagitems, taglist, tagload, tagsave, text, thumbscacherename, thumbsconf, timestamp, toolbar, topindex, trayballoon, trim U unset, unstep, update, urldecode, urlencode, utf8decode, utf8encode V varname, vartype, view W wait, wipe, writefile, writepv Z zip_add, zip_extract, zip_list2

abs()

Returns the absolute value of a number.

Syntax

```
abs(number)
```

number	Must be a number. A string will raise an error.
return	The absolute value of the number.

Examples

```
echo abs(1);  // 1
```

```
echo abs(0);  // 0
echo abs(-1); // 1
echo abs(-2.3456); // 2.3456
```

age()

Converts a number of milliseconds into a human-friendly age format.

Syntax

```
age(msecs, [maxfields=2], [sepfields=", "])
```

msecs	Number of milliseconds.
maxfields	Maximum number of fields. Defaults to 2.
sepfields	Separator between fields. Defaults to ", ".
return	Age string.

Remarks

- The last shown field is rounded.
- Months (and years to a lesser extent) are a tricky unit here because they have different lengths. This is solved by using an average length of a month of 30.436875 days (365.2425 / 12). Note that the results of this strategy are sometimes counterintuitive when the input is in the form of calendar dates:

```
echo age(datediff("2024-08-08", "2024-09-09", ms));  //1 month, 2 days
echo age(datediff("2024-09-08", "2024-10-09", ms));  //1 month, 1 day
```

Examples

```
echo age(); //0 secs
echo age(1000642);     //16 mins, 41 secs
echo age(1000642, 3);  //16 mins, 40 secs, 642 ms
echo age(1000000642);     //11 days, 14 hrs
echo age(1000000642, 5);  //11 days, 13 hrs, 46 mins, 40 secs, 642 ms
echo age(1000000642, 5, " & ");  //11 days & 13 hrs & 46 mins & 40 secs & 642 ms
echo age(2^81, 7); //76,618,668,343,801 years, 4 months, 20 days, 3 hrs, 42 mins, 39 secs, 808 ms
```

SC age() can be nicely combined with datediff() using the ms (milliseconds) interval:

```
echo age(datediff("2008-08-08", "2024-09-30" , ms)); //16 years, 2 months
```

ageclasses()

Sets or retrieves age class definitions.

Syntax

```
ageclasses([definitions], [separator="|"])
```

definitions	New definitions, separated by separator. Pass "" to reset to factory defaults. Pass nothing to just retrieve the current definitions.
separator	Separator of definitions.
return	Current definitions (using separator if passed).

Examples

```
ageclasses("d>79BB53//today|1 d>599B43//yesterday|11 d>9B5943//11 days ago");
echo ageclasses(); //show current definitions (separated by |)
text ageclasses(, <crlf>); //show current definitions (separated by CRLF)
```

aid

Controls whether dereferencing is allowed in interpolation (AID for AllowInterpolatedDereferencing).

Syntax

```
aid [allow=1]
```

allow	1: [default] allowed 0: not allowed

Note

Also affects dereferencing in HEREDOCs.

Example

```
 $v = "vampire"; $vd = '$v'; $c = "cow";
    echo *$vd . ", " . *$c . ", *$vd, *$c, $v, $c"; //vampire, cow, *$v, *cow,
vampire, cow
    aid;
    echo *$vd . ", " . *$c . ", *$vd, *$c, $v, $c"; //vampire, cow, vampire, cow,
vampire, cow
    aid 0;
    echo *$vd . ", " . *$c . ", *$vd, *$c, $v, $c"; //vampire, cow, *$v, *cow,
vampire, cow
```

air

Controls the density of the user interface of the main window. Identical to Configuration | Colors and Styles | Styles | Overall spacing.

Syntax

```
air [value=2]
```

value	Number of pixels to insert at various locations. Default is 2. Valid values range from -4 to 8. Invalid values are silently adjusted.

Example

```
air 8;
```

appicon

Lets you pick the app icon on the fly.

Syntax

```
appicon [selector="AAA"]
```

selector	AAA: [Default] Use the classic standard icon, aka BlackOrange. ABA: Use the red version of the 2022 icon, aka BlueBox. ABB: Use the blue version of the 2022 icon, aka BlueBox. other: Same as AAA. missing: Pop a dialog showing the choices.

Remarks

- You have a choice between icon resources that embedded in the executable. No external icons are possible.
- The choice is remembered across sessions.

Examples

```
appicon; //pop a dialog showing the choices
appicon "AAA"; //select the classic standard icon (black, white, orange)
appicon "ABA"; //select the 2022 icon, red box version
appicon "ABB"; //select the 2022 icon, blue box version
```

applyfolderview()

Applies any of the saved folder views to the current list.

Syntax

```
applyfolderview([path])
```

path	Path of the source folder view. If it does not look like a path it's checked for a partial match. If empty or missing then the default folder view is applied, if available.
return	0: No folder view found. 1: Folder view found and successfully applied.

Remarks

- You find a list of the available folder views here: *View* | *Folder View Settings* | *Manage Folder Views...*
- You could create a small collection of folders in a corner of your AppData folder, each with its own stored folder view. Then, you could use these folders as a source for quickly applying a specific, complex style to the current list. For example, you could use a custom toolbar button as a one-click solution for this purpose.

Examples

```
applyfolderview("E:\Test\FolderView2"); //matches exactly this path
```

```
applyfolderview("Jimmy"); //matches the first folder view that contains "Jimmy" in the path
```

asc()

Returns the Unicode codepoint of a character. This function complements chr().

Syntax

```
asc(string, [format])
```

string	character If string is larger than one character the first character is used. If string is empty asc() returns "" (empty).
format	[missing]: [Default] Return decimal value. H: Return raw hex value. X: Return prefixed hex value (eg 0x9F8D). U: Return Unicode Number (eg U+9F8D). B: Return hex values of bytes (Little Endian) instead of codepoint. Makes a difference only with surrogate pairs (codepoints beyond 16 bits). A: Enforce ANSI code page.
return	Unicode codepoint, range 0 to 1114111 (U+0000 to U+10FFFF).

Examples

```
echo asc("a"); //97
echo asc(chr(20000)); //20000
echo asc("龍");   //40845
echo asc("𝔗");    //120191
echo asc("𝔗", H); //1D57F
echo asc("𝔗", U); //U+1D57F
echo asc(" ", B); //DD7FD835
echo asc("€");    //8364
echo asc("€", X); //0x20AC
echo asc("€", A); //128 (might differ depending on locale)
echo asc(chr(0x1D57F), X); //0x1D57F
```

assert

Interrupts processing on certain conditions.

Syntax

```
assert condition, [message], [continue=0]
```

condition: [required] any expression that evaluates to True or False

message: [optional] message that's displayed if condition evaluates to False

continue: [optional] set to 1 to allow continuing the script after displaying the message; else the "Continue" button is disabled; default is 0

Usage

This is mainly a debugging feature to interrupt processing on certain conditions. If the condition evaluates to True the program flow is not interrupted, otherwise you are notified and can investigate what went wrong and decide whether to continue and cancel (if continue=1). So, "assert" can be very useful to keep your scripts under some control.

Apart from debugging you may use it to check (continue=1) or enforce (continue=0) critical preconditions for your scripts to run successfully.

Examples

```
$a="A"; assert $a=="A", '$a has wrong value!'
```

= no message (condition is True)

```
$a="A"; assert $a=="B", '$a has wrong value!', 1
```

= message (condition is False)

```
$a="A"; assert $a=="B", '$a has wrong value!', 0
```

= message (condition is False) and no option to continue

```
$count = get("CountSelected"); assert $count!=0, "You must select a file before running this script!"
```

= if no files are selected: message

attrstamp()

Sets the attributes for files or folders.

Syntax

```
attrstamp([attr], [mode=1], [itemlist])
```

attr:	a = FILE_ATTRIBUTE_ARCHIVE (32) h = FILE_ATTRIBUTE_HIDDEN (2) r = FILE_ATTRIBUTE_READONLY (1) s = FILE_ATTRIBUTE_SYSTEM (4) They can be combined in any order and any case, e.g. "raH".
mode:	0=get, 1=set [Default], 2=replace, 3=toggle, 4=remove.
itemlist:	CRLF- or |-separated list of items (full path) to attrstamp; if empty then the currently selected list items are attrstamped.
return:	Old attributes of the last processed item. The return value is made from summing up the attributes bit values, e.g. RAH = 1 + 32 + 2 = 35.

Remarks

CRLF- or |-separated list of items: If itemlist contains CRLF(s) then the list is parsed by CRLF. CRLF = Carriage Return Line Feed = End of Line ("EOL") character = 0x0D0A = \r\n. This means you can pass the items one per line. Otherwise it is parsed by pipe characters (|).

Examples

```
attrstamp("r");     //set readonly to selected items
attrstamp("rh");    //set readonly and hidden to selected items
attrstamp("r", 2);  //set readonly to selected items (any other existing attributes are removed)
attrstamp("r", 3);  //toggle readonly in selected items
attrstamp("R", 3);  //toggle readonly in selected items (same as above)
attrstamp("r", 4);  //remove readonly from selected items
attrstamp("r", , "E:\Test\x"); //set readonly to "E:\Test\x"
```

If no attributes are passed it just returns the old attributes of the last processed item:

```
text attrstamp();   //e.g. 5 = 1+4 (readonly and system)
```

automaxcolumn

Adjusts the width of a column so that it takes up all of the remaining space when all the other columns are visible without scrolling horizontally.

Syntax

```
automaxcolumn [column], [switches], [minwidth=64], [keepempty], [pane=a]
```

column	The column to adjust, identified by its canonic name or by its current display name (both case-insensitive). Defaults to "Name".
switches	a: Autosize all columns first.
minwidth	The column won't get smaller than this. Defaults to 64 pixels.
keepempty	Number of pixels to keep empty on the right.
pane	a: [default] active pane i: inactive pane 1: pane 1 2: pane 2

Remarks

- For normal columns the canonic name is identical to their English name. Referencing special columns is a bit harder. You can use this script to learn about the canonic names of the currently visible columns:
  `text setcolumns();`
- The command is only effective in views with columns ("Details" and "Details with Thumbnails").
- The absolute hard minimal width for columns is 19 pixels. You cannot go below that even if you try.

Examples

```
automaxcolumn;                  //max Name
automaxcolumn "name";           //max Name
automaxcolumn "comment";        //max Comment
automaxcolumn "name", a;        //autosize first, then max Name
automaxcolumn "name", , 175;     //max Name, keep at least a width of 175 pixels
automaxcolumn "name", , 175, 20; //max Name, keep at least a width of 175 pixels, keep empty 20 pixels
```

Note how you can nicely toggle the max width of two columns by alternating these lines:

```
automaxcolumn "name", a;      //autosize first, then max Name
automaxcolumn "comment", a;   //autosize first, then max Comment
automaxcolumn "Nombre"; //using the current display names works as well
```

autosizecolumns

Adjusts the width of one or more columns to their content.

Syntax

```
autosizecolumns columns, [pane=a]
```

columns	Comma-separated list of columns, identified by their canonic names or by their current display names (both case-insensitive).
pane	a: [default] active pane i: inactive pane 1: pane 1 2: pane 2

Remarks

- For normal columns the canonic name is identical to their English name. Referencing special columns is a bit harder. You can use this script to learn about the canonic names of the currently visible columns:
  `text setcolumns();`
- The command is only effective in views with columns ("Details" and "Details with Thumbnails").

Examples

```
autosizecolumns "name";
autosizecolumns "modified,size";
autosizecolumns ":s-image.dimensions";  //special properties column
autosizecolumns ":d-10";    //custom column
autosizecolumns "Extra 1,Extra 3,Extra 5";  //some extra columns
autosizecolumns "Nombre,Tamaño"; //using the current display names works as well
```

backupto

See moveto.

base64decode()

Decodes a Base64-encoded string or file.

Syntax

```
base64decode(text, [file], [targetfile], [utf8=1])
```

text	Text to decode. Ignored if "file" is not empty.
file	If not empty: file to decode. Path defaults to current path.
targetfile	If not empty: write decoded bytes to this file. Path defaults to current path. On exist file is overwritten without asking.
utf8	1: Convert from UTF8 after decoding. Defaults to 1.
return	Decoded string.

Remarks

This is the reverse function: base64encode().

Examples

```
text base64decode("Ki4/"); // *.?
```

```
text base64decode(, "Base64.txt"); //(input from file) *.? (if the file contains "Ki4/")
```

```
text base64decode("Ki4/", , "Text.txt"); //write output "*.?" to file
```

```
text hexdump(base64decode(<clipboard>, 3:=0)); //base64-decode the clipboard contents without intermediate UTF8-decoding, and show as hex dump
```

base64encode()

Encodes a string or file content in Base64.

Syntax

```
base64encode(text, [file], [targetfile], [utf8=1])
```

text	Text to encode. Ignored if "file" is not empty.
file	If not empty: file to encode. Path defaults to current path.
targetfile	If not empty: write decoded bytes to this file (full path spec). Path defaults to current path. On exist file is overwritten without asking. On exist file is overwritten without asking.
utf8	1: Convert to UTF8 before encoding. Defaults to 1.
return	Encoded string.

Remarks

This is the reverse function: base64decode().

Examples

```
text base64encode("*.?"); // Ki4/
```

```
text base64encode(, "Text.txt"); //(input from file) Ki4/ (if the file contains "*.?")
```

```
text base64encode("*.?", , "Base64.txt"); //write output "Ki4/" to file
```

beep

Generates a simple tone on the motherboard speaker.

Syntax

```
beep [frequency=800], [duration=200]
```

frequency	The frequency of the sound, in hertz. This parameter must be in the range 37 through 32,767. Defaults to 800.
duration	The duration of the sound, in milliseconds. Defaults to 200.

Example

```
beep 800,100; beep 400,100; beep 600,100;
```

Notes

The function is synchronous; it does not return control to its caller until the sound finishes.

box

Define the color for the Boxed Branch feature of the current location on Tree, or turn it off.

Syntax

```
box [color (rrggbb)], [folder]
```

color: The new box color in format RRGGBB. If empty or missing the current box color is removed.

folder: [optional] if stated then the highlighting is applied to that folder, else it is applied to the current folder.

Examples

```
box "00FFFF"
```

Will set the box color to cyan for the current location.

```
box
```

Will disable the Boxed Branch feature for the current location.

```
box "000000"
```

Will set the box color for the current location to the default Box color.

```
box "BBDDBB", "Desktop"
```

Set green box to Desktop folder.

br

Redefine the default line breaker to be used with scripting commands copytext, msg, and text.

Syntax

```
br linebreaker
```

linebreaker: The new line breaker.

Usage

The default line breaker is
 (as in HTML). If this does not suit you, use br to specify another line breaker.

Examples

```
br "#" ; msg "line 1#line 2";
```

Will use a one-character new line-breaker ("#")

```
br " # "; msg "line 1#still 1 # line 2";
```

Will use a three-character new line-breaker (" # ")

```
br;
```

Prevent any line breaking in commands copytext, msg, and text.

break

Ends execution of a control structure (while loop).

Syntax

```
break [levels=1]
```

levels	[optional] Numeric argument which specifies how many nested enclosing structures are to be broken out of. The argument ignores IF blocks.

Example

```
// script will only show "done"
  $i = 1;
```

```
while ($i <= 3) {
  while ($i <= 2) {
    break 2; //jump to echo "done";
    echo $i;
  }
  $i++;
}
echo "done";
```

button

Emulates a click on a toolbar button.

Syntax

```
button key, [action=1]
```

key [required] Key that identifies the button.
You can retrieve the existing keys using the SC toolbar, or using the Customize Toolbar command from a user button's context menu.

action [optional]

0 = do nothing

1 = left-click [default]

2 = right-click

8 = dropdown arrow

256 = customize toolbar dialog, preselect button "key"

Examples

```
button "mru";          //pops MRU menu at key
button "minitree";     //toggles MiniTree
button "minitree", 2; //pops ctx menu of MiniTree button
button "back", 8;      //pops dropdown menu of Back button
button "minitree", 256; //open customize toolbar dialog at MT
```

Usage

Note that using button you can trigger also buttons that are not part of the current toolbar! The command can become useful in the context menu definition of User Buttons. E.g., you can wrap a number of rarely used buttons into the context menu of a single button and thus save space on the toolbar. For example:

```
"Recent Locations|:mru" button "mru"
```

```
"Hotlist|:hotlist" button "hotlist"
-
"CKS|:cks" button "cks"
"UDC|:udc" button "udc"
-
"Exit no save|:exitnosave" button "exitnosave"
```

buttonset

Sets or gets the current toolbar button set.

Syntax

`buttonset([index])`

index	1-4: Index of the set to switch to. -1: Cycle the sets. Other values: No switch happens. Missing: No switch happens.
return	Index of the current set (before any switch).

Remark

You can switch to a set that is not reachable via UI, e.g. to set #4 while "Number of Button Sets" is set to 3. The power of scripting.

Examples

```
echo buttonset(); //return current set index
buttonset(-1); //cycle the sets
buttonset(4); //switch to set 4
```

catalogexecute

Executes a Catalog item.

Syntax

`catalogexecute [index=-1]`

index	Index of the Catalog item to execute. Defaults to the currently selected item.

How to find out the index of a Catalog item:

- It's shown in the caption of the Catalog item's Properties dialog.

- It's shown in the tooltip of the item's icon when you hold down CTRL.

Examples

```
catalogexecute;
```

```
catalogexecute 375;
```

Remarks

- The item is executed as if you press Enter on the item. So, if it's a Category its expansion state will be toggled.
- If you pass an invalid index, you get an error message "Invalid parameter".

Warnings

- This command enables you to bind a keyboard shortcut to a specific catalog item, or to the current one. But be careful with the latter one: It's a bit risky to have a keyboard shortcut trigger whatever happens to be the currently selected item.
- And if you specify an item by its index and then switch to a different catalog, a completely different item is likely to be referenced by that index.

catalogload

Loads a Catalog file.

Syntax

```
catalogload file, [switches=sp]
```

file	The Catalog file to load. Can be relative to the <xycatalogs> folder (by default it is <xydata>\Catalogs). Also portable paths ("?:\") are supported. If empty the default Catalog (<xycatalogs>\catalog.dat) is (re)loaded.
switches	(can be concatenated in any order): s: Save any changes in the current Catalog. p: Make [file] the permanent Catalog, i.e. load this Catalog on next app start. n: Create a new empty Catalog named [file]. i: Include (temporarily merge) an external catalog in the current one. e: Expand an included catalog. u: Unload an included catalog. [missing]: Defaults to "sp". Note: If i or u is present all other switches are ignored.

Examples

```
//save the current Catalog, load <xydata>\catalog2.dat, and make it the new permanent
```

```
Catalog
catalogload "catalog2.dat";
```

```
//do not save the current Catalog, load <xydata>\catalog2.dat, do no make it
permanent
catalogload "catalog2.dat", "";
```

```
//save the current Catalog, create a new empty Catalog named catalognew.dat in
<xydata>, and make it the new permanent Catalog
catalogload "catalognew.dat", "spn";
```

```
//discard the current Catalog and reload the default Catalog <xydata>\catalog.dat,
and make it permanent
catalogload , "p";
```

```
//include a catalog, and unload it again
catalogload "cat_include.dat", "i";
catalogload "cat_include.dat", "u";
```

cataloglocation()

Gets or sets the location of a catalog item.

Syntax

`cataloglocation([index=-1], [location])`

index	The index of the catalog item for which to set/get the location. Defaults to the currently selected item.
location	New location to set. If omitted, the current location is not touched.
return	Location of currently selected item.

How to find out the index of a catalog item

- It's shown in the caption of the catalog item's properties dialog.
- It's shown in the tooltip of the item's icon when you hold down CTRL.

Examples

```
echo cataloglocation(712);  //get location of #712
cataloglocation(712, "X:\");  //set location of #712
echo cataloglocation();  //get location of currently selected item
cataloglocation(, "X:\");  //set location of currently selected item
```

catalogreport()

Creates a report on the Catalog.

Syntax

```
catalogreport(template_category, template_item, [currentcategory=0])
```

template_category Template for categories

The following variables are supported:

{Caption} = Caption field.

{Index} = Item index (a fixed numeric ID, independent of the position).

{Location} = Location field ("Description" in Categories)

{RGB Text} = Text color in RRGGBB format.

{RGB Back} = Background color in RRGGBB format.

template_item Template for items.

Variables see template_category above.

currentcategory

0: [Default] Report on whole catalog.

1: Report on current category.

return The report.

Remarks

- The variables are case-sensitive: "{caption}" won't work.
- If a field is empty the variable is left unresolved.
- If "Apply Colors" in Catalog Properties is OFF then the color variables are left unresolved.
- Multi-line data in location fields (i.e. scripts) are folded into one line. The CR+LF sequences are replaced by PILCROW+TAB.

Examples

```
text catalogreport("{Caption}","- {Caption}: {Location}");
```

```
text catalogreport("Category: {Caption}", "  {Caption}, {Location}, {RGB Text}, {RGB Back}");
```

```
text catalogreport("Category: {Caption}", "  #{Index}: {Caption}, {Location}, {RGB Text}, {RGB Back}");
```

cea()

Sets or gets actions and scripts of the various Custom Event Actions (CEA).

Syntax

```
cea(index, [action], [script])
```

index Index of the CEA.

0 = CEA_TreeDoubleClickOnWhite

1 = CEA_ListDoubleClickOnWhite

2 = CEA_TabDoubleClickOnWhite

3 = CEA_CrumbDoubleClickOnWhite

4 = CEA_TreeMiddleClickOnWhite

5 = CEA_ListMiddleClickOnWhite

6 = CEA_TreeRightClickOnWhite

7 = CEA_ListRightClickOnWhite

8 = CEA_DoubleClickOnTab

9 = CEA_MiddleClickOnTab

10 = CEA_DoubleClickOnStatus

11 = CEA_LeftClickOnStatus

12 = CEA_MiddleClickOnStatus

13 = CEA_MiddleClickOnFolder

14 = CEA_MiddleClickOnFile

15 = CEA_ABMiddleClickIcon

16 = CEA_ABRightClickIcon

17 = CEA_MouseButtonBack

18 = CEA_MouseButtonForward

19 = CEA_BrowseBefore

20 = CEA_BrowseAfter

21 = CEA_BrowseAfterPaint

22 = CEA_AfterFileOperation

23 = CEA_LineNumD

24 = CEA_LineNumL

25 = CEA_LineNumM

26 = CEA_LineNumR

27 = CEA_ListLeftClickOnWhite

28 = CEA_DriveAddedRemoved

29 = CEA_TabMiddleClickOnWhite

30 = CEA_CrumbMiddleClickOnWhite

31 = CEA_LineNumHeaderD

32 = CEA_SwitchTabs

33 = CEA_SwitchPanes

34 = CEA_Exit

35 = CEA_Start

36 = CEA_RightClickOnTab

action	Index of the action to set. First index is 0. Missing: Action remains unchanged. -1: Return the current action index. -2: Return the current script.
script	Script to set. Standard line feeds (CRLF) are supported. Missing: Script remains unchanged.
return	The current action index or script (see action parameter).

Examples

```
cea(32, 0); //set Switch Tabs action to "None"
cea(32, 2); //set Switch Tabs action to "Run Script"
cea(32, , 'echo <curpath>;'); //set Switch Tabs script to "echo <curpath>;"
echo cea(32, -1); //get Switch Tabs action index
echo cea(32, -2); //get Switch Tabs script
```

ceil()

Rounds up fractions.

Syntax

```
ceil(number)
```

number	Number to ceil.

Examples

```
echo ceil(3.2);   //4
echo ceil(-3.2);  //-3
echo ceil(3);     //3
```

ces()

Defines a sound that will be played on certain events. CES stands for Custom Event Sound.

Syntax

```
ces(event, [soundfile], [threshold])
```

event	Event that triggers the sound. Currently the following events are supported (lower case or upper case both work): FOF: Foreground copy/move done. FOFE: Foreground copy/move done (errors). FOB: Background copy/move done. FOBE: Background copy/move done (errors). FOQ: Background copy/move queue done. FOQE: Background copy/move queue done (errors). NEW: New item created. REN: Item(s) renamed. DEL: Item(s) deleted. FIND: Search finished. Not played on switching to a Branch View, nor on switching toward a search results tab. FOP: File operation prompt. CLP: Clipboard update. Played when the clipboard is updated while XYplorer is the top window. Missing: Show "Edit Event Sounds" dialog, a GUI to manage all event sounds. Tip: The defined sound is played when you double-click the Event cell.
soundfile	Full or relative path to an audio file. WAV is recommended because it decodes the fastest, but all playable audio formats are supported (playable by Quartz.dll aka DirectX) . Path defaults to the Sounds folder in app data path (<xydata>\Sounds). *1: Use internal sound #1 (embedded in the executable). *2: Use internal sound #2, etc. Currently it goes up to *5. *[EventLabel]: Play a system event sound, specified by an event label, eg SystemAsterisk. Empty: Don't play anything at this event. Missing: Setting remains unchanged.
threshold	Number of seconds the operation has to take at least in order to trigger the sound. If the operation takes less no sound is played. 0: Always play the sound regardless of the duration of the operation. -1: Don't play anything at this event. Missing: Setting remains unchanged.
return	Current "[event key]: [soundfile] | [threshold]", e.g. FOFE: Cowbell.wav | 0.

Remarks

- Event Sounds are only played if Configuration | General | Controls & More | Miscellaneous | Play a sound on certain events is enabled.
- The sound is played after you set it, or when you just show the definition.

- The volume is controlled by the Windows "System Sounds" volume slider.
- There is feedback in the status bar, e.g. FOFE: Cowbell.wav | 0.
- Event labels can be found in your registry here: HKEY_CURRENT_USER\AppEvents\EventLabels\

Examples

```
ces("fof", "Blurp.wav", 0);  //always play "<xydata>/Blurp.wav" after a foreground job
ces("fof", "");   //remove the soundfile setting: don't play anything at this event
ces("fof", , -1); //keep the soundfile setting but don't play it
ces("fof", , 60); //keep the soundfile, set the threshold to 60 seconds
echo ces("fof"); //show the current settings, and play the sound
ces(); //show "Edit Event Sounds" dialog
ces("REN", "*SystemAsterisk");  //play "SystemAsterisk" after a rename
```

Stopping a sound

To stop any playing Custom Event Sound (in case it's a long one), you can use this command:

```
sound ""; //stop any sound
```

Also any new Custom Event Sound will stop the previous one.

charview

Shows the characters of a string in a vertical list in various encodings and notations.

Syntax

```
charview string
```

string: The string to view the characters of.

Examples

```
charview "abcá";
charview <clipboard>; // show string in clipboard
```

The first example pops this table:

#	dec	hex	utf8	char
1	97	61	a	a
2	98	62	b	b
3	99	63	c	c
4	225	E1	Ã¡	á

chr()

Returns a character specified by its codepoint. This function complements asc().

Syntax

```
chr(codepoint)
```

codepoint: Valid range 0 to 1114111 (U+0000 to U+10FFFF).

return: Character.

Examples

```
echo chr(88).chr(89); //XY
echo chr(28888); //烘
echo chr(0x1D57F); //  = Mathematical Bold Fraktur Capital T
echo chr("U+1D57F"); //  = Mathematical Bold Fraktur Capital T
echo chr(asc(" ")); //
```

colorfilter()

Activates one or more color filters in Tree and List.

Syntax

```
colorfilter([filters], [separator=||])
```

filters:	Filter, or list of filters separated by separator. If empty ("") any color filters are deactivated. If missing the function just returns the currently active filters.
separator:	Separator for list of filters.
return:	Currently active filters.

Examples

```
colorfilter("+lent:>12 //long filenames>BF6000,");
colorfilter("+len:>260 //overlong items>FFFFFF,FB4F04");
colorfilter("+dateM: dw 6-7 //last modified on a weekend>FFFFFF,B83E30");
colorfilter("+ageC d:d //folders created today>EEEEEE,4A97EC||+dir:xy*>FFFF00,FF8000");
colorfilter("prop:#AspectRatio: 4:3>FFFFFF,FB4F04");
colorfilter(""); //deactivate any color filter
text colorfilter(), 800; //return the currently active filters
```

Remarks and Usage

The filters applied via colorfilter() are functionally identical to Instant Color Filters. Details see Instant Color Filters.

columnlayout()

Loads or saves a column layout.

Syntax

```
columnlayout(file, [mode="load"])
```

file	File to save the column layout to, or to load it from. Either full path or resolved relative to <xydata>\Columns\. If missing you are prompted by a standard Windows dialog. If no extension is passed the extension defaults to ".txt".
mode	[optional] load = [Default] Load column layout from file. save = Save column layout to file. get = Just return the current column layout.
return	The column layout loaded or saved.

Examples

```
columnlayout("default"); //loads columns from <xydata>\Columns\Default.txt
columnlayout("default", "save"); //saves columns to <xydata>\Columns\Default.txt
columnlayout(); //loads columns, file is prompted
columnlayout(, "save"); //saves columns, file is prompted
text columnlayout(, "get");
```

compare()

Compares two strings.

Syntax

```
compare(string1, string2, [method=b])
```

method:

b:	[default] binary, bytes: strings are compared alphabetically
i:	same as b, but case-insensitive: A=a
n:	numeric: strings are converted to numbers (fractions use dot) and then compared numerically
v:	version: compares file versions of formats like #.## or #.##.#### or #.##.##.##
d:	date: compares dates, optionally including times; both string1 and string2 default to the current date/time.

Valid date syntax depends on your locale. Fractions of a second (up to 7 decimal digits) are supported.

return:

-1	if string1 < string2
0	if string1 == string2
1	if string1 > string2

Examples

```
echo compare("a", "b"); // -1
echo compare("a", "A"); // 1 (a is sorted after A)
echo compare("a", "A", "i"); // 0
echo compare("2", "12", "b"); // 1
echo compare("2", "12", "n"); // -1
echo compare("10.20.0025", "7.60.0026", "b"); // -1
echo compare("10.20.0025", "7.60.0026", "v"); // 1
echo compare("24.08.2012", "25.08.2012", "d"); //-1
echo compare("24.08.2012", "24.08.2012", "d"); //0
echo compare("25.08.2012", "24.08.2012", "d"); //1
echo compare("24.08.2012 08:43:01", "24.08.2012 08:43:02", "d"); //-1
echo compare("24.08.2012 08:43:02", "24.08.2012 08:43:01", "d"); //1
echo compare("24.08.2012 08:43:01.000", "24.08.2012 08:43:01.001", "d"); //-1
```

conf()

Sets or gets certain settings otherwise only accessible through the configuration dialog (F9).

Syntax

```
conf([setting], [value1], [value2])
```

setting	Setting to configure. Currently implemented (case-insensitive A==a): MiddleTruncation: = Configuration | Colors and Styles | Styles | Columns | Truncate filenames in the middle PreviewVideosAsThumbnail: = Configuration | Preview | Preview | Video preview | Preview as thumbnail ShowIconOverlays: = Configuration | General | Refresh, Icons, History | Icons | Show icon overlays SortHeadersAlways: = Configuration | General | Sort and Rename | Sort | Show sort headers in all views ThumbsOverlaySpecs: (not in the configuration dialog) ThumbsOverlayContent: (not in the configuration dialog)

value1	1st value. empty: Toggle values 0/1 (if it's an on/off setting). missing: Only return the current value.
value2	2nd value, used for toggling value1/value2.
return	Current value.

Examples for MiddleTruncation etc

```
echo conf("middletruncation");  //get (value is also shown in status bar)
conf("middletruncation");       //get (value is shown in status bar)
conf("middletruncation", 1);    //set
conf("middletruncation", 0);    //unset
conf("middletruncation", 0, 1); //toggle 0/1
conf("middletruncation", "");   //toggle 0/1, alt syntax
```

The others (PreviewVideosAsThumbnail, SortHeadersAlways) work the same way.

ThumbsOverlaySpecs

These are various custom properties of the thumbnail overlay that's shown when Configuration | Preview | Thumbnails | Show dimensions of original is ticked.

Syntax: `TextColor (RRGGBB),BackColor (RRGGBB),Opaqueness (0-255),ThousandSeparator (0/1), WhichImageSize(0=original,1=embedded,2=both),FontSizePercentage (50-200)`

Examples

```
echo conf("ThumbsOverlaySpecs"); //get current value
conf("ThumbsOverlaySpecs", "FFFF00,FF0000,128,1"); //yellow on semitransparent red with ThousandSeparator
conf("ThumbsOverlaySpecs", ""); //back to default (white on semitransparent black)
```

Note that you can set certain values while leaving the others untouched:

```
conf("ThumbsOverlaySpecs", ",,,,,120"); //bigger text (120%)
```

Remark: Just make sure that Configuration | Preview | Thumbnails | Show dimensions of original is checked. Then paste the above examples into the address bar and press ENTER. The thumbnail list will immediately update to reflect the changes.

ThumbsOverlayContent

Here you can define a custom content for the thumbnail overlay that's shown when Configuration | Preview | Thumbnails | Show dimensions of original is ticked. By default the dimensions of the original are shown, but you can show anything you like, and all XYplorer native variables are supported in your definition. Note that you can use `<crlf>` for line breaks.

You can specify a special section for video files. If the section marker `|video:|` is present in the definition, then everything after it is used for videos, everything before it is used for images.

Examples

```
echo conf("ThumbsOverlayContent"); //get current value
conf("ThumbsOverlayContent", '<prop #image.datetaken>'); //date taken
conf("ThumbsOverlayContent", ""); //back to default (i.e. show dimensions of original)
```

Note the use of single and double quotes. Outer single quotes protect the variable from being resolved before it is passed to SC conf. Now double quotes can and must be used inside to protect the space after ©:

```
conf("ThumbsOverlayContent", '<prop #image.datetaken "© ">');
```

A luxurious three-liner for photographers:

```
conf("ThumbsOverlayContent", '<prop #image.dimensions><crlf><prop #image.focallength> <prop #image.fstop> <prop #image.exposuretime> <prop #image.isospeed><crlf><prop #image.datetaken "© ">');
```

Finally and example with a special section for video files:

```
conf("ThumbsOverlayContent", '<prop #AspectRatio>|video:|<prop System.Video.TotalBitrate "Bitrate: *kbps">');
```

Remark: Just make sure that Configuration | Preview | Thumbnails | Show dimensions of original is checked. Then paste the above examples into the address bar and press ENTER. The thumbnail list will immediately update to reflect the changes.

confirm()

Pops message box with specified buttons, returns an integer identifying pressed button.

Syntax

```
confirm(text, [linebreaker="<br>"], [default_button=1], [buttons_icons=1])
```

text — message

linebreaker — [optional] Any character sequence to be replaced by a line break. Defaults to "<br>". Pass "" to prevent any replacing.

default_button — Index of initially focused button, counting from left.
1: [Default] First button
2: Second button
and so on.

buttons_icons (bit field)
Buttons:
1: [Default] OK/Cancel; returns 1/0
2: Abort/Retry/Ignore; returns 3/4/5
3: Yes/No/Cancel; returns 1/0/2
4: Yes/No; returns 1/0
Icons:
16: Critical
32: [Default] Question

48: Exclamation (sic: 16 + 32, MS invented that)
64: Information

return 1 on OK, 0 on Cancel; exceptions see buttons_icons above.

Examples

```
msg confirm("Ok?");
msg "You pressed " . (confirm("Ok?")==1?"OK":"Cancel") . ".";
msg confirm("Ok?") . confirm("Are you sure?");
echo confirm("Should#I#stay#or#should#I#go#?", "#");
echo confirm("Should#I#stay#or#should#I#go#?", "#", 2);
echo confirm("Ok?", , , 4);
echo confirm("Click Yes if you like this dialog.", , , 4+64); //information icon
```

continue

Skips the rest of the current control structure (foreach loop or while loop) and continues execution at the condition evaluation.

Syntax

```
continue [levels=1]
```

levels [optional] Numeric argument which specifies how many levels of enclosing loops it should skip to the end of.
The argument ignores IF blocks. Only foreach loops and while loops count as levels.

Example

```
// script will only show "done"
  $i = 1;
  while ($i <= 3) {
    $i++;
    continue;
    echo $i;
  }
  echo "done";
```

controlatpos()

Returns the control at a certain screen position.

Syntax

```
controlatpos([x], [y], [flags])
```

x	(optional) Defaults to the X mouse position on screen.
y	(optional) Defaults to the Y mouse position on screen.
flags	
	0: [Default] If x and y are passed, they are the position on screen.
	1: If x and y are passed, they are the position on XYplorer.
return	Control Short Form (see table here below).

Returns per Control

Control	Control Short Form	
List 1	L 1	
	ScrollX L 1	
	ScrollY L 1	
List 2	L 2	
	ScrollX L 2	
	ScrollY L 2	
Tree	T	
	ScrollX T	
	ScrollY T	
Catalog	C	
	ScrollY C	
Address Bar	A	
Toolbar	TB|<button index>	The first button has index 1. Separators count as buttons.
Tab Bar 1	TAB 1|<tab index>	The first tab has index 1.
Tab Bar 2	TAB 2|<tab index>	The first tab has index 1.
Breadcrumb 1	BC 1	
Breadcrumb 2	BC 2	
Status Bar	SB	
Live Filter Box	LFB	
Preview Pane	PP	

Examples

```
echo controlatpos(); //control at current mouse position on screen
echo controlatpos(592, 662); //control at arbitrary mouse position on screen
echo controlatpos(73, 108, 1); //control at arbitrary mouse position on XYplorer
```

controlposition()

Returns the position and dimensions of a control.

Syntax

```
controlposition(control, [flags])
```

control — The control to investigate.
Use same Control Short Form as returned by ControlAtPos.
Additionally:

List Active	L

It's case-insensitive (A==a).

flags

0: [Default] Position is relative to screen.

1: Position is relative to XYplorer.

return — A string of format "X|Y|Width|Height" (no quotes; values in pixels).
If the control is invisible then nothing is returned.

Remarks

An additional "control" is supported: "XY" returns the position and dimensions of XYplorer itself.

Examples

```
echo controlposition("T");  //position and dimensions of the Tree, relative to screen
echo controlposition("T", 1);  //position and dimensions of the Tree, relative to XY
echo controlposition("XY");  //position and dimensions of XYplorer, relative to screen
```

copier()

Defines an external program to handle copy and move operations.

Syntax

```
copier([definition])
```

definition: Defines which program to use and how.
General syntax: `Caption|Executable|Switches`
Details see External Copy Handlers.
Set to "" to use XYplorer's Custom Copy.

return: the current definition.

Examples

```
copier("FastCopy (AutoClose)||/auto_close"); //set copy handler
```

```
copier(""); //back to XYplorer's Custom Copy
echo copier(); //show current definition (don't change anything)
$old = copier(""); //store current definition, then reset it to ""
```

Notes

Custom Copy has to be enabled to actually use the external copy handler. Else Windows Shell copy is used.

copy

Copies item(s) to the clipboard.

Syntax

```
copy itemlist, [effect=copy|cut], [append]
```

itemlist	CRLF- or |-separated list of items (files or folders) items support environment variables, XYplorer native variables, relative (to application path) and portable paths.
effect	Lets you specify whether the items are marked as copied or cut, i.e. whether a subsequent paste operation will copy or move the items. copy: [Default] mark as copied cut: mark as cut
append	1 = Append *itemlist* to the items already in clipboard. The effect (copy or cut) of the items already in the clipboard is kept if you leave the "effect" argument empty.

Remarks

- The command will only copy those items that do exist/are available in this moment. Other items are silently ignored. If none of the passed items are available you get a message.
- No smartness whatsoever is applied when appending. So you can create crazy clipboard contents with items being present multiple times, containing each other, or from totally different locations. You are in control.

Usage

Allows you to copy any items to the clipboard without the need to go to their locations first to fetch them.

Also allows you to copy items from distributed locations in one go which is not possible by any other means.

Also can be used for converting textual clipboard contents to item clipboard contents, and also to change the marking of current clipboard contents from Copied to Cut and vice versa.

Examples

```
copy "E:\Test\Test.txt|E:\Test\a\alpha.png";
copy "D:\TestD.txt|E:\TestE.txt|F:\TestF.txt";
copy <clp>;         //copy (text or items in clipboard) as items, marked as copied
copy <clp>, "cut";  //copy (text or items in clipboard) as items, marked as cut
copy <curitem>, , 1;  //append current item to clipboard, keep effect
copy <curitem>, "copy" , 1; //append current item to clipboard, set effect to "copy"
copy <curitem>, "cut" , 1;  //append current item to clipboard, set effect to "cut"
```

copyas, moveas

Copies item(s) under a pattern-based name.

The moveas command works analog to the copyas command, with the only obvious difference that the items are moved instead of copied.

Syntax

```
copyas [pattern], [targetpath], [itemlist]
```

pattern: [optional] Name pattern where
* stands for the file base, and
? stands for the file extension (without dot).
Variables are allowed.
If empty then you get prompted.

targetpath: [optional] Path to copy the items to.
If empty then the current list path is used.
The path is silently auto-created if it does not exist.

itemlist: [optional] CRLF- or |-separated list of items (full path, or relative to current path) to copy.
If empty then the current list selections are copied.

Incremental suffixes are appended as needed to avoid collisions.

Examples

Copies all selected items to the current list path under the name ["Copy" + original Extension]:

```
copyas "Copy.?";
```

Creates copies named [original name]_backup_20100117.[original extension] if today is 17-Jan-2010:

```
copyas "*_backup_<date yyyymmdd>.?";
```

Same as doing menu File | Duplicate | Copy Here As...:

```
copyas;
```

Copies all selected items to D:\ under their original name:

```
copyas "*.?", "D:\";
```

Copies two specific files to D:\ under a pattern-based name:

```
copyas "*-copy.?", "D:\", "E:\Test\alpha.png|E:\Test\beta.png";
```

You can run a line like this to locally backup each item with its modified date in the target name. Note that you have to pass the argument in single quotes to make it work as expected -- otherwise the date variable is resolved to the modified date of the *current item* before the argument is passed to copyas:

```
::copyas '*_backup_<datem yyyymmdd>.?';
```

copydata

Sends data to another window.

Syntax

```
copydata hwnd, data, mode
```

hwnd: Handle of the target window.

data: Text data to send.

mode:

0: Nothing special, simply send the text data.

1: Text data is an XYplorer script to be executed by the receiving window (which in this case, of course, has to be XYplorer.exe).

2: Resolve variables in data and return to sender immediately. Variables are XYplorer native and environment variables.

3: Pass the value of the data argument as location to another XYplorer instance. Whatever XYplorer accepts in the Address Bar is acceptable here.

Examples

Run a small script in another XYplorer (197078):

```
copydata 197078, "::echo 'hi';", 1;
```

Return the contents of variable <curitem> from another XYplorer (197078) to this window (1573124), using copydata first in this XYplorer process and then again in the other XYplorer process for the return:

```
copydata 197078, '::copydata 1573124, <curitem>;', 1;
```

Determine <curitem> in another XYplorer instance (hWnd 197078). Note that the single quotes in the example are essential else <curitem> would be resolved in *this* instance of XYplorer before being sent to the other instance:

```
copydata 197078, '<curitem>', 2; echo <get copieddata 3>;
```

Go to "C:\" in the XYplorer instance with hWnd 525048:

```
copydata 525048, "C:\", 3;
```

Go to the current path of this instance:

```
copydata 525048, <curpath>, 3;
```

Run a script (note that the command only returns when the script is completed in the other instance!):

```
copydata 525048, 'echo "hi!";', 3;
```

Notes

- The command only returns when the receiving window has fully processed the data. For example if you send a script the command will return only after the script has terminated.
- The mode parameter in SC CopyData simply selects different dwData:
  If called with mode 0 then cds.dwData == 4194304 (0x00400000)
  If called with mode 1 then cds.dwData == 4194305 (0x00400001)
  If called with mode 2 then cds.dwData == 4194306 (0x00400002)
  If called with mode 3 then cds.dwData == 4194307 (0x00400003)
  So any application can use these dwData values to trigger a specific reaction in XYplorer when it receives data via WM_COPYDATA.

copyitem

Creates a copy of a file or folder (with all contents).

Syntax

```
copyitem [item], [copy]
```

item: [optional] Full path/name of the item to copy; if missing defaults to current item; can be relative to current path.

copy: [optional] Full path/name of the copy to create; if missing defaults to current item plus an automatic incremental suffix.
Can be relative to current path.
Can contain * to stand for the base of item.
Can contain ? to stand for the extension of item.

Examples

If the current item is "blah.txt" then this will create "blah-01.txt" in the current folder:

```
copyitem;
```

Duplicates "blah.txt" as "blub.txt" in another location:

```
copyitem "C:\Temp\blah.txt", "D:\blub.txt";
```

Will create folder "new\" if necessary:

```
copyitem "C:\Temp\blah.txt", "D:\new\blub.txt";
```

Duplicates "blah.txt" in the current path as "blah-copy.txt":

```
copyitem "C:\Temp\blah.txt", "*-copy.?";
```

Backup XY data folder to [e.g.] "...\appdata_backup_20090515":

```
copyitem "<xydata>", "<xydata>_backup_<date yyyymmdd>";
```

Duplicates folder "C:\Temp" with all its contents as "C:\Temp-01":

```
copyitem "C:\Temp";
```

Remarks

(1) The command uses the shell copy engine for the job. Hence you get the usual prompts about overwriting and no support for overlong filenames (> 260 characters).

(2) Any non-existing folders in the target are silently created without asking you politely for your approval.

(3) Note the difference to the command "CopyTo": In CopyTo the "location" argument is the destination where the source is copied to. In CopyItem the "copy" argument is the name of the copy itself.

(4) You cannot pass a drive, server, or share, as argument.

(5) Trailing slashes for folders are optional.

(6) The "copyitem" command partly overlaps functionally with the "new" command when you pass a source argument to the latter. However, "copyitem" is straighter to use for the job and has additional powers (e.g. using wildcards * and ?). Also, "new" will never overwrite an existing file or folder whereas "copyitem" will (unless you omit the "copy" argument).

(7) The command sets last target (used by menu View | Go to Last Target).

copytext

This command allows you to copy or append text to the clipboard.

Syntax

```
copytext text, [append], [linebreaker="<br>"]
```

text	The text to be copied/appended to the clipboard.
append	[optional]
[empty]:	Copy text to clipboard
a:	Append text to clipboard
linebreaker	[optional] Any character sequence to be replaced by a line break. Defaults to " ". Pass "" to prevent any replacing.

Usage

Simply call the command with whatever text you want to copy into our clipboard, or add parameter a to append it. You can also, of course, copy any variables to the clipboard, which are resolved before being added. For multiline text you can use the line breaker <br> which will be converted to a CRLF (0D0A).

You can change the default line breaker (<br>) by using command br first.

Examples

```
copytext "Hey, it's some other Galaxy!";
```

Copies Hey, it's some other Galaxy! to the clipboard.

```
copytext """Hey, it's some other Galaxy!""";
```

Copies "Hey, it's some other Galaxy!" (with quotes) to the clipboard.

```
copytext "The current XYplorer running is:"; copytext "<br><xypath>\<xyexe>", a;
```

Copies XYplorer is running from: to the clipboard, and appends XY's app path and EXE name on a new line to the clipboard.

In the end, the clipboard would contain something like this:

```
The current XYplorer running is:
C:\Program Files\XYplorer\XYplorer.exe
```

copyto

See moveto.

count()

Retrieves the number of elements of an array.

Syntax

```
count(variable)
```

variable	Array variable name, e.g. $a or $a[] (trailing square brackets are optional).
return	Count of elements. If variable is no array: -1. If variable does not exist: -2.

Examples

```
$a = array("Banana", "Cherry", "Apple"); $b = "b"; echo count($a); //3
$a = array("Banana", "Cherry", "Apple"); $b = "b"; echo count($b); //-1 (no array)
$a = array("Banana", "Cherry", "Apple"); $b = "b"; echo count($c); //-2 (no variable)
$a[5] ="cat"; echo count($a); //6 (they first 5 elements have been silently created)
```

ctbicon()

Set/Get the icon of a Custom Toolbar Button (CTB).

Syntax

```
ctbicon([icon], [button_index])
```

icon	Whatever can be entered into the Icon field in a CTB definition, e.g.: - Path to an icon resource (ICO, ICL, EXE, or DLL file). Full path or relative to <xyicons>. - Toolbar icon key of XYplorer's internal toolbar icons; e.g. ":home". If empty the function only returns the current icon. Optionally append a background color in RRGGBB format, separated by an asterisk. The color is drawn as a circle by default, but can be rectangular by appending an "r" after a comma. You can limit the drawing of the background to the dark mode by appending "d" separated by a comma.
button_index	Index of the button. If empty then the index of the button owning the script is used. If the script is called from outside a CTB you *have* to state a button index, else you will get an error message.
return	The current icon (before any new icon was set).

Examples

```
ctbicon("Apple.ico*#F6F6F6,r"); //icon, with background color in rectangular shape
```

```
ctbicon("Apple.ico*#F6F6F6,dr"); //icon, with background color in rectangular shape IF in dark mode
```

Use this script in the On Click event of a CTB; you need icons Apple.ico and Apple_Half.ico in <xyicons>:

```
// apple toggle
  if (ctbicon() == "Apple.ico") {
    ctbicon("Apple_Half.ico");
  } else {
    ctbicon("Apple.ico");
  }
```

Notes

- The new icon set via scripting is treated just like an icon set via UI; so it is as well remembered across sessions.
- You can even set icons of buttons that are not shown in the current toolbar.

ctbname()

Set/Get the name of a Custom Toolbar Button (CTB).

Syntax

```
ctbname([name], [button_index])
```

name	The new name. If empty the function only returns the current name.
button_index	Index of the button. If empty then the index of the button owning the script is used. If the script is called from outside a CTB you *have* to state a button index, else you will get an error message.
return	The current name (before any new name was set).

Example

```
text ctbname("New Name", 1); // set name of CTB #1, return old name
```

ctbstate()

Set/Get the pressed state of a Custom Toolbar Button (CTB).

Syntax

```
ctbstate([state], [button_index])
```

state	0 = unpressed 1 = pressed
button_index	Index of the button. If empty then the index of the button owning the script is used. If the script is called from outside a CTB you *have* to state a button index, else you will get an error message.
return	The current state (before any new state was set).

Example

Use this script in the On Click event of a CTB:

```
// state toggle
  if (ctbstate() == 1) {
    ctbstate(0);
  } else {
    ctbstate(1);
  }
```

Notes

- The state is remembered across sessions.
- You can even set the state of buttons that are not shown in the current toolbar.

dark

Toggles Dark Mode.

Syntax

```
dark [enable=-1], [darkness=-1], [darkcontrast=-1], [darkadaptive=-1], [darktint=-1]
```

enable	Turn Dark Mode on or off. -2 = toggle -1 [default] = keep current setting 0 = off 1 = on
darkness	Level of darkness. Valid values are 0 to 50 (factory default is 25). -1 [default] = keep current setting
darkcontrast	Text contrast in Dark Mode. Valid values are 0 to 30 (factory default is 15).
darkadaptive	Turn the Dark Adaptive mode on or off (see Remarks). -2 = toggle -1 [default] = keep current setting 0 = off 1 = on
darktint	Color tint of the darkness. Valid values are 0 to 360 (0 = no tint). -1 [default] = keep current setting

Remarks

The Dark Adaptive mode makes custom colors a bit less bright in Dark Mode and thus easier to the eye.

Examples

```
dark 1;          // enable dark mode
dark 0;          // disable dark mode
dark -2          // toggle dark mode
dark 1, 25, 15; // enable dark mode and set darkness level=25, text contrast=15
dark 3:=-2;      // toggle DarkAdaptive
dark 3:=0;       // set DarkAdaptive=0
dark 3:=1;       // set DarkAdaptive=1
dark 1, , , 1;  // enable dark mode and DarkAdaptive tweak
dark 1, 3:=1;   // same as above
```

```
dark 4:=220;    //blue tint
dark 4:=120;    //green tint
dark 4:=20;     //red tint
```

datediff()

Returns the number of time intervals between two dates.

Syntax

```
datediff(date1, [date2=now], [interval=d])
```

date1	[required] First (earlier) date/time.
date2	[optional] Second (later) date/time. Defaults to now.
interval	[optional] Time interval you want to use as the unit of difference between date1 and date2. Defaults to "d" (days).

y = years

m = months

w = weeks

d = days [default]

h = hours

n = minutes

s = seconds

ms = milliseconds

us = microseconds

ns = nanoseconds

return	Number of time intervals between date1 and date2. Negative if date1 is later than date2.

Remarks

The date expressions should be formatted either in the current locale, or in the universal ISO 8601 format (YYYY-MM-DD hh:nn:ss).

Examples

```
text "The Beijing Olympics were opened " .  datediff("2008-08-08") . " days ago.";
text "The current file is " . datediff("<datec>",,"h") . " hours old.";
text datediff("2008-08-08 14:22:50.222", "2008-08-08 14:22:50.224", "ms"); //2
```

datepicker()

Pops a Date Picker dialog and returns the selected date.

Syntax

```
datepicker([date=today], [format])
```

date	[optional] Initial date. Defaults to today.
format	[optional] Format of the returned date.
return	Selected date when OK was pressed.

Examples

```
text datepicker("2008-01-12"); //12.01.2008
text datepicker(); //01.08.2015 (in German locale)
text datepicker(, "yyyy-mm-dd"); //2015-08-01
text datepicker(, "yyyy-mm-dd hh:nn:ss"); //2015-08-01 00:00:00
```

dectohex(), hex()

Converts a signed decimal integer into hexadecimal.

Syntax

```
dectohex(integer, [rlen])
hex(integer, [rlen])
```

hex() is an alternative short name of the command

integer	The signed decimal number to convert. The highest allowed integer is 2147483647 (0x7fffffff), the lowest is -2147483648 (0x80000000).
rlen	Return this many rightmost characters from return value, if the value is smaller left-pad with zeros. Meaningful range 1 - 8. Other values are silently corrected.
return	The hexadecimal number, UPPERCASE and without any prefix.

Examples

```
text dectohex(-1); // FFFFFFFF
text dectohex(1234567890);  // 499602D2
text dectohex(1234567890, 4); // 02D2
text dectohex(15, 8); // 0000000F
text hex(15, 8);    // 0000000F
```

```
text hex(15, 8000); // 0000000F

text hex(15, 1);    // F

text hex(15, 0);    // F

text hex(15, -10);  // F
```

delete

Delete selected Tree or List item(s).

Syntax

```
delete [recycle=1], [confirm], [itemlist]
```

recycle

1 = [default] use Recycle Bin

0 = delete (no Recycle Bin)

confirm

1 = confirmation prompt

0 = no confirmation prompt

[empty] = [default] user setting in configuration

itemlist CRLF- or |-separated list of items (full path) to delete. The separator or the items may be surrounded by any number of blanks. Folders should not have a trailing slash. You can use wildcards (* ?) to delete multiple items at once. Any non-existing items passed in itemlist are silently ignored.

[empty] = [default] current Tree or List selections are deleted, depending on the focus.

:tree = delete current tree selection

:list = delete current list selection(s)

Remarks

- If the confirm argument is set to 0 then the setting of "Configuration | General | Safety Belts, Network | Safety Belts | Confirm delete operations" is ignored (there will be no prompt even if it is checked).
- In Paper Folders delete behaves like pressing DEL (see below Examples when in Paper Folder).

Examples

```
delete 1, 0, ":list"
```

Send all selected list items to the Recycle Bin, no questions.

```
delete 0, 1, ":tree"
```

Delete the selected tree folder (NO Recycle Bin), ask before.

`delete` or `delete 1`

Delete selected Tree or List items (depending on focus), use Recycle Bin, confirmation prompt depending on user settings.

```
delete 0, 0
```

Delete selected Tree or List items (depending on focus), NO Recycle Bin, NO questions.

```
delete 1, 1, "E:\TestFiles\Temp\droptest\resource.h | E:\TestFiles\Temp\droptest\MainFrm.h"
```

Delete those two files, use Recycle Bin, ask before.

```
delete 1, 1, "E:\Folder\*.tmp | E:\Folder\*.bak";
```

Delete all INI and BAK files in E:\Folder, use Recycle Bin, ask before.

Examples when in Paper Folder

When the List is a showing Paper Folder and "On Delete Remove Items from Paper Folder" is ON these will behave like pressing DEL now and remove the selected items from the Paper Folder, not from the file system:

```
delete 1, 1, ":list"; //with prompt
```

```
delete 1, 0, ":list"; //without prompt
```

If the List has the focus, it also work without the ":list" argument:

```
delete 1, 1; //with prompt
```

```
delete 1, 0; //without prompt
```

dlog

Pops a dialog showing the current debug log.

Syntax

```
dlog [switches]
```

switches r: clear the log (remove current contents)

Remarks

- It's basically a shorthand for: `text get("debuglog");` but with a little more information in the header.
- The time passed from the previous to the next log entry is shown first in the line. For example:

```
          375    = 375 milliseconds passed
        4.047    = 4 seconds and 47 milliseconds passed
     12:03.687   = 12 minutes, 3 seconds and 687 milliseconds passed
   1:42:41.595   = 1 hour, 42 minutes, 41 seconds and 595 milliseconds passed
```

- The debug log does not grow forever but auto-cycles at 1024 entries (= when a new entry is added the oldest one is dumped).

Examples

```
dlog; //show log
```

```
dlog r; //clear log
```

download

Download a file from the internet to a specified location.

Syntax

```
download url, [targetfile], [options]
```

url	[required] any URL (http://, https://, or ftp://)
targetfile	[optional] full path/file of desired target; UNC works as well Defaults: - if empty: [current path]\sourceurlfilename - if only file (or relative path): [current path]\targetfile - if only path: [targetfile]\sourceurlfilename - if sourceurl has no filename: "Download[current date].htm"
options	[optional] [empty] = On collision: ask before overwriting. o = On collision: overwrite without asking. i = On collision: auto-increment targetfile. b = Check byte count after download. On no-match the download fails. s = Silent: No status bar progress and success messages.

Examples

```
download "https://www.xyplorer.com/tour/images/pfa.png";
```

Download "pfa.png" to "[current path]\pfa.png".

```
download "https://www.xyplorer.com/tour/images/pfa.png", , i;
```

Download "pfa.png" to "[current path]\pfa.png". Add increment (e.g. pfa-01.png) if pfa.png already exists.

```
download "https://www.xyplorer.com/download/xyplorer_full.zip","xy.zip";
```

Download "xyplorer_full.zip" to "[current path]\xy.zip".

```
download "https://www.xyplorer.com/";
```

Download "index.htm" to "[current path]\Download-20080905.htm".

```
download "ftp://ftp.editplus.com/epp230hlp.zip", "D:\Download\";
```

Download "epp230hlp.zip" to "D:\Download\epp230hlp.zip".

Remarks

The application waits until the download is complete. You can abort the download at any time by pressing ESC.

The downloaded data size is not limited. Files are streamed onto disk (you can see them growing in the file list).

The download progress is shown in the status bar.

If there is no internet connection, your system might attempt to establish one (if you configured it like that) and then continue with the download.

If sourceurl is ftp:// then there is currently no test for existence of the remote file, nor can its size be determined. Nevertheless the download works (but you cannot know how long it will take).

Only anonymous FTP is supported.

After a successful download the Last Target Path is set (menu Go | Go To Last Target).

The format of [current date] is defined in Configuration | Templates under Filename affixes.

echo

Shows a simple message box.

Syntax

```
echo text, [topic], [subtopic]
```

text	Text.
topic	[optional] Topic. Defaults to "Echo".
subtopic	[optional] Subtopic.

Examples

```
echo "hi!";
```

```
echo <date>;
```

```
echo hash("sha384", <drop>, 1), "SHA-384", <drop>; //drop-on script for a bookmark button
```

Remarks

- The dialog features a Copy button, and Ctrl+C will copy the message as well.
- Any NULL characters in the displayed text are replaced by spaces.
- `e` works as an alias for `echo`.

editconf

Displays a configuration file as editable list of lines.

Syntax

```
editconf [flags], [filter], [file]
```

flags — (bit field)

1: Before reading the file, the current configuration (usually "XYplorer.ini") is saved.
That way you ensure that the file you are about to edit represents the current state of the application.

2: Before reading the file, all settings are saved (including the configuration file, so this includes flag 1).
That way you ensure that the file you are about to edit represents the current state of the application. And, if you later "restart without saving" the application is in the same state as now (apart from your edits).

4: After OK-ing the dialog automatically restart XYplorer without saving. This will apply the changes you made. Of course, this only makes sense if you have been editing "XYplorer.ini".

filter — Preset the filter.
Allows you to focus particular lines in the file right away.

file — Full path to the configuration file to edit.
Defaults to this XYplorer's main INI file (usually "XYplorer.ini").

Notes

- In Lines Mode you cannot add or remove any lines, but just modify the existing ones. The Edit button or key F2 will go into line-wise edit mode.
- You can switch to Editor Mode by clicking the "Editor Mode" button at the bottom left of the window, or by pressing key F6. Then you have all liberties and all responsibilities... good luck!
- If you OK the list the file is immediately rewritten with the new data without further questions.
- Note that if you are editing "XYplorer.ini" then saving the file alone will not change anything in the configuration you are currently seeing (which is all in memory, not on disk). To actually apply your changes you have to call "Restart without Saving". You can automate this by passing flag 4.
- Using this command you can actually edit any file in this line-by-line fashion. You have to know what you are doing, of course...
  WARNING: If you think editing binary files using this command is a good idea... no, it's not! Limit yourself to text files...
- Even if you don't plan to edit anything this new command serves as a nice file viewer since you can filter the list.

Examples

```
editconf;       //edit XYplorer.ini, no auto-restart
editconf 1;     //edit XYplorer.ini, save config
editconf 1 + 4; //edit XYplorer.ini, save config + auto-restart
editconf , , "<xypath>\Startup.ini"; //edit this file
editconf 4
editconf , "ZipPath7zip=*";   //filter and preselect this line in XYplorer.ini
```

```
editconf 5, "ZipPath7zip=*";  //save XYplorer.ini; filter and preselect this line; auto-restart
```

end

Terminates a running script.

Syntax

```
end condition, [message], [scope=0], [linebreaker=<br>]
```

condition	If True the script is ended else nothing happens (True = anything but 0 or empty)
message	Message displayed before ending.
scope	0 = [default] end whole script stack 1 = (or any other value) end this script
linebreaker	Any character sequence to be replaced by a line break. Defaults to . Pass "" to prevent any replacing.

Examples

```
end 1==2; msg "Still here!";  //"Still here!"
end 1==1; msg "Still here!";  //ends silently
end 1==1, "Bye!"; msg "Still here!";  //"Bye!", then ends
end confirm("Are you sure to continue?") == 0, "Bye!"; msg "Still here!";
end confirm("Are you sure to continue?") == 0, "Bye!<br><br>Yours, Don"; msg "Still here!";
```

eval()

Evaluates an expression.

Syntax

```
eval(expression)
```

expression	The expression to evaluate.
return	The evaluated expression.

Examples

```
$a = "1 + 1";
  echo $a;        //1 + 1
  echo eval($a);  //2
```

```
$a = '<xypath>';
  echo $a;        //<xypath>
  echo eval($a);  //C:\Progs\XYplorer
```

```
// Little calculator
$math = input("Little calculator", "Paste your math:", "1 + 1");
  echo "The result is: " . eval($math);
```

```
// Also functions are resolved in eval(). Result: This is the time: 19:11:53!
  $v = '" $x " . substr("exact time:", 6) . " <date hh:nn:ss>"';
  $x = "the";
  $s = eval($v); // will also evaluate the substr(), and concatenate it with the
rest!
  text "This is" . $s . "!";
```

exists()

Checks whether a file or directory or URL exists.

Syntax

```
exists(item)
```

item	Full path (local or UNC) to a file system item, or URL. Relative paths are resolved relative to the current path.
return	0 = does not exist 0|404 = URL returned 404 1 = exists (file; URL) 2 = exists (folder; drive; share; server)

Examples

```
echo exists("C:\autoexec.bat"); //1
echo exists("C:"); //2
echo exists("C:\"); //2
echo exists("C:\Windows"); //2
echo exists("C:\Windows\"); //2
echo exists("C:\*.bat"); //0 - wildcards are NOT recognized
echo exists("paper:foo"); //"2" if paper folder exists, "0" if not
```

With URLs it's either 1 or 0, no further distinction between files and folders:

```
echo exists("https://www.xyplorer.com/"); //1
echo exists("https://www.xyplorer.com/xyfc/"); //1
echo exists("https://www.xyplorer.com/xyfc/index.php"); //1
```

```
echo exists("https://www.xyplorer.com/nosuchpath/"); //0|404
```

Note on checking URLs

The function can take some seconds on first URL access.

Remarks

The API function used to determine the existence and nature of the item is inherently tolerant with trailing spaces, and it resolves relative syntax (\..\.. etc) before doing the check.

Some of those special cases:

```
echo exists("C:\autoexec.bat       "); //1
```

```
echo exists("C:\autoexec.bat\.."); //2 (drive C:\ exists)
```

exit

Exit XYplorer, optionally with restart.

Syntax

```
exit [mode]
```

mode	Exit mode. [empty]: exit (= #192) Saving depends on user setting in Configuration | Startup & Exit | Save settings on exit. s: exit with saving (no menu command) n: exit without saving (= #191) sr: restart with saving (no menu command) nr: restart without saving (= #190) sre: restart elevated with saving nre: restart elevated without saving

Notes

- When you cancel the elevation prompt the previous XYplorer instance is still closed.
- It is not possible to do the reverse (run an unelevated instance from an elevated one).

Examples

```
exit; //default exit, like clicking X-close
```

```
exit "nr"; //restart without saving
```

```
exit "sre"; //restart elevated with saving
```

explode()

Maps a list onto an array.

Syntax

```
explode($array, list, [separator="|"], [flags])
```

$array	Array variable name, e.g. $a or $a[] (trailing square brackets are optional).
list	List of items separated by separator.
separator	Separator of received list items. Defaults to |.
flags	e = skip empty
return	Number of items in the array.

Examples

```
explode($a, "a,b,c", ","); echo $a[0]; //a
explode($a, "a,,c", ",");      echo implode($a);        //a||c
explode($a, "a,,c", ",", "e"); echo implode($a);        //a|c
```

extlist()

Lets you customize certain extension lists.

Syntax

```
extlist(type, [extensionlist], [switches])
```

type	Which extension list to customize (upper/lower case is ignored): FolderThumbs: File types that can be used for folder thumbnails. HoverBoxSkip: File types for which no Hover Box should be displayed. NoThumb: File types for which no thumbnails are generated. ShowFolderSizesExclude: Paths that are to be excluded from folder size calculation.
extensionlist	List of extensions, dot-separated. ?: Just return the old extension list. Empty: Reset all extensions. Missing: Open list management dialog to customize the list in a GUI.
switches	a: Add extensionlist to the current list. r: Remove extensionlist from the current list.
return	Old extension list.

Examples

```
extlist("folderthumbs"); //open "Folder Thumbnails - File Extensions" dialog
```

```
echo extlist("folderthumbs", "?");  //show current Folder Thumbnails File Extensions
extlist("folderthumbs", "ai.eps.svg");  //set Folder Thumbnails to these extensions
extlist("HoverBoxSkip", "");  //reset any Hover Box Skip extensions
echo extlist("HoverBoxSkip", "?");  //show current Hover Box Skip file extensions
extlist("folderthumbs", "ai.eps.svg", "a");  //add these extensions to the Folder Thumbnails
extlist("folderthumbs", "ai.eps.svg", "r");  //remove these extensions from the Folder Thumbnails
extlist("NoThumb");  //open "No Thumbnails - File Extensions" dialog
extlist("showfoldersizesexclude"); //open "No Folder Size Calculation - Paths" dialog
```

extracttext()

Extracts pure text from complex files (e.g. DOC, DOCX, ODT, PDF).

Syntax

```
extracttext([file], [bitness], [flags])
```

file	File to extract text from. Defaults to the current file.
bitness	This argument is currently obsolete. Only 64-bit IFilters are used.
flags	(bit field) 1: Suppress IFilter errors (just return an empty string).
return	Extracted text.

Remarks

This function makes use of IFilters. It depends on your system which ones are available.

Examples

```
text extracttext("E:\Test\test.pdf");
text extracttext("E:\Test\test.doc");
text extracttext("E:\Test\test.htm");
text extracttext(<curitem>, , 1); //auto-bitness; suppress IFilter errors
```

extratag()

Sets or gets an Extra Tag definition.

Syntax

```
extratag([id], [definition])
```

id	Extra Tag index: 1 to 16. OR: Extra Tag field prefix: ex1 to ex16. OR: Current column caption. If missing: Return the definitions for all Extra Tags, one per line.
definition	Extra Tag definition. Format 1: Caption|Type|Prop|DefaultIndex|List[|Reserved|Reserved|Reserved|Reserved] This will change the whole definition. Format 2: Caption This will change just the column caption. If missing then only the current definition is returned.
return	The previous definition.

Examples

```
echo extratag(4); //get Extra Tag 4
extratag(4, "Colors|5|0|0|blue;red;yellow;green||||"); //set Extra Tag 4 to Type 5 (Pop-up List)
echo extratag(4, "Colors|5|0|0|blue;red;yellow;green||||"); //set/get Extra Tag 4
extratag(4, "Shades"); //set caption of Extra Tag 4
extratag("ex4", "Shades");    //set caption of Extra Tag 4 to Shades
extratag("Shades", "Sharks"); //set caption of Extra Tag "Shades" to "Sharks"
text extratag();  //return the definitions for all Extra Tags
```

Notes

Changes, e.g. new column captions, are immediately applied to the list when applicable.

favs()

Sets or retrieves a Favorites list.

Syntax

```
favs([type=d|f], [items], [separator=CRLF], [action=a|aa|p|pp|s|d])
```

type	Specifies which favorites list. d: [Default] Folders/Directories f: Files
items	Favorite items separated by separator. If missing the function just returns the current favorites.
separator	Separator for list of items. Defaults to CRLF (line feed).
action	Action to take with the items.

a: [Default] Append to current list.
No dupes: If item is already present then nothing happens.
aa: Append to current list. Dupes allowed.
p: Prepend to current list.
No dupes: If item is already present then nothing happens.
pp: Prepend to current list. Dupes allowed.
s: Set/Replace current list.
d: Delete from current list.

return — The list of current favorites for the specified type (before any change).

Examples

```
text favs('f'); // return the current list of favorite files
favs('d', <curpath>); // add current path to favorite folders
favs('d', <curpath>, , 'd'); // remove current path from favorite folders
favs('d', '', , 's'); //set favorite folders to nothing
favs('f', <focitem>, , 'p'); // add focused item to top of favorite files
paperfolder('Favorites', favs('f') . <crlf> . favs('d')); // create Paper Folder from favs
```

filesequal()

Compares two files for identity.

Syntax

```
filesequal([file1], [file2], [algo])
```

file1 — First file. Defaults to first selected file.

file2 — Second file. Defaults to first or second (if first file defaulted to first selected file) selected file.

algo — Hash algorithm; one of the following:
(empty) Compare files bytewise.
md5: MD5.
crc32: CRC-32.
sha1: SHA-1.
sha256: SHA-256.
sha512: SHA-512.

return — -1 = Function failed.
0 = Files are different.
1 = Files are identical.

Examples

```
echo filesequal();  //first two selected files, bytewise
echo filesequal("D:\a.txt", "D:\b.txt", "sha256");
```

filesize()

Retrieves the size of the specified file, in bytes.

Syntax

```
filesize([filename])
```

filename	Full path/name, or relative to current path. Defaults to the current file.
return	The file size in raw bytes.

Examples

```
echo filesize();
echo filesize("C:\WINDOWS\explorer.exe");
```

filetime()

Retrieves any of the three filetimes of the specified file.

Syntax

```
filetime([filename], [type="m"], [format="yyyy-mm-dd hh:nn:ss"])
```

filename	Full path/name, or relative to current path. Defaults to the current file.
type	m: [Default] Modified c: Created a: Accessed
format	Format of the returned date, defined by the usual letters (ymdhns, and w for week). Defaults to "yyyy-mm-dd hh:nn:ss". There is a special value "hex" which returns an 8 byte little endian hex value. This hex value provides the full file time resolution of NTFS down to 100 nanoseconds. It can be written by SC timestamp.
return	The file date/time in ISO 8601 format (yyyy-mm-dd hh:nn:ss).

Examples

```
echo filetime();          //modified date of current file, in ISO 8601, eg: 2023-10-16 14:42:22
echo filetime(, "c");     //created date of current file, in ISO 8601, eg: 2023-10-16 14:42:22
echo filetime(,, "yyyy"); //modified date of current file, year only, eg: 2023
echo filetime(,, "dddd"); //modified date of current file, as weekday, eg: Monday
```

```
echo filetime(,, "ww");   //modified date of current file, week number, eg: 42
echo filetime(,, "hex");  //modified date of current file, as 8 byte little endian hex value, eg: 01DA002E35164B00
echo filetime("C:\Windows\explorer.exe", "c"); //created date of C:\Windows\explorer.exe
```

filetype()

Analyzes a file by its contents.

Syntax

```
filetype(file)
```

file	full path/file name to analyze; defaults to the current file.

return

Ascii	It's an ASCII text file.
UTF-8	It's a UTF-8 file.
UTF-8 BOM	It's a UTF-8 BOM file.
UTF-16LE	It's a UTF-16LE file.
UTF-16BE	It's a UTF-16BE file.
Binary	It's a binary file.
Empty	It's an empty file.
Cannot	Cannot open file.
Nofile	It's not an existing file.

Example

```
echo filetype(<curitem>);
```

filter

Apply a visual filter or a live filter to the current tab, or a global visual filter to all tabs. Or remove an existing filter.

Syntax

```
filter [filterspec], [flags]
```

filterspec	The Visual Filter to apply to the List. If empty or missing any existing Visual Filter is removed. See Visual Filters for syntax.
flags	(bit field)

1: Handle open square brackets, i.e. convert "[" to "[[]" to allow matching "[". Without this conversion "[" and "]" are interpreted as group markers by the pattern parser. Note that setting this flag is necessary only if the closing square bracket is present as well right of the opening bracket. Otherwise there is no ambiguity and the necessary things are done automatically even without the flag.
2: Set the filter as Global Visual Filter.
If filterspec is missing: Toggle last GPF on/off.
4: Do not toggle on same filter (i.e. guarantee the filter is set regardless of the current state of the list).
8: Apply filterspec as live filter.

Examples

```
filter "*.txt"         // show only *.txt files
filter "*.zip;*.rar"; // show only ZIP and RAR files
filter "*[*";          // show only items with [
filter "*[ab]*", 0;    // show only items with "a" or "b"
filter "*[ab]*", 1;    // show only items with "[ab]"
filter;                // remove any filters (show all files)
```

Examples for Global Visual Filters

```
filter "*.txt", 2;          //globally show only TXT files
filter "ageM: m", 2;        //globally list only files modified this month
filter "ageM: <= 7 d", 2;   //globally list only items modified in the last 7 days
filter "ageM: <= 7 d", 6;   //set above GVF, don't toggle
filter , 2;                 //toggle last Global Visual Filter
filter "", 2;               //remove any Global Visual Filter
```

Examples for Live Filters:

```
filter "k*", 8;
filter "kyo", 8;
filter "size:0", 8;
filter "ageM:d", 8;
filter "", 8;               //remove any live filter
```

flattenfolder()

Flattens a folder. To flatten a folder means to move all files from its subfolders into the folder itself, and then delete the now empty subfolders.

Syntax

```
flattenfolder([folder], [flags], [separator_name="-"], [separator_return=CRLF])
```

folder	(optional) Folder to flatten. If missing then: If List has focus and a folder is focused AND selected in List, then this folder is flattened. Else: Defaults to the current folder.
flags	(in any order) n: No safety prompt. Nerves like steel. m: Return name mappings: old > new (else just new names are returned). p: Preview only: Just return names/mappings, no actual flattening. f: Prefix folder names to file names: Maps the folder structure to the filenames. Advantage: Avoids any collisions and keeps the information with the files. Disadvantage: Names can get long.
separator_name	Separates the folder names in the filename if flag "f" is passed. Defaults to "-".
separator_return	Separates the items in the return value. Defaults to CRLF.
return	List of contents of the folder after being flattened; items with full path, separated by separator. On flag "m", name mappings are returned (see above).

Remarks

- The function will auto-avoid filename collisions by affixing increments to the new name according to the user settings (Configuration | Templates | Filename Affixes).
- Undo/Redo is supported (in 2 steps: ; restore deleted empty folders; move files back into them).
- The tags database is updated for the moved items.
- Warning: This command can easily destroy a complex folder structure. Therefore a safety prompt has to be OK-ed before the rumble begins (unless you pass flag "n").

Examples

```
flattenfolder();    //flatten current folder, safety prompt
flattenfolder(, "n"); //same, but without prompt
text flattenfolder(); //flatten and show return
text flattenfolder(, "mp"); //just show return (name mappings), no flattening
// flatten this folder, prefix folder names separated by "+", show prompt:
flattenfolder("E:\Test\flatten-01", "f", "+");
```

floor()

Rounds fractions down.

Syntax

```
floor(number)
```

number Number to floor.

Examples

```
echo floor(3.7);  //3
echo floor(-3.7); //-4
echo floor(3);    //3
```

focus

Sets the focus to a specified control of the XYplorer main window, or to a specified window.

Syntax

```
focus [control (L|A|T|C|P1|P2|PI|LFB|PP|FP|XY)]
```

control: The control to focus

L: [Default] List

T: Tree

C: Catalog

A: Address Bar

P1: Pane 1 (left/top pane)

P2: Pane 2 (right/bottom pane)

PI: Inactive Pane

LFB: Live Filter Box

PP: Preview Pane or Preview Tab (wherever the preview currently happens)

FP: Floating Preview window

XY: XYplorer window

Remarks

- The command does NOT show a control if it's not visible already; in that case the focus won't move from its current position.
- PP does not do much apart from taking the focus away from wherever it is now because the actual preview is contained in a child of the preview pane which at the moment cannot be focused.

Examples

```
focus; //Set focus to the List
focus "L"; //Set focus to the List
focus "C"; //Set focus to the Catalog
```

```
focus "FP"; //focus the Floating Preview window (nothing happens if there is none)
```

folderreport()

Creates a report on a folder.

Syntax

```
folderreport([type=dump], [target=clipboard], [folder], [flags], [outputfile], [separator], [depth=-1])
```

type	Type of report, one of the following: dump: [Default] Classic directory dump. Includes version if flag v is set. bcsv: Basic info to CSV. Includes version if flag v is set. ecsv: Extended info to CSV. tcsv: Tree of items (folders and files) with basic data in CSV format: Raw Bytes;Modified Date[;Version]. Includes version if flag v is set. tcsvdirs: Tree of items (folders only) with basic data in CSV format: Raw Bytes;Modified Date[;Version]. Includes version if flag v is set. tree[:template]: Tree structure, folders only. (see Remarks) treeitems[:template]: Tree structure, folders and files. (see Remarks) dirs[:template]: List of subfolders (like a flat Tree). (see Remarks) dirsrel: List of subfolders relative to current folder. files[:template]: List of files in current folder (absolute paths). (see Remarks) filesrel: List of files in current folder (relative paths). items[:template]: List of items in current folder (absolute paths). (see Remarks) itemsrel: List of items in current folder (relative paths). list: The current list contents.
target	Target of report, one of the following: u, popup: Show in popup. c, clipboard: Copy to clipboard. f, file: Write to file. r, return: Return data to running script.
folder	Folder to report on; defaults to the current folder.
flags	A combination of these in any order: a: Append to output file. Only for target "file". j: Include Junctions. p: Prompt to view output file. Only for target "file". r: Include subfolders. Not for type "list". v: Include version information. Only for types "dump", "bcsv", "tcsv", "tcsvdirs".
outputfile	Output file. Only for target "file".
separator	Separator between records. Only for types tree, dirs, dirsrel, items, itemsrel, files, filesrel. If missing records are returned one per line.

depth	Depth of recursion. Ignored if [flags] does not contain "r" (include subfolders). -1 = unlimited (default) 0 = no recursion 1 = one level 2 = two levels etc
return	The report data (if target is "return").

Remarks

- The function provides scripting access to the commands available in the Report tab in the Info Panel. But, in particular through the folder parameter, and through format templates (see below), it goes well beyond the UI commands.
- The behavior is not fully controlled by scripting, but also by settings in Configuration | Report & Data, and in Info Panel | Report. However, the properties that *are* controllable by scripting overwrite the interface settings. E.g. the state of the "Include subfolders" checkbox has no impact on the behavior of folderreport; this property is only controlled by the "flags" parameter.
- Types "tree", "treeitems", "dirs", "files", and "items" support a format template. The template is appended to the selector separated by a ":" (colon), and it fully supports the syntax of the template argument in report.
  When the template contains the `{size}` field the size of each listed folder is freshly calculated! This can take time with large folders.

Examples

```
folderreport("dump", "clipboard", "E:\Test\", "r");
folderreport("dump", "file");
folderreport("dump", "file", <xypath>, "a", "C:\temp\test.txt");
folderreport("bcsv", "c", "C:\Program Files\XYplorer\", "v");
text folderreport("dump", "r", "C:\Program Files\XYplorer\", "v");
text folderreport("tcsv", r, , r);     //tree structure with basic data, files and folders
text folderreport("tcsvdirs", r, , r); //tree structure with basic data, folders only
```

This returns a pipe-separated list of all files in the current path (including subfolders):

```
text folderreport("files", "r", , "r", , "|");
```

Shows all files in the current path, includes only the immediate subfolders:

```
text folderreport("files", "r", , "r", , "<crlf>", 1);
```

Examples for using format templates

Recursive (i.e. including subfolders) report on the current folder:

```
text folderreport("dirs:{fullname}; {size kbr}; {modified yyyy-mm-dd}", "r", , "r");
```

Non-recursive report on C:\Temp:

```
text folderreport("dirs:{name} ({size} bytes)", "r", "C:\Temp");
```

Recursive tree report (folders only) on the current folder:

```
text folderreport("tree:{name} ({size kbr}, {count} items)", "r", "", "r");
```

Recursive tree report (folders and files) on the current folder:

```
text folderreport("treeitems:{dir \{name} ({size mb}, {count} items)|{name}; {size kbr}; {modified yyyy-mm-dd_hh:nn:ss}}", "r", "", "r");
```

Return similar to dos command 'dir /b /s':

```
text folderreport("items", "r", , "r");
```

Returns path and name for an folder, additional size and mod date for files:

```
text folderreport("items:{dir {fullname}|{fullname}; {size kbr}; {modified yyyy-mm-dd}|}", "r", , "r");
```

Returns CSV with user chosen info and self-defined separator TAB:

```
text folderreport("items:{fullname}<tab>{size kbr}<tab>{modified}", "r", , "r");
```

foldersize()

Counts subfolders, files, and bytes in a folder.

Syntax

```
foldersize([folder], [template="<d>|<f>|<b>"], [recursive=1])
```

folder	Folder to investigate. Defaults to the current list path.
template	Format of the returned string. <d> is replaced by the folder count. <f> is replaced by the file count. <b> is replaced by the byte count (formatted, rounded up). <r> is replaced by the byte count (raw, exact). Defaults to "<d>|<f>|<b>".
recursive	0 = non-recursive 1 = recursive, i.e. including subfolders (Default)
return	(depends on the template)

Remarks

Can take time with huge folders. The process can be stopped by ESC.

Examples

```
echo foldersize();
```

```
echo foldersize("<xyscripts>");
```

```
echo foldersize("<xyscripts>", "<b>"); //return just the bytes
```

```
echo foldersize("<xyscripts>", "<r>"); //return just the raw bytes
echo foldersize(, "Folders: <d><crlf>Files: <f><crlf>Bytes: <b>"); //friendly
echo foldersize(, , 0); //current folder non-recursive
```

font()

Sets or gets any of the main application fonts.

Syntax 1

```
font([area=m], [name], [size], [bold], [italics])
```

area	Each area has its own font: m: Main font [default]. Only this font supports arguments "bold" and "italics". c: Controls font (aka "Buttons and Labels"). e: Editor font. r: Regular Expressions font.
name	Font name.
size	Font size. Either absolute value (fractional values work), or increase or decrease amount prefixed with + or -. + or -: Increase / decrease to the next valid font size (the algorithm steps by 0.25 until it reaches a valid value).
bold	Font bold. 0=off, 1=on; [omit]=keep.
italics	Font italics. 0=off, 1=on; [omit]=keep.
return	Definition of the current font (before any change). Format: area;name;size;bold;italics.

Syntax 2

```
font(definition)
```

definition	;-separated string of font properties as returned by this function. Format: area;name;size;bold;italics.

Examples for Syntax 1

```
text font("m", "Consolas", 12, 0, 1); //set main font to Consolas 12 italics, show old font in text box
font(, "Segoe UI", 9, 0, 0);  //set main font to the current factory default
font("e", , 12);    //just change the size of the editor font
font("e", , "+2");  //just increase the size of the editor font ("+2" needs the quotes!)
```

```
font("mce", , 12);  //yep, you can even set all 3 areas at once
status font();      //just show the current main font in the status bar
font("r", , "+"); //increase RegExp font size to the next valid value
font("r", , "-"); //decrease RegExp font size to the next valid value
```

Examples for Syntax 2

```
font("m;Consolas;12;0;1");  //set main font to Consolas 12 italics
font("m;Segoe UI;9;0;0");   //set main font to the current factory default
font("e;Courier New;9.75"); //set editor font name and size
font(";Courier New");       //set main font name
font(";Courier New;12");    //set main font name and size
font(";;12");               //set main font size
font(";;+2");               //increase main font size by 2 points
font(";;-2");               //decrease main font size by 2 points
font(";;;0;0");             //main font turn off bold and italics
font("mce;;9");             //yep, you can even set all 3 areas at once
font(";;+"); //increase main font size to the next valid value
font(";;-"); //decrease main font size to the next valid value
```

format()

Returns a string formatted.

Syntax

```
format(string, style)
```

string	String to format: can be a number or a date.
style	Description of the desired format.
return	Formatted string.

Remarks

Date expressions should be formatted either in the current locale, or in the universal ISO 8601 format (YYYY-MM-DD hh:nn:ss).

Examples

```
text format(1, "000"); //001
text format(1, "@@@"); //  1 (leading spaces)
text format(1234.56, "#,0.00"); //1,234.56
text format(<date>, "yyyy"); //2010
text format(<date>, "dd-mmm-yy"); //20-Sep-10
```

```
text format(<date>, "dddd"); //Monday (in your locale)
```

```
// create a number of empty TXT files in the current folder
  $numfiles = input("Number of files to create", , "10");
  $i = 1;
  while ($i <= $numfiles) {
    new("file" . format($i, "000") . ".txt");
    $i++;
  }
```

formatbytes()

Formats a number as bytes, and converts byte formats.

Syntax

```
formatbytes(bytes, [format=FLEX], [decimals=-1])
```

bytes	The number to format. Can be a raw number or a string like "2 KB".
format	The desired formatting. One of the following: FLEX: [Default] Flexible (best unit is auto-selected). FLEXR: FLEX, but rounded up. KB, MB, GB, TB, or PB: usual bytes unit. KBR: KB, but rounded up. BB: Bytes (with unit) B: Bytes (no unit) RAW: Raw number (no unit, no thousand separators).
decimals	The number of decimal points. Ignored if format is B, BB, KBR, or FLEXR. -1: System default.
return	Formatted number.

Remarks

- The returned numbers are rounded to 2 digits and have thousand separators (unless "R" is stated as format).
- Note that the function is locale-specific! Decimal and thousand separators may be different from locale to locale. However, the first argument accepts the dot and the local decimal separator as decimal separator. This is relevant only in locales where the decimal separator is not the dot, of course. For example:

  ```
  echo formatbytes("1.5 KB", "bb"); //1.536 bytes -- works in any Windows
  echo formatbytes("1,5 KB", "bb"); //1.536 bytes -- works in German Windows
  ```

Examples for US locale:

```
text formatbytes("1.34 MB", "KB");     //1,372.16 KB
text formatbytes("1,372.16 KB", "MB"); //1.34 MB
```

Examples for German locale:

```
text formatbytes("1,34 MB", "KB");     //1.372,16 KB
text formatbytes("1.372,16 KB", "MB"); //1,34 MB
```

More examples for German locale:

```
text formatbytes("2 kb", "b"); //2.048
text formatbytes("2 kb", "bb"); //2.048 bytes
text formatbytes(2560);             //2,50 KB
text formatbytes("2560");           //2,50 KB
text formatbytes("2560", "FLEX");   //2,50 KB
text formatbytes("2560", "FLEXR");  //3 KB
text formatbytes(2048 * 1024);        //2,00 MB
text formatbytes(2048 * 1024 . "KB"); //2,00 GB
text formatbytes("1,5 MB", "RAW"); //1572864
```

Rounding (German locale):

```
text formatbytes(1, "KB");    //0,00 KB
text formatbytes(1, "KB", 0); //0 KB
text formatbytes(1, "KB", 2); //0,00 KB
text formatbytes(1, "KBR");   //1 KB
text formatbytes(512, "KB");    // 0,50 KB
text formatbytes(512, "KB", 0); // 1 KB
text formatbytes(512, "KB", 2); // 0,50 KB
text formatbytes(512, "KBR");   // 1 KB
```

formatdate()

Returns a date/time expression in a specific format, optionally shifted by a specific interval.

Syntax

```
formatdate([date], [format], [shift_unit], [shift_num])
```

date	[optional] Date expression to be formatted. Defaults to now. The date expression should be formatted either in the current locale, or in the universal ISO 8601 format (YYYY-MM-DD hh:nn:ss).
format	[optional] e.g. yyyymmdd_hhnnss, see examples below.

Also supports the named format ISOWeek.
Defaults to general system date/time format.

XYplorer-only special values (useful for advanced scripters in some extreme situations):
4char: Returns a 4 character (8 byte) string where the bytes exactly mirror the bytes in an 8-byte Visual Basic Date variable corresponding to the input date.
8char: Same like 4char but a wide string (8 character, 16 byte), where every second byte is 0 (null).

shift_unit [optional] unit to shift date by
y = years
m = months
w = weeks
d = days
h = hours
n = minutes
s = seconds
Legal results: between 01.01.0100 and 31.12.9999, else error

shift_num [optional] number of units to shift date by;
negative to go back in time;
must be an integer value.

Examples

```
text formatdate(); //16.08.2009 08:27:05
text formatdate(, "yyyymmdd_hhnnss"); //20090816_082217
text formatdate("", "yyyymmdd_hhnnss"); //Error: Invalid date
text formatdate(, "dddd");  //Monday (if it is Monday)
text hexdump(formatdate("24.01.2018 11:08:02", "8char"),,"r"); //69 0E 5E D8 8E 0E E5 40
```

Examples for shifted dates

```
// returns 16.08.2009 08:00:03
text formatdate("16.08.2009 08:00:00", , "s", 3);
// returns 16.08.2009 08:03:00 (+ 3 minutes)
text formatdate("16.08.2009 08:00:00", , "s", 180);
// returns 16.08.2009 07:59:59
text formatdate("16.08.2009 08:00:00", , "s", -1);
// returns 16.08.2009 11:30:00 (+ 3.5 hours)
text formatdate("16.08.2009 08:00:00", , "n", 210);
// returns 16.08.2009 08:00:00 with error "Date shifting failed"
text formatdate("16.08.2009 08:00:00", , "y", 7991);
```

Usage

A practical example would be to shift the timestamp of your photos after coming back from a holiday in a different time zone:

```
//set modified date of current file to its EXIF + 6 hours
timestamp m, formatdate("<dateexif>", , "h", 6);
```

formatlist()

Formats a list of items.

Syntax

```
formatlist(list, [format], [separator="|"], [param], [flags])
```

list	The list to format.
format	Switches for the desired formatting. Can be freely combined in any order. (Of course, the combinations should be meaningful.) t = trim (remove surrounding spaces) u = unquote the items s = sort ascending (by default case-insensitive: A==a) r = sort descending (think "reversed order") c = sort case-sensitive (A!=a) n = sort naturally (can only be case-insensitive!) v = reverse sort (reverse the current order, whatever it is) x = shuffle (randomize order) e = remove empty items d = remove duplicate items (think "dedupe") f = filter (wildcards) F = filter (no wildcards) q = quote the items p = perforate (split by length of parts, not by separator; the separator is inserted between the parts)
separator	Separates the items in the list; defaults to | (pipe). If you pass "" (empty) the list is treated as a list of single characters.
param	On format x: Seeds the randomizer used for shuffling. Can be any number. If a seed is passed, the randomizer always generates the same random sequence for this seed. On format p: Length of parts the list shall be split into. If missing or smaller than 1 it is set to 1. On format f (and F): A list of filters, separated by separator (3rd argument). The filters support the usual pattern wildcards (*?#). Format f (and F) also supports inversion by a prefixed "!". If inverted, the filter returns all non-matching items. If more than one filter is stated (OR-ed list of filters) the inversion is applied to combined result of all filters. You cannot invert individual filters. See examples below.
flags	f = On format f: Filter a list of full paths only by the filenames, i.e. the filter given in "param" is only matched against the last components of the paths. (It is not verified

whether the last components are existing files or folders.)

return The formatted list.

Notes

- Trimming is done before unquoting.
- Unquoting is done before sorting.
- Sorting is done before removing duplicates.
- Sorting is by default case-insensitive (A==a). Passing switch "c" forces it to be case-sensitive, unless "n" is passed as well which overwrites "c" and forces case-insensitive sorting.
- Sorting naturally (3 < 20 < 100) is only supported from XP onwards.
- Passing "r", "c", or "n" implies "s".
- Passing "x" (shuffle) overwrites any other sorting flags.
- Removing empty items is done before removing duplicates.
- Removing duplicates only works on adjacent items, so if the list is not sorted non-adjacent duplicates remain in the list.
- Duplicates are matched case-insensitive (A==a).
- Quoting is done as the last step. Items that are already quoted in the source are not quoted again.
- If "list" is empty then an empty (and unquoted) string is returned, even if the "q"-switch is passed.
- If "format" is empty then "list" is returned unchanged.

Examples

```
text formatlist("b|a|c", "s"); //a|b|c
text formatlist("b|a|c", "r"); //c|b|a
text formatlist("b|a|c", "v"); //c|a|b
text formatlist("b||a|c", "s"); //|a|b|c
text formatlist("b||a|c", "se"); //a|b|c
text formatlist(" b  |   |a  |   c", "s"); //   |   c| b  |a
text formatlist(" b  |   |a  |   c", "set"); //a|b|c
text formatlist('b|"a"|c', "s"); //"a"|b|c
text formatlist('b|"a"|c', "su"); //a|b|c
text formatlist("bac", "s", ""); //abc (empty separator)
text formatlist("3b|10a|200c", "s");  //10a|200c|3b
text formatlist("3b|10a|200c", "n");  //3b|10a|200c
text formatlist("3b|10a|200c", "nr"); //200c|10a|3b
text formatlist("b|a|C|D", "s");  //a|b|C|D
text formatlist("b|a|C|D", "c");  //C|D|a|b
text formatlist("b|a|a|a|c", "s");  //a|a|a|b|c
```

```
text formatlist("b|a|a|a|c", "sd"); //a|b|c
text formatlist("b|b|a|b|c", "d");      //b|a|b|c
text formatlist("b||b|a|b|c", "ed");    //b|a|b|c
text formatlist("b||b|a|b|c", "eds");   //a|b|c
text formatlist("b||b|a|b|c", "edsq");  //"a"|"b"|"c"
text formatlist("ba|a|c", "f", , "a|b");  //a
text formatlist("ba|a", "f", , "a*");     //a
text formatlist("ba|a", "f", , "*a");     //ba|a
text formatlist("b|a|c", "q"); //"b"|"a"|"c"
text formatlist("b", "q");  //"b"
text formatlist("", "q");   //returns empty string, *not* quoted
text formatlist("b|a|c", "x"); //a|c|b ... b|c|a ... c|b|a ... (no seed passed)
text formatlist("b|a|c", "x", , 19); //always b|a|c (seed 19 passed)
text formatlist("abcdefg", "pr", , 2); //g|ef|cd|ab
text formatlist("abcdefg", "p", "." , 1); //a.b.c.d.e.f.g
```

Format f and inversion by prefixed "!": To use "!" as a normal character in the first position of the filter list you can escape it by "\":

```
text formatlist("a|ba|ac|ed|!a", "f", , "a*|*d");     //a|ac|ed
text formatlist("a|ba|ac|ed|!a", "f", , "!a*|*d");    //ba|!a
text formatlist("a|ba|ac|ed|!a", "f", , "\!a*|*d");   //ed|!a
```

Format f and Flags f:

```
text formatlist("C:\A.txt|C:\C.txt", "f", "|", "*c*"); //C:\A.txt|C:\C.txt
text formatlist("C:\A.txt|C:\C.txt", "f", "|", "*c*", "f"); //C:\C.txt
```

Filter f (wildcards) versus F (no wildcards):

```
text formatlist("a*|ab", "f", , "a*");     //a*|ab
text formatlist("a*|ab", "f", , "!a*");    //[nothing]
text formatlist("a*|ab", "F", , "a*");     //a*
text formatlist("a*|ab", "F", , "!a*");    //ab
```

fresh

Runs a new XYplorer instance with factory defaults.

Syntax

```
fresh [startpath]
```

startpath: Start path in the first pane.
Defaults to the factory default start path ("This PC").

Remarks

Fresh mode sets the Application Data Path to subfolder "Fresh" of the normal Application Data Path. This way it's impossible that any of your everyday settings are inadvertently overwritten by the fresh instance.

Example

```
fresh;        //start fresh instance in "This PC"
fresh "C:\";  //start fresh instance in C:\
```

freshhere

Like fresh but opened to current path.

Syntax

```
freshhere
```

Example

```
freshhere;    //start fresh instance in current path
```

get()

See Scripting Command Get.

getkey()

Gets the value of a configuration key in the current INI file, or of any INI file.

Syntax

```
getkey(key, section, [INIfile], [flags])
```

key	The name of the key.
section	The name of the section.
INIfile	[optional] the name of the INI file. Expects a filename with extension, and can be absolute or relative to the running script's path. If the parameter is missing it defaults to XYplorer's current INI file. Of course, the file to be loaded is expected to conform to the format of INI files.
flags	(bit field) 1: Use XYplorer's native algorithm (see Notes below). Note that key and section are *case-sensitive* with this algorithm!

2: Decode UTF-8 if there is any.
Only works in combination with flag 1!

return — The value.

Usage

Note that getkey pairs with setkey. The getkey/setkey commands support reading/writing values of up to 32,766 characters.

Notes on XYplorer's native algorithm

XYplorer's native algorithm for reading INI values will return leading Tab characters. The standard Windows algorithm will not.
XYplorer's native algorithm is optimized for reading hundreds of keys out of huge INI files. It's not made for picking out single keys as the getkey() function does. So, performance-wise you will suffer a bit for the added functionality.

Examples

```
$a = getkey("StartPath", "General"); msg $a;
```

Sets variable $a to the value of key StartPath in section General of the current INI file; then displays it in a message box.

```
$a = getkey("timeout", "boot loader", "C:\boot.ini"); msg $a;
```

Sets variable $a to the value of key timeout in section boot loader of C:\boot.ini; then displays it in a message box.

Examples for flags=1 (XYplorer's native algorithm)

In these examples the key is `Year=[TAB]2012`, where [TAB] = Tab character.

```
text getkey("Year", "Vacation", "<curpath>\holiday.ini");    // "2012"
text getkey("Year", "Vacation", "<curpath>\holiday.ini", 1); // "[TAB]2012"
```

getpathcomponent(), gpc()

Returns the specified component of a path.

Syntax

```
getpathcomponent([path], [component], [index=1], [flags])
```

path — The path to parse; can be a folder or a file. Defaults to the current folder or file.

component — The component to return; can be any of the following:

path: [default] The path without any file, unslashed. Drives are returned with colon.

drive:The drive or share. Drives are returned without colon, i.e. the drive letter only.

parent: The parent folder.

file, name: The file (or folder) name without path.

base: The file without path and without extension.

ext: The extension.

lext: The extension in lower case.

server: The server in case of an UNC path, else the drive (without colon).

full: Return all components of the input path.

count: The count of components.

component: The component referred to by index.

index — The index of the component to return. Only used if component is "component". The first component is 1, the last component is -1.
Invalid indices or index 0 (zero) return nothing.
UNC paths are stripped off the leading "\\" before applying the index.

flags — (bit field)
1: Return from start.
2: Return till end.
4: Mind folders when returning "base" or "ext". (Path is tested for being a folder; folders have no extensions.)
Flags 1 and 2 only apply if component is set to "component".
8: Convert long format to DOS/8.3.
16: Convert DOS/8.3 to long format.

Notes

- Both backward "\" and forward slash "/" are supported as component separators.
- Double-slashes are converted to single slashes before parsing.
- Paths are returned unslashed.
- DOS/8.3 format: Note that in Windows the 8.3 file naming can be enabled/disabled per drive. If it is disabled then a conversion to DOS/8.3 using this function returns the (valid part of the) input unchanged (apart from removing any trailing backslash).

Examples

```
echo getpathcomponent(, "file");                  // current item without path
echo getpathcomponent("E:\x\y.txt", "ext");       // txt
echo getpathcomponent("E:\x\y.txt", "base");      // y
echo getpathcomponent("E:\x\y.txt", "file");      // y.txt
echo getpathcomponent("E:\x\y.txt", "name");      // y.txt     (identical to "file")
echo getpathcomponent("E:\x\y.txt", "parent");    // x
echo getpathcomponent("E:\x\y.txt", "path");      // E:\x
echo getpathcomponent("E:\y.txt", "path");        // E:
echo getpathcomponent("E:\x\y.txt", "drive");     // E
```

```
echo getpathcomponent("\\Mars\Venus", "drive");   // \\Mars\Venus
echo getpathcomponent("\\Mars\Venus", "path");    // \\Mars
echo getpathcomponent("\\Mars\Venus", "server");  // Mars
echo getpathcomponent("\\Mars\Venus", "count");   // 2

echo getpathcomponent("E:\x\y.txt", "component");     // E:
echo getpathcomponent("E:\x\y.txt", "component", 1);  // E:
echo getpathcomponent("E:\x\y.txt", "component", 2);  // x
echo getpathcomponent("E:\x\y.txt", "component", 3);  // y.txt
echo getpathcomponent("E:\x\y.txt", "component", 4);  // <nothing>
echo getpathcomponent("E:\x\y.txt", "component", -1); // y.txt
echo getpathcomponent("E:\x\y.txt", "component", -2); // x
echo getpathcomponent("E:\x\y.txt", "component", -3); // E:
echo getpathcomponent("E:\x\y.txt", "component", -4); // <nothing>
echo getpathcomponent("\\Mars\Venus", "component");   // Mars

echo gpc("D:\Users\%USERNAME%\Music", "component", -2, 0);  //Donald
echo gpc("D:\Users\%USERNAME%\Music", "component", -2, 1);  //D:\Users\Donald
echo gpc("D:\Users\%USERNAME%\Music", "component", -2, 2);  //Donald\Music
```

If the currently selected item is the folder "Stuff.001":

```
echo getpathcomponent("<curitem>", "base", , 0); //"Stuff"
echo getpathcomponent("<curitem>", "base", , 4); //"Stuff.001"
```

DOS/8.3 to/from long format:

```
echo getpathcomponent("C:\Program Files", "full", 3:=8);  //C:\PROGRA~1
echo getpathcomponent("C:\PROGRA~1", "full", 3:=16); //C:\Program Files
echo getpathcomponent("C:\Program Files\AutoHotkey\AutoHotkeyU64.exe", "file", 3:=8);  //
AUTOHO~3.EXE
```

getsectionlist()

Returns a list of sections in an INI file or a list of "key=value" pairs for the specified section.

Syntax

```
getsectionlist([section], [INIfile], [separator=CRLF])
```

section [optional] The name of the section.

	If omitted a list of sections is returned.
INIfile	[optional] The name of the INI file. Expects a filename with extension, and can be absolute or relative to the running script's path. If omitted XYplorer's current INI file is used.
separator	[optional] Separates the items in the returned list. Defaults to CRLF (line feed).
return	The list of sections or 'key=value' pairs in the specified section.

Examples

Displays the list of sections in XYplorer's current INI file:

```
text getsectionlist();
```

Displays the list of "key=value" pairs in the "General" section of XYplorer's current INI file:

```
text getsectionlist("General");
```

gettoken()

Returns a substring by index.

Syntax

```
gettoken(string, [index=1], [separator=" "], [format], [flags])
```

string	Source string; not changed by procedure.
index	Index of token, 1-based; defaults to 1. Negative indices reference items from the right end of the list. "count": returns the count of tokens in the string. Returns 0 if string is empty.
separator	Separator of tokens; defaults to " ". If empty ("") then the string is separated char by char.
format	Switches for the desired formatting. Can be freely combined in any order. t = trim returned string (crop leading and trailing spaces)
flags (bit field)	
	1: Return from start. 2: Return till end. 4: Count empty tokens as well.
return	The token.

Examples

```
echo gettoken("Donald Duck"); // Donald
```

```
echo gettoken("a,b,c,d,e", 4, ","); // d
```

```
echo gettoken("https://www.xyplorer.com", 2, "//"); // www.xyplorer.com
```

```
echo gettoken("a,b,c,d,e", "count", ","); // 5
```

```
echo gettoken("", "count", ","); // 0
echo gettoken("one,two,three,four,five", -1, ","); //five
echo gettoken("one,two,three,four,five", -4, ","); //two
```

Using the flags parameter:

```
echo gettoken("a,b,c", 2, ",");       // b
echo gettoken("a,b,c", 2, ",", , 1);  // a,b
echo gettoken("a,b,c", 2, ",", , 2);  // b,c
```

Empty separator:

```
text gettoken("abcd", 1, ""); //a
text gettoken("abcd", 4, ""); //d
text gettoken("abcd", -1, ""); //d
text gettoken("abcd", 2, "", , 1); //ab   (Return from start)
text gettoken("abcd", 2, "", , 2); //bcd  (Return till end)
```

Counting empty tokens as well:

```
text gettoken(",,a,b,c,", "count", ",");       //3
text gettoken(",,a,b,c,", "count", ",", 4:=4); //6
```

gettokenindex()

Returns the first index of a token in a token list, or the count of its occurrences.

Syntax

```
gettokenindex(token, tokenlist, [separator=|], [switches="iw"], [start=1]);
```

token	Token to find in list of tokens.
tokenlist	List of tokens.
separator	Separator used in list of tokens.
switches	(lower case letters in any order; defaults to "iw" if missing) i: Matching is case insensitive (A==a). w: Token has wildcards. If missing then token is treated literally. If included and token does not contain wildcards it is treated as *token*. c: Return count the occurrences.
start	Token index from which to start searching. The index returned is still relative to the beginning of the tokenlist. Defaults to 1 (= 1st index).
returns	

0: If token was not found in the list.
Otherwise return the index of the token within the list (1 = first position).
On switch c: Count the occurrences.

Examples

```
echo gettokenindex("xls", "docx|doc|xlsx|xls");  //3 (*xls*)
echo gettokenindex("red;*", "Blue;12|Red;15|Green;28"); //2
echo gettokenindex("red", "Blue;12|Red;15|Green;28"); //2 (*red*)
echo gettokenindex("red", "Blue;12|Red;15|Green;28", , ""); //0 (red)
echo gettokenindex("xls*", "docx|doc|xlsx|xls", , "wc");  //2 (return count)
```

Examples for the start parameter

```
echo gettokenindex("e", "a,b,e,d,e", ",");      //3
echo gettokenindex("e", "a,b,e,d,e", ",", , 3); //3
echo gettokenindex("e", "a,b,e,d,e", ",", , 4); //5
echo gettokenindex("e", "a,b,e,d,e", ",", , 5); //5
echo gettokenindex("e", "a,b,e,d,e", ",", , 6); //0 (not found)
```

ghost

Defines a Ghost Filter, toggles its state.

Syntax

```
ghost(patterns, toggle)
```

patterns	(optional) List of wildcard patterns separated by |. missing: keep current patterns empty (""): clears filter else: set new patterns
toggle	(optional) missing/empty: keep current state 0: turn Ghost Filter OFF 1: turn Ghost Filter ON -1: toggle Ghost Filter
return	current/old patterns (before setting new patterns)

Remarks

The new patterns will completely replace the previous patterns just as if you manually entered them through the GUI. So they will be toggled by the "Ghost Filter" toolbar button, and remembered between sessions.

Examples

```
ghost(".*", 1); //hide all items beginning with a dot
ghost(, 0); //turn off any ghost filter
ghost(, -1); //toggle ghost filter
text ghost(); //view current ghost filter definition
ghost(""); //remove all patterns
ghost("", 0); //remove all patterns and turn the filter OFF
```

global

Define one or more variables as global.

Syntax

```
global variable(s)
```

variable(s) A single variable, or a comma-separated list of up to 10 variables. If a variable has already been defined as global before, then the global command initializes it to its current global value, else it is initialized to an empty string.

Usage

By default, all variables in XY scripting are local, i.e. they are not shared between scripts. The command "global" can be applied to share a variable between all scripts that used "global" on that variable.

Examples

```
global $foo, $bar;
```

= Define $foo and $bar as global variables.

Here are two scripts inside one script resource where the first is calling the second:

```
"script1"
  $foo = 4; // set local value: new local var $foo is created
  msg $foo; // 4 (local)
  // now make $foo global
  // it's initialized to [empty] (assuming nothing created
  // a global $foo before with a different value, which is
  // possible when "script1" has been called by yet another
  // script that has declared $foo as global)
  global $foo;
  msg $foo; // [empty] (global)
  $foo = 8; // set global value
  msg $foo; // 8 (global)

  sub script2; // go down
```

```
  msg $foo; // 16 (global, as set in script2)
  msg $bar; // $bar (uninitialized, not a variable)
  // make $bar global
  // it's set to "tequila" (as previously set in script2)
  global $bar;
  msg $bar; // tequila

"script2"
  msg $foo;   // $foo (uninitialized, not a variable)
  $foo = 15;  // set local value: new local var $foo is created
  msg $foo;   // 15 (local)
  // now make $foo global
  // it's set to 8 (as previously set in script1)
  // also create a new global $bar
  global $foo, $bar;
  msg $foo;   // 8 (global)
  $foo = 16;  // set global value
  $bar = "tequila";  // set global value
```

Optionally you can immediately assign initial values to the declared variables:

```
global $a = "a"; echo $a;
```

```
global $a = "a", $b = "b"; echo "$a, $b";
```

These assigned values can be variables themselves, or even expressions or functions:

```
global $a = <clipboard>; echo $a;
```

```
global $a = 5*7; echo $a;
```

Notes

- You can have as many global variables as you like, but it can be no more than 10 per "global" statement for internal reasons. IOW, add a new "global" line for each pack of 10 variables if you really need that many.
- It's recommended that you reflect the global nature of a variable in the variable name, for example by prepending $g_ to global variables. It will make your code easier to read and maintain.

goto

Changes the current to a new location.

Syntax

```
goto location, [reuseexistingtab]
```

location: [Required] A path, a file, a Quick Search, a Visual Filter, or a URL ... whatever is accepted in the Address Bar, Favorites, Catalog, and User-Defined Command Go To. Including Environment Variables and XYplorer native variables.
Relative paths are resolved relative to <xypath> (the path of XYplorer.exe).

reuseexistingtab:

0: [default] overwrite current tab

1: reuse existing tab (else overwrite current)

Examples

```
goto "C:\";
```

Go to C:\.

```
goto "Desktop";
```

Go to the Desktop folder.

```
goto "Desktop?*.bmp";
```

Quick Search: List all *.bmp files in Desktop.

```
goto "Desktop?*.bmp;*.jpg";
```

Quick Search: List all *.bmp and *.jpg files in Desktop.

```
goto "C:\|*.bat";
```

Visual Filter: Go to C:\ and show only BAT files.

```
goto "C:\|";
```

Visual Filter: Go to C:\ and show all files (removing any previous Visual Filter).

```
goto <xypath>;
```

Go to the current application path.

```
goto "<curpath>\..";
```

Go one level up.

```
goto %temp%;
```

Go to the system temporary path.

hash()

Creates hash of a string or file.

Syntax

```
hash([algo="md5"], [string], [flags])
```

algo: Hash algorithm; one of the following:
md5: [Default] MD5.

crc32: CRC-32.
sha1: SHA-1.
sha256: SHA-256.
sha384: SHA-384.
sha512: SHA-512.

string — Data or file to be hashed (see flags).
Defaults to current list file if flags AND 1.

flags — (bit field)
0: [default] String is data to hash.
1: String is a file spec whose contents to hash.
2: Show progress and allow break out by ESC. Only applied if "1" is also set.
4: Convert Unicode strings to UTF8 before hashing. Note: This should be the way to go for Unicode strings, although it's difficult to find authoritative information about it.

return — Hash.

Notes

- The maximum supported size for strings is 2GB data. For files the size is not limited (the file is streamed).
- You cannot abort the process by ESC. XYplorer will be blocked while the process is running.
- hash("md5" ...) is about 20 times faster than md5().
- md5() is kept for compatibility, and for systems where hash is not supported (which might be the case on older Windows).

Examples

```
text hash("md5", ""); //d41d8cd98f00b204e9800998ecf8427e
text hash("sha1", ""); //da39a3ee5e6b4b0d3255bfef95601890afd80709
text hash(,,1); //md5 of current list file
text hash(,,3); //md5 of current list file, with progress
text hash("md5", "李振藩", 0); //0d1b08c34858921bc7c662b228acb7ba = probably incorrect
text hash("md5", "李振藩", 4); //5f5f4736a7c75238c23f8c9601c796cf = probably correct
```

hashlist

Calculates and shows the hash values for a list of files.

Syntax

```
hashlist [algo], [filelist]
```

algo — Hash algorithm; one of the following:
md5: MD5 [Default].
crc32: CRC-32.

sha1: SHA-1.
sha256: SHA-256.
sha384: SHA-384.
sha512: SHA-512.

filelist — CRLF- or |-separated list of files.
Defaults to all currently selected files.

Examples

```
hashlist; //shows MD5 values for all selected files
hashlist "sha256", "D:\a.txt|D:\b.txt";
```

hexdump()

Returns a string as hex dump.

Syntax

```
hexdump(string, [unicode], [switches])
```

string — [required] String to display.

unicode — (bit field) [optional]
0 = [Default]
left section: show one hex byte per character
right section: show one character per hex byte
1 = left section: show two hex bytes per character
2 = right section: show one character per two hex bytes

switches — [optional]
r = raw
i = inverse

return — The hex dump.

Notes

- With the "r" switch (raw) you can return a raw hex string: Hex values separated and terminated by a space.
- With the "i" switch (inverse) the spaces separating and terminating the Hex values are optional. Either you use them (all) or you don't (none).
- The inverse function also supports lower case Hex strings.

Examples

```
text hexdump(chr(20000), 1), 700;
text hexdump("<clipboard>", 1), 700; //show current clipboard
text hexdump("abc", , r);   //"61 62 63 "
```

```
text hexdump("abc", 1, r);  //"61 00 62 00 63 00 "
text hexdump("E5 65 2C 67 9E 8A ", 1 , ri); // "Nihongo" in Japanese characters
text hexdump("E5652C679E8A", 1 , ri); // "Nihongo" in Japanese characters
text hexdump("e5652c679e8a", 1 , ri); // "Nihongo" in Japanese characters
```

Effect of the unicode argument:

```
text hexdump("text", 0);  //00000000: 74 65 78 74                                          text
text hexdump("text", 1);  //00000000: 74 00 65 00 78 00 74 00                             t.e.
x.t.
text hexdump("text", 2);  //00000000: 74 65 78 74                                          整瑣
text hexdump("text", 3);  //00000000: 74 00 65 00 78 00 74 00                             text
```

Usage

Useful when you want to know what's really on the clipboard.

hextodec()

Converts a hexadecimal number into a signed decimal number.

Syntax

```
hextodec(hexnumber)
```

hexnumber	The hexadecimal number to convert. Can be in any of these formats: "0x499602D2" (0x-prefix as used in C++, PHP etc) "&H499602D2" (&H-prefix as used in VB) "499602D2" (no prefix) Anything above 8 digits (after the prefix) is ignored.
return	The signed decimal number.

Notes

The highest value is 0x7fffffff (2147483647), the lowest value is 0x80000000 (-2147483648).

Examples

```
text hextodec("0x499602D2"); // 1234567890
text hextodec("ffffffff"); // -1
```

highlight

Defines the color for the Highlighted Folder feature of the current location on Tree, or turns it off.

Syntax

```
highlight [color (rrggbb)], [folder]
```

color: The new highlight color in format RRGGBB. If empty or missing the current highlight color is removed.

folder: [optional] if stated then the highlighting is applied to that folder, else it is applied to the current folder.

Examples

```
highlight "FF8000"
```

Will set the highlight color to orange for the current location.

```
highlight
```

Will disable the Highlighted Folder feature for the current location.

```
highlight "000000"
```

Will set the highlight color for the current location to the default highlight color.

```
highlight "FF0000", "C:\"
```

Set red highlight to C:\.

```
highlight , "C:\"
```

Unset any highlight from C:\.

hotkeyshowapp()

Sets or gets the hotkey to show/minimize the application.

Syntax

```
hotkeyshowapp([key])
```

key	The hotkey to show/minimize the application. The hotkey consists of an optional modifier (combination of Win, Ctrl, Shift, Alt) and a key.
return	The current/previous hotkey.

Remarks

- The hotkey is used to show (restore, foreground) the application when it's in the background, minimized to the taskbar, or minimized to the tray.
- If the application is already showing, the key will minimize it.
- The command takes effect immediately, no restart required.

Examples

```
echo hotkeyshowapp();              //show the current hotkey
hotkeyshowapp("Ctrl+Shift+O");     //set Ctrl+Shift+O as new hotkey
hotkeyshowapp("Win+Alt+Y");        //set Win+Alt+Y as new hotkey
hotkeyshowapp("");                 //unset current hotkey
```

html()

Displays an HTML formatted string.

Syntax

```
html([html], [width=800], [height=400], [caption=XYplorer])
```

html	HTML formatted string, or name of one. If no "<" character is contained it is interpreted as a URL or filename pointing to an HTML formatted document, or any type of document that can be displayed in Internet Explorer (PDF, DOC, etc.). Default is the current file.
width	Width of window in pixels (minimum: 250; maximum: screen width; default: 800). Percentages of screen width are supported, e.g. 75%.
height	Height of window in pixels (minimum: 150; maximum: screen height; default: 400). Percentages of screen height are supported, e.g. 75%.
caption	Caption on window's title bar; default: "XYplorer".
return	A special hypertext reference when a link starting with "xys:" was clicked (see example below). On Close button or ESC, an empty string is returned.

Examples

```
html("<html><body>Hi!</body></html>");
```

```
html("https://www.xyplorer.com/", 800);
```

Shows a message box "Yes!" or "No!" depending on the link you click:

```
echo(html('<html><body><a href="xys:Yes!">Say "Yes!"</a><br><a href="xys:No!">Say "No!"</a></body></html>'));
```

Note that <xypath> is resolved *before* the argument is passed to html, so it is auto-interpreted correctly as filename:

```
html("<xypath>\XYplorer.pdf");
```

```
html("<b>Size is relative to screen!</b>", 75%, 75%);
```

htmlencode()

Converts non-ANSI Unicode characters to numeric HTML entities.

Syntax

```
htmlencode(text)
```

text	String of characters to convert.

Remarks

- The general entity format returned is: `&#xHHHH;` . HHHH here stands for the hexadecimal Unicode code point.
- ANSI characters (ordinal 0-255) remain unchanged.

Examples

```
text chr(0x5FEB) . " -> " . htmlencode(chr(0x5FEB)); //快 -> &#x5FEB;
text htmlencode("快速入门指南"); //&#x5FEB;&#x901F;&#x5165;&#x95E8;&#x6307;&#x5357;
copytext htmlencode(<clipboard>); //convert current clipboard contents
```

id3tag()

Writes ID3v1 tags to files that support this. Returns the current tags.

Syntax

```
id3tag([file], [tagslist], [separator="|"], [flags])
```

file	File to tag. Defaults to the current file.
tagslist	[optional] List of tags to write. Format: Name=Value|Name=Value|... Supported names and max length of values in ID3v1: title 30 characters artist 30 characters album 30 characters year 4 characters comments 28 characters track 0-255 (numeric) genre 0-255 (numeric) The names are case-insensitive (A==a). The order is irrelevant.
separator	[optional] Separates the tags in the passed tagslist. Defaults to "|" (pipe).
flags	(bit field) 1: Return ID3v1 tags even if ID3v2 tags are present.

Otherwise ID3v2 tags are returned even if ID3v1 tags are present.

return — The current tags (before the tagging).
If ID3v2 are present they are returned instead of ID3v1 tags.

Examples

```
id3tag(, "album=On The Beach|artist=Neil Young"); //tags the current file
id3tag(, "ARTIST=Neil Young|ALBUM=On The Beach"); //identical to above
text id3tag();  //just return the ID3v1 (or ID3v2 if present) tags of the current file
text id3tag(3:=1);  ///just return the ID3v1 tags (even if ID3v2 tags are present)
text id3tag("E:\Test\Sugar.mp3"); //return the ID3v1 (or ID3v2 if present) tags of that file
```

Remarks

- Only the tags you pass are written, any other tags are left untouched.
- The tags "track" and "genre" are written and returned as numbers.
- Works with all files that support ID3v1/ID3v2 tags, e.g. MP3, MP4, M4A.
- While ID3v2 tags can be read they cannot be written by this command. Use the id3v2tag() function for that.

id3v2tag()

Writes ID3v2 tags with full Unicode support to compatible files. It returns the current tags.

Syntax

```
id3v2tag([file], [tagslist], [separator="|"], [flags])
```

Remarks

- This function is identical to the id3tag() function, except that the latter only writes ID3v1 tags.
- ID3v2 tags can be much larger than ID3v1 tags. They have a maximum total size of 256MB, and individual tags can be up to 16MB.

Examples

```
id3v2tag(, "album=On The Beach|artist=Neil Young"); //tags the current file
id3v2tag(, "genre=Laid-Back Country Funk|year=543210"); //tags the current file
```

implode()

Maps an array onto a list.

Syntax

```
implode($array, [$list], [separator="|"], [flags])
```

$array	Array variable name, e.g. $a or $a[] (trailing square brackets are optional).
$list	Variable name of the list, e.g. $a. If omitted, the function returns the list.
separator	Separator of the returned list items. Defaults to |.
flags	e = skip empty
return	Number of items in the list, or the list itself if $list is omitted.

Examples

```
explode($a, "a,b,c", ","); $a[1] = "X"; echo implode($a); //a|X|c
explode($a, "a,,c", ",");      echo implode($a);        //a||c
explode($a, "a,,c", ",");      echo implode($a,,, "e"); //a|c
```

incr

Increments a numerical value.

Syntax

```
incr outputvar, [value], [increment=1]
```

outputvar	[Required] The variable that will receive the output.
value	Value defaults to outputvar(!); can be negative.
increment	Defaults to 1; can be negative.

Examples

```
incr $a;             // returns $a + 1
set $a, 5; incr $a; // returns 6 (5 + 1)
incr $a, 1;        // returns 2 (1 + 1)
incr $a, 1, 2;     // returns 3 (1 + 2)
incr $a, -1, 2;    // returns 1 (-1 + 2)
incr $a, 1, -2;    // returns -1 (1 - 2)
incr $a, -1, -2;   // returns -3 (-1 - 2)
incr $a, "Paul";   // returns 1 (0 + 1)
incr $a, "3Paul"; // returns 4 (3 + 1)
```

indexatpos()

Returns the fixed internal item index within a control at a certain screen position.

Syntax

```
indexatpos([x], [y], [flags])
```

x	(optional) Defaults to the X mouse position on screen.
y	(optional) Defaults to the Y mouse position on screen.
flags	
	0: If x and y are passed, they are the position on screen.
	1: If x and y are passed, they are the position on XYplorer.
return	Item index. The index is usually 0-based, i.e. the first index is "0". Returns "-1" if the X/Y coordinates do not point to any item.

Remarks

Supported controls are:
- Tree (position of hovered item)
- Catalog (position of hovered item)
- List (position of hovered item)
- Toolbar (position of hovered button)
- Tab Bar (position of hovered tab)

With Breadcrumb Bar and Status Bar, the Index value is meaningless (it's always 1 less than the position index).

Examples

```
echo indexatpos(); //item position at current mouse position on screen
echo indexatpos(592, 662); //item position at arbitrary mouse position on screen
echo indexatpos(73, 108, 1); //item position at arbitrary mouse position on XYplorer
status posatpos() . " - " . indexatpos(); //show position and index in the Status Bar
```

input()

Shows an input dialog to the user and returns user input.

Syntax

```
input(topic, [notes], [default], [style=s|e|m|w], [cancel], [width=800], [height=400], [items], [icon])
```

topic	Topic line, printed in bold on white.
notes	on style=s/e: text, can be multiline, printed on grey on style=m/w: text, singleline, printed on white
default	Text that will be given as default value.
style	Style of the input dialog:

s: [default] singleline
e: singleline, notes in Edit Text font (fixed space)
m: multiline non-wrapped
w: multiline wrapped

cancel	Value to be returned when user cancels. If missing the script is terminated on cancel.
width	Width of window in pixels (minimum: 250; maximum: screen width; default: 800).
height	Height of window in pixels (minimum: 150; maximum: screen height; default: 400). For multiline input only.
items	A pipe or <CRLF> separated list of item(s) for the combo-box entries. If CRLF is present then CRLF is used, else pipe is used. Lets you turn the input field into a combo box where you can select items from a dropdown.
icon	The key to a toolbar icon, prefixed by ":", e.g. ":dice". It also accepts an image file (GIF, JPG, PNG). It can be a full path, or just a filename (auto-resolved to the default icons path <xyicons>).
return	User input.

Examples

Ask the user for an extension to filter out, store it inside variable $ext. Then set a Visual Filter to hide all files with the given extension:

```
$ext = input("Enter the file extension you do NOT want to see on List", "Just the extension without dot, e.g. txt." , "zip"); filter "!*.$ext";
```

Ask the user to enter a script in a multiline text box; then load it:

```
$a = input("Enter Script", , 'echo "Hi!"', "m"); load $a,,"s";
```

Use the items parameter to turn the input field into a combo box (also showing the use of the icon parameter) :

```
echo input("Enter Greeting", "Try to be nice..." , "Hi!", e, 7:="Hello|Konbanwa|Go away|WTF", 8:=":dark");
```

The width parameter works with single line style:

```
echo input("Test", 5:=1000);
```

inputfile()

Shows a common "Open File" dialog to the user and returns the full path/name of the selected file.

Syntax

```
inputfile([path], [extension], [caption])
```

path	The default path for the dialog. When left empty or non-existent, path defaults to the

current path. In the latter case you are prompted whether to continue the script.

extension	A list of extensions (separated by pipe (|) or semi-colon (;), no dots) of the files to list on the window.
caption	The text that will be displayed as title of the window.
return	The full path/name of the selected file.

Usage

If you press Cancel, the script execution will be aborted.

Examples

```
$file = inputfile("<curpath>", "rar;zip", "Select an archive to extract in the current folder"); open quote("winrar") . " x " . quote($file) . " " .  quote("<curpath>\");
```

Asks the user to select an archive (RAR or ZIP), puts its full name (with path) in variable $file. Uses WinRAR (must be a registered application on Windows) to extract its content to current location.

```
$file = inputfile("Desktop", "gif|jpg|png", "Select Image File"); text $file, 600, 200;
```

Asks the user to select an image file in folder Desktop. Then shows the selected file name in a textbox.

inputfolder()

Shows a common "Browse For Folder" dialog to the user and sets a variable to the full path/name of the selected folder.

Syntax

```
inputfolder([path], [caption])
```

path	The default path for the dialog. When left empty or non-existent, path defaults to the current path. In the latter case you are prompted whether to continue the script.
caption	The text that will be displayed as title of the window.
return	The full path/name of the selected file.

Usage

If you press Cancel, the script execution will be aborted.

Paths are returned without trailing backslash

The paths returned in scripting, either through variables or commands (like inputfolder), are never backslashed (no final \) so that you can use them in a predictable way. E.g. if the current location is C:\ then <curpath> will only return "C:".

Examples

```
$folder = inputfolder("C:\", "Select Folder"); copytext $folder;
```

Asks the user to select a folder (starting the dialog in C:\) and puts its full name in variable $folder. Then copies the folder name to clipboard.

```
$source = inputfolder("C:\", "Select folder to copy into current location"); copyto "<curpath>", $source;
```

Asks the user to select a folder (starting the dialog in C:\) and puts its full name in variable $source. Then copies the folder into the current location.

inputselect()

Pops a list of strings from which the selected one is returned.

Syntax

```
inputselect(header, listdata, [separator="|"], [style=1], [cancel], [width=800], [height=400], [windowcaption], [preselectprefix], [icon], [preselectitem])
```

header	First line is printed in bold; any other lines are printed non-bold; leave empty to show no header.
listdata	String of list items, separated by separator.
separator	Separator; defaults to |.
style	1 = Show file system icons for the items (= default). 2 = Show checkboxes; prefix items with "+" to pre-check them. 4 = Show empty list if there is only one empty item. 8 = Show generic icons (just from extension, files need not exist). 16 = Allow user to re-order items with drag & drop. Also Ctrl+Up/Down can be used. 32 = Auto select first item in list. 64 = Focus filter box initially. 128 = Return selected index (first item = "1"). If nothing is selected, "0" (zero) is returned. Note: Cannot be meaningfully combined with style 16. 256 = Show large (32x32) icons. Note that style bit 1 must be set as well (to show icons at all). 512 = Return all items in their current order. Of course, this is only useful when combined with style 16, otherwise the order is always the one from the original *listdata*. 1024 = The *listdata* argument has fields (see Examples for style 1024). 2048 = Show file items without the path (filename only). 4096 = Show line numbers in the list. 8192 = Hide filter box. 16384 = Use Edit Text font (instead of the default Buttons and Labels font). 32768 = Pre-check all checkboxes. 65536 = Position window at mouse pointer.

cancel	Value to be returned when user cancels; if not set then cancel will terminate the script.
width	Width of window in pixels (minimum: 150; maximum: screen width; default: 800). Percentages of screen width are supported, e.g. 75%.
height	Height of window in pixels (minimum: 150; maximum: screen height; default: 400). Percentages of screen height are supported, e.g. 75%.
windowcaption	Window caption.
preselectprefix	One or more characters that are prefixed to the item to preselect. Only one item should be marked like this. Any following items (checked from left to right) are ignored.
icon	The key to a toolbar icon, prefixed by ":", e.g. ":dice". It also accepts an image file (GIF, JPG, PNG). It can be a full path, or just a filename (auto-resolved to the default icons path <xyicons>).
preselectitem	This item in listdata will be preselected. The first match wins (checked from left to right), any following matches are ignored. Anything set in preselectprefix is overridden by preselectitem.
return	The selected list item. If checkbox style then the checked items are returned, separated by separator.

Notes

- If the list shows only one item OK will trigger it even if it is not selected.
- The preselected item is centered vertically in the visible portion of the list, allowing you to see adjacent items above and below.

Examples

Show header and a list of drives; then go to the selected drive:

```
goto inputselect("Select Destination", "C:|D:|E:");
```

Same with two-line header:

```
goto inputselect("Select Destination".<crlf>."Choose drive:", "C:|D:|E:");
```

No header:

```
goto inputselect(, "C:|D:|E:");
```

Use checkboxes and icons (1 OR 2 = 3):

```
text inputselect("Select Drives", "C:|D:|E:", , 3);
```

Use checkboxes, pre-check "D:"); return "-" on cancel:

```
text inputselect("Select Drives", "C:|+D:|E:", , 3, "-");
```

Show checkboxes and return selected indices:

```
text inputselect("Test", listfolder(<curpath>), , 1 + 2 + 128);
```

Shows one empty item:

```
inputselect("Test 3/1",  listfolder(%windir%, "s*", 3), , 1);
```

Shows empty list:

```
inputselect("Test 3/5",  listfolder(%windir%, "s*", 3), , 5);
```

Auto select + Focus filter:

```
text inputselect("Test",  listfolder(<curpath>), , 96);
```

Examples for preselecting drive "D:":

```
text inputselect("Select Drives", "C:|+D:|E:", , 1, "-",,,,"+");
```

```
text inputselect("Select Drives", "C:|###D:|E:", , 1, "-",,,,"###");
```

```
text inputselect("Select Drives", "C:|<U+9>D:|E:", , 1, "-",,,,"<U+9>");
```

Examples for small/large icons:

```
goto inputselect("Select Destination", "C:|D:|E:",, 1);     //small icons
```

```
goto inputselect("Select Destination", "C:|D:|E:",, 1+256); //large icons
```

Example for preselecing an item. Parameter "style" is set to 32: if preselectitem has no match the first item is preselected (won't happen in this example):

```
echo inputselect("Select Country", "Afghanistan;Albania;Algeria;Andorra;Angola", ";", 32, 10:
="Angola");
```

Example for "Return all items in their current order":

```
text inputselect("Test", "a|b|c|d", 3:=512+16);
```

Example for listing the current folder without paths:

```
text inputselect("Test", listfolder(), , 1 + 2048);
```

Examples for font options:

```
text inputselect("Test", listfolder(), , 1);          // Buttons and Labels font
```

```
text inputselect("Test", listfolder(), , 1 + 16384);  // Edit Text font
```

```
text inputselect("Test", listfolder(), , 0b1000000000000001);  // Edit Text font (binary
number)
```

Examples for icons:

```
goto inputselect("Select Destination", "C:|D:|E:", 9:="rock.ico");
```

```
goto inputselect("Select Destination", "C:|D:|E:", 9:="rain.jpg");
```

```
goto inputselect("Select Destination", "C:|D:|E:", 9:="D:\pics\rain.jpg");
```

```
goto inputselect("Select Destination", "C:|D:|E:", 9:=":dice");
```

Examples for pre-check:

```
text inputselect("Select Food", "Soup|Cheese|+Cake",, 2); //Cake pre-checked
```

```
text inputselect("Select Food", "Soup|Cheese|+Cake",, 2+32768); //all pre-checked
```

Examples for style 1024

You can define up to three predefined fields in the *listdata* argument, separated by |: `Caption|Data|Icon`

Caption	Caption to display in the list.
Data	String returned for the selected list item on OK. If the Data field is missing/empty the Caption field is returned.
Icon	Full path to a file item used for the icon that's shown in the list. Also supports XYplorer Toolbar icons by using the ":iconkey" syntax. You can use "*.<DIR>" to specify the generic folder icon. If the Icon field is missing/empty the Data field is used for the icon. If the Data field is also missing/empty the Caption field is used for the icon. Note that "style" needs bit 1 to actually show any icons.

Remark: You obviously cannot use the default separator "|" to separate the items from each other when you use these "|"-separated fields. Any separator without the "|" character will do.

Example:

```
  $list = <<<>>>
C:\Program Files (x86)\7-Zip\7zFM.exe
7-Zip|C:\Program Files (x86)\7-Zip\7zFM.exe
7-Zip|7-Zip Selected|C:\Program Files (x86)\7-Zip\7zFM.exe
>>>;
  text inputselect("Test: Caption|Data|Icon", $list, <crlf>, 1 + 1024);

  // 1st entry: Caption           -> Caption, Return, Icon: C:\Program Files (x86)\7-Zip\7zFM.exe
  // 2nd entry: Caption|Data      -> Caption: 7-Zip; Return, Icon: C:\Program Files (x86)\7-Zip\7zFM.exe
  // 3rd entry: Caption|Data|Icon -> Caption: 7-Zip; Return: 7-Zip Selected; Icon: C:\Program Files (x86)\7-Zip\7zFM.exe
```

Using XYplorer Toolbar icons:

```
text inputselect("Test", "Delete|User selected Delete|:del<crlf>Ghost|User selected the Ghost|:ghost", <crlf>, 1 + 1024);
```

interfacecolors, ifc

Sets or gets custom interface colors.

Syntax

```
interfacecolors([colors], [type=0])
```

colors	New custom interface colors. Format per color: type 0 (Default): RRGGBB[|RRGGBB] (backcolor|textcolor)

type 1, 2: [RRGGBB,]RRGGBB (textcolor,backcolor)
type 3: AT1,AB1,IT1,IB1,AT2,AB2,IT2,IB2 (A=Active,I=Inactive,T=Text, B=Back,1=Pane1,2=Pane2; each color in RRGGBB format; omitted values remain unchanged)
Empty: Reset interface colors to Windows defaults.
Missing: Just return current custom interface colors.

type	Type of colors: 0: The so-called "Button Face Color/Button Text Color" which Microsoft describes as the "Background/Text color of controls" (controls with variable content like Edit Boxes and Lists are not included here). 1: The Tree Section Colors. Sections in the colors parameter: `[Reserved]|SpecialFolders|UserFolders|Drives|PortableDevices|RecycleBin|Network` Leave a section empty to use the default tree colors in that section. 2: The Folder Row Colors. 3: The Breadcrumb Colors.
return	Current custom interface colors of the selected type.

Remarks for Type 0

- Colors are to be stated in hexadecimal RRGGBB format (red, green, blue).
- If you set the textcolor then checkboxes and radio buttons lose their Windows Theme Style. It's the only way to change their text color.
- It is recommended to choose a backcolor that is lighter than the textcolor, otherwise you will have readability problems in some areas of the interface.
- Dark Mode is completely unaffected by the colors defined here.

Remarks for Type 1

- Colors are to be stated in hexadecimal RRGGBB format (red, green, blue).
- Tick Tools | Customize Tree | Show Section Colors to actually show the colors in the tree. See also there.

Examples for Type 0

```
echo interfacecolors(); //show current custom interface colors

interfacecolors("");    //reset interface colors to Windows defaults

interfacecolors("E9E6E3|124578"); //set backcolor and textcolor

interfacecolors("EC8F32|C07532"); //New York City smoke mode

interfacecolors("|124578");       //set textcolor only

interfacecolors("E9E6E3|");       //set backcolor only

interfacecolors("E9E6E3");        //set backcolor only
```

Examples for Type 1

```
echo interfacecolors(,1);  //show the current tree section colors definition

interfacecolors("", 1); //reset all tree section colors to the factory defaults

interfacecolors("|0055AA,F5F8FC|||773311,F8F0E0|006622,D0F8E0|225544,EFFAFF", 1); //set these tree section colors
```

Examples for Type 2

```
echo interfacecolors(, 2); //display the current folder row colors definition

interfacecolors("224488,DDEEFF", 2); //make them blue

interfacecolors("", 2); //reset the folder row colors to the factory defaults
```

Examples for Type 3

```
echo ifc(, 3); //display the current breadcrumb colors definition

ifc("", 3); //reset all breadcrumb colors to the factory defaults

ifc("EEFFEE,5C879E,FFFFEE,A1B6CC,FFEEEE,A06072,FFFFEE,C9A4B1", 3); //set all breadcrumb colors to custom values

ifc(",5C879E,,A1B6CC,,A06072,,C9A4B1", 3); //modify just the back colors

ifc(",7CB76E", 3); //modify just the back color of the active breadcrumb in pane 1
```

internetflags

Tunes internet commands (Download, ReadURL).

Syntax

```
internetflags name, [value]
```

name	[Required] INTERNET_FLAG_NO_COOKIES (no other flags are currently supported).
value	[optional] 1: On [default] 0: Off

Examples:

The default is to use cookies:

```
$a = input("Enter URL",, <clipboard>);
  download $a;
```

This will not use cookies:

```
internetflags "INTERNET_FLAG_NO_COOKIES";
  $a = input("Enter URL",, <clipboard>);
  download $a;
```

isset()

Determines if a variable is set.

Syntax

```
isset($var)
```

$var	The variable to be checked.
return	0: The variable is not set. 1: The variable is set.

Examples

For example, check whether a permanent variable is set before using it:

```
$contents = isset($p_filenamestore)?$p_filenamestore:"";
```

isunicode()

Analyzes a text string for the presence of wide characters.

Syntax

```
isunicode(string, [mindcodepage])
```

string	Full path/file name to analyze.
mindcodepage	[optional] 1: do not count wide characters that have a narrow pendant in the current codepage.
return	1 if wide characters are present in string, else 0.

Examples

```
echo isunicode("abc"); //0

echo isunicode("abc", 1); //0

echo isunicode(chr(20000)); //1

echo isunicode(chr(20000), 1); //1

echo isunicode("€"); //1 (U+20AC EURO SIGN)

echo isunicode("€", 1); //0! e.g. on 1252 Western codepage,
  where the Euro sign is mapped to 0x80
```

Also these refer to the Euro sign:

```
echo isunicode(chr(8364), 0); //True
```

```
echo isunicode(chr(8364), 1); //False on 1252 Western codepage
```

itematpos()

Returns the item at a certain screen position.

Syntax

```
itematpos([x], [y], [flags])
```

x	(optional) Defaults to the X mouse position on screen.
y	(optional) Defaults to the Y mouse position on screen.
flags	
	0: If x and y are passed, they are the position on screen.
	1: If x and y are passed, they are the position on XYplorer.
	2: Return special path (eg "Downloads\Alice") if the item at that position is displayed that way, else always convert to real path (eg "C:\Users\Donald\Downloads\Alice").
return	The item that would be selected by a click on this position.
	Nothing if no item would be selected.

Returns per Control

Control	Return	Control Short Form
List 1	Item at X/Y	L 1
List 2	Item at X/Y	L 2
Tree	Item at X/Y	T
Catalog	Item/Whatever at X/Y	C
Address Bar	Current Text	A
Toolbar	Button Caption at X/Y	TB
Tab Bar 1	Location of Tab at X/Y	TAB 1
Tab Bar 2	Location of Tab at X/Y	TAB 2
Breadcrumb 1	Path at X/Y	BC 1
Breadcrumb 2	Path at X/Y	BC 2
Status Bar	Section Text at X/Y	SB

"Control Short Form" is returned by controlatpos().

Remarks

- The function returns nothing if XYplorer is not the foreground window.

- Folders are returned without trailing slash.
- In case of special folders, the function returns the real path: "C:\Users\Donald\Desktop", not "Donald\Desktop".

Examples

```
echo itematpos(); //item at current mouse position on screen
echo itematpos(592, 662); //item at arbitrary mouse position on screen
echo itematpos(73, 108, 1); //item at arbitrary mouse position on XYplorer
```

lax()

Allows to define a string in a lax way.

Syntax

```
lax(string)
```

string	String expression treated in a lax way. Variables are resolved. Everything else is just left as it is. All single and double quotes are preserved. No "Dubious Syntax" warnings.

Use

Makes it easier to define complex arguments with lots of quotes.

Examples

```
echo lax(hi...          there!);  //no quotes necessary, spaces are preserved
echo lax(hi, hi, hi);  //commas are ignored
echo lax('<curitem>');  //<curitem> is resolved, quotes are preserved
run lax("C:\Program Files (x86)\WinRAR\WinRAR.exe" x "<curitem>");
```

listfolder()

Lists the contents of a folder (non-recursive, i.e. not including subfolders).

Syntax

```
listfolder([path=<curpath>], [pattern=*], [flags], [separator="|"])
```

path	Path of folder to list; defaults to current list path. Special paths ("Desktop") and portable paths ("?:\") are supported. Recycle Bin (%recycler%) is not supported.
pattern	Wildcard (?, *) pattern to filter returned items; defaults to "*" (show all). If no wildcards are contained, the pattern is auto-embraced by "*" (unless flag

	NoAutoWildcards is set). Can as well be a list of patterns separated by | or ;.
flags	0: return folders and files with full paths [default] 1: NoFolders (= return only files) 2: NoFiles (= return only folders) 4: NoPaths (= return the filenames without paths) 8: NoAutoWildcards (= allow exact matches) 16: NoCharacterLists (= useful when you want to match square brackets) 32: ReturnCount (= return the count instead of the items) 64: KeepSpecial (= return special paths)
separator	Separates the items in the returned list; defaults to | (pipe).
return	List of items.

Note

The order of items is determined by the OS or file system. For XP/NTFS and later it's alphabetical, files and folders mixed.

Examples

List all items in the current list folder:

```
text listfolder();
```

List all items in %windir% that contain "y" in the name:

```
text listfolder(%windir%, "y", , <crlf>);
```

List all items in %windir% that contain "x" or "y" in the name:

```
text listfolder(%windir%, "x;y", , <crlf>);
```

List all folders in %windir% that begin with "s":

```
text listfolder(%windir%, "s*", 2, <crlf>);
```

List all folders in %windir% that begin with "s"and return the names without path:

```
text listfolder(%windir%, "s*", 2 + 4, <crlf>);
```

Using NoCharacterLists (16):

```
inputselect("Test", listfolder(, "[e]", 0));  // match all *e*
```

```
inputselect("Test", listfolder(, "[e]", 16)); // match all *[e]*
```

Using ReturnCount (32):

```
echo listfolder(, "*.jpg", 1 + 32); //count of *.jpg files in current folder
```

Using KeepSpecial (64):

```
inputselect("Test", listfolder("%personal%", , 2));  //lists subfolders using real paths (eg "C:\Users\Donald\Documents\Guitar")
```

```
inputselect("Test", listfolder("%personal%", , 2 + 64));  //lists subfolders using special paths (eg "Documents\Guitar")
```

List all DLL files in %windir% (typically C:\Windows\) and give a short report on them:

```
$itemlist = listfolder(%windir%, "*.dll", 1);
  text report("{Name}, {Size B} bytes, {Modified yyyy-mm-dd hh:nn:ss},
  ver {FileVersion}<crlf>", $itemlist);
```

List all DLL files in %windir%, select one to go to:

```
$itemlist = listfolder(%windir%, "*.dll", 1);
  goto inputselect("DLLs in %windir%", $itemlist, , 1);
```

listpane()

Lists the contents of a pane.

Syntax

```
listpane([pane=a], [pattern=*], [flags], [separator="|"])
```

pane	Pane to list. a: [default] active pane i: inactive pane 1: pane 1 2: pane 2
pattern	Wildcard (?, *) pattern to filter returned items; defaults to "*" (show all). Also "character lists" in square brackets are supported, e.g. [ab] to match "a" OR "b". If no wildcards are contained, the pattern is auto-embraced by "*" (unless flag NoAutoWildcards is set).
flags	(bit field) 0: return folders and files with full paths [default] 1: NoFolders (= return only files) 2: NoFiles (= return only folders) 4: NoPaths (= return the filenames without paths) 8: NoAutoWildcards (= pattern must be exact match) 16: NoCharacterLists (= useful when you want to match square brackets)
separator	Separates the items in the returned list; defaults to | (pipe).
return	List of items.

Notes

- As you might have noted, this command is analogue to listfolder() with the exception of the first argument.
- The command also works on Search Results, Drives, Recycle Bin and whatever list modes we will get in the future.
- The order of items is determined by the current list order.

Examples:

```
inputselect("Items on pane 1", listpane(1));
inputselect("Items on pane 2", listpane(2));
inputselect("Items on active pane", listpane());
inputselect("Items on inactive pane", listpane("i"));
inputselect("Items a* on active pane", listpane(, "a*"));
//list all folders that are named "code" in active pane:
inputselect("Test", listpane(, "code", 2 + 8));
```

Examples for Character Lists and flag NoCharacterLists (16):

```
//list all items that contain "a" or "b" in the name:
inputselect("Test", listpane(, "[ab]"));
//list all items that contain "[ab]" in the name:
inputselect("Test", listpane(, "[ab]", 16));
```

llog

Pops a dialog showing the current load log (aka load times).

Syntax

```
llog [threshold=10]
```

threshold — Threshold in milliseconds to filter the load times listing. The default is 10 ms, i.e. only events that took at least 10 ms are listed.

Example

```
llog;   //10 msecs threshold
llog 0; //no threshold
```

load

Loads a script or a script file and either pops up a menu listing the scripts contained, or directly executes one of the scripts.

Syntax

```
load resource, [labels], [resource_type]
```

resource — [Required] Can be one of the following:
(1) The path/name of a script file.
(2) A script or multi-script resource, actually anything that could work as contents of a script file.
Note 1: If you use load inside a resource you can also use * as resource, to specify the

current resource. This is very useful to ensure that your scripts will keep self-referencing even if the script file is renamed, or to allow a resource to load its menu even when not used from an actual script file (for example from User-Defined Command "Run Script" or menu Scripting | Try Script...)
Note 2: If the load command is within a loaded script file, you can omit the path of the loaded file if it is in the same location as the containing script file.

labels — This can either be
(1) A script label: must be a valid label of one of the scripts contained in the resource.
(2) A list of script labels, separated by semi-colons (;) or by CRLF. To add a menu separator you may put "-" instead of a label. Note that, of course, the label list must be in quotes in order to parse the ";" as expected.
(3) A number prefixed by #: must be the index of which script to execute. Note than only scripts get an index, while menu separators do not. The first index is 1.
(4) An asterisk (*) to show all (hidden and visible) scripts in the resource that have a caption.
Note that if the given labels cannot be found, XY will pop up the menu from the file as if no labels had been specified.

resource_type — Defines the type of resource:
f: [Default] File
fc: File with caching
s: Script

Loading a Script File

The file name can be specified with full or relative path. In the latter case the script file to load is looked for

(1) in the path of the calling script file (if any)

(2) in the Scripts subfolder of the application data path (variable <xyscripts>)

(3) in the application data path (variable <xydata>)

If the given filename does not contain an extension, XY will automatically add .XYS as its extension ("XYplorer Scriptfile"). So, to load the file MyScript.xys located in the application data folder, all you need to specify as parameter is MyScript: `load "MyScript"`

For the syntax of XYplorer Script Files see here.

Loading a Script File with Caching

Loads files from an internal cache instead of reading them from disk again and again.

- Only useful in scripts where the same resource file is loaded repeatedly.
- Always the last loaded file contents are cached. So if you load the same file again (with no other file loaded in between) the cache will be used.
- The cache does not survive between scripts.
- You won't note much of a difference normally since the Windows disk read cache is pretty effective. But in case you turned off that cache, or your disk reading is slow for whatever other reason using

the "fc" setting will speed up your script immensely.

Examples:

```
load 'dostuff.xys';          // without caching
load 'dostuff.xys', , 'fc'; // with caching (if loaded previously)
```

Loading a Script

Simply put the script as first argument. The syntax for the script is exactly the same as for Script Files, it can contain more than one script in which case a menu will be loaded and popped up, unless you have specified a script to be executed (using index or label).

This are some simple examples:

```
load 'goto "C:\"', , s
load 'msg "A loaded script.";', , s
```

Of course, these examples don't make a lot of sense since you could just as well call the scripts directly. Loading a script this way starts making sense when you create the script on the fly.

Examples

```
load "myscript"
```

Will load the file myscript.xys located in the calling script's folder, or if not found (or not used from a script file) then in the application folder.

```
load "?:\XYscripts\goto.xys"
```

Will load the file "Goto.xys" (see below) located in the folder "XYscripts" on the drive XYplorer is currently running from, and pop up a menu presenting all scripts contained in that file.

```
load "?:\XYscripts\goto.xys", "system"
```

Will load the file "Goto.xys" (see below) located in the folder "XYscripts" on the drive XYplorer is currently running from, and execute the script whose label is "system".

The contents of goto.xys could look like this:

```
"Go to C:\ : croot"
  goto "C:\"
"Go to System Folder : system"
  goto %winsysdir%
"Go to XYplorer Folder : xy"
  goto <xypath>
```

Using a list of labels

All the following examples use the above displayed script file goto.xys, and expect it to be located in the default Scripts path.

(1) Make no use of the labels argument:

```
::load "goto.xys"
```

Pops up menu with all scripts contained in the file in original order:

Go to C:\

Go to System Folder

Go to XYplorer Folder

(2) Use the labels argument to pick only two of the scripts:

```
::load "goto.xys", "croot;xy"
```

Pops up menu:

Go to C:\

Go to XYplorer Folder

(3) Use the labels argument to change the order and add a menu separator:

```
::load "goto.xys", "xy;-;system;croot"
```

Pops up menu:

Go to XYplorer Folder

Go to System Folder

Go to C:\

loadlayout()

Loads or saves a layout.

Syntax

```
loadlayout(file, [mode="load"])
```

file	File to save the layout to, or to load it from. Either full path or resolved relative to <xydata>\Layouts\. If missing you are prompted by a standard Windows dialog. If no extension is passed the extension defaults to ".txt".
mode	
load: [Default] Load layout from file.	
save: Save layout to file.	
return	The layout loaded or saved.

Examples

```
loadlayout("default"); //loads layout from <xydata>\Layouts\Default.txt
loadlayout("default", "save"); //saves layout to <xydata>\Layouts\Default.txt
loadlayout(); //loads layout, file is prompted
loadlayout(, "save"); //saves layout, file is prompted
```

See also setlayout.

loadsearch

Load a previously stored search template and (optionally) run a search based on it.

Syntax

```
loadsearch template, [options=rl]
```

template [required] search template name as displayed in the search template dialog

options [optional, default = rl]

r = Run Search: Run search after the template is loaded. Else just the Find Files tab is adjusted to the loaded template.

l = Load Search Location: Search the location that's stored with the template. Else search the current location.

e = Load Excluded Folders: Exclude folders as stored with the template. Else use the currently excluded folders.

c = Load Cached Results (if any): Instead of doing a fresh search load the results previously saved in the template. If this option is set the other options (r, l, e) are auto-implied.

x = (none of the above)

Examples

```
::loadsearch "Created or Modified This Hour", r
```

Load template "Created or Modified This Hour" and run a search in the current location.

```
::loadsearch "Reset Find Options", x
```

Load template "Reset Find Options" and do nothing else.

```
::loadsearch "Videos in Temp Internet Files"
```

Load template "Videos in Temp Internet Files" and run a search in the stored location (Temporary Internet Files).

Remarks

Search templates are INI-files that are all stored in a predefined folder: <xydata>\FindTemplates. The template name (as used in loadsearch) is usually identical to the base name of the template file. Template names, however, may contain characters that are illegal for file names; these characters are

encoded in the filename using the scheme "% & Hex(charcode)". You create and manage search templates using menu Edit | Search Templates.

The loadsearch command is the nexus of two of XYplorer's strongest features: Scripting and File Search. In combination with the Catalog, User-Defined Commands, and XYS files, you can easily build handy libraries of complex live searches:

(1) First configure your search settings on the Find Files tab.

(2) Then save the settings as a search template using menu Edit | Search Templates.

(3) Then write a script `::loadsearch "[name of your template]"` and test it.

(4) Finally wrap the script in a Catalog item, User-Defined Command, or XYS file. Or simple add it to your Favorite Folders (yes, they support one-line scripts).

loadsettings

Restarts XYplorer with different settings.

Syntax

```
loadsettings path, [flags]
```

path	Path of new settings. Can be: - INI file base only: "new" [defaults to "<xydata>\new.ini"] - INI file only: "new.ini" [defaults to "<xydata>\new.ini"] - path to new app data folder: "E:\path" [defaults to "E:\path\XYplorer.ini"] - path to new app data folder plus INI file: "E:\path\new.ini"
flags	(bit field) 1: Keep current instance open (else it's closed). Works even if this is OFF: Configuration | Startup & Exit | Allow multiple instances

Remarks

Contrary to File | Settings Special | Load Configuration... (which only loads a new INI file) SC LoadSettings can set a new app data folder. This means it can load a whole different set of settings (Catalog, Tags, Keyboard Shortcuts, Tabsets, etc).

Examples

Load just new INI file:

```
loadsettings "new.ini"; //INI file <"xydata>\new.ini"
loadsettings "new";     //same as above (extension defaults to *.ini)
```

If the path is included this path will be the new appdata path:

```
loadsettings "E:\XYplorer\appdata_screenshots\";  //default to "XYplorer.ini" in new appdata path
loadsettings "E:\XYplorer\appdata_screenshots";   //IF the path exists: same as above
                                                  //If NOT: load "E:
```

```
\XYplorer\appdata_screenshots.ini"
loadsettings "E:\XYplorer\appdata_screenshots\new.ini"; //"new.ini" in new appdata path
```

Keeping the current instance open:

```
loadsettings "E:\XYplorer\appdata_screenshots\", 1;
loadsettings <curpath>, 1;
```

loadtree

Loads a specific Mini Tree.

Syntax

```
loadtree [pathlist], [modify=0], [flags]
```

pathlist — CRLF- or |-separated list of paths.
XYplorer native variables and Environment variables are supported;
if modify = 0 then the current tree folder is set to the first path;
defaults to the current path if missing.

modify

0: [default] fresh Mini Tree from scratch
1: add paths to current Mini Tree
2: hide paths from current Mini Tree
3: reduce the Mini Tree to just the paths of all tabs in the current pane

flags — (bit field)
1: do not trigger browsing the selected path after the tree is loaded
2: Expand the selected node (if it is not already expanded).

Remark

With modify = 1 and the current tree still non-Mini, loadtree will convert the tree to a Mini Tree without losing any of its present folders. Interesting option! You can convert a Maxi Tree to a Mini Tree without adding any paths using this line: `loadtree , 1;`

Examples

```
loadtree "C:|D:";
loadtree "<xydata>\scripts";
loadtree "<curpath>|%programfiles%|<xypath>";
loadtree "%temp%", 1; //add TEMP dir to the current tree
loadtree "E:|F:", 2; //hide drives E: and F:
loadtree "", 3; //minimize tree to just the paths of all tabs in the current pane
```

logon

Pops a logon dialog if the network location is password protected and not already unlocked.

Syntax

```
logon network_path
```

network_path	UNC path or mapped drive path.

Note

If the location is not password protected or already unlocked then nothing happens at all.

Examples (abstract):

```
logon "\\server\share[\sub]";
logon "X:\Mapped\Drive\Path";
```

makecoffee

Makes a good cup of coffee.

Syntax

```
makecoffee
```

Example

```
makecoffee;
```

Remark

The coffee is served on the current pane. Call again for another cup.

md5()

Calculates the md5 hash of a string.

THIS COMMAND IS DEPRECATED! Use hash() instead.
md5() is kept only for older Windows versions where hash might not be supported.

Syntax

```
md5(string, [flags])
```

string	String to md5. Either the data itself, or a file spec (depends on flags).
flags	(bit field) 0: string is data that will be md5-ed. The maximum allowed string size is 512 MB.

1: string is a file spec whose contents will be md5-ed. The maximum allowed file size is 512 MB.

Examples

```
echo md5(""); //d41d8cd98f00b204e9800998ecf8427e
echo md5("XYplorer"); //72b70e05bade8928abc97cc4a53abf48
echo md5("<xypath>\<xyexe>", 1);    //md5 of the current application
```

moveto, copyto, backupto

Move/Copy/Backup the selected items to the specified location.

Syntax

```
moveto [location], [source], [rootpath], [flags], [autorichop], [on_collision],
[preserve_dates], [create_log], [pop_stats], [custom_flags], [verify], [show_progress]
```

```
copyto [location], [source], [rootpath], [flags], [autorichop], [on_collision],
[preserve_dates], [create_log], [pop_stats], [custom_flags], [verify], [show_progress]
```

```
backupto [location], [source], [on_collision], [preserve_dates], [create_log], [pop_stats],
[custom_flags], [verify], [show_progress], [filter]
```

location	[optional] The destination where the selected items should be moved/copied/backed up to. Defaults to the current path.
source	[optional] The item that will be copied or moved. Can be a list of items, separated by CRLF or |. Alternatively it can point to the source control (":tree", ":list", or ":catalog"), or to a text file containing a list of source items ("*filename"; see below). If missing then the selected items of the focused control (List, Tree, Catalog) are used.
rootpath	[optional; moveto/copyto only] Common root for a rich operation (source paths are recreated in the target location relative to rootpath). Slashed or unslashed makes no difference.
flags	[optional; moveto/copyto only] (bit field) 0 = [Default] 1 = FilesOnly. For sources containing wildcards only files are copied. 2 = SkipPromptToCreate. Don't prompt, but create any non-existing destination folder without asking.
autorichop	[optional; moveto/copyto only] 0 = [Default] Ask 1 = Always

2 = Never
This flag temporarily overwrites the global flag AutoRichFileOps.

Note for copyto / moveto: If you use any of the following parameters from [on_collision] to [show_progress] then the operation is forced to be Custom Copy / Move because Shell Copy/Move does not support these settings. For all non-passed parameters the global user settings are used as specified in Configuration | File Operations | Custom Copy Operations | Configure....

on_collision [optional]
-1 = Ask
0 = [Default] Use global setting from Configuration | File Operations | Backup Operations | On name collisions
1 = Overwrite if newer
2 = Overwrite
3 = Skip
4 = Suffix number to copy
5 = Affix current date to copy
6 = Affix last modified date to copy
7 = Suffix number to existing
8 = Affix current date to existing
9 = Affix last modified date to existing
10 = Overwrite if different size or date
11 = Overwrite if different contents (The contents are compared by comparing the SHA-256 hash of each file.)

preserve_dates [optional] Preserve all three file dates (only applied if the file itself is copied).
Empty = Use global setting from Configuration.
0 = Whether/which dates are copied depends on OS.
1 = All three dates are copied.

create_log [optional] Create a detailed log file about the backup operation.
Empty = Use global setting from Configuration.
0 = Don't.
1 = Do.

pop_stats [optional] Pop up a summary message about what was copied / overwritten / not overwritten.
Empty = Use global setting from Configuration.
0 = Don't.
1 = Do.

custom_flags [optional] (bit field)
Empty = Use global settings from Configuration.
0 = Copy junctions as folders.
1 = Skip junctions.

2 = Copy junctions as junctions (admin rights required).
4 = Exclude hidden files and folders (referring to the HIDDEN file attribute).
If items with the HIDDEN attribute are among the selected items (or among the items explicitly specified as source items) then they are NOT excluded.
Affects backupto and copyto only; ignored with moveto.

verify [optional] Verify each copy operation on the fly.
Empty = Use global setting from Configuration.
0 = None (no verification)
1 = Byte-to-byte
2 = MD5
3 = SHA-1
4 = SHA-256
5 = SHA-512

show_progress [optional] Show progress dialog.
Empty = Use global setting from Configuration.
0 = Don't.
1 = Do.

filter [optional; backupto only] List of patterns or full paths, separated by "|", used to include or exclude files or folders in/from the operation by name.
See sync for a detailed description.

Usage

Non-Existing Paths

You can use non-existing paths as destination. If you do so, you will be prompted whether or not you want to create that path before continuing with the file operation.

Note that all parts can be new, not just the last subfolder. So D:\new\new too\also new will work just fine.

Variables

You can also use Environment Variables (XY supports both common Windows & XY-specific environment variables) as part of your destination, as well as XY variables and, when scripting, user-variables. For more on all those variables, please refer to Script Variables.

Relative Paths

You can use relative paths too. Unlike everywhere else on XY, relative paths here on destination for your Move/Copy/Backup To operations refer to the current location, and not XY's application path.

So to quickly move/copy/backup into a newly-created subfolder (of the current location), simply enter its name. If the subfolder doesn't exist, you'll be prompted whether to create it or not!

Using Date Variables

You can also use date variables in the destination, that will be resolved with the current date! The syntax here is identical to the one you know from the Batch Rename, and the date format syntax is the same you know from other XY date terms, like in File Find.

Examples:

```
NewName<date yyyy>                  NewName2006
NewName<date yyyy-mm-dd_hh-nn-ss>   NewName2006-08-16_15-49-03
```

Used together with the ability to enter non-existing paths, and relative ones, this is an extremely valuable addition! For example, you now can easily make regular dated backups by using a destination like this: D:\Daily Backups\<date yyyy-mm-dd> Then, when you Move/Copy/Backup To this destination:

(1) If not existing already, the folder (eg: D:\Daily Backups\2006-10-07) will be created

(2) The selected items are moved/copied/backed up there

Note that you even can have more than one date variable in the destination field. So you can also use something like: D:\Backups\<date yyyy>\<date mm>\<date dd>\<date hh'nn> to get a folder for the year, one for the month, one for the day, and one for the hour! (e.g.: D:\Backups\2006\10\07\10'42)

Source

You can specify a source for the operation. The source must be the path to a folder or a file, or a control. You can use relative paths (relating to XYplorer's application folder), and variables (see examples below). When the source is a path, a trailing backslash is allowed but not necessary.

You can also use multiple sources in a CRLF- or |-separated list, for example:

```
backupto "D:\Backup\XY folders\<date yyyy-mm-dd>", "<xydata>\Scripts|<xydata>\FindTemplates"
```

The source(s) may contain wildcards, for example:

```
copyto "D:\Backup\XY files\<date yyyy-mm-dd>", "<xydata>\*.ini|<xydata>\*.dat"
```

Copies all *.ini and *dat file from the XYplorer data path to D:\Backup\XY files\2008-04-22\ (if that's the date today).

Note: Any Tags are not moved/copied along when the sources contain wildcards.

Control as Source

The "source" argument can as well point to the source control by setting it to ":list", ":tree", or ":catalog". This makes the selections used as sources independent of the current input focus. Examples:

```
//copy all selected list items to "C:\Temp"
copyto "C:\Temp", ":list";
//copy the selected tree item to "C:\Temp"
copyto "C:\Temp", ":tree";
```

Source Items File as Source

You can pass a file containing a source items list as source. The source items file should be an ASCII or UNICODE text file, and containing one source item per line. Empty lines are ignored. Folders can be slashed or not. Example for the contents of a source items file:

```
E:\Test\file-01.txt
E:\Test\file-02.txt
C:\Temp\folder\
```

To mark the source argument as source items file simply prefix it with "*" (asterisk). Example:

```
copyto "E:\Target\", "*E:\Test\Sources.txt";
```

Remarks:

- This feature allows you to easily trigger "distributed file operations", i.e. operations on a set of items that cannot easily be defined by selecting the items in the file list. This can mean an enormous usability boost.
- The *-prefix was borrowed by TeraCopy, thanks for that!

Using Variables referring to the Source

When you specify a source, you will be able to use all variables <src...> in your destination location, which will refer to the first item in source:

<srcname>	the full name of the first source item without path
<srcbase>	the base name of the first source item without extension
<srcext>	the extension of the first source item
<srcver>	the version number of the first source item (if it has one) (so-called "fixed file version" format, eg "23.60.0.5")
<srcvers>	the version number of the first source item (if it has one) (so-called "string file version" format, eg "23.60.0005")
<srcdatem ...>	a date variable (usual XY date syntax) based on the Modified date of the first source item, e.g. <srcdatem yyyy-mm-dd>. Equally <srcdatec ...> for the Created date and <srcdatea ...> for the accessed date.

Examples

Copies all currently selected items to folder "D:\Stuff":

```
copyto "D:\Stuff";
```

Moves all currently selected items to folder "E:\Test\", on collision "Affix current date to copy":

```
moveto "E:\Test\",,,,,5;
```

Creates a copy of the currently running XYplorer.exe in D:\Backup\XYplorer v17.10.0000 (on 2016-09-02), where v17.10.0000 is the current version, and 2016-09-02 the current date. The folder is created on the fly:

```
backupto "D:\Backup\XYplorer v<srcver> (on <date yyyy.mm.dd>)", "<xypath>\XYplorer.exe";
```

Copies a junction as junction ("Junction to SomeFolder" is a junction):

```
copyto "T:\TestTarget", "E:\Test\junc\Junction to SomeFolder", 9:=2;  //2 = Copy as junctions
```

Exclude hidden files and folders:

```
backupto "T:\Test\Target", "T:\Test\Source\*", 6:=4;  //backup all under source to target;
```

```
exclude hidden
```

```
copyto "T:\Test\Target", "T:\Test\Source\*", 9:=4;  //copy all under source to target;
exclude hidden
```

Using rootpath for Rich Operations

By passing a common root through the rootpath argument you can trigger Rich Copy/Move Operations directly from a script. This allows you to copy/move files together with parts of their folder structure in a fully controlled yet automatic way.

Note: If a source file is not located under rootpath then its full path is recreated in the target location; colons (:) after drive letters and "\\" before UNC paths are removed to conform to general filename syntax.

Examples

Copies "D:\Download\New\Test.txt" to "E:\Test\Download\New\Test.txt":

```
copyto "E:\Test", "D:\Download\New\Test.txt", "D:\";
```

Copies "D:\Download\New\Test.txt" to "E:\Test\New\Test.txt":

```
copyto "E:\Test", "D:\Download\New\Test.txt", "D:\Download";
```

Copies "D:\Download\New\Test.txt" to "E:\Test\D\Download\New\Test.txt" because the source file is not under the given rootpath "E:\":

```
copyto "E:\Test", "D:\Download\New\Test.txt", "E:\";
```

Creates a copy of all selected files under "E:\Test" recreating their full source paths. Note that a dummy string "*" is passed as rootpath: it will never work as an actual rootpath, so all source paths are fully recreated:

```
copyto "E:\Test", , "*";
```

Using the FilesOnly and SkipPromptToCreate flags

To move/copy files only set the flags parameter to 1, AND use the source argument with wildcard * anywhere in the pattern. To skip any "create?"-prompts, OR the value 1 with 2 which gives 3.

```
copyto "c:\temp", "e:\test\*", , 1; //copies only files
```

```
copyto "c:\temp", "e:\test\*";      //copies all
```

```
copyto "c:\tempNew", "e:\test\*", , 3; //only files; no prompt
```

Examples for backupto

Backup selected items to E:\Test; on collision affix current date to the copies; preserve dates; create log:

```
backupto "E:\Test", , 5, 1, 1;
```

Backup all selected items to E:\Test; on collision number-suffix the existing items in E:\Test:

```
backupto "E:\Test", , 7;
```

Backup all selected items to E:\Test; on collision ask what to do:

```
backupto "E:\Test", , -1;
```

Using the filter:

```
backupto "E:\Temp", 9:="*.jpg|*.png";   //only JPG and PNG
backupto "E:\Temp", 9:="-*.jpg|-*.png"; //all apart from JPG and PNG
```

mru()

Adds an item to an MRU list and returns the last used item from the list.

Syntax

```
mru(type, [add])
```

type	Type of list. vf: Visual Filters (currently this is the only supported list).
add	[optional] Item to add to the top of the list (= last used). If missing or empty, the current list is not altered.

Examples

```
echo mru("vf"); //show the last used Visual Filter
mru("vf", "*.txt"); //set "*.txt" as last used Visual Filter
```

msg

Shows a message box.

Syntax

```
msg text, [buttons_icons], [linebreaker="<br>"], [window_title]
```

text	The text to be shown in the window.
buttons_icons	(bit field) Defines what buttons and icons to show in the dialog: 0: [Default] Only one button, "Ok". 1: Two buttons, "Ok" and "Cancel". When you click "Cancel" the script execution will end there (useful for confirmations). 2: Three buttons, "Abort", "Retry", "Ignore". When you click "Abort" the script execution

will end there (useful for confirmations). The other options continue the script (they make no difference since the command does not return a value).

3: Three buttons, "Yes", "No", "Cancel". When you click "Cancel" or "No" the script execution will end there.

4: Two buttons, "Yes" and "No". When you click "No" the script execution will end there.

5: Two buttons, "Retry" and "Cancel". When you click "Cancel" the script execution will end there.

6: Three buttons, "Cancel", "Retry", "Continue". When you click "Cancel" the script execution will end there.

16: Critical icon

32: Question icon

48: Exclamation icon (sic: 16 + 32, MS invented that)

64: [Default] Information icon

linebreaker	[optional] Any character sequence to be replaced by a line break. Defaults to " ". Pass "" to prevent any replacing.
window_title	[optional] Window title. Defaults to "XYplorer" plus current version number.

Usage

Simply call the command with whatever text you want to show the user in a pop-up window. You can of course use any variables in your text, as well as put your text over multiple lines, using the line breaker <br> which will be converted to a CRLF (0D0A).

You can change the line breaker (<br>) by using command br first.

Remarks

Any NULL characters in the displayed text are replaced by spaces.

Examples

```
msg "XYplorer is running from <xypath><br>The application data are stored in <xydata>";
```

Shows a message box with the application's folder (path to running XYplorer.exe) and the application's data folder (where all settings are stored), and an OK button.

```
$dest = "D:\Backup\XYplorer Settings - <date yyyy-mm-dd>"; msg "Do you want to backup your XYplorer settings to ""$dest"" ?", 1; copyto $dest, "<xydata>\*.ini|<xydata>\*.dat|<xydata>\FindTemplates";
```

(1) Sets variable $dest to a subfolder of D:\Backup called "XYplorer Settings - " followed by the current date, eg. "XYplorer Settings - 2008-02-10".

(2) Asks the user whether to continue or not? Pressing "Cancel" will abort script execution here.

(3) Copies all INI and DAT files as well as subfolder FindTemplates. The target folder, e.g. D:

\Backup\XYplorer Settings - 2008-03-13\, will be created if necessary.

```
msg "Exclamation with OK and Cancel!", 48 + 1;
msg "Are you sure you want to betray our allies?", 4 + 32; //Yes/No with question mark
```

new()

Creates a new file or folder.

Syntax

```
new(name, [type=file|dir|link|symlink|hardlink|junction], [source], [flags])
```

name	Name of the new item to be created. If no path is passed, the current path is taken as default. If the target folder does not exist it is created on the fly. The "last target" path (used by Ctrl+Alt+F7) is set to the target folder.
type	Defines what kind of item to create: file: [Default] a new file, or dir: a new folder. Note that this parameter will be ignored when a source is specified, obviously. link: Create a shortcut to the item stated in the "source" argument (which now actually serves as the *target*). symlink: Create a symbolic link to the item stated in the "source" argument (which now actually serves as the *target*). Note that you need Admin rights to create a symbolic links! See also Remarks on Types link and symlink below. hardlink: Create a hardlink to the item stated in the "source" argument. This function works for files only. junction: Create a junction to the item stated in the "source" argument. This function works for folders only.
source	Path of the item to "duplicate" when creating a new one. For more about this, please see below.
flags	A combination of these in any order: r: After the new item has been created, Rename mode in invoked (much like when using the New feature manually); ignored if new item is not in current path. u: Allow Undo.
return	The full path/name of the newly created item (if any). If the rename argument is set to "r" then the returned value is the name of the created item *before* any rename.

Create Empty Files Or Folders

Let's start with the basics. All you need to do is to enter a Name for the newly created items, as well as choose its Type (File or Folder).

Examples

```
new("file.txt");
```

Creates a new file called "file.txt" in the current path.

```
new("NewFolder", "dir");
```

Creates a new folder called "NewFolder" in the current path.

Use Variables For New Name

Instead of having to use a fixed default new name, XY brings even more power by allowing the use of a few variables. Those variables are relating to the item currently both focused and selected (this item is displayed in the status bar).

<curfolder>	the name of the current folder
<curname>	the name (without path) of the item
<curbase>	the base name of the item without its extension
<curext>	the extension of the item
<curver>	the version number of the item (if it has one)

Examples for using variables in the name parameter:

If the currently focused & selected file is C:\Downloads\Updates\xyplorer_full.zip then:

name: <curfolder> - <curbase>.txt

resolved: Updates - xyplorer_full.txt

name: <curtitle>.txt

resolved: xyplorer_full.zip.txt

name: <curbase>_<curext>.txt

resolved: xyplorer_full_zip.txt

If the currently focused & selected file is the application file for XYplorer v6.60.0042 (D:\Program Files\XYplorer\XYplorer.exe) then:

name: <curbase> v<curver>.txt

resolved: XYplorer v6.60.0042.txt

name: <curtitle> (<curver>).txt

resolved: XYplorer.exe (6.60.0042).txt

Example scripts:

```
new("<date yyyy-mm-dd>.txt");
```

Creates a new file called "2008-03-13.txt" (or the date today) in the current path.

If the currently focused & selected file is the application file (XYplorer.exe) for XYplorer v6.60.0042, modified December 4th, 2007 at 11:11, then:

```
new("<curbase>_<datem yyyymmdd>.<curext>");
```

Creates a copy called "XYplorer_20070412.exe".

```
new("<curbase>_<curver>.<curext>");
```

Creates a copy called "XYplorer_6.60.0042.exe".

```
new("<curbase> v<curver> (<datem mm.dd.yyyy>).<curext>");
```

Creates a copy called "XYplorer v6.60.0042 (04.12.2007).exe".

Name Collisions

If the name specified already exists then the default name presented on rename mode will use a suffix number, according to the format defined in Configuration | Templates | Incremental suffix.

Cloning a source: Create Files Or Folders With Content

In the same way as the New Items feature works, you can also define a source that will be "cloned" (e.g. copied) when a new item is created. Basically what XYplorer will do when using a source is to copy the specified item (Source) to the current location using whatever Name was specified as default (if left empty, the original name (of the clone item) will be used). So if the source was...

...a file, a new file with the same content will be created;

...a folder, a new folder with the same full content (files and subfolders) will be created.

When defining a Source you don't have to worry about the type parameter, as it will obviously be ignored.

With this feature you will be able to clone files and folders whenever you want, from wherever you are, simply by using one click or one keyboard shortcut.

What this means is, you can easily create a new file with content (e.g. a Word document with your usual headers, layouts, signature block, etc.) or even a full folder with content (e.g. a folder with the usual files and subfolders you always use when creating a new project)! There are absolutely no limitations.

Cloning the current item

You can also use the currently selected and focused item as source of cloning, by putting <curitem> as Source. That way, the current item will be copied and rename mode will be activated afterwards with whatever was defined on Name as default.

```
new(, , "<curitem>");
```

Creates a copy of the current item in the current path. Collisions are avoided by an auto-appended number.

```
new(, , "<curitem>", "r");
```

The same, but invoking rename mode after creation.

```
new("<srcbase>-<srcdatem yyyy-mm-dd>.<srcext>", , "XYplorer.exe");
```

Creates a copy of XYplorer.exe called "XYplorer-2008-03-12.exe" in the current path (see below "Variables referring to the source").

Variables referring to the source

The following variables in the name parameter will relating to the source item (set in Source field):

<srcname>	The full name of the first source item without path.
<srcbase>	The base name of the first source item without extension.
<srcext>	The extension of the first source item.
<srcver>	The version number of the first source item (if it has one).
<srcdatem ...>	A date variable (usual XY date syntax) based on the Modified date of the first source item, e.g. <srcdatem yyyy-mm-dd>. Equally <srcdatec ...> for the Created date and <srcdatea ...> for the accessed date.

Finally some examples using the full path and the return value

Creates new empty file "Kabul.txt" in folder "E:\Afgh\":

```
$newfile = new("E:\Afgh\Kabul.txt");
```

Creates new path "E:\Afgh\Kabul\EastEnd\":

```
$newfolder = new("E:\Afgh\Kabul\EastEnd\", "dir");
```

Copies "C:\Temp\Kabul.png" to "E:\Afgh\Kabul-copy.png":

```
$newfile = new("E:\Afgh\Kabul-copy.png", , "C:\Temp\Kabul.png");
```

Copies the current item to "E:\Afgh\", suffixed with "-copy":

```
$newfile = new("E:\Afgh\<curbase>-copy.<curext>", , "<curitem>");
```

Remarks on Types link and symlink

- The "name" argument can be omitted; in that case it will be taken from the "source" argument.
- The "name" argument can be relative to the current path.
- In case of type "link" the extension ".lnk" will be appended to the created file if not already passed in the "name" argument.
- The "source" argument MUST be stated (full path).
- The function will auto-avoid filename collisions by affixing the new name according to the user settings (Configuration | Templates | Filename Affixes).
- The "flags" argument is ignored with types "link" and "symlink".
- The "source" defaults to the current list item.

```
new("newlink", "link", "E:\Test\junc\bird.jpg");
new(, "link", "E:\Test\junc\bird.jpg");
new("SymLink to bird.jpg", "symlink", "E:\Test\junc\bird.jpg");
new(, "symlink", "E:\Test\junc\bird.jpg");
new(, "link"); //creates shortcut to currently selected item
new(, "symlink"); //creates SymLink to currently selected item
new(, "hardlink"); //creates hardlink to currently selected file
new(, "junction"); //creates junction to currently selected folder
```

newwindow

Opens a path in a new window.

Syntax

```
newwindow [path]
```

path:	Path to open a new window (clone of the current window). Defaults to current path.

Remarks

You can hold CTRL+SHIFT while clicking menu File | Open Throw Away Clone to achieve the same as `newwindow;`.

Example

```
newwindow;       //open the current path a new window
newwindow "C:\"; //open C:\ a new window
```

now()

Returns the current date/time.

Syntax

```
now([format])
```

format	Either the usual date format, e.g. "yyyymmdd", or "msecs" which returns the number of milliseconds that have elapsed since the system was started (after 49.7 days it wraps back to start from zero). If missing the default system time format is used.

Examples

```
echo now();         // 27.04.2011 18:46:12
```

```
echo now("hh:nn");  // 46:12
echo now("yyyy");   // 2011
echo now("yyyy-mm-dd hh:nn:ss.fffffff");   // maximum resolution down to the 1/10 microsecond
echo now("msecs");  // 1221097715
```

Notes

The slash stands for the current system date separator, which might be set to a dot. To force a slash you have to escape it with a backslash:

```
echo now("dd/mm/yyyy");   // 27.04.2011 in Germany
text now("dd\/mm\/yyyy"); // 27/04/2011 in Germany
```

obfuscate

Obfuscates item names in Tree, List, Address Bar, Catalog, Tabs, Breadcrumb Bars, Status Bar, Info Panel, and Window caption.

Syntax

```
obfuscate [mode], [bullet], [list], [flags]
```

mode	[optional] (missing): [Default] toggle 0 > 1, or not 0 > 0 (bit field) 0: No obfuscation. 1: Obfuscation of all path/file names visible on the surface (excl. the Ext column). 2: Obfuscation of all tooltips and dropdowns. 4: Obfuscation of all special columns (Properties, Special Properties, Custom Columns, Ext column).
bullet	[optional] Character to use. Defaults to Unicode Character 'BULLET' (U+2022). Can also be a list of characters from which a random character is picked on each replacement.
list	[optional] List of strings to be obfuscated, separated by |. Strings are processed from left to right, and matching is case-insensitive (A==a). Strings may contain basic wildcards (*, ?). If missing everything is obfuscated.
flags	(bit field) 1: Toggle obfuscation when the same "mode" is passed again.

Remarks

- Some special characters are excluded from obfuscation: `.\|/<>"".,;:()[]{}+-*%^$@?!` and `Space` and `CR` and `LF`.

These excluded characters help to understand the text structure without revealing private information.

- The obfuscation state is not remembered across sessions.
- If you save settings in obfuscated state then the interface is auto-de-obfuscated, otherwise some items would be written obfuscated to the INI which is not anything you want.
- Also the title bar is obfuscated. Application title and version number are excluded to assist debugging (which is the purpose of obfuscate after all).

Examples

```
obfuscate;
obfuscate 0;
obfuscate 1; //obfuscate all path/file names (excl. the Ext column)
obfuscate 4; //obfuscate all special columns (incl. the Ext column)
obfuscate 5; //obfuscate all path/file names and all special columns
obfuscate 1, "x";
obfuscate , chr(9608);    //toggle Unicode Character 'FULL BLOCK' (U+2588)
obfuscate , chr(0x2588);  //same in Hex format
obfuscate , , "Donald|Duck";  //toggle obfuscation of "Donald" and "Duck"
obfuscate , , "Yamaha*"; //will completely obfuscate "Yamaha YZR500.png"
obfuscate , , "*.jpg";   //will completely obfuscate all JPGs
obfuscate , "abcdefghijklmnopqrstuvwxyz0123456789"; // obfuscate everything with letters and numbers
obfuscate , "abcdefghijklmnopqrstuvwxyz" , "*.jpg"; // obfuscate all JPGs with random letters
obfuscate , "0123456789", "ä|ö|ü"; // replace all umlauts by random numbers

obfuscate 3; //thorough obfuscation of everything
obfuscate 3, , "Donald|XYplorer"; //thorough obfuscation of the words "Donald" and "XYplorer"
obfuscate 3, , , 1; //toggle thorough obfuscation of everything
```

open

Runs an application or opens a document with an associated application.

Syntax

```
open item, [switches];
```

item	[Required] Path to an application to start or a document to open. Defaults to the current list item.
switches	(missing): [Default] Use XYplorer's Custom File Associations to determine the opening

application.
e: Spawn the opening application in an elevated processes (= run as administrator).
w: Use Windows associated application.

Examples

```
open "C:\Temp\file.txt";  //Open the file C:\Temp\file.txt
open "C:\Program Files (x86)\Mozilla Firefox\firefox.exe";  //Run FireFox
open "firefox";  //Run FireFox (if it is a registered application)

open "D:\test.bat", e;  //use Custom File Association, elevated process
open "D:\test.bat", we; //use Windows association, elevated process
open """D:\test.bat"" ""hi there""", "we"; //how to pass a parameter to an elevated process
```

`open '"winzip32" -min'` or `open """winzip32"" -min";`

Run Winzip minimized (if it is a registered application). Note that the application must be double-quoted when command line parameters are used! If the whole argument is single-quoted, the contained double-quotes don't have to be doubled.

openwith

Opens the currently selected (or list-defined) items with the specified application.

Syntax

```
openwith application, [mode (s|m)], [itemlist]
```

application	[Required] Path to the application to open the selected items with.
mode	Defines which mode to use when sending the parameters (item's names) to the application. s: [Default] Single instance (pass all the items to the same instance of the application). m: Multiple instance (pass each of the items to its own instance of the application).
itemlist	CRLF- or |-separated list of items (full path) to open (overriding any List selections).

Examples

`openwith "UEdit32"` or `openwith "UEdit32", s;`

Open the selected list items with UltraEdit (registered as "UEdit32"), single instance.

```
openwith "UEdit32", m;
```

Open the selected list items with UltraEdit (registered as "UEdit32"), one instance for each item.

```
openwith """D:\Program Files\WinZip\Winzip32"" -e <items> ""<base>""", m;
```

You can also create commands to extract currently selected archives. The above script will extract all selected archives, each into a folder with archive's name, minus the extension (e.g. "My Backup.zip" will be extracted to a subfolder "My Backup").

Note that the application must be quoted when command line parameters are used! Since the whole argument is quoted the inner quotes should be doubled.

```
openwith "UEdit32", , "<xydata>\XYplorer.ini";
```

Open XYporer.ini with UltraEdit.

outputfile()

Lets you select a location for a file via the Windows Save As common dialog.

Syntax

```
outputfile([path], [file], [caption])
```

path	The default path for the dialog. When left empty or non-existent, path defaults to the current path. In the latter case you are prompted whether to continue the script. With or without trailing slash, it does not matter.
file	Initial name of the file. Can be changed within the dialog.
caption	The text that will be displayed as title of the window. If omitted it will be "Save As" in the local language.
return	The full path/name of the selected existing or new file.

Remarks

If you press Cancel, the script execution will be aborted.

The file is not saved, just the name is returned. Any saving has to be done by a separate command.

Examples

```
echo outputfile(,"Peace.jpg");
```

```
echo outputfile(,"Peace.jpg", "Select Location...");
```

```
echo outputfile("C:\Love","Peace.jpg", "Select Location...");
```

paperfolder()

Sets or gets Paper Folder contents.

Syntax

```
paperfolder(name, [itemlist], [separator=CRLF], [mode=nl])
```

name	Name of the Paper Folder, e.g. "test". May have prefix. e.g. "paper:test". The name is resolved relative to '<xypaper>'. Can also be full path/file, e.g. "E:\Test\foo.txt".
itemlist	List of items to set and (optionally) load.
separator	Separator used in itemlist.
mode	n: Load items in itemlist as new Paper Folder. Overwrites any existing Paper Folder of that name without asking. a: Append items in itemlist to Paper Folder. Creates new one if "name" does not exist. d: Delete items in itemlist from Paper Folder. If itemlist is omitted then the Paper Folder is emptied. Items are not deleted from the file system. l: Load Paper Folder into current list if not loaded anyway. This is done after processing itemlist. s: Show results in status bar (how many items were added/removed). Note: Will be overwritten by subsequent status if "l" is set as well. r: Return current contents of Paper Folder. o[z|p|d|e|i]: Set option (and return old setting). The 2nd letter stands for the option: oz: Allow Zombies op: Always Show Path Column od: On Delete Remove Items from Paper Folder oe: Explicit Save Only oi: Show Information Bar in the List The 3rd character is number 0 ("off") or 1 ("on"), e.g.: oz1, see below Examples for "o"-Mode Cannot be combined with any other modes (other modes are ignored).
return	The current contents of Paper Folder (before any new ones are set).

Remarks

When adding items the function takes care that no duplicates are created.

Examples

```
text paperfolder(, , , "r"); //return contents of currently shown paper folder (if any)

text paperfolder("test", , , "r"); //return contents of paper folder "test"

paperfolder("test", "E:\Test\LuckyLuke.jpg|C:\Users\Donald\Desktop\Desk\clap.png", "|"); //
load 2 items as PF into list

paperfolder("test", <clipboard>); //set items from clipboard (one per line) to paper folder
"test", and load it

paperfolder("test", <clipboard>, , "a"); //add items from clipboard (one per line) to paper
```

```
folder "test"
paperfolder("test", "E:\Test\LuckyLuke.jpg", , "ds"); //remove item from paper folder "test", show status
paperfolder("test", , , "ds"); //empty paper folder "test"
paperfolder("test", , , "l"); //load paper folder "test" into the current list
```

Examples for "o"-Mode

```
text paperfolder(, , , "oz"); //return current setting (0 or 1) of "Allow Zombies"
text paperfolder(, , , "oz0"); //set "Allow Zombies" to 0 (= unticked), return old setting
text paperfolder(, , , "oz1"); //set "Allow Zombies" to 1 (= ticked), return old setting
paperfolder(, , , "oe1"); //set "Explicit Save Only" to 1 (= ticked)
paperfolder(3:="oi0"); //hide information bar in the list
```

pasteto

Paste the clipboard contents to the specified destination.

Syntax

```
pasteto [destination], [flags]
```

destination	Folder to paste the clipboard contents to. Defaults to current path.
flags	(bit field) 1: Paste also textual clipboard contents.

Examples

```
pasteto;      //paste here (items on clipboard)
pasteto , 1;  //paste here (items or textual clipboard contents)
pasteto "C:\Test";    //paste to C:\Test (items on clipboard)
pasteto "C:\Test", 1; //paste to C:\Test (items or textual clipboard contents)
```

patchimage

Replaces an internal image, e.g. of a toolbar button, with one from a file.

Syntax

```
patchimage key, [file], [switches="r"]
```

key	Image key. How to find out an image key of a toolbar button: Hold CTRL while hovering a button and the tooltip will show the button key and the current image key or image path (if

different from button key).

file	Image file (full path/name, or relative to current path). Defaults to the selected/focused file. If empty ("") the patch is removed and the original image is used again.
switches	(lower case letters in any order) r: Refresh the Toolbar when the replacement is done. [Default] s: Use this image for the small size (16x16). l: Use this image for the larger sizes (24x24 and higher). Note: If both "s" and "l" are missing then both small and large size are set.

Remarks

- The file can be larger than the image in which case it is automatically shrunk.
- A non-small patch is used for all sizes, unless there is a small patch (which is then used for the small size). If there is only a small patch then the original image is used for all non-small sizes.
- These changes are not permanent but last as long as the session.

Examples (all using the selected image file)

```
patchimage "flatview";            //set small and large patch
patchimage "flatview", , "rsl";   //set small and large patch
patchimage "flatview", , "rs";    //set small patch only
patchimage "flatview", , "rl";    //set large patch only
patchimage "flatview", "";        //remove small and large patch
patchimage "flatview", "", "rsl"; //remove small and large patch
patchimage "flatview", "", "rs";  //remove small patch only
patchimage "flatview", "", "rl";  //remove large patch only
```

Specifying a file by full path:

```
patchimage "flatview", "C:\Cactus.png";  //replace small and large "flatview" image with "Cactus.png"
```

pathreal

Returns the real path for a given path.

Syntax

```
pathreal([path])
```

path	Full real or special path to an item. Defaults to the currently focused list item.

Remarks

The return is always without trailing backslash.

Remarks on Special Paths

See below at pathspecial.

Examples

All return "C:\Users\Donald\Downloads\Alice\6foot.jpg":

```
echo pathreal("Downloads\Alice\6foot.jpg");
echo pathreal("Donald\Downloads\Alice\6foot.jpg");
echo pathreal("C:\Users\Donald\Downloads\Alice\6foot.jpg");
```

pathspecial, pathvirtual

Returns the special path for a given path.

The alternative form "pathvirtual" is deprecated.

Syntax

```
pathspecial([path], [flags])
```

path	Full real or special path to an item. Defaults to the currently focused list item.
flags	0: Returns the shortest special version of the path. 1: Mind the visibility in the current tree. Returns the shortest special version of the path where the top node is part of the current tree. 2: Mind the current tree location. Returns the special version of the path that matches currently selected tree location. If that location is a real path then the real path version is returned.

Remarks

The return is always without trailing backslash.

Remarks on Special Paths

Some paths support to be referred to by a shortened path spec, the so-called special path. In XYplorer these paths are Desktop, Documents, Downloads, Links, and <user>. So there are at least two ways to refer to these paths, mostly even three, since all but <user> are contained in <user>. For example:

```
Special path 1: Downloads\Alice\
Special path 2: Donald\Downloads\Alice\
Real path:      C:\Users\Donald\Downloads\Alice\
```

Examples

All return "Downloads\Alice\6foot.jpg" regardless of what's visible in the current folder tree:

```
echo pathspecial("Downloads\Alice\6foot.jpg");
echo pathspecial("Donald\Downloads\Alice\6foot.jpg");
```

```
echo pathspecial("C:\Users\Donald\Downloads\Alice\6foot.jpg");
```

All return "Downloads\Alice\6foot.jpg" if a special "Downloads" folder exists in the current folder tree, else they return "Donald\Downloads\Alice\6foot.jpg" if a special "Donald" folder exists, else they return "C:\Users\Donald\Downloads\Alice\6foot.jpg".

```
echo pathspecial("Downloads\Alice\6foot.jpg", 1);
echo pathspecial("Donald\Downloads\Alice\6foot.jpg", 1);
echo pathspecial("C:\Users\Donald\Downloads\Alice\6foot.jpg", 1);
```

All return "Donald\Downloads\Alice\6foot.jpg" if the current tree location is anywhere in the "Donald" branch (else they return "Downloads\Alice\6foot.jpg" or the real path "C:\Users\Donald\Downloads\Alice\6foot.jpg"):

```
echo pathspecial("Downloads\Alice\6foot.jpg", 2);
echo pathspecial("Donald\Downloads\Alice\6foot.jpg", 2);
echo pathspecial("C:\Users\Donald\Downloads\Alice\6foot.jpg", 2);
```

perm

Define one or more variables as permanent.

Syntax

```
perm variable(s)
```

variable(s)	A single variable, or a comma-separated list of up to 10 variables. If a variable has already been defined as global before, then the perm command makes it permanent. If a variable did not exist before the perm command initializes it to "" (empty string). Note that you optionally can initialize the variables to certain values (see below).

Usage

Permanent Variables stay alive in memory during the whole XYplorer session, and hence can be easily shared between scripts.

Examples

Run the following scripts one after the other through the Address Bar:

```
perm $foo; $foo = "hi!";
echo $foo;  //"hi!"
unset $foo; echo $foo;  //$foo
```

These two scripts illustrate that you can "perm" a global variable even after it was set:

```
global $foo; $foo = "hi!"; perm $foo;
echo $foo;  //"hi!"
```

Optionally you can immediately assign initial values to the declared variables:

```
perm $a = "a"; echo $a;
```

```
perm $a = "a", $b = "b"; echo "$a, $b";
```

These assigned values can be variables themselves, or even expressions or functions:

```
perm $a = <clipboard>; echo $a;
```

```
perm $a = 5*7; echo $a;
```

Remarks

- Permanent Variables are always also global. However, contrary to normal globals, they don't need to be initialized to the current scope in order to be accessed. They are always available everywhere.
- To remove a Permanent Variable from memory use the "unset" command.
  Note: The "global" command will *not* make it non-permanent; to use "global" on a Permanent Variable is not necessary and has no effect.
- The number of Permanent Variables is not limited.
- It's recommended that you reflect the nature of a Permanent Variable in its name, for example by prepending $p_. It will make your code easier to read and maintain.

popupcontextmenu

Pops the shell context menu for an arbitrary item.

Syntax

```
popupcontextmenu [item], [bitness], [flags]
```

item	File or folder to pop the shell context menu for. Can also be a list of items separated by CRLF or | (the separators must not be mixed). The items can be located in different places (Win7+). Defaults to the currently focused list item.
bitness	Bitness of the shown menu. This argument is currently obsolete. Only the 64-bit menu is shown.
flags	(bit field) 1: Hide shell extensions (see Remarks).

Remarks

- The menu is popped at the mouse cursor.
- Hide shell extensions: The effect is identical to ticking "Configuration | General | Menus, Mouse, Usability | Context Menus | Hide shell extensions from shell context menu".

Examples

```
popupcontextmenu;
```

```
popupcontextmenu , , 1; //pop a basic menu for selected items (Hide shell extensions)
popupcontextmenu "C:\Users\Donald\Desktop\Desk\Preview.bas";
popupcontextmenu <xy>;
popupcontextmenu "Moto G (4)\SD-Karte von SanDisk\Test\Sub\IMG_0060.JPG"; //Items on portable devices are supported
popupcontextmenu "<xydata>\XYplorer.ini|<xydata>\ks.dat"; //same location
popupcontextmenu "%winsysdir%\Kernel32.dll|%SystemRoot%\explorer.exe"; //different locations
```

popupmainmenu

Shows the main menu as popup menu.

Syntax

```
popupmainmenu [submenu], [x=-1], [y=-1]
```

submenu	The caption of any of the main sub menus to pop just this menu. If missing the whole main menu is popped.
x	X position; -1 to use current mouse X.
y	Y position; -1 to use current mouse Y.

Examples

```
popupmainmenu;
popupMainMenu , 0, 0; //top left corner of screen
popupmainmenu "file";
popupmainmenu "edit";
```

popupmenu()

Pops a menu and returns the selected item.

Syntax

```
popupmenu(itemlist, [x=-1], [y=-1], [start=1], [count=-1], [flags=0], [sep_itemlist="|"], [sep_item=";"], [on_cancel=""])
```

itemlist	List of items separated by sep_itemlist. Each item is made of a Caption and optionally a Data part, an Icon part, and a State part, all parts being separated by sep_item: "Caption;Data;Icon;State" Caption: Menu caption. Use "-" (no quotes) to specify a menu separator. Data: Returned when item is clicked. If missing or empty then Caption is returned. Icon: Supports various sources for icons Toolbar icons, prefixed by ":", e.g. :paper

Generic file system icons, e.g. *.png
Specific files, e.g. C:\Program Files\XYplorer\XYplorer.exe
Icon files (defaulting to <xyicons>, e.g. 64_bit.ico
Image files (defaulting to <xyicons>, e.g. D:\pics\ball.png
XYplorer native and environment variables, e.g. <xy>
State: (bit field)
1 = default (bold)
2 = checked
4 = disabled

x	X position; -1 to use current mouse X. Or: Toolbar button key to place the menu directly below that button (in that case the y parameter is ignored). If the button is not present in the visible rows of the current toolbar the menu opens at the mouse pointer.
y	Y position; -1 to use current mouse Y.
start	Index of first item to show; 1 = first item.
count	Count of items to show; -1 = show all.
flags	1: Return index (instead of caption/data). 2: Process paths (go to folders, open files, run executables). 4: Show name only, without full path (only when combined with flag 2). 8: Show fast generic icons in the menu.
sep_itemlist	Separates items; can have more than one char. Defaults to CRLF if itemlist contains any CRLF. Else defaults to "|".
sep_item	Separates caption from data; can have more than one char.
on_cancel	String returned when menu is cancelled (ESC).
return	Data of selected item; caption if no data defined; index if flags AND 1.

Examples

```
text popupmenu("a|-|b|c");  //- is a menu separator
text popupmenu("a;data returned on a|-|b;data b|c;data c");
text popupmenu(<get selecteditemspathnames |>);
text popupmenu("Megan;Hello Megan!;*.png;1|Betty;Hello Betty!;*.jpg|Checked;;;2");
text popupmenu("a|-|b|c", "back");  //place menu at Back button
```

Tip: You can use `*.<DIR>` as a pseudo extension for folders in the icon slot, i.e. you can use it to show the generic folder icon. Example:

```
echo popupmenu("Something with Folder Icon;Now *.<DIR> shows the folder icon.;*.<DIR>");
```

Processing paths:

```
echo popupmenu("<xydata>|<xypath>|<xy>");   //these paths are just strings
popupmenu("<xydata>|<xypath>|<xy>", 5:=2);  //process paths, show full path
popupmenu("<xydata>|<xypath>|<xy>", 5:=6);  //process paths, show name only
```

Adding XYplorer Menu Commands and Toolbar Buttons

You can add XYplorer menu commands to your scripted menu, identified by simply their ID. You get the original captions (in your language) with keyboard shortcuts, and even the checked state (where applicable) is shown. When you click a command it is executed as if clicked in the main menu.

You can as well add XYplorer toolbar buttons to your scripted menu, identified by their key. The captions of the buttons are used as menu captions, and the button graphics as menu icons.

- The menu commands have to be stated by the command ID prefixed with a number sign (#).
- The toolbar buttons have to be stated by the button key prefixed with a colon (:).
  Tip: Hold CTRL while you click "Customize Toolbar..." to see the button keys in the button listing.
- Note that the on/off (pressed/unpressed) state of the toolbar buttons cannot be displayed in the menu. This is about the only minor drawback here.
- You can freely mix your own menu items with XYplorer menu commands and toolbar buttons.
- You can add icons to XYplorer menu commands and toolbar buttons (overwriting the original button graphics) if you like.

Examples

```
popupmenu("#101|#102|-|#800|#801|#802"); // menu commands only

popupmenu(":copypath|:dp|:pp");          // toolbar buttons only

popupmenu("#101|:pp"); // mixed commands

popupnested("Copy Commands| :copypath| #102|Dual Pane Commands| #800| #801| #802");
 // mixed commands nested

popupnested("Copy Commands| #101;;:copypath| #102|Dual Pane Commands| #800;;:dp| #801|
#802"); // mixed commands nested with toolbar icons
```

Adding Custom Captions

You can define custom captions for menu commands defined as XYplorer Menu Commands or Toolbar Buttons. Simply put the function definition in the "Data" part of the item definition (Caption;Data;Icon;State) and use the "Caption" part for your caption.

Examples

```
popupmenu("My Caption for :copypath;:copypath;Copy.ico");

popupmenu("My Caption for #102;#102;Copy.ico");
```

Adding Scripts

You can also add scripts to your scripted menu, identified by the script marker "::". On selecting the menu item the script is executed.

In this case the menu item supports the following formats (assuming sep_item=";"):

```
::Script                  (here Script is also used as Caption)
::Caption;Script
::Caption;/Icon;Script    (/ functions as icon marker)
```

Examples

```
popupmenu("::Hi;echo 'hi'|::Ho;echo 'ho'");
popupmenu("::echo 'hi'|::echo 'ho'");
popupmenu("::Caption for a script;/:sync;e 'Hello'");    //using the internal "sync" icon
popupmenu("::Caption for a script;/Fun.png;e 'Hello'");  //icon "Fun.png" defaults to the <xyicons> path.
```

There can be multiple statements per item and a trailing ";":

```
popupmenu("::""Hi Ho!"";echo 'hi';echo 'ho';");
```

All sorts of menu items can be freely mixed. Here are three ways to code a "Copy Path" function:

```
popupmenu(":copypath|#101|::'Copy Path';#101;");
```

Remarks

It might be hybris to pack a script into a substring of an argument of another script and can surely open the doors of hell. But OTOH it's a cute little option for the Hamburger menu...

popupnativecontextmenu

Pops a native context menu for all selected list items.

Syntax

```
popupnativecontextmenu.
```

Remarks

- It's a lightning-fast 100% native context menu (no shell involved).
- The command won't pop any menu when there are no selections in the list.
- If there are selections in the list the command will set the input focus to the list (if it's not there already). Otherwise various menu commands would not work as expected. So, SC PopupNativeContextMenu ONLY works for items in the file list, not in the folder tree.

Examples

```
popupnativecontextmenu;
```

This script in CEA "Left-click on status bar" will pop a native context menu on left-clicking the first section of the status bar:

```
if (<CEA_ClickedItem> == 1) {popupnativecontextmenu;}
```

popupnested()

Pops a nested menu and returns the selected item.

Syntax

Identical to popupmenu. The only difference to popupmenu is that indented captions are translated to a submenu structure.

Examples

Nested popup menus are passed to the function as an indented itemlist. The first indent encountered sets the gear for all following indents. So these two lines generate identical menus:

```
text popupnested("Top| Sub| Sub|  Sub2|  Sub2|Top");        //indent 1 space per level
text popupnested("Top|  Sub|  Sub|    Sub2|    Sub2|Top");  //indent 2 spaces per level
```

While the above one-liners perfectly work the HEREDOC syntax comes in handy with larger menus. The structure is clearly readable and additional props like icons and states can easily be added to each line:

```
        $menu = <<<MENU
Top;;:paper;1
  Sub;;;4
  Sub
    Sub2;;*.ini
    Sub2
    Sub2;;;2
Top;;<xy>
MENU;
          text popupnested($menu);
```

To improve readability the top level does not have to be absolutely left-bound. Instead an overall indent can be applied to all items. Example:

```
      $menu = <<<MENU
                Top;;:paper;1
                  Sub;;;4
                  Sub
                    Sub2;;*.ini
                    Sub2
                    Sub2;;;2
                Top;;<xy>
              MENU;
        text popupnested($menu);
```

Remarks

- Nesting is limited to a depth of 64 levels.

- It's possible to create invalid indenting. Here, for example, Sub2 is indented too far:
  `text popupnested("Top| Sub|   Sub2");`
  In consequence the "Sub" submenu will not work. XYplorer does NOT attempt to auto-correct this. This scripter is responsible.

posatpos()

Returns the item position index within a control at a certain screen position.

Syntax

`posatpos([x], [y], [flags])`

x	(optional) Defaults to the X mouse position on screen.
y	(optional) Defaults to the Y mouse position on screen.
flags	
	0: If x and y are passed, they are the position on screen.
	1: If x and y are passed, they are the position on XYplorer.
return	Position index. First position is "1". Returns "0" if the X/Y coordinates do not point to any item.

Remarks

Supported controls are:
- Tree (position of hovered item)
- Catalog (position of hovered item)
- List (position of hovered item)
- Toolbar (position of hovered button)
- Tab Bar (position of hovered tab)
- Breadcrumb Bar (position of hovered path component)
- Status Bar (position of hovered section)

Examples

```
echo posatpos(); //item position at current mouse position on screen
echo posatpos(592, 662); //item position at arbitrary mouse position on screen
echo posatpos(73, 108, 1); //item position at arbitrary mouse position on XYplorer
```

preview

Previews a file in the Preview Pane.

Syntax

```
preview [file], [mode], [guid]
```

file	Full path of file to preview. Defaults to the currently focused file.
mode	How the preview is generated. The bitness-related arguments are currently obsolete but kept for backward compatibility. Only 64-bit stuff is used. n: [Default] Normal UI preview as if you selected the file. p: Use PreviewHandler (64-bit). p32: (same as p) p64: (same as p) t: Use ThumbnailProvider (64-bit), method and bitness depend on current user settings. te32: Use IExtractImage (64-bit). tf32: Use 64-bit IShellItemImageFactory. tf64: Use 64-bit IShellItemImageFactory. tg32: Use 64-bit GDI+ (Graphics Device Interface). Tip: Only works for image files. x: Close the Preview Pane.
guid	CLSID of the PreviewHandler to use, eg {CF822AB4-6DB5-4FDA-BC28-E61DF36D2583}. Only used with modes p, p32, p64.

Remarks

- The command ensures that the Preview Pane is visible.
- Apart from mode "n", all modes ignore the settings in "Configuration | Preview | Previewed Formats". Every file type is attempted to be previewed.
- The time needed is displayed in the status bar.
- A powerful command that allows you to preview files that are not currently listed.
- Also useful as a debugging device for shell-generated previews and thumbnails, and as a test bed for CLSIDs.

Examples

```
preview "E:\Test\Preview\Katakana.pdf";       //normal preview

preview "E:\Test\Preview\Katakana.pdf", p;    //PreviewHandler (64-bit)

preview "E:\Test\Preview\Katakana.pdf", p32;  //PreviewHandler (64-bit)

preview "E:\Test\Preview\Katakana.pdf", p64;  //PreviewHandler (64-bit)

preview "E:\Test\Preview\Katakana.pdf", t;    //ThumbnailProvider (64-bit)

preview "E:\Test\Preview\Katakana.pdf", tf32; //ThumbnailProvider IShellItemImageFactory (64-bit)

preview "E:\Test\Preview\Katakana.pdf", tf64; //ThumbnailProvider IShellItemImageFactory (64-bit)

preview "E:\Test\Preview\Katakana.pdf", te32; //ThumbnailProvider IExtractImage (64-bit)

preview; //normal preview of current file

preview , x; //close the preview pane
```

```
//PreviewHandler (64-bit), GUID of PDF-XChange PDF Preview Provider:
preview "E:\Test\Preview\Katakana.pdf", p32, "{CF822AB4-6DB5-4FDA-BC28-E61DF36D2583}";
```

previewcheck

Checks various ways to preview a file and displays a final report.

Syntax

```
previewcheck [file]
```

file	Full path of file to preview. Defaults to the currently focused file.

Remark

It automatically cycles through all modes supported by SC preview.

Example

```
previewcheck; //check the currently focused file
```

property()

Retrieves a particular property from a file or folder.

Syntax

```
property([property], [item])
```

property	[optional] One of the following: (1) name of the shell property (2) index of the shell property (number prefixed by #); suffix " n" to return the name of the property. (3) name of the XYplorer-specific property (name prefixed by #) Defaults to showing Shell Properties dialog.
item	[optional] Name of the file or folder. Defaults to current path / item.
return	The desired property.

Examples

```
echo property("FileVersion", "%winsysdir%\msvbvm60.dll");
echo property("CameraModel", "F:\IMG_1926.JPG");
echo property("WhenTaken", "F:\IMG_1926.JPG"); //= EXIF date
msg property("Artist", "1.mp3"); //1.mp3 in current path
```

```
msg property("FileVersion"); //version of current file

property(, "<xypath>\<xyexe>"); //show Properties for XYplorer

property(); //show Properties for current item
```

Examples using the Format Template

The format template is only returned if the property returns a value. Simply suffix the template (you can freely put in there whatever you want) within single quotes to the property name argument, and use * (asterisk) as placeholder for the returned value:

```
echo property("#image.dimensions 'Size: * px'");

echo property("CameraModel 'Camera: *'");
```

You can add formatting instructions to the value placeholder. Here are some examples using the <prop ...> variable (see below):

```
echo <prop System.Video.TotalBitrate 'Bitrate: *'>;     //Bitrate: 409600     (no formatting)

echo <prop System.Video.TotalBitrate 'Bitrate: *n'>;    //Bitrate: 409,600    (number
formatting)

echo <prop System.Video.TotalBitrate 'Bitrate: *b'>;    //Bitrate: 400.00 KB  (bytes
formatting)

echo <prop System.Video.TotalBitrate 'Bitrate: *kbps'>; //Bitrate: 462kbps    (kb per sec
formatting)
```

You can also add duration formatting instructions to the value placeholder:

```
echo <prop System.Media.Duration "Duration: *">;      //Duration: 2957410000  (no formatting)

echo <prop System.Media.Duration "Duration: *sec">;   //Duration: 04:55       (seconds
precision)

echo <prop System.Media.Duration "Duration: *msec">;  //Duration: 04:55.741   (milliseconds
precision)
```

Examples for using the Property Index

Note that Microsoft traditionally changes the indices with each new OS.

These two lines return the same under WinXP:

```
msg property("FileVersion", "%winsysdir%\msvbvm60.dll");

msg property("#37", "%winsysdir%\msvbvm60.dll");
```

Suffix " n" to return the (locale aware) name of the property:

```
msg property("#37 n", "%winsysdir%\msvbvm60.dll");
```

Under Vista you need to pass another index:

```
msg property("#145", "%winsysdir%\msvbvm60.dll");
```

Under Win7 you need to pass yet another index:

```
msg property("#156", "%winsysdir%\msvbvm60.dll");
```

Let XYplorer find the Property Index

Prefix an asterisk (*) to the property name and it is internally resolved to the matching index. The

name can be any of the items listed in *Configuration | File Info Tips & Hover Box | Show these fields*. Note that the name is localized (but not case-sensitive), so in a German Windows you would do this to retrieve the width of an image file...

```
text property("*breite");
```

... and this in an English Windows:

```
text property("*width");
```

Note that spaces in property names must be passed as underscores:

```
text property("*Letzter_Zugriff");
```

Alternative usage as native variable <prop ...>

Everything you can feed into `property(property)` you can also feed into the variable `<prop property>` (the `item` parameter is not supported in the variable). Examples:

```
<prop #audio.bitrate>
<prop #audio.samplerate>
<prop #tag.composer>
<prop #mp3.year>
<prop #date.created>
<prop #version>
<prop #hash.sha256>
<prop FileVersion>
```

Those variables can be used in scripts (e.g. `echo <prop FileVersion>;`) and templates (e.g. in Status Bar Templates).

Windows canonical properties (from Vista onwards only!)

Tip: The command property() , the Find Files selector prop:, and the <prop ...> variable all supports the locale-independent Windows canonical properties as listed here:

https://learn.microsoft.com/en-us/windows/win32/properties/props?redirectedfrom=MSDN

For example, select an image file and run this through the Address Bar:

```
echo property('System.Image.HorizontalSize') . 'x' . property('System.Image.VerticalSize');
```

Note that these canonical properties are case-sensitive.

XYplorer-specific Special Properties

A couple of XYplorer-specific named arguments are also supported. They all have to be preceded by # to distinguish them from Windows shell properties. They are not case-sensitive.

```
#Attr          = File Attributes as Letters, eg "AI" or "DJI"
#AttrDOS       = File Attributes as Letters in DOS style, eg "//R---A-T--OI--------"
#AttrDec       = File Attributes as decimal value, eg 12577
#AttrHex       = File Attributes as hexadecimal value, eg 0x00003121
```

#LinkTarget = Shortcut Target

#ShortcutTarget = Shortcut Target (same as #LinkTarget)

#JunctionTarget = Junction Target

#ResolveJunctions = Path with all junctions resolved, one pass (see below "Resolving Junctions")

#ResolveJunctionsAll = path with all junctions resolved, many passes

#ItemCount = Number of items (files and folders) in a folder (non-recursive, i.e. not including subfolders); "" (nothing) if folder unavailable

#Label = Label

#Tags = Tags

#Comment = Comment

#HardLinks = Number of Hard Links

#AspectRatio = Aspect ratio of an image file; also works for videos

#audio.bitdepth = Bit Depth

#audio.bitrate = Bit Rate

#audio.channels = Channels

#audio.length = Length

#audio.samplerate = Sample Rate

Audio Tags. Supported tag formats: **ID3 tags** (MP3 files), **Vorbis Comments** (FLAC and OGG files).

#tag.album = Tag Album

#tag.artist = Tag Artist

#tag.Comments = Tag Comments

#tag.Composer = Tag Composer

#tag.genre = Tag Genre

#tag.title = Tag Title

#tag.track = Tag Track

#tag.year = Tag Year

Vorbis Comments (FLAC and OGG files) are freely definable: You can pass whatever tag you want. If the tag is actually in the file you will get the value in return. Examples:

#tag.performer or #tag.PERFORMER (functionally identical)

#tag.encoder

#tag.whateveryouwant

#date.created, #date.c = Created Date

#date.modified, #date.m = Modified Date

#date.accessed, #date.a = Accessed Date

#image.dimensions = Dimensions (Width x Height); also works for videos

#image.width = Width; also works for videos

```
#image.height        = Height; also works for videos
#image.datetaken     = Date Taken
#image.cameramodel   = Camera Model
#image.focallength   = Focal Length
#image.fstop         = F-Stop
#image.exposuretime  = Exposure Time
#image.isospeed      = ISO Speed
#image.exposureprog  = Exposure Program
#image.exposurebias  = Exposure Bias

#image.hash          = Image Hash (see Image Hash)

#image.fuji.filmmode        = Film / Film Simulation
#image.fuji.highlighttone   = HTone / Highlight Tone
#image.fuji.shadowtone      = STone / Shadow Tone
#image.fuji.color           = Color /Color (Saturation)
#image.fuji.sharpness       = Sharp / Sharpness
#image.fuji.whitebalance    = WB / White Balance

#empty = Emptiness (see below)
#nofiles = Not containing any files in the branch, but possibly empty subfolders (see Remarks
below)
#nosubs = Not containing subfolders (see Remarks below)
#contains.[pattern] = Does a folder contain files matching a simple wildcard pattern (see
Remarks below)
#version = Version

#hash.md5 = MD5 Hash value of a file
#hash.sha1 = SHA1 Hash value of a file
#hash.sha256 = SHA256 Hash value of a file
#hash.sha512 = SHA512 Hash value of a file
#hash.crc32 = CRC32 Hash value of a file
```

Remarks

- Return values of #empty, #nofiles, and #nosubs:

  X = Accessed Denied
  E = Any error except "Access Denied"
  0 = item not found
  1 = empty file
  5 = non-empty file
  2 = empty folder (#nosubs: folder without subfolders) (#nofiles: branch without files)
  6 = non-empty folder (#nosubs: folder with subfolders) (#nofiles: branch with files)

- Return values of #contains.[pattern]:

  X = Accessed Denied
  E = Any error except "Access Denied"
  0 = Folder does not contain matching files
  1 = Folder contains matching files
  2 = Item is not a folder

- The property #image.datetaken supports matching partial dates:

```
prop:#image.datetaken:2005-06-28          //will match 2005-06-28 19:51:58 (and any other time on that day)
prop:#image.datetaken:6/28/2005           //will match 2005-06-28 19:51:58 (and any other time on that day)
prop:#image.datetaken:6/28/2005 07:51 PM  //will match 2005-06-28 19:51:58 (and any other second in that minute)
```

- The property #image.datetaken supports comparison operators (>, >=, <=, >) to define date ranges in your searches:

```
prop:#image.datetaken: >= 4/9/2007    //match all photos taken on or after that date
prop:#image.datetaken: <= 4/9/2012    //match all photos taken on or before that date
```

  Match all photos taken between those two dates:

```
prop:#image.datetaken: >= 4/9/2007 AND prop:#image.datetaken: <= 4/9/2012
```

Examples

```
text property("#attr");   //defaults to current list item
text property("#label");  //defaults to current list item
text property("#linktarget", "E:\Test\funstuff\Icons.lnk");
text property("#AspectRatio");  //defaults to current list item
text property("#Hash.SHA512");  //SHA512 value of the current list item
text property("#Hash.MD5", "E:\Test\funstuff\goodfellas.jpg");  //MD5 value of "goodfellas.jpg"
text property("#nofiles");  //returns 2 if this is a branch without files (else it returns 6)
text property("#contains.*.jpg", "E:\Test\funstuff\");  //returns 1 if "E:\Test\funstuff\" contains JPG files
```

Usage

This powerful function gives you direct and ultra-fast access to all available properties of all available files and folders in the system without any browsing involved!

Which particular property is available (and how it is called!) depends on the file, the file type, and on your OS. You have to experiment to find out what works.

Remarks

(1) Note that file times ("Write", "Create") are returned as Coordinated Universal Time (UTC), which is usually not identical to the System Filetime (unless you live around London).

(2) Also the formatting of the returned data is completely controlled by the Shell. E.g. the "FileVersion" of XYplorer.exe is returned as "8.10.0.4" as opposed to "8.10.0004" or "8.10.00.04".

Resolving Junctions

Example of a complex junction structure, where a part (C:\Backup\) of a junction target is itself a junction to yet another target (N:\):

```
C:\Dati\Condivisi\Musica\ = junction to C:\Backup\Condivisi\Musica\
                           and
C:\Backup\                = junction to N:\
```

Now this returns C:\Backup\Condivisi\Musica\:

```
text property("#ResolveJunctions", "C:\Dati\Condivisi\Musica\");
```

And this returns N:\Condivisi\Musica\:

```
text property("#ResolveJunctionsAll", "C:\Dati\Condivisi\Musica\");
```

Note that"#ResolveJunctionsAll" has built-in protection from endless recursion caused by cyclic junction structures.

quickfileview

Open the Quick File View for the specified file.

Syntax

```
quickfileview [filename], [codepage], [hexviewbytes]
```

filename	full path/name, or relative to current path; defaults to the current file.
codepage	codepage to use for decoding the file
empty:	use system default codepage
else:	use the passed codepage
hexviewbytes	
> 0:	number of bytes to show in hex view No problem if the file is shorter. Superfluous bytes are ignored.
0:	none (no forced hex view)
-1:	show all bytes of the file in hex view

Note

Hex view via hexviewbytes is internally limited to a maximum of 1MB of data. Reason: Speed.

Examples

```
quickfileview , 932;    //view the current file as Japanese (Shift-JIS):
```

```
quickfileview , 65001;  //view the current file as UTF-8
quickfileview , , -1;   //hex view the current file:
```

quicksearch()

Finds items.

Syntax

```
quicksearch([query="*"], [path], [separator="<crlf>"], [flags])
```

query	[optional] Search pattern, can have switches. Defaults to "*" (match all).
path	[optional] Search location. Defaults to the current path. You can pass several paths separated by "|" or ";" in the path argument (aka multi location search).
separator	[optional] Separates items in the returned search results list. Defaults to CRLF.
flags	[optional] String of flags to modify the function. m: Return metadata. The format is fixed: `name|size|modified|created|accessed|attributes` - name: full path name - size: raw bytes - dates: ISO 8601 - attributes: string of hyphens/letters (just like in the file list) n: Names only. Return matching items without path. l: Return directories with a trailing backslash. (l=lower case L) o: Sort results alphabetically in ascending order. These settings are honored by the sort procedure: - Configuration | General | Sort and Rename | Sort | Sort method - Configuration | General | Sort and Rename | Sort | Sort filenames by base s: Show status. Displays the status of the search in the Status Bar. t: Include top folders in multi location search results, i.e. top folders that match the search criteria are returned in the search results.
return	Search results list.

Remarks

- QuickSearch is probably the most powerful and useful of all scripting commands. It works 100% identical to GUI Quick Search apart from not affecting the GUI: The search results are returned by the function in a string. QuickSearch lets you quickly collect specific stuff from the most remote corners of your file system by means of simple scripts that can be stored and shared. The stuff is returned in a format ready for further automated processing. One of the most powerful file search engines compressed into a one-liner.

- The search is by default recursive (includes subfolders).
- Found items are always returned with full path.
- Paper Folders are supported.
- Portable Devices are supported.
- Multi location searches are supported: You can pass several paths separated by "|" or ";" in the path argument.
- You can use the Search In List aliases # and + in the path argument.
- The running function will pretty much block XYplorer.
- ESC can abort a search.

Examples

Search the current path for all JPG files:

```
text quicksearch("*.jpg");
```

The same using SC inputselect:

```
inputselect(, quicksearch("*.jpg",,"|"));
```

Search the current path for all JPG files, and show progress in the status bar:

```
text quicksearch("*.jpg", , , "s");
```

Search the whole computer for all items tagged Rock:

```
text quicksearch("tags:Rock", "*");
```

Search the current path for all items modified on a weekend:

```
text quicksearch("dateM: dw 6 | dw 7");
```

Search drive D: for all items larger 1 MB and smaller 2 MB:

```
text quicksearch("size: > 1MB AND size: < 2MB", "D:");
```

Search paperfolder "echo" non-recursively, i.e. not including subfolders:

```
text quicksearch("*.jpg /n", "paper:echo");
```

Search a specific path for all JPG files:

```
text quicksearch("*.jpg", "E:\Test");
```

Search two paths for all JPG files:

```
text quicksearch("*.jpg", "E:\Test;D:\Test");
```

Generate a full branch view of the current path:

```
text quicksearch();
```

Also Search In List (SIL) is supported:

```
text quicksearch("*.txt", "E:\Test.jpg;E:\Test2.txt");  //E:\Test2.txt
text quicksearch("*.txt", "<selitems ;>");
```

Return file names only (without path):

```
text quicksearch("jpg", 3:="n");
```

Find all JPG files in the current path and return them with metadata:

```
text quicksearch("*.jpg", , , "m");
```

Sample Return:

```
E:\Test\AltFacts.jpg|94078|2017-01-23 09:42:44|2017-01-23 09:42:43|2017-01-23 09:42:44|----A-----I-
E:\Test\OrangeOne.jpg|39138|2015-11-01 21:11:00|2017-01-27 14:19:58|2017-01-27 14:19:58|----A-----I-
```

Using the sorting switch "o". This makes a subtle difference on NTFS and a huge difference on FAT32/exFAT:

```
text quicksearch("*.txt", 3:="");   //unsorted
text quicksearch("*.txt", 3:="o");  //alphabetically ascending
```

Tip: By default GUI Quick Search (and hence the QuickSearch command) honors the current visibility settings ("Configuration | General | Tree and List | Items in Tree and List | Select Items... | Hidden files and folders" etc.) To return all matching items regardless of any GUI settings you can pass the /a switch:

```
text quicksearch("*.jpg");     //show visible only
text quicksearch("*.jpg /a");  //show all
```

quote()

Double-quotes a string.

Syntax

```
quote([string], [flags=0])
```

string	[optional] string
flags	0: Quote. 1: Unquote.
return	quoted string

Examples

```
echo quote("a"); // "a"
echo quote(1);   // "1"
echo quote();    // ""
echo quote('"Hola!"', 1); // Hola!
```

raf()

Sets, adds, or removes Rapid Access Folders (RAF).

Syntax

```
raf(itemlist, [mode=a], [separator=CRLF])
```

itemlist	List of items, separated by separator, to set, add, or remove.
mode	Type of action. a: [Default] Add to current RAF. r: Remove from current RAF. First it compares the passed items with the display names (left of ">" in the definition). If there is no match, it compares them with the target paths (right of ">" in the definition). s: Set as new RAF (dump previous).
separator	List separator. Defaults to CRLF.
return	The current RAF, separated by separator.

Examples

```
raf("C:\Windows"); //add Windows folder
raf("C:\Windows|Downloads", a, "|"); //add Windows and Downloads folder
raf("<xydata>"); //add XYplorer data folder (automatic display name)
raf("XY Config><xydata>"); //add XYplorer data folder (explicit display name)
text raf(<clp>, s); //set whatever is in the clipboard, return current RAF
raf("Windows", r); //remove Windows folder (by display name)
raf("C:\Windows", r); //remove Windows folder (by target path)
text raf(); //just return current RAF
```

rand()

Generates an integer random number.

Syntax

```
rand([low=0], [high=1])
```

low	[optional] the lowest value to return (default: 0)
high	[optional] the highest value to return (default: 1) should be same or higher than low
return	random number

Examples

```
echo rand(); //returns 0 or 1
echo rand(2, 4); //returns 2 or 3 or 4
echo rand(-1, 1); //returns -1 or 0 or 1
```

readfile()

Read the contents of a file into a variable.

Syntax

```
readfile(filename, [mode], [numbytes], [codepage], [start=1])
```

filename — Full path/name, or relative to current path.
Defaults to the currently selected file.

mode

t: [default] text; whether file is ASCII or UTF-8 or Unicode is auto-detected
t_BLU8: text, check for BOM-less UTF-8;
of course, UTF-8 with BOM will also be detected and decoded in this mode

b: binary (converts bytes according to the active codepage)
corresponds to mode "b" in WriteFile()
r: raw bytes (binary safe, no conversion)
corresponds to mode "r" in WriteFile()
ru: like "r" but converted to Unicode (2 bytes per character) and thus can be displayed as readable text (which in XYplorer is always in Unicode)

numbytes

empty: read whole file

else: number of bytes to read; if longer than file size then the number is silently set to file size;
if <= 0 the function returns an empty string (and does not even try to read the file)

codepage — Codepage to use for decoding the file.

empty: use system default codepage

else: use the passed codepage

start — Byte position to start reading from.
1 = beginning of file

Remarks

- This command will not read more than 100 MB at a time. To read larger files use the "start" parameter and do it portion by portion.
- Note that the "start" parameter points to the *character* position in case of text files (in double-

byte encodings this is different from the byte position).

- The wildcard * is supported in the filename parameter. The first matching item is used according to NTFS item order.
- When mode is explicitly set to "t" (Text) or "t_BLU8" (Text, check for BOM-less UTF-8) the file is checked for UTF-8 even if its extension is not among the Text Files extensions defined in Configuration | Preview | Previewed Formats.

Examples

Reads text from "test-in.txt" (ASCII or UNICODE) and writes it back to "test-out.txt" (ASCII or UNICODE):

```
$a = readfile("test-in.txt"); writefile("test-out.txt", $a);
```

Reads text from "test-in.txt" (ASCII or UNICODE) and writes it back to "test-out-A.txt" in ASCII (1 byte per char):

```
$a = readfile("test-in.txt"); writefile("test-out-A.txt", $a, , "ta");
```

Reads text from "test-in.txt" (ASCII or UNICODE) and writes it back to "test-out-U.txt" in UNICODE (2 bytes per char):

```
$a = readfile("test-in.txt"); writefile("test-out-U.txt", $a, , "tu");
```

Read first 4 bytes of the current file:

```
echo readfile( , , 4);
```

Decode the current file as Japanese (Shift-JIS):

```
text readfile(, , , 932);
```

Decode the current file as UTF-8:

```
echo readfile(, , , 65001);
```

Read current file from position 100:

```
text readfile(, , , , 100);
```

You can actually use ReadFile() and WriteFile() for file copying on a low level (bypassing the Shell):

Copy "test-in.txt" to "test-out.txt" (text file):

```
writefile("test-out.txt", readfile("test-in.txt"));
```

Copy "xy.png" to "xy-out.png" (could be any file, binary or text); source and target will be 100% identical independently of your locale:

```
writefile("xy-out.png", readfile("xy.png", "r"), , "r");
```

Showing the first 50 bytes of the current file as hex dump:

```
text hexdump(readfile( , "b", 50));     //locale-specific conversions possible
```

```
text hexdump(readfile( , "r", 50), 1);  //you see the true bytes
```

```
text hexdump(readfile( , "ru", 50));    //you see the true bytes
```

When using the wildcard * the first matching item is used according to NTFS item order:

```
text readfile("<xypath>\*.txt");    //opens the first TXT file
```

```
text readfile("<xypath>\XY*.txt");  //opens the first matching TXT file
```

readonly

Runs a new XYplorer instance with the current configuration as saved on disk, and will not overwrite that configuration.

Syntax

```
readonly [startpath]
```

startpath:	Start path in the first pane. Defaults to the start path as saved on disk.

Example

```
readonly;        //start readonly instance
```

```
readonly "C:\";  //start readonly instance in C:\
```

readonlyhere

Like readonly but opened to current path.

Syntax

```
readonlyhere
```

Example

```
readonlyhere;    //start readonly instance in current path
```

readpv

Reads permanent variables (PVs) from a file.

Syntax

```
readpv [file]
```

file	The file to read from. Can be relative to the current path. If empty it defaults to "<xydata>\pv.dat".

Examples

```
readpv;  //read from "<xydata>\pv.dat"
```

Remarks

- The file should have been previously created by writepv, otherwise it will not match the expected format.
- The PVs will be added to the current PVs. Current PVs of the same name will be overwritten.

readurl()

Reads the contents of a web file into a variable.

Syntax

```
readurl(url, [nocookies], [StripHTML], [flags], [codepage])
```

url	[required] URL (supported are http:// , https:// , ftp://, file:///).
nocookies	[optional] 0: use cookies 1: don't use cookies
StripHTML	[optional] 0: return unchanged HTML 1: strip HTML comments and tags, scripts and styles, convert some common HTML entities, and remove excessive tabs, spaces and empty lines
flags	(bit field) 1: [reserved] 2: silent mode (no error messages) 4: Check byte count after download. On no-match the download fails. 8: No status bar progress and success messages.
codepage	Code page.

Some common code pages:

```
874     'Thai
932     'Japanese
936     'Simplified Chinese (PRC, Singapore)
949     'Korean (EUC)
950     'Traditional Chinese (BIG5)
65000   'UTF-7
65001   'UTF-8
```

return	The contents of the web file.

Examples

Displays the contents of the given URL (HTML code in this case) in a text box:

```
text readurl("https://www.xyplorer.com/index.php");
```

Displays the pure text of that URL in a text box:

```
text readurl("https://www.xyplorer.com/download.php", 0, 1);
```

Displays the decoded contents of a Japanese website encoded in UTF-8:

```
text readurl("https://ja.wikipedia.org/",,,,65001);
```

Reading local HTML files (the file:/// prefix is optional):

```
text readurl("file:///<curitem>", , 1, , 65001); //1=StripHTML, 65001=UTF-8 codepage
```

```
text readurlutf8(<curitem> , , 1);
```

Notes

The data size is limited to 100MB.

readurlutf8()

Reads the contents of a web file into a variable. Identical to readurl, but defaults to decoding the contents as UTF-8.

Syntax

```
readurlutf8(url, [nocookies], [StripHTML], [flags], [codepage])
```

Remarks

Many (if not most) websites are UTF-8 encoded, so this offers a simpler way to handle this than passing "65001" for codepage.

Example (a Japanese website encoded in UTF-8):

```
text readurlutf8("https://ja.wikipedia.org/");
```

recase()

Changes the case of a string.

Syntax

```
recase(string, [mode], [flags], [titlecase_exceptions_lower], [titlecase_exceptions_keep])
```

string	String to change case of.
mode	One of the following (full word, or first letter): l, lower: [default] to lower case. u, upper: to upper case.

t, title: to Title Case (first letter of each word to upper case, other letters to lower case).
c, camel: like "title" but preserves any caMel-case.
i, invert: invert the case of each letter.
s, sentence: first letter upper case, all others lower case.
r, removediacritics: strip accents and the like from letters, e.g. converts "à" to "a".
n, none: don't change anything.

flags (bit field)
1: set extensions to lower case

titlecase_exceptions_lower
missing: Use the global defaults (as stored at key RenameTitleCaseExceptions).
empty: Do not use any exceptions.
else: Use this ";"-separated list of exceptions, e.g. "a;an;the".

titlecase_exceptions_keep
missing: Use the global defaults (as stored at key RenameTitleCaseExceptionsKeep).
empty: Do not use any exceptions.
else: Use this ";"-separated list of exceptions, e.g. "ABBA;LaTeX;QUEEN;UFO".

return String with changed case.

Examples

```
text recase("the caMel can't.", "lower"); //the camel can't.
text recase("the caMel can't.", "upper"); //THE CAMEL CAN'T.
text recase("the caMel can't.", "title"); //The Camel Can't.
text recase("the caMel can't.", "camel"); //The CaMel Can't.
text recase("the caMel can't.", "invert"); //THE CAmEL CAN'T.
text recase("the caMel can't.", "sentence"); //The camel can't.
text recase("Motörhead", "removediacritics"); //Motorhead
text recase("This is a test.HTM", "title", 1); //This Is A Test.htm
text recase("This is a test.HTM", "n", 1);     //This is a test.htm
```

Examples for Title Case

```
text recase("the caMel BITES the horse.", "t");                 //The Camel Bites the Horse. (factory default exceptions)
text recase("the caMel BITES the horse.", "t", , "");           //The Camel Bites The Horse. (no exceptions)
text recase("the caMel BITES the horse.", "t", , "the;bites");  //The Camel bites the Horse. (custom exceptions)
text recase("ABBA - dancing QUEEN.mp3", "t", 1, , "QUEEN");              //Abba - Dancing QUEEN.mp3
text recase("ABBA - take a chance on me.mp3", "t", 1, , "ABBA");          //ABBA - Take a Chance on Me.mp3
text recase("ABBA - take a chance on LaTeX.mp3", "t", 1, , "ABBA;LaTeX"); //ABBA - Take a
```

```
Chance on LaTeX.mp3
```

Examples for Camel Case

```
text recase("the caMel BITES the horse.", "c");                 //The CaMel BITES the Horse. (factory default exceptions)
text recase("the caMel BITES the horse.", "c", , "");           //The CaMel BITES The Horse. (no exceptions)
text recase("the caMel BITES the horse.", "c", , "the;bites");  //The CaMel bITES the Horse. (custom exceptions)
```

refresh

Refreshes the contents of various controls.

Syntax

```
refresh [control="list"]
```

control — Control to refresh. These are supported:
list [Default]: The file list of the active pane. Note: Will remove a search and go into browse mode.
tree: The folder tree.
toolbardrives: The toolbar drive buttons.

Examples

```
refresh;  //refresh active list
refresh "toolbardrives"; //refresh toolbar drive buttons
```

refreshlist

Refreshes the list on any of the panes.

Syntax

```
refreshlist [pane="a"]
```

pane
a: [default] active pane
i: inactive pane
1: pane 1
2: pane 2

Examples

```
refreshlist;      //refresh active pane
refreshlist "i";  //refresh inactive pane
```

regexmatches()

Returns a list of all matches of a regular expression pattern in a given string.

Syntax

```
regexmatches(string, pattern, [separator="|"], [matchcase=0])
```

string	String to work on (haystack).
pattern	The RegExp pattern to search for in string (needle).
separator	Separator between matches in the returned list. Defaults to | (pipe).
matchcase	[Defaults to 0] Compare method. 0: A=a 1: A<>a
return	List of matches.

Example

```
text regexmatches("tiger,frog,ball", "[^,]+"); //tiger|frog|ball
```

regexreplace()

Replaces parts of a string, using a regular expression pattern.

Syntax

```
regexreplace(string, pattern, [replacement], [matchcase])
```

string	String to work on (haystack).
pattern	The RegExp pattern to search for in string (needle). The caret (^) & dollar ($) match the beginning and end of every *line* (not the very start and very end of the entire string).
replacement	String to replace with. If missing or empty then all occurrences of pattern will simply be removed (replaced by nothing).
matchcase	[Defaults to 0] Compare method. 0: A=a 1: A<>a
return	The new string.

Examples

```
$a = RegExReplace("Image[1].png", "(?#Remove common IE suffix)^(.+)\[[0-9]+\](\..+)$",
```

```
"$1$2");
```

Sets $a to "Image.png", removing the common IE suffix [1] from the filename.

```
$a = RegExReplace("bob", "b$", "p");
```

Sets $a to "bop" (the final "b" is replaced by "p").

releaseglobals

Releases all global/permanent variables from memory.

Syntax

```
releaseglobals [flags=3]
```

flags	(bit field, defaults to 3) 1: release non-perm globals 2: release perm globals

Examples

The following script illustrates that the Permanent Variable $foo is gone (unset, no variable anymore) or not gone after "releaseglobals" and depending on the flags:

```
// to release or not to release
  perm $foo; $foo = "hi!"; releaseglobals;
  echo $foo;  //$foo
  perm $foo; $foo = "hi!"; releaseglobals 1;
  echo $foo;  // "hi!"
  perm $foo; $foo = "hi!"; releaseglobals 2;
  echo $foo;  //$foo
```

rename

Renames the currently selected List item(s) according to the defined pattern and options.

Syntax

```
rename [mode (b|r|s|k|e|l)], pattern, [preview (p)], [itemlist], [flags=1],
[illegalcharsreplacewith]
```

mode	Defines the mode for the rename operation: b: [Default] Batch Rename r: RegExp Rename s: Search and Replace k: Keep Particular Characters e: Set Extension

l: Rename the selected items using a list of names passed in the "pattern" argument (see Remarks below)

CamelCase: A* A*.aaa
ProperCase: Aaa Aa.aaa
ProperCase_NX: Aaa Aa.aaa (Same as "ProperCase" but without applying the exceptions defined in tweaks RenameTitleCaseExceptions and RenameTitleCaseExceptionsKeep.)
LowerCase: aaa aa.aaa
UpperCase: AAA AA.AAA
LowerCaseExt: *.aaa
UpperCaseExt: *.AAA
SpacesToUnderscores: Spaces to _
UnderscoresToSpaces: _ to Spaces
RemoveDiacritics: Remove Diacritics
ConvertToASCII: Convert to ASCII
UrlEscape: UrlEscape (Space > %20 ...)
UrlUnescape: UrlUnescape (%20 > Space ...)
UnicodeToUTF8: Unicode to UTF-8
UTF8ToUnicode: UTF-8 to Unicode

NameToID3_ArtistTitle: Filename (Artist - Title.mp3) to ID3 Tag
NameToID3_TrackArtistTitle: Filename (Track - Artist - Title.mp3) to ID3 Tag
NameToID3_ArtistAlbumTrackTitle: Filename (Artist - Album - Track - Title.mp3) to ID3 Tag
NameToID3_ArtistYearAlbumTrackTitle: Filename (Artist - Year - Album - Track - Title.mp3) to ID3 Tag
ID3ToName_ArtistTitle: ID3 Tag to Filename (Artist - Title.mp3)
ID3ToName_TrackArtistTitle: ID3 Tag to Filename (Track - Artist - Title.mp3)
ID3ToName_ArtistAlbumTrackTitle: ID3 Tag to Filename (Artist - Album - Track - Title.mp3)
ID3ToName_ArtistYearAlbumTrackTitle: ID3 Tag to Filename (Artist - Year - Album - Track - Title.mp3)

pattern	[Required] The pattern to be applied. Its syntax may vary depending on which mode is used, see below for more.
preview	Defines whether or not to show the Rename Preview before the actual rename operation. [empty]: [Default] No Preview. Omit this parameter to rename the files directly, without preview or confirmation first. p: Show Preview. The rename preview will be shown before anything happens, allowing you to verify, change the pattern or cancel the operation at will.
itemlist	CRLF- or |-separated list of items (full path) to rename. If empty then current list selections are renamed. The separator may be surrounded by any number of blanks.

flags (bit field, defaults to 1)
1: PromptOnFailure (prompt on each failed rename attempt)
2: ShowListOfUntouched (show list of all items that kept their old name)
4: Replace illegal characters (with the value of argument illegalcharsreplacewith) *after* passing it to the internal rename function. Any switches will work as expected, since they are processed before replacing illegal characters.
8: Increment on collision. Note that this does the same as the /i switch, but it's much easier to handle when taking patterns e.g. from the clipboard.
16: Clean the input (replace invalid characters) *before* passing it to the internal rename function. Note: Any switches will NOT work as expected as the "/" will be replaced.
32: Uppercase the first letter.
64: Suppress the "Rename canceled." error message when user cancels any dialog related to rename, e.g. the Rename Preview.
128: Overwrite on collision. No questions, no undo.

illegalcharsreplacewith Replace illegal characters with this string, IF flags has the "4" bit set.

Remarks on Mode l ("list"):

- Renames the selected items using a list of names passed in the "pattern" argument. If no list is passed then the "Edit Item Names" dialog is opened pre-filled with the currently selected names.
- The selected list items are renamed top to bottom (just like with the "Edit Item Names" GUI command).
- Tip: If "Allow move on rename" is ticked (Configuration | General | Sort and Rename | Rename | Allow move on rename) then you can pass absolute or relative (to the files current path) paths in the "pattern" argument, and the files will be moved accordingly. Moving works even to other volumes and any missing paths are silently created. Example:
```
// files are renamed and moved to "Logs" subfolder, with preview
rename "l", "Logs\Don2017.jpg|Logs\Don2018.jpg", "p";
```

Examples

Note the single quotes around the date variable: This ensures that it is only resolved inside the rename procedure on a per-file basis. If it were double-quoted then all selected files would be appended the modified date of the currently focused item.

```
rename b, '*-<datem yyyymmdd>', p; //Appends the modified date to each of the selected items. With preview.

rename b, '*-<datem yyyymmdd>'; //Appends the modified date to each of the selected items. Without preview.

rename , '*-<datem yyyymmdd>'; //Same as above since the first parameter defaults to "b".

rename , '<dateexif yyyymmdd.hhnnss>-*', p; //Prepends the EXIF date to each of the selected items.

rename r, "\.html$ > .htm", p; //RegExp Rename replacing html extensions by htm.

rename s, "ü/ue"; //Search & Replace Rename replacing all "ü" by "ue".

rename k, "0123456789", p; //Keep Particular Characters Rename, keeping only the numbers
```

```
present in a filename.
rename e, "htm"; //Change any current extension to "htm".
```

Rename a specific file (here: E:\TestFiles\asia.jpg) to a hardcoded new name (africa.png), using itemlist and the /e switch (for renaming the extension as well):

```
rename , "africa.png /e", p, "E:\TestFiles\asia.jpg";
```

Rename from clipboard:

```
rename b, <clipboard>, p, , 4;  //drop illegal characters
rename b, <clipboard>, p, , 4, "_"; //replace them with _
rename b, <clipboard>, p, , 12, "_"; //same as before with increment
```

Rename by list:

```
rename "l";                                  //dialog is pre-filled with current item names
rename "l", "Don2017.jpg|Don2018.jpg";       //dialog is skipped; no preview
rename "l", "Don2017.jpg|Don2018.jpg", "p"; //dialog is skipped; preview
```

Examples for drop-on bookmarks in the toolbar, with rename preview; note that multiple items can be dropped on these bookmarks:

```
rename "LowerCaseExt", , p, <drop>;
rename "ID3ToName_ArtistYearAlbumTrackTitle", , p, <drop>;
```

For the functionality of the different rename operations see also here.

renameitem()

Renames a file or folder.

Syntax

```
renameitem(newname, [sourceitem=<curitem>], [flags=0], [numsuffix])
```

newname	The new name (will be used as base and/or extension, depending on flags). The newname argument supports the following placeholders: * = original base ? = original extension
sourceitem	The item to rename. Absolute or relative to current list path. Defaults to the current item.

flags	0: [Default] Smart (keep extension unless extension is passed) 1: Rename base (keep extension) 2: Rename extension (keep base) [1 + 2 = 3: Rename all (base and extension)] 4: Show dialog (Suffix / Overwrite) on name collision. 8: Silent overwrite on name collision. 16: Force appending the numsuffix. If numsuffix is not passed as 4th argument then it defaults to the incremental suffix set in Configuration. Notes: 1 and 2 are only applicable if sourceitem is a file, not a folder. On 4 numsuffix defaults to the incremental suffix defined in Configuration | Templates. If 8 is passed then 4 is ignored.
numsuffix	Number template to auto-suffix on collision. Set the start value and use "0"s as placeholders for numerical increments (e.g. "-01", starting with "-01", "-02", "-03", etc.) or "a"s as placeholders for alphabetical increments (e.g. "-aa", starting with "-aa", "-ab", "-ac", etc.)
return	The *new name* if the rename was successful, else *nothing* (empty string).

Examples

Rename current file to "John[.ext]" (or current folder to "John") in current folder (fail on collision):

```
renameitem("John");
```

Same as above, but auto-suffix on collision:

```
renameitem("John", , , "-01");
```

Rename "Paul.txt" to "John.txt" in current folder:

```
renameitem("John", "Paul.txt", , "-01");
```

Rename current file" to "<clipboard>[.ext]" in current folder:

```
renameitem(<clipboard>);
```

Rename current item to "[base.]jpg" (i.e. change the extension):

```
renameitem("jpg", , 2);
```

Rename the current item to "John"; on name collision show the dialog with options to Suffix, Overwrite, or Cancel.

```
renameitem("John", , 4, "-001");
```

Same as above using the incremental suffix defined in Configuration | Templates.

```
renameitem("John", , 4);
```

Overwrite existing file without any warning:

```
renameitem("John.jpg", , 8);
```

Examples for using placeholders * and ?

This renames a current file "hash.jpg" to "Copy of hash.jpg":

```
renameitem("Copy of *");
```

This renames a current file "hash.jpg" to "(jpg) Copy of hash.jpg":

```
renameitem("(?) Copy of *");
```

Examples for flag 16:

Context: A file named "John.jpg" already exists in the path and is selected.

```
renameitem("John.jpg", , , "-01");    //file not changed
renameitem("John.jpg", , 16, "-01");  //file renamed to John-01.jpg
```

Remark

Contrary to SC rename, SC renameitem is able to rename items that are not currently listed or selected, and it can auto-suffix numbers on collision. This gives you a number of fascinating new options.

replace()

Replace parts of a string.

Syntax

```
replace(string, search, [replacement=""], [matchcase=0], [start=1], [count=-1])
```

string	String to work on (haystack).
search	String to be replaced (needle).
replacement	String to replace with. Default is empty string "": replace by nothing.
matchcase	Compare method. 0: A=a [Default] 1: A<>a
start	Position from left where the replacement starts.
count	Maximal number of replacements.
return	The new string.

Usage

When no replacement is defined, all occurrences of search will simply be removed (replaced by nothing).

Examples

```
echo replace("Maxi", "max", "Min");    // "Mini"
echo replace("Maxi", "max", "Min", 1); // "Maxi"
echo replace("aaa", "a", "b", , 2);     //abb
```

```
echo replace("aaa", "a", "b", , 2, 1);  //aba
echo replace("aaa", "a", "b", , , 1);   //baa
```

The following line transforms the current location to a Unix-style path and copies it to the clipboard. For example, if the current location is S:\usr\local\apache2\conf then this script will copy /usr/local/apache2/conf to then clipboard:

```
$p = substr("<curpath>", 2); copytext replace($p, "\", "/");
```

replacelist()

Replaces substrings by list.

Syntax

```
replacelist(string, searchlist, [replacelist], [separator], [matchcase], [scope])
```

string	String to work on (haystack).
searchlist	Substrings to be replaced (needles).
replacelist	Substrings to replace with. empty: All substrings in searchlist are removed. Special case: If you state only one replace unit, all items in the search list are replaced with that one unit. Consequently if the replace list is empty then all items in the search list are removed from the string.
separator	Separates the strings in the lists. empty: Each character is taken as a substring.
matchcase	Compare method. 0: A=a [Default] 1: A<>a
scope	Matching scope (what to match in string) 0: substring [default] 1: whole words 2: whole string
return	The new string.

Examples

```
text replacelist("Taxi", "ai", "ia"); //Tixa
text replacelist("Taxi", "ax,i", "-,Rex", ","); //T-Rex
text replacelist("Taxi", "ai"); //Tx (a and i removed from Taxi)
text replacelist("Taxi", "ai", "o"); //Toxo (a and i replaced by o)
text replacelist("Taxi", "a,i", "ou", ","); //Touxou (a and i replaced by ou)
```

Examples for scope:

```
text replacelist("Hello, Taxi Driver!", "He|Hello|Hello, Taxi Driver!", "Ha|Hi|Taxi!", "|", 0, 0); //Hallo, Taxi Driver!
```

```
text replacelist("Hello, Taxi Driver!", "He|Hello|Hello, Taxi Driver!", "Ha|Hi|Taxi!", "|", 0, 1); //Hi, Taxi Driver!
```

```
text replacelist("Hello, Taxi Driver!", "He|Hello|Hello, Taxi Driver!", "Ha|Hi|Taxi!", "|", 0, 2); //Taxi!
```

Note how in lines 1 and 2 there are two replacements (Hello->Hi; TaxiDriver->Taxi Driver):

```
text replacelist("Hello, TaxiDriver!", "He|Hello|TaxiDriver|Hello, TaxiDriver!", "Ha|Hi|Taxi Driver|Taxi!", "|", 0, 0); //Hallo, Taxi Driver!
```

```
text replacelist("Hello, TaxiDriver!", "He|Hello|TaxiDriver|Hello, TaxiDriver!", "Ha|Hi|Taxi Driver|Taxi!", "|", 0, 1); //Hi, Taxi Driver!
```

```
text replacelist("Hello, TaxiDriver!", "He|Hello|TaxiDriver|Hello, TaxiDriver!", "Ha|Hi|Taxi Driver|Taxi!", "|", 0, 2); //Taxi!
```

Notes

- As you see from the 2nd example, the strings in search-replace pair can have different lengths.
- If the item count in both list differs, then the smaller count is used and the surplus items are ignored.
- The string is walked one time from left to right; any replaced parts are not processed again. So in this example, "Rex" will not be replaced by "Bone" in a second pass:
  `text replacelist("Taxi", "ax,i,Rex", "-,Rex,Bone", ","); //T-Rex`
- The substrings are processed from left to right, first match wins. Therefore:
  `text replacelist("Taxi", "a,ax", "i,ox"); //Tixi`
  `text replacelist("Taxi", "ax,a", "ox,i"); //Toxi`

Usage

This command can be used for interesting things, e.g. to (roughly) transliterate e.g. Cyrillic to Latin, or do some simple encryption, or clean file names from undesired characters.

report()

Returns a report of the current file list contents.

Syntax

```
report([template], [itemlist], [header], [footer])
```

template	Defines the layout of one line (file record); for each reported file one line is created based on this template. If missing or empty: Take the current list data as is.
itemlist	What to report on: 0: [Default] All current list items. 1: All currently selected list items.

2: The currently focused list item.
3: The currently hovered list item.
[else]: Items in a CRLF- or |-separated list of items (full path).
Only files and folders, drives are not supported here.

header — Any string data to be put at the top of output.

footer — Any string data to be put at the bottom of output.

return — The report.

Omitting the Template

If you omit the template argument (or pass "") the report you get is identical to the list you see: Same types and sequence of columns, same file data in same format. The fields are by default separated by Tab characters.

You may overwrite the default separator by passing a different separator through the template argument.

Examples

Use the default field separator Tab: `text report();`

Use ", " as field separator: `text report(", ");`

Use "{Name}. " as template (see next paragraph): `text report("{Name}. ");`

Using the Template

If you use the template argument you are in full control and can decide what is shown, where it is shown, and how it is shown. The various file data are defined by fields, corresponding to the columns the file list. A field consists of a field name and, in some cases, an optional format definition, separated from the field name by a single blank. The fields are enclosed in curly brackets {}, and they are not case-sensitive (i.e. a=A). Example for a template argument:

```
"No. {#}, {name}, {size kb} ({size b}), {modified yyyy-mm-dd}"
```

The following fields are currently supported (they obviously match the file list's column headers):

Browse Mode: {#}, {Name}, {Ext}, {Size}, {Type}, {Created}, {Modified}, {Accessed}, {Attr}, {Len}, {Label}, {Tags}, {Comment}

Find Mode: {#}, {Name}, {Ext}, {Size}, {Type}, {Created}, {Modified}, {Accessed}, {Attr}, {Len}, {Label}, {Tags}, {Comment}, {Path}

Drives Mode: {#}, {Name}, {Type}, {Total Size}, {Used Space}, {Free Space}, {Full %}, {Free %}, {Per Cluster}, {Vol Serial}, {File System}

Optional format definitions for the {Size} field:

```
{Size KBR}    KB (rounded up)
{Size KB}     KB
{Size B}      Bytes (no suffix)
```

```
{Size BB}     Bytes (with suffix)
{Size FLEXR}  Flexible (rounded up)
{Size FLEX}   Flexible
{Size RAW}    Raw bytes (no thousand separators)
{Size MB}     MB
{Size GB}     GB
{Size TB}     TB
{Size PB}     PB
{Size}        [no format: use current Size Column Format]
```

Optional format definitions for {Created}, {Modified}, {Accessed}:

```
{Created yyyy-mm-dd hh:nn:ss}
```

etc. ... the usual date format definitions...

```
{Created}     [no format: use current Date Column Format]
```

Optional format definitions for {#}:

```
{# @@@@}      fill with leading blanks
{# 000}       fill with leading zeroes
```

Note that you cannot combine @ and 0 to have first leading blanks and then leading zeroes.

Metadata (aka Extended Properties)

The so-called "Extended Properties" (the complete set of available metadata of file) are fully supported using the field `{prop:[property]}`. `[property]` can be a numeric as well as a literal property identifier.

For example, the following line reports the image dimensions for image files in the current list using the literal property identifier "dimensions":

```
text report("{name} ({prop:dimensions})<crlf>");
```

This line yields the same results in a different way:

```
text report("{name} ({prop:ImageX} x {prop:ImageY})<crlf>");
```

This line yields the same results in yet another way, namely with numeric property identifiers (note that these numbers only work in Win7!) :

```
text report("{name} ({prop:#162} x {prop:#164})<crlf>");
```

Note that MS made a total mess here so that each Windows version has a completely reshuffled set of numbers and keywords. But there is help: The line numbers in "Configuration | File Info Tips & Hover Box | Show these fields" give you the actual values of all available extended properties in your current Windows version. Literal property identifiers are not documented at all by MS, so you can only guess, trial, and error.

Additional Special Fields

`{DocTitle}, {DocSubject}, {DocAuthor}, {DocCategory}, {DocComments}`

- These fields that will retrieve the respective meta properties from all file types that support meta properties.

{FileVersion} = version of EXEs, DLLs, etc.

{WhenTaken} = EXIF date of photos

{Dimensions} = width x height of images

- Note that these fields only work under XP! The labels have been changed in Vista and later.

{Fullname}

- The field is set to the full path/name of the file.
- Obviously only for Browse and Find modes, not Drives.

{Fullpath}

- Field is set to the full path of the item.
- The returned path has no backslash.
- In Find mode, the return is identical to the field {Path}.
- In Drives mode, "Computer" (or so) is returned.
- In Network mode, "Network Places" (or so) is returned.

{Basename}

- The field is set to the base name of the file (file name without extension).
- Folder names are returned unchanged.

{LabelID}

- Set to the item's Label ID. If the item has a comment but no color label assigned {LabelID} returns "0"; if the item is not tagged at all {LabelID} returns "".

{Count}

- Returns the (recursive, i.e. including subfolders) count of items (files and folders) contained within a folder. For files it returns nothing.

Note that calculating the count can take time with large folders.

Example, using the meta-field {dir} to fork returns between folders and files:

```
text report("{name}{dir , <DIR> with {count} items| ({size kbr})}<crlf>");
```

{Dir dir_value|file_value|drive_value}

- Field is set to "dir_value" if item is a directory.

- Field is set to "file_value" if item is a file.
- Field is set to "drive_value" if item is a drive.

Note that the field {Dir ...} may contain other fields (but not contain itself). For example, in

```
text report("{Name} - {Dir -|{Size B} ({Size FLEX})|}<crlf>");
```

the {Dir ...} field will return:

- for directories: -
- for files: {Size B} ({Size FLEX}) [resolved, of course]
- for drives: [nothing]

{Zebra odd_value|even_value[|divisor=2]}

- Field is set to "odd_value" in all odd lines.
- Field is set to "even_value" in all even lines.
- The divisor defaults to 2 but you can set it to any larger value so that e.g. the "even_value" (the value that divides by the divisor without remainder) is set to every 5th line, the "odd_value" to all other lines.

You can use this field to apply alternate formatting, e.g. to achieve a zebra striping effect if you use HTML for your report, or to group the output into blocks of a certain size. See examples below.

Extra Columns

Extra Columns are referenced by their case-insensitive canonic names (independent of any translations), e.g. "extra 1".

Example:

```
text report("{name}: {extra 1}, {extra 2}, {extra 3}, {extra 4}, {extra 5}<crlf>");
```

Columns of type Date can be formatted in the usual way:

```
{extra 5 yyyy-mm-dd}
```

Columns of type Checkbox can be formatted like this:

```
{extra 4 yn}    = Show localized "Yes" (non-empty) or "No" (empty)
{extra 4 y}     = Show localized "Yes" (non-empty) or nothing (empty)
{extra 4}       = Show the raw contents
```

There is also a fully customizable format for the non-empty/empty states:

```
{extra 4 OnNonEmpty|OnEmpty}  = Show "OnNonEmpty" on non-empty, "OnEmpty" else
```

Example (assuming Extra 4 is a Checkbox column):

```
text report("{name}: {extra 4 Done|To Do}<crlf>");
```

Custom Columns

Custom Columns can be referenced by their case-insensitive canonic names (independent of any translations), e.g. "custom 1", e.g.:

```
text report("{name}: {custom 1}, {custom 2}, {custom 3}, {custom 4}, {custom 5}<crlf>");
```

Columns of type Date can be formatted in the usual way, e.g.:

```
text report("{name}: Year {custom 5 yyyy}<crlf>");
```

Custom Columns and Extra Columns can as well be referenced by their case-insensitive current captions as long as they are shown in the current list. In this case no formatting is possible though. E. g.:

```
text report("{name}: {Dimensions}, {Aspect Ratio}, {Camera Model}<crlf>");
```

Other:

```
text report("{custom 17}", 2); //show the Custom Column #17 cell for the focused list item
```

```
text report("{custom 17}", 3); //show the Custom Column #17 cell for the hovered list item
```

Examples

```
text report();
```

Displays the current file list "as is" in a multiline textbox.

```
text report("{modified yyyy-mm-dd hh:nn:ss}<crlf>", 1);
```

Displays the modified date of all selected items in a multiline textbox. Note that it's your turn to supply a carriage return / line feed if needed. In the example, the internal variable `<crlf>` (resolved to 0xd0a0) ends the line.

```
text report("{modified yyyy-mm-dd hh:nn:ss.fffffff}<crlf>", 1);
```

The same as above with maximum resolution down to the 1/10 microsecond.

```
text report("{name}; {size KBR}; {modified}<crlf>", , "Report on <curpath>, <date><crlf>",
"---");
```

Displays a mini-report of the current file list in a multiline textbox.

```
text report('The item "{name}" has {size BB}.' . "<crlf>");
```

As you see, these fields have nothing to do with scripting variables, hence they are resolved whether single-quoted or not.

```
text report('The file "{fullname}" has {size B} bytes.' . "<crlf>");
```

Displaying the full name.

```
text report("{name} is a {dir folder|file|drive}.<crlf>");
```

Showing whether item is file, folder, or drive.

```
text report("{# @@@@} {name}<crlf>{zebra |<crlf>|5}");
```

Fill leading blanks to the line number, and add an empty line after every 5 lines. Note that the XY variable <crlf> works within the double-quoted argument.

```
text report("{name}, {free %}, {total size gb}<crlf>");
```

Mini-report for the drives listing (Computer node).

```
writefile("Report.htm",
  report("<tr><td>{Name}</td>
  <td align=right>{Size B} {dir [DIR]|bytes|}</td>
  <td>{Created}</td><td>{Modified}</td></tr><crlf>", ,
  "<table cellpadding=4>", "</table>"));
```

Generate an HTML table of the current folder and save it to Report.htm.

```
text report("{Name}, {Size B} bytes, {Modified yyyy-mm-dd hh:nn:ss},
  ver {FileVersion}", "%winsysdir%\shell32.dll");
```

Report on a single specific item.

Other examples:

```
text report("{Label}", 3); //show the Label for the hovered list item
```

Remarks

(1) The sort order is the one currently present in the file list.

(2) The {#} field (line number) is not filled with the line number each file currently has in the file list, but simply numbers the lines in the report starting with 1. This coincides with the actual line numbers in the file list only if the report features all files (not just a subset of selected ones).

(3) You can use any number of fields in the template, and it's no problem to use the same field more than once with different format, as in "... {size kb} ({size bb}) ..."

(4) The Report() function allows you to create full blown HTML-reports of any parts of your file system or of search results! You may even show icons for file types if you provide the image files and give them names that match the extension, e.g. type-txt.gif, type-png.gif, etc. which then would be referred to like "C:\Your\Path\type-{ext}.gif".

(5) In cooperation with function WriteFile() you are able to easily write the created reports to your hard disk.

resolvepath()

Converts a relative, special, or portable path to an absolute path.

Syntax

```
resolvepath([path], [base=<xypath>], [flags])
```

path	Relative, special, or portable path. If empty or missing the function returns *base*.
base	Base path; defaults to <xypath>.
flags	(bit field)

0: Normal function.
1: Reversed function: path is absolute, return path is relative to *base*.
2: The default for "base" is <curpath> (else it's <xypath>).

return — Absolute or relative path (depending on *flags*).

Notes

- The path does not have to exist.
- A trailing backslash in the input is kept in the output.
- A missing trailing backslash in the input is also missing in the output.
- You can pass a fully resolved absolute path. It will be returned unchanged.
- You don't need to provide a trailing backslash for the base (but it does no harm).
- The wildcard * is supported in the path parameter. The first matching item is used according to NTFS item order.

Examples

Assuming <xypath> = C:\Program Files\XYplorer; <curpath> = E:\Test:

```
echo resolvepath();       //C:\Program Files\XYplorer      = <xypath>
echo resolvepath(2:=2);   //E:\Test                        = <curpath>
echo resolvepath("foo");  //C:\Program Files\XYplorer\foo
echo resolvepath("foo\"); //C:\Program Files\XYplorer\foo\
echo resolvepath("\foo"); //C:\foo
echo resolvepath("\foo\");//C:\foo\
echo resolvepath("\");    //C:\
echo resolvepath("..");   //C:\Program Files
echo resolvepath("?:");   //C:
echo resolvepath("?:\");  //C:\
echo resolvepath("?:\foo\bar\");  //C:\foo\bar\

echo resolvepath("?:", "D:\foo\bar\");  //D:
echo resolvepath("foo", "D:\foo\bar\"); //D:\foo\bar\foo
echo resolvepath("", "D:\foo\bar\");    //D:\foo\bar\

echo resolvepath("Desktop"); //C:\Users\Donald\Desktop
echo resolvepath("Downloads\NotThere\"); //C:\Users\Donald\Downloads\NotThere\
```

Note the single quotes. These variables will arrive at resolvepath() as is, and then be resolved:

```
echo resolvepath('%tmp%');  //C:\Users\Donald\AppData\Local\Temp
echo resolvepath('<xydata>'); //C:\Users\Donald\AppData\Roaming\XYplorer
```

Absolute to relative:

```
echo resolvepath("C:\Apps\XY\XYplorer.exe", "C:\Apps\", 1);  //XY\XYplorer.exe
```

Using wildcard *: In a typical installation, the following two lines return "C:\Program Files\XYplorer\License.txt":

```
text resolvepath("<xypath>\*.txt");
text resolvepath("*.txt");  //resolvepath defaults to <xypath>
```

return

A) Statement used in a User-Defined Function: Sets the return value for the function, and exits the function.

Syntax

```
return data
```

data	Data to return.

Example

Simple function to return the sum of two numbers:

```
function sum($x, $y) {
 return $x + $y;
 // lines below return are not executed
 echo "Nobody will read this.";
}
```

B) Statement used in a Custom Column script: Defines the cell data and ends the script.

Syntax

```
return data
```

data	Data to show in the cell.

Example

Displays the first 12 characters of the content (!) of each file:

```
return readfile(<cc_item> , , 12);
```

rotate

Rotates JPG images. Rotation is lossless for typical digital camera images.

Syntax

```
rotate [mode=90|180|270|h|v], [jpgfile_src], [jpgfile_trg], [only_if_lossless=0|1], [flags]
```

mode	rotation in degrees (clockwise), or horizontal/vertical flip orientation; defaults to 90
jpgfile_src	JPG file to rotate; defaults to current file if empty; relative to current path if path missing
jpgfile_trg	target file name; defaults to jpgfile_src if empty; relative to current path if path missing

only_if_lossless

0:	[default] rotate always
1:	error if rotation cannot be done lossless

flags

0:	[default] no refresh
1:	refresh list, preview, and thumbnails.

Remarks

(1) ONLY if the width and height of the image are both multiples of 16, then indeed the rotation is lossless! Luckily, digital camera image dimensions are usually multiples of 16. MS has buried some deeper info on this here:
https://learn.microsoft.com/en-us/windows/win32/gdiplus/-gdiplus-transforming-a-jpeg-image-without-loss-of-information-use
If the image dimensions are not multiples of 16 then colors and sharpness will slightly fade if you do many consecutive rotations on the same image.

(2) And yes, the function will ONLY work on JPG images.

(3) EXIF data are preserved.

(4) Thumbnails are refreshed only when in thumbnails mode.

Examples

```
rotate;     //rotate current image by 90 degrees (clockwise)
rotate 180; //rotate current image by 180 degrees (clockwise)
//rotate "E:\my.jpg" by 90, but only if it can be done lossless
  rotate , "E:\my.jpg", , 1;
//flip current image horiz., save to "flipped.jpg" in cur path:
  rotate h, , "flipped.jpg";
//save vertically flipped "D:\pic.jpg" to "E:\pic-v.jpg":
  rotate v, "D:\pic.jpg", "E:\pic-v.jpg";
```

Example for POM

Add the following item to your Custom File Associations (menu Tools), and you'll have a command to rotate the image 90° clockwise on the Portable Openwith Menu (POM) of each JPG file:

```
|"Rotate 90° clockwise" jpg>::rotate
```

round()

Rounds a number.

Syntax

```
round(number, [precision=0])
```

number	Number to round.
precision	Number of digits after the decimal point. precision can also be negative or zero (default). A negative precision refers to digits before the decimal point.

Examples

```
echo round(172.7368, -3); //0
echo round(172.7368, -2); //200
echo round(172.7368, -1); //170
echo round(172.7368, 0);  //173
echo round(172.7368, 1);  //172.7
echo round(172.7368, 2);  //172.74
echo round(172.7368, 3);  //172.737
echo round(172.7368, 4);  //172.7368
```

rtfm

Opens a specific page in the Help file XYplorer.chm.

Syntax

```
rtfm page
```

page	Name of the HTML page, optionally with #anchorname.

Examples

```
rtfm "idh_find.htm";
RTFM "idh_find.htm#idh_findtabtags";
```

run

Works (almost) identical to Windows Start | Run, i.e. you can start executables, open documents with the OS-associated applications, open websites, and much more.

Syntax

```
run command, [directory], [wait=0], [show=1]
```

command: Anything that works in Windows Start | Run, usually a module name plus (optionally) arguments. Note that module names containing blanks must be quoted to correctly separate name and any arguments.

directory: The working directory, allows you to skip the path in command; defaults to current path.

wait:

0: [Default] Return immediately and continue the script.
1: Wait, with message: Only continue when the shelled process has finished, show message box "Please wait..." while the process is going on.
2: Wait, no message: Same as above but without message box.

show:

0: Hides the window.
1: [Default] Activates and displays the window.
2: Activates the window and displays it as a minimized window.
3: Activates the window and displays it as a maximized window.

The differences to the original Windows Start | Run are:

(1) Different error messages in case of failure.

(2) Quoting items with blanks slightly differs:

```
run '"C:\With Blank.txt"';
```

Names with blanks must be quoted. Note that the quotes have to be wrapped in single-quotes because an argument's outer quotes are auto-stripped by XYplorer's script parser!

```
run '"C:\Program Files\Winzip32.exe" -min';
```

Also wrapped in single-quotes because otherwise the "-" in "-min" would be interpreted as a math operator.

Tip: See also runq for an alternative way to handle quoting.

(3) Using "run" with wait=1 differs even more to the Windows Run command. Not all sorts of commands work, there is less smartness built in. For example, you cannot call documents that will be opened by the associated application. This will NOT work:

```
run "C:\Program Files\XYplorer\Readme.txt", , 1; //NO JOY
```

You also cannot use short forms like this:

```
run "winzip", , 1; //NO JOY
```

However this works (maybe because it's in the path):

```
run "notepad", , 1; //OKAY
```

See more examples below under Using the wait parameter.

Examples

```
run "calc";
```

Registered apps are recognized via "Prog ID" which is usually the base name of the *.exe file.

```
run "winzip32 -min";
```

No blanks in the filename: No quotes necessary...

```
run '"winzip32" -min';
```

... but quotes are allowed.

```
run "www.xyplorer.com";
```

Open an URL.

```
run "control";
```

Show Control Panel.

```
run "charmap";
```

Show character map utility.

```
run "regedit";
```

Open registry editor.

```
run "winver";
```

Display the Windows verson installed on the computer.

```
run "mailto:support@xyplorer.com?subject=Wow!&body=Hi!";
```

Open email client with some fields prefilled.

```
run "rundll32.exe shell32.dll,Control_RunDLL timedate.cpl";
```

Show Windows Time/Date dialog.

```
run "rundll32.exe shell32.dll,OpenAs_RunDLL <curitem>";
```

Open the Windows Open With dialog for the current item. Note that in this case <curitem> has not to be quoted even if it contains blanks; rundll32.exe is smart enough to handle it.

```
run """C:\Program Files (x86)\WinMerge\WinMergeU.exe"" /e /r ""<get path 1>"" ""<get path 2>""";
```

Run WinMerge (a popular comparison tool) in order to compare the current panes.

Fast Folder Delete using RMDIR

This will quickly remove the currently selected folder (selected in the file list pane!) and ALL of its contents, WITHOUT UNDO, and NO QUESTIONS ASKED:

```
run lax("cmd" /c rmdir /s /q "<curitem>");
```

If you want a Y/N prompt and see any return message use this way:

```
run lax("cmd" /k rmdir /s "<curitem>");
```

WARNING: Be careful with RMDIR:

- It fails for pathnames longer than 256 characters.
- You may need to "Run As Administrator" to get this to work, depending on the drive.

- If you have symlinks to directories in the folder then it will delete the actual directories and not symlinks!

Emulating Shell context menu clicks using run

User Buttons (see Custom Toolbar Buttons) allow you to get the action done quicker and with less clicks than by using the often slow Shell context menu. Here are some useful mini scripts (both use the lax() function for easier definition of the quotes):

For NIS to scan a selected folder:

```
run lax("C:\Program Files (x86)\Norton Internet Security\Engine\17.0.0.136\navw32.exe" "<curpath>\*");
```

For WinRAR to extract from multiple consecutive .RAR files by selecting only the first file:

```
run lax("C:\Program Files (x86)\WinRAR\WinRAR.exe" x "<curitem>");
```

The argument x in this example is WinRAR's switch to extract to the current folder and preserve the folder structure within the compressed file.

Using the wait parameter

If wait=1 there's a modal dialog while the shelled process is being active. It blocks you from doing anything but wait or cancel the waiting and continue with the script.

```
run '"C:\Program Files (x86)\WinZip\WINZIP32.EXE" -min', , 1; msg "Done!";
```

Shows "Done!" when WinZip is closed.

```
run "notepad C:\Program Files\XYplorer\Readme.txt", , 1; msg "Done!";
```

Shows "Done!" when notepad is closed.

Using the show parameter

Note that it depends on the called application whether it honors the show parameter. Some ignore it.

```
run "notepad <xypath>\startup.ini", , , 0;  //hidden
run "notepad <xypath>\startup.ini", , , 1;  //normal
run "notepad <xypath>\startup.ini", , , 2;  //minimized
run "notepad <xypath>\startup.ini", , , 3;  //maximized
```

runq

Syntax and behavior is identical to command run with one exception: runq puts the "command" parameter in quotes before it's passed to the system. This can greatly simplify your syntax depending on the context.

Examples

```
run "E:\Test\Has Space.txt";         //fails (because of the space in the file path)
```

```
run quote("E:\Test\Has Space.txt");  //works
runq "E:\Test\Has Space.txt";        //works
runq quote("E:\Test\Has Space.txt"); //fails (because it will be quoted two times)
```

runret()

Runs a command line and returns the standard output.

Syntax

```
runret(command, [directory], [codepage], [flags])
```

command	Anything that works in Windows Start | Run, usually a module name plus (optionally) arguments. Note that module names containing blanks must be quoted to correctly separate name and any arguments.
directory	The working directory, allows you to skip the path in command; defaults to current path.
codepage	Expected codepage of the console. Will be converted to Unicode in the return value. If missing or zero the currently active OEM codepage is assumed.
flags	(bit field) 1: Return a hex dump of the raw data in the pipe. Useful mainly for debugging. 2: Return the exit code of the process. Useful mainly for debugging.
return	The standard output.

Examples

```
text runret("cmd /c dir c:\");
text runret("ping localhost");
```

List the current directory with full Unicode support going via the UTF8 codepage (65001):

```
text runret("cmd /c chcp 65001 & cmd /c dir", , 65001);
```

savesettings

Saves all or part of the current settings.

Syntax

```
savesettings [settings=all], [ini]
```

settings	(bit field) 0 = (none) 1 = XYplorer.ini and pane data 2 = catalog.dat 4 = udc.dat 8 = ks.dat

16 = servers.ini (cached servers)
32 = fvs.dat
64 = tag.dat
128 = action.dat (action log, undo/redo data)
256 = pv.dat (permanent variables)
512 = Language.ini (holds name of current language file)
1023 = [Default] all above settings

ini — Alternate ini file.
Path defaults to <xydata>.
If this argument is set then the "settings" argument is totally ignored and only this INI file is written.
NOTE: Any same-named existing file is overwritten without questions!

Remark

The command is equivalent to menu File | Save Settings and saves the current *.ini (unless ini parameter is set) and *.dat files, including the pane data.

Examples

```
savesettings; //saves all settings
savesettings 3; //saves ini and catalog
savesettings , "<curpath>\test.ini";
savesettings , "%TEMP%\dummy.ini";
savesettings , "trashme.ini";
```

savethumb()

Saves the thumbnail of a file to a file.

Syntax

```
savethumb([file=<curitem>], [thumbnail_file="*_thumb"], [widthbox], [heightbox], [format="jpg"], [border_width], [flags], [transparency=2], [color_canvas])
```

file — [optional] The name of the source file.
Defaults to the current list item.
Can be any format that has a thumbnail image (including video files).
Can be a list of files separated by CRLF; for this to work, the thumbnail_file MUST contain a * character (which will be replaced with the base name of the source image file).
Set to <clipimg> to use the current image in the clipboard.

thumbnail_file — [optional] The name of the target file.
Defaults to "*_thumb" (see remarks).
The path is resolved relative to the path of >file<.
The extension is set to >format< if missing.

widthbox	[optional] The width of the thumbnail's bounding box. Defaults to 500 if widthbox AND heightbox are both missing. The argument can also be used to pass a percentage by which the thumbnail is shrunk from the original, e.g. `50%`. See Remarks.
heightbox	[optional] The height of the thumbnail's bounding box. Defaults to 500 if widthbox AND heightbox are both missing.
format	[optional] Format of the thumbnail file. Can be one of the following: jpg, png, gif, bmp, tif Defaults to "jpg". In case of "jpg", the desired JPG quality value (1-100) can optionally be appended to the format tag, e.g. "jpg75".
border_width	[optional] Absolute or relative size of a white border around the thumbnail. Append % to mean percentage. The percentage applies to the mean image dimension of the thumbnail: (width + height) / 2.
flags	[optional] (bit field) 1 = Drop Shadow on white border (border_width has to be set). 2 = Zoom To Fill: Zoom the thumb to fill the whole box. If heightbox is missing it defaults to widthbox, and vice versa (= square box). 4 = Arguments "widthbox" and "heightbox" are taken as size of a canvas in which the input image ("file") is centered. If necessary the image is shrunk to fit the canvas. The resulting image will have these dimensions no matter the size of the input image.
transparency	[optional] -1 = Preserve any transparency Note that with this option, border_width as well as the flags 1 (Drop Shadow) and 2 (Zoom To Fill) are ignored. So it's just scaling up or down. 0 = Neutral (here: the current window backcolor, e.g. of the tree, dark mode aware) 1 = Grid 2 = White [Default] 3 = Black 4 = The color stated in color_canvas
color_canvas	[optional] Canvas background color in hex RRGGBB. If missing it defaults to the current window backcolor (dark mode aware).
return	The full path/name of the saved thumbnail file.

Remarks

- The thumbnail_file argument supports a couple of inline variables:

```
<width>     = actual thumbnail width (without any borders)
<height>    = actual thumbnail height (without any borders)
<widthall>  = actual thumbnail width with borders
<heightall> = actual thumbnail height with borders
<widthbox>  = bounding box width, or canvas width
<heightbox> = bounding box height, or canvas height
```

```
<percent>   = applied percentage
<quality>   = JPG quality
*           = base name of the source image file
```

- You can pass only one dimension, widthbox OR heightbox, and the other dimension is calculated proportionally so that the thumbnails fully fills the box. Only when both arguments are missing width and height default to 500 pixels each.
- The widthbox argument can also be used to pass a percentage by which the thumbnail is shrunk from the original. Simply attach "%" to the value. In that case the heightbox argument is ignored. Note that percentages > 100 are supported, so this is a way to enlarge images (which, of course, dramatically reduces their quality).
- border_width: The border is added to the box after the thumbnail is generated, so a 500 pixel wide thumbnail with a 10 pixel border will create a 520 pixel wide image file.
- If the original is smaller than the bounding box it is NOT stretched.
- If a file of the same name exists it is overwritten without asking.
- If no thumbnail can be created from "file" the function raises an error.
- Works for all files that show a thumbnail in thumbnails view, not only images but also e.g. videos or PDF files.

Examples

```
savethumb(); //creates autonamed JPG within bounding box 500x500
savethumb(, "*-<width>x<height>", 400, 400, "jpg");
savethumb("E:\Test.pdf", , 400, 500, "png"); //creates E:\Test_thumb.png from E:\Test.pdf
savethumb(, "*_t", 550);     //max width 550, height is automatic
savethumb(, "*_t", , 100);   //max height 100, width is automatic
savethumb(, "*_t");          //max width 500, max height 500
savethumb(, , "50%");  //half size
savethumb(, "*-<percent>%", "200%", , "png");  //double size
savethumb(, , "50%", , , "7%");  //half size, 7% border
savethumb(, "*-<width>x<height>", 400, 300, "png" , 10); //10 pixel border
savethumb(, "*-<widthall>x<heightall>", 550, , , 10); //10 pixel border, overall dimensions in the filename
savethumb(, "folder", 300, 200, "jpg", , 2); // Zoom To Fill, 300x200 JPG
savethumb(, "*_slice", 100, 300, "jpg", "7%" , 2); // Zoom To Fill, 7% white border, 100x300 JPG (yep, vertical slice)
savethumb(, "*-<width>x<height>", 300, , "jpg", "7%", 3); // Zoom To Fill + Drop Shadow, 7% white border, 300x300 square JPG
savethumb(, "*-Q<quality>", 300, , "jpg100"); // 300 pixel wide JPG, quality 100
```

Examples for transparency (on focused list item, also works when it's an ICO file)

```
savethumb(, "*_t", 256, , "png", 7:=-1);    //preserve transparency
```

```
savethumb(, "*_t", 256, , "png", 0, 0, 1);  //grid
savethumb(, "*_t", 256, , "png");           //white
savethumb(, "*_t", 256, , "png", 0, 0, 3);  //black
```

Examples for using canvas

Save selected image centered in white 1080x1080 PNG, transparent areas also white:

```
savethumb(, "*_<widthbox>x<heightbox>", 1080, 1080, "png", , 4, 4, "FFFFFF");
```

Save selected image centered in 1080x566 PNG, background and transparent areas in window backcolor (dark mode aware):

```
savethumb(, "*_<widthbox>x<heightbox>", 1080, 566 , "png", , 4, 0);
```

Save a copy of the image in original size with transparent areas in magenta (FF00FF):

```
savethumb(, "*_<width>x<height>", "100%" , , "png", , 0, 4 , "FF00FF");
```

Examples for saving images from the clipboard

```
savethumb(<clipimg>, "clp_<date yyyymmdd_hhnnss>", "100%", , "jpg"); //save clipboard image as JPG, full size
savethumb(<clipimg>, "clp_<date yyyymmdd_hhnnss>", "50%", , "png"); //save clipboard image as PNG, half size
```

Examples for processing multiple images in one go

```
savethumb(<clp>, "*_<width>", 500, , "png"); //when clipboard has a list of files
savethumb(<drop>, "*_<width>", 500, , "png"); //for drop-on bookmark
savethumb(<selitems <crlf>>, "*_<width>", 500, , "png"); //all selected list items
```

searchtemplate()

Saves or loads Find Files settings to/from a search template.

Syntax

```
searchtemplate(name, [mode="load"], [flags=0])
```

name	Template name - All characters allowed - Auto-converted to a valid filename in this location: <xydata>\FindTemplates\<name>.ini OR: Full path/file name in any location you want.
mode	load: [Default] Load template. See flags below. save: Save current Find Files settings to template. Use flags 1 to also save the current search results (if any).
flags	(bit field) 1: Cache results (on load this flag logically implies flags 2+4+8).

2: Use stored location (on load only).
4: Use exclude folders (on load only).
8: Run search at once (on load only).

Remarks

Any existing template of the same name is overwritten without questions.

Examples

Save current search to "test":

```
searchtemplate("test", "save");
```

Save current search to "test", store the search results as well:

```
searchtemplate("test", "save", 1);
```

Load search in original location, use exclude folders, run at once:

```
searchtemplate("test", , 14);
```

Load cached results (logically implies: use original location, use exclude folders, run at once):

```
searchtemplate("test", , 1);
```

Load the stored Find Files settings without running any search:

```
searchtemplate("test");
```

sel

Select and focus one item in the file list by position or pattern matching. Optionally select any number of the items following.

Syntax

```
sel [position/pattern], [count], [startfromtop]
```

position

(number): Define the new line to be focused and selected. Either an absolute line number, or a relative (to the currently focused) one when prefixed with + or -. Note that the argument must be quoted when + is prefixed! See examples below.

[pattern]: When this parameter is in between square brackets ([ ]), then the item to be focused and selected will be the first one matching the given pattern, starting on current position.

a: Select all.

i: Invert selection.

f: Select all files (no folders).

[empty]: Deselect all.

count

The number of items to get selected, starting with the one on position (that just got focused as well). Ignored when the first parameter is empty, or set to a, i or f. Defaults to 1.

startfromtop

0: [Default] start from current position

1: start from top of list
startfromtop is ignored if the first argument is not a pattern.

Examples

```
sel 1    // Select first item in list
sel 2    // Select second item in list
sel "+5"   // Select item 5 positions after current; note the quotes (else +5 evaluates to 5)!
sel -1   // Select item before current
sel +, 3 // Select 3 items starting with the currently focused one.
```

Use the command sortby to have more control about the results of sel.

```
sortby size, d; sel 1, 5;
```

Sorts the List by Size in descending order, and selects the five first items. Focus will be on the first item on List.

```
sel f; sel i;
```

Selects all files (no folders), then inverts the selection = Selects all folders. Note that no items will be selected if there are no folders in the current location, or if they are not shown due to options (like Show Folders in List being disabled) or if a Visual Filters have them hidden.

```
sortby "Name", a; sel "[readme.*]";
```

Sorts the List by Name, ascending, then selects the first file whose name is "readme", regardless of its extension.

```
sortby "Name", a; sel 1,0; sel "[[ab]*.txt]";
```

Sorts the List by Name, ascending.

Puts focus on the first item.

Selects the first text file (extension .TXT) whose name begins with either a "A" or a "B".

```
sel "[Test.txt]", , 1;
```

Selects "Test.txt" in the file list (also if it is above the current focus position). On no match any current

selections are unselected.

```
sel -1, , 1; //select the last item in the list
sel -2, , 1; //select the pre-last item in the list
```

You can identify items from the end of the list by using negative positions and setting the startfromtop argument.

selectitems

Selects items in the file list by a list of names.

Syntax

```
selectitems itemlist, [flags], [focus=1], [mode], [pane]
```

itemlist — [required] CRLF- or |-separated list of item names; the names can be title-only or full path.
Wildcards * and ? are allowed.

flags — (bit field)
0: [Default] Mind extensions when matching the names.
1: Ignore extensions.
2: Match full paths.

focus — (bit field)
0: Keep the focus where it is.
1: [Default] Auto-move the focus to the first selected file.
2: Push the focused and selected item (see Notes / Push below).

mode
n: [Default] New selection (drop current selections).
a: Add the matches to any current selections.
r: Remove the matches from any current selections.
t: Toggle the selection state of the matches.

pane
a: [default] active pane
i: inactive pane
1: pane 1
2: pane 2

Notes

- The matching is only done against the file titles; when paths are passed they are ignored.
- The matching is case-insensitive (A=a).
- If itemlist is empty all items are unselected.
- Push here means that the focused and selected item is treated as if it had been actively clicked on by the user. It becomes the "current item" internally, which means the Info Panel is filled with its

properties, and the Preview is triggered. Push only works on items that are focused *and* selected.

Remark

Using this command you can easily save and (later) restore a selection:

```
$selections = get("selecteditemsnames", "|");  //store
selectitems $selections; //restore
```

Examples

```
selectitems "Fiji.txt|Tahiti.txt";  //selects 2 islands
selectitems "Tahiti.txt", 1; //also selects Tahiti.wav (if exists in current list)
selectitems "<clipboard>", 2; //Match full paths from a list on clipboard (one item per line)
selectitems "C:\Fiji.txt"; //selects Fiji.txt in current list (no matter which folder)
selectitems "Canary.txt|Fiji.txt", , , "a"; //add to current selections
selectitems "test.txt", 3:=t; //toggle selection state
selectitems "<clipboard>", 3:=t; //toggle selection state
selectitems "Test.jpg", , 3;  //select, focus, push
selectitems "*.png";    //select all PNG files
selectitems "*.png|4*"; //select all PNG files and files beginning with "4"
```

selectthumbs()

Selects items according to the properties of their thumbnails.

Syntax

```
selectthumbs([mode=1])
```

mode	0: Select all items without thumbnails. 1: [default] Select all items with good thumbnails. 2: Select all items with bad thumbnails. ("Can't decode" etc). 3: Select all items with thumbnails.
return	Number of selected items. -1 if not in thumbnails mode.

Remark

If the list is not in thumbnails mode it returns -1, and the current selections are not changed.

Examples

```
selectthumbs(1); //select all good thumbnails
selectthumbs(2); //select all bad thumbnails
selectthumbs(3); //select all thumbnails
echo selectthumbs(3); //select all thumbnails; show number of selections
```

self()

Get info about the currently running script or script file.

Syntax

```
self(info, [level])
```

info	Any of the following: file: current script file (empty if none) path: current script file's path (app data if none) base: current script file's base (empty if none) script: current script caption: caption of the current script icon: icon resource of the current script label: label of the current script level: level of the current script (1 = top level)
level	0: self [Default] 1: 1st level (top level) 2: 2nd level, etc -1: caller of self (one level up) -2: caller of caller of self (two levels up), etc
return	Info.

Remarks

If the desired level does not exist the function returns nothing (no error message).

Examples

```
msg self("path");
```

```
msg self("script");
```

Of course, the info types file, path, and base only make sense when the script is running from script file.

Example of using the Level parameter

```
// test SC self
   echo self("level");  //self level (1 = top level)
   // go down to a user function
   godowntowork1();

function godowntowork1() {
   echo self("level");  //self level (1 = top level)
   godowntowork2();
}
function godowntowork2() {
```

```
   echo self("level");  //self level (1 = top level)
   echo self("script");  //self
   echo self("script", -1); //caller (one level up)
   echo self("script", -2); //caller of caller (two levels up)
   echo self("script", 1);  //1st level
   echo self("script", 2);  //2nd level
   echo self("script", 3);  //3rd level
   echo self("script", 4);  //4th level -- returns nothing here
}
```

Using the Caption of a Script: self("caption")

A script's caption can be part of the script itself (typically in a multi-script resource) but also be stated in its container, e.g. a Catalog Item or a User-Defined Command. If the current script has no own caption then self("caption") returns the caption of its parent script (the script that called it, if any), or the parent of the parent etc. until a caption is found. The latter makes it possible to load script files in a Catalog Item via load() and still refer to the Catalog Item's caption by self("caption").

Here's an example for a Catalog-based script that tags the dropped items taking the actual tags (cats, dogs) from the caption of the script:

Caption:

```
Tag dropped items: cats, dogs
```

Script:

```
// Caption example = Set tag to dropped items: cats, dogs
  $tags = gettoken(self("caption"), 2, ":", "t");
  tag $tags, <get drop |>, 1;
```

Items dropped on this Catalog Item are tagged with "cats" and "dogs". Simply change the caption of the Catalog Item (e.g. to `Tag dropped items: birds`) to change the functionality of the drop event.

selfilter

Selects list items using a pattern.

Syntax

`selfilter [pattern], [type (f|d)], [column="Name"], [mode], [flags]`

pattern	The pattern to filter which items to select. If empty or missing then all items will be selected. If column is "#" then `[pattern]` is `"[start=1],[step=1]"` (see examples below).
type	The type of items to get selected [Default] Files and folders f: Files only d: Folders (Directories) only

column	Column name. Defines the column which the pattern will be treated against. Defaults to "Name". Tip: A partial match is used as a fall back if there is no full match for the "column" parameter, so to match the column "Modified" you could use either "m", "Mod", or "modified", for example. #: Select items by their line number in the current listing (see examples below).
mode	0,n: [Default] New selection (drop current selections). 1,a: Add the matches to any current selections. 2,r: Remove the matches from any current selections. 3,t: Toggle the selection state of the matches.
flags	(bit field) 1: Focus the first selected item and scroll it into view.

Examples

```
selfilter "readme";
```

Selects all items whose name contains "readme" (eg: readme, readme.txt, !readme.now).

```
selfilter """readme""";
```

Selects item named "readme" and only that one item.

```
selfilter h, , "attr";
```

Selects all hidden items (if hidden items are shown on List, obviously).

```
selfilter "2008", f, "mod";
```

Selects all files on List modified in 2008.

Note that this will only work if the Modified column display years in 4-digits format.

```
selfilter "###,## MB", , "size";
```

Selects all items of at least 100,00 MB. Folders will also be selected IF their sizes are calculated and shown on column Size.

Note that this will only work if the Size column uses format "MB".

```
selfilter "*.txt"; selfilter "a*", , , 1;
```

First selects all TXT files, and then adds all items starting with "a" to the selection.

```
selfilter "*.", f;
```

Selects all files without extension.

```
selfilter "*.jpg", 4:=1;
```

Focus the first selected item and scroll it into view.

Examples for remove and toggle

```
selfilter "*.jpg", 3:=r;  //remove all JPG files from the current selection
selfilter "*.jpg", 3:=2;  //same
selfilter "o*", 3:=t;     //toggle the selection state of all items starting with "o"
selfilter "o*", 3:=3;     //same
```

Examples for column=#

```
selfilter "3", , "#";   //select all from position 3
selfilter "3,", , "#";  //(same as above)
selfilter "3,1", , "#"; //(same as above)
selfilter "3,2", , "#"; //select every 2nd from position 3
selfilter ",4", , "#";  //select every 4th from position 1
selfilter "", , "#";    //select all from position 1
selfilter "0,0", , "#"; //(same as above)
```

seltab

Selects a tab by position.

Syntax

```
seltab [position], [switches]
```

position:	The position of the tab to be selected (opened, activated):
[empty]	Activates the default tab (if any).
(position)	The position of the tab to select, counting from left (as shown on Tab list - menu Window | Tabs). The first tab is number 1. A negative position points to tabs from the right end (-1 points to the right-most tab). 0 points to the current tab.
+ (plus)	Selects the next tab.
- (minus)	Selects the previous tab.
switches:	If position is relative ("+" or "-") then switches controls which tab is selected next.
0	[Default] Left-right order.
1	Last used order.
i	Position is tab ID.

Remarks

The tab ID always remains with a tab, no matter where it is moved. To find out the ID of a tab, hold down the SHIFT key while hovering over the tab header; the ID is displayed in the tooltip.

Examples

```
seltab 3;    //select the third tab (from the left)
seltab 3, i; //select the tab with ID 3
seltab +; //select the next tab.
seltab; //select the default tab.
seltab "+", 1; //select the next tab, using MRU.
seltab "-", 1; //select the previous tab, using MRU.
```

```
seltab -1; //select the right-most tab.
seltab 0; //select the current tab (reverts to the tab's saved settings).
```

set

Sets a variable to a defined value.

Syntax

```
set output, value, [reprocess]
```

output: The output variable.

value: The value to set the variable to.

reprocess: [optional]
r = Post-process the value set in the variable to resolve any contained script variables (`$a`), and environmental variables (`%tmp%`), and native XYplorer variables (`<clipboard>`).

Usage Tip

See Using the equal-operator (=) below for a more common way to set variables to a value.

To unset a variable, see command unset.

The reprocess argument

Compare the following two examples to see the difference the reprocess "r" parameter makes.

```
copytext '%tmp%'; set $msg, <clipboard>; msg $msg;
```

Copy to the clipboard the string %tmp% (in single quotes: not resolved!).

Put the content of the clipboard (%tmp%) into $msg.

Show the content of $msg (%tmp%).

```
copytext '%tmp%'; set $msg, <clipboard>, "r"; msg $msg;
```

Copy to the clipboard the string %tmp% (in single quotes: not resolved!).

Process the content of the clipboard (%tmp%) to resolve any variable, and put the results (e.g. C:\TMP) into $msg.

Show the content of $msg (e.g. C:\TMP).

Using the equal-operator (=)

There's a common alternative for the set command. To assign a value to a variable you can simply use the following syntax:

`output = value`, e.g. `$a = "b";`

The optional reprocess argument is also supported, e.g.:

```
::copytext '%tmp%'; $a = <clipboard>, "r"; msg $a;
```

Displays "C:\Temp" (or whatever it is).

Adding strings (concatenation)

Set (or the equal-operator) can also be used to add strings (or variables):

`set $a, "1"."2";` OR `$a = "1"."2";`

Set $a to "12".

`set $a, "tr"; set $b, "ee"; set $c, $a.$b; msg $c;` OR

`$a = "tr"; $b = "ee"; $c = $a.$b; msg $c;`

Display "tree".

Examples

`set $year, "2008"; msg $year;`

Set $year to "2008". Then show it in message box.

`set $year, <date yyyy>; msg $year;`

Set $year to the current year. Then show it in message box.

`set $path, <curpath>; msg $path`

Set $path to the current path. Then show it in message box.

setcolumns()

Sets or gets the list's columns.

Syntax

`setcolumns([columns], [flags=0], [position])`

columns	Comma-separated list of canonic column names. May optionally include visibility prefix "+" as '+Name'. May optionally include the column width (in pixels) as 'Name.Width'. If missing only the current columns are returned. If "" then the current columns are ordered in the default column sequence. See Extended Syntax for "Soft Columns" below.
flags	Options for set and get. 0: Only visible columns. Prefixes and Widths are ignored (but permitted), and not returned. 1: Mind/Return visibility of the columns with prefixed "+". 2: Mind/Return column widths, suffixed to the names separated by a period. 4: Set only the widths, not visibility or position. 8: Toggle column(s): Append to/remove from current columns. 16: Autosize columns. 32: Toggle column visibility. Only one column at a time, identified by its case-insensitive caption (A==a), or by its canonic name (e.g. "Extra 4"). Can only be combined with the Autosize columns flag (16). 64: Return captions for Custom Columns, instead of extended syntax identifiers (see below).

position	Position where the column(s) should be placed. 0: (default) append the column(s) 1: first position 2: second position -1: pre-last position
return	The previous columns (as comma-separated list of canonic column names). Depending on the flags with visibility prefixes and widths suffixes.

Examples

```
setcolumns("Name,Size"); //shows Name and Size
```

```
setcolumns("Name,+Size", 1); //shows only Size
```

Set exact pixel widths; where the width is missing or 0 the current width of the column is used:

```
setcolumns("Name.223,Ext.43,Size.48,Created.115,Modified.115,Accessed.115,Label.18,Tags.53,
Comment.82", 2);
```

```
setcolumns("Name.250", 4); //set Name column width to 250 pixels
```

```
setcolumns("Name.250,Size.100", 4); //set the width of two columns
```

```
setcolumns(":v-dimensions", 8); //toggle Dimensions column
```

Autosize columns:

```
setcolumns(":s-hash.sha1", 24); // toggle SHA1 column and ensure it's fully visible
```

```
setcolumns(, 16); // just autosize the columns
```

Toggle column visibility:

```
setcolumns("path", 32); //toggle the Path column
```

```
setcolumns("Mixed", 32); //toggle the Mixed column
```

```
setcolumns("Custom 8", 32); //toggle the Custom 8 column
```

Position:

```
setcolumns(":v-dimensions", 8, 1); //toggle Dimensions column, first position
```

```
setcolumns(":v-dimensions", 8, 2); //toggle Dimensions column, second position
```

```
setcolumns(":v-dimensions", 8, -1); //toggle Dimensions column, pre-last position
```

Show current columns:

```
text setcolumns(); //just the captions
```

```
text setcolumns(, 3); //captions, visibility, width
```

Examples for flag 64

```
// might return ":d-12,Name,Ext,Size,Modified,Created,Tags"
```

```
text setcolumns(":d-12", 8, 2);
```

```
// might return "Last Name,Name,Ext,Size,Modified,Created,Tags"
```

```
text setcolumns(":d-12", 72, 2);    //64 + 8 = 72
```

Notes

You cannot hide all columns. If you try then the Name column will be shown.

Extended Syntax for "Soft Columns"

The comma-separated list of canonic column names also supports the extended possibilities that XYplorer also uses internally. You can directly create and show new custom columns, even with custom captions, by means of the "columns" argument.

These on-the-fly Custom Columns are called "Soft Columns" to distinguish them from the "hard" Custom Columns "Custom 1" to "Custom 10". There are 4 types of Soft Columns:

```
[caption]:n-10              = numeric shell property #10
[caption]:v-dimensions      = verbal shell property "dimensions"
[caption]:s-audio.bitrate   = special property "audio.bitrate"
[caption]:d-10              = definition of Custom Column 10
   ^ Optional caption; if missing then an automatic caption is coined depending on the contents of the column.
           ^ Prefix n, v, s, or d, separated with a "-" just for better readability. The prefix determines the type of the column.
             ^ Content definition, depending on the type.
                                 ^ What this column would display.
```

- Type v suffers from the problem that MS does not document anywhere which verbs actually exist. You have to guess what works and what doesn't.
- Type v supports the "asterisk syntax" for property names where localized column names can be used (see examples below).
- The last type (d) links to one of the 64 available Custom Column definitions. They either referenced by index (1-64) or by their caption.
- Important: The Soft Columns are not as persistent as the other columns. Whenever you set a new set of columns to a list, any soft columns are automatically discarded. Otherwise, the lists would just grow more and more columns (there is currently no special scripting command to remove a column). So, to remove one or more soft column you simply pass a list of columns without those columns.

Examples for "Soft Columns"

Columns for audio files:

```
setcolumns("Name,:s-audio.bitdepth,:s-audio.bitrate,:s-audio.channels,:s-audio.samplerate,:s-audio.length,:s-mp3.artist");
```

Columns for image files:

```
setcolumns("Name,:s-image.dimensions,:s-image.datetaken,:s-aspectratio");
```

Setting a custom caption, with width:

```
setcolumns("Name,The Man:v-owner.200", 2);
```

Toggle Dimensions column:

```
setcolumns(":v-dimensions", 8);
```

Asterisk syntax: Toggle Width and Height columns (English Windows):

```
setcolumns(":v-*width,:v-*height", 8);
```

Asterisk syntax: Toggle Width and Height columns (German Windows):

```
setcolumns(":v-*Breite,:v-*Höhe", 8);
```

This example toggles a soft column at position 2 assigned to the 12th custom column:

```
setcolumns(":d-12", 8, 2);
```

This example toggles a soft column at position 2 assigned to the first custom column named "View":

```
setcolumns(":d-view", 8, 2);
```

seticons()

Sets or gets the current Custom File Icons.

Syntax

```
seticons([definitions], [mode=a], [position=1], [separator=CRLF])
```

definitions	List of Custom File Icon patterns, separated by separator. Each may optionally start with a '+' to indicate they are enabled.
mode	a: [Default] Add Definitions (at position) s: Set Definitions (discards all current definitions!) r: Replace Definitions (from position; may go beyond the original bounds) d: Delete Definitions (identified by string compare)
position	Where to add or replace new definitions. Only used if mode is "a" (Add) or "r" (Replace). 1: [Default] Top of list. n: Before nth item. 0: Append to list (same for both modes: Add and Replace)
separator	Separator between definitions (also used in return value). Defaults to CRLF.
return	The current Custom File Icons (before any new ones are set).

Examples

```
seticons("+*.txt>kiss.ico");        //add kiss.ico to the top of CFI
seticons("+*.txt>kiss.ico", , 0);   //add kiss.ico to the end of CFI
seticons("+*.txt>kiss.ico", s); //set kiss.ico to TXT files
seticons("+*.txt>kiss.ico|+*.jpg>omelet.ico", s, , "|"); //set these two definitions
seticons("+*.txt>kiss.ico", r);     //replace the first definition
seticons("+*.txt>kiss.ico", r, 2);  //replace the second definition
seticons("+*.txt>kiss.ico", r, 0);  //add kiss.ico to the end of CFI (sic: same as add)
seticons("+*.txt>kiss.ico|+*.jpg>omelet.ico", d, , "|"); //remove these two definitions
```

setkey

Sets the value of a configuration key in the current INI file, or of any INI file.

Syntax

```
setkey value, key, section, [INIfile], [flags]
```

value	The value to be set to the key. Empty or missing: the key is removed (unless flag 1 is set).
key	The name of the key. Empty or missing: the section is removed.
section	The name of the section. Empty or missing: command does nothing.
INIfile	[optional] the name of the INI file. Expects a filename with extension, and can be absolute or relative to the running script's path. If the parameter is missing it defaults to XYplorer's current INI file. If the file does not exist yet, it will be created.
flags	(bit field) 0: Automatic quoting: quote value when necessary. - when the value is quoted - when the value has leading or trailing spaces 1: Write the value in quotes. 2: Don't write the value in quotes.

Usage

Note that setkey pairs with getkey. The getkey/setkey commands support reading/writing values of up to 32,766 characters.

Examples

Set key StartPath in section General to value "C:\" in the current INI file:

```
setkey "C:\", "StartPath", "General";
```

Note: Only the key on disk is changed, not the current setting in memory! To realize any changes made via setkey you should use "Restart without Saving" in menu File!

Set key Year in section Vacation to value "2009" (resolved date now, if now is year 2009) in the INI file "C:\holiday.ini".

```
setkey "<date yyyy>", "Year", "Vacation", "C:\holiday.ini";
```

The INI file, if it was freshly created, will look like this:

```
[Vacation]
Year=2009
```

Similar to above, but value quoted:

```
setkey "2012", "Year", "Vacation", "<curpath>\holiday.ini", 1;
```

```
[Vacation]
Year="2012"
```

setlayout()

Sets or gets the current layout.

Syntax

```
setlayout([layout], [separator=","])
```

layout	The layout definition, being a list of key/value pairs in the format "key1=value1, key2=value2,...". The key/value pairs can be listed in any order; the keys are not case-sensitive. If missing then only the current layout is returned.
separator	Separator between key/value pairs, defaults to comma.
return	The current layout (before any new layout is set).

Remark

The supported values are not listed here. They can easily be seen in any saved layout file (menu Window | Save Layout As...).

See also loadlayout.

Examples

```
text setlayout(); //show the current layout
text setlayout(, <crlf>); //show the current layout, line by line
setlayout("ShowNav=0,ShowInfoPanel=0"); //hide Navigation Panel and Info Panel
```

Tip: For Boolean values (0 or 1) you can pass "!" to mean "toggle":

```
setlayout("ShowNav=!,ShowInfoPanel=!");  //toggle Navigation Panel and Info Panel
setlayout("ShowMainMenu=!");  //toggle main menu
setlayout("DPSwapPanes=!");     //toggle Swap Panes
setlayout("DPSwapPanes=1");     //Swap Panes
setlayout("DPSwapPanes=0");     //Unswap Panes
```

setthumb

Sets a specific thumbnail for one or more currently listed items.

Syntax

```
setthumb [item], [thumb], [mapfile], [deep]
```

item	Item to set the thumbnail for. The path defaults to the current list path. This item has to be present in the current list. If omitted then the thumbnail is set for all currently selected items.

thumb — File to be used as thumbnail.
The path defaults to <xyicons>.
Set to "" to reset to original thumbnail.

mapfile — File with a list of "item > thumb" mappings.
The path defaults to <xydata>.

deep
0: Deep injection OFF.
1: Deep injection ON (see Remarks below).

Remarks

- The list has to be in a view with thumbnails for this command to work.
- The injected thumbnails will also be remembered in the cache (but only for this particular thumbnails size).
- The injected thumbnails will not survive a "Refresh Thumbnails".
- You can use this command on files (or folders) that you need a different thumbnail for, or files (or folders) that for some reason don't get a thumbnail at all.
- On deep injection the thumbnails injected via setthumb or thumbnails mapfile behave more like original thumbnails: They support Hover Box and MDBU (but both only over the thumbnails, not over the file icons), i.e. those features will show the injected image (if any), not the original image of the hovered/clicked file.
  Deep injection is remembered across sessions.
  Tip: You can invert the "deep" setting on-the-fly by holding SHIFT while pressing the mouse button down. That way you can quickly inspect the other thumbnail on items that have an original one and an injected one.

Examples

```
setthumb , "E:\Test\thumbs\stars.jpg"; //set it for all selected files
setthumb , "stars.jpg"; //it's in path <xyicons>
setthumb "E:\Test\Meskalin.exe", "\\Vega\shared\Test\Pics\BlowUp.jpg";
setthumb "Meskalin.exe", <clp>; //pull thumb from clipboard
setthumb , "";  //reset to original thumbnail
```

Remarks on the map file

- By setting a map file you can permanently patch the thumbnails of any items on your system. This includes setting thumbnails for items that otherwise would have none.
- You only have to set it once with SetThumb. It then sticks in the INI file until you change it to another map file or set it to "".
- Syntax: One mapping per line, generic format: item > thumb. "item" can be full path, a mere filename, or a wildcard pattern. The lines are processed from top to bottom. First match wins.
  Examples:
  `D:\Test\Java.psd > E:\Test\thumbs\injection\palmoil.jpg`

```
E:\Space\* > E:\Test\thumbs\stars.jpg
XY-HoverBoxKeyBoardTricks.txt > E:\XY\XYlogo.png
*.exe > E:\Test\thumbs\injection\motor.jpg
* > E:\Icons\item.ico
```

- Updating the map file does not automatically update the thumbnails cache nor the currently shown thumbnails. If you want to see an immediate effect you have to call #501 (View | Caches | Refresh Thumbnails):

```
setthumb 2:="XYthumbsMap.txt"; #501;  //set map file + refresh thumbnails
setthumb 2:=""; #501;                 //reset map file + refresh thumbnails
```

setting

Alter some of XYplorer's user settings, temporarily or permanently. Typically used in order to ensure the expected execution of a script, which might depend on certain settings.

Syntax

```
setting name, [value (1|0|r)], [permanent (p)]
```

name

[Required] One of the following (case insensitive):

ShowNav: Show Navigation Panels (Tree and, optionally, Catalog)

ShowTree: Show Tree

ShowFloppies: Show Floppy drives

ShowHiddenDrives: Show Hidden Drives

ShowHiddenItems: Show hidden files and folders

ShowSystemItems: Show system files and folders

ShowNethood: Show My Network Places

HideFoldersInList: Hide folders in list

ShowFolderSizeInList: Show folder sizes in file list

SortFoldersApart: Sort folders apart

KeepFoldersOnTop: Keep folders on top

SortMethod: Sort Method (0=Binary, 1=Textual, 2=Natural)

SortNatural: Natural numeric sort order (XP and higher) -- *deprecated, replaced by SortMethod, only kept for backward compatibility*

ResortAfterRename: Resort list immediately after rename

AutoRefresh: Auto-refresh

LockTree: Lock Tree

WatchDuringFileOp: Auto-refresh during file operations

RegexpRenameSep: Set the separator used in a Regexp Rename Expression

AllowRecursion: Show no warning on script recursions

UseCustomCopy: Enable Custom Copy for file operations

BackgroundFileOps: Enable background processing (XYcopy)

BackgroundedFileOps: Define the file operations that should be backgrounded

EnableFolderViewSettings: Enable Folder View Settings

FindExtendedPatternMatching: Enable Extended Pattern Matching

CacheThumbsReadOnly: Show Cached Thumbnails Only

SafeOverwrite: Safe Overwrite (Custom Copy)

PlayEventSounds: Play a sound on certain events

value

How to set the specified setting:

1: [Default] Enable the option.

0: Disable the option.

r: Restore the previous value (from before the script was executed).

Note that some settings have values other than 1/0/r:

With RegexpRenameSep, you pass the separator, e.g.

```
setting "RegExpRenameSep", ">>";
```

With BackgroundedFileOps, the value is OR-combined from the following values

```
  1 [x] Backup (currently unused; has yet to be coded)
  2 [x] Copy
  4 [x] Cross-volume move
  8 [ ] Intra-volume move  (always parallel = not queued!)
 16 [ ] Delete             (always parallel = not queued!)
```

Example:

```
setting "backgroundedfileops", 31; //= all fileops
```

AllowRecursion supports a value 2 that can be useful in custom column scripts that use `<cc_trigger>` and run into recursion because the script triggers a column refresh:

```
setting "AllowRecursion", 2;
```

permanent

Defines whether the state for the specified option should be "permanent", or restored on script execution end.

[empty]: [Default] Makes the change temporary, so the option will be restored to its previous value when the script execution ends.

p: Makes the change "permanent", so it stays that way even after script execution is over.

Note: You can also use command settingp to set options permanently.

Usage

Simply specify the option you want to change, and whether you want to enable or disable it. Obviously if the option is already on that state, no change will be done.

Note that restoring options to their previous values, or user values, is not required: all modified options are restored automatically by XY when script execution ends (unless specified otherwise by using "p" as third parameter) regardless of how the script ended: normal termination, error, user cancellation...

Examples

```
setting "shownethood", 1;
```

Ensure the network neighborhood is shown. A change of the setting is not permanent. If the network neighborhood was not shown before then it will automatically hidden again when the script has ended.

```
setting "HideFoldersInList"; msg "Folders are hidden!";
```

Hide folders in the list, show a message (to give you the chance to verify that the folders are actually hidden). After the script is over, the folders will be shown again (if they were shown originally).

```
setting "HideFoldersInList"; setting "HideFoldersInList", r;
```

Hide folders in the list, then immediately and explicitly restore to the original setting. If folders were hidden anyway this script will do absolutely nothing.

settingp

Alter some of XYplorer's user settings permanently.

Syntax

```
settingp setting name, [value=1]
```

name

[Required] The name of the setting. See command setting for a list.

value

How to set the specified setting:

1: [Default] Enable the option.

0: Disable the option.

Usage

Simply specify the option you want to change, and whether you want to enable or disable it. Obviously if the option is already on that state, no change will be done.

Command settingp is an alias to using command setting with parameter p (for "permanent"). See there for examples.

shellopen

Opens locations by the shell.

Syntax

```
shellopen location
```

location — Can be a real path, or a special path.
Defaults to the current list item.

Examples

```
shellopen "C:\Windows";
shellopen "%USERPROFILE%";
shellopen "shell:Libraries";
shellopen "shell:DocumentsLibrary";
shellopen "shell:MusicLibrary";
shellopen "shell:PicturesLibrary";
shellopen "shell:VideosLibrary";
shellopen "shell:sendto";
shellopen "shell:appsfolder";
shellopen "shell:Fonts";
```

Remarks

- Usually locations are opened in File Explorer.
- There are countless such "shell:..." locations to be found in scattered places all over the web. Some work, some don't.

shellprops

Displays the shell properties dialog for a given item.

Syntax

```
shellprops [item]
```

item — The item for which shell properties are to be displayed.
Or :addressbar, :tree, :list, or :catalog for the focused item in these controls.

Defaults to the focused item in the focused control (Address Bar, Tree, List, or Catalog).

Examples

```
shellprops;           //for the focused item in the focused control
shellprops ":list";   //for the focused list item (even if the list itself has no focus)
shellprops ":tree";   //for the focused tree item (even if the tree itself has no focus)
shellprops "C:\";     //for C:\
shellprops <curitem>; //for the currently focused and selected list item
```

showhash

Shows the most common hash values (MD5, SHA-1, SHA-256, SHA-512) for a particular file.

Syntax

```
showhash [file]
```

file	The file. Defaults to the current file, or the focused file if no file is selected.

Examples

```
showhash; //shows hash values for current file
showhash "D:\a.txt";
```

showintree

Shows a path in the tree.

Syntax

```
showintree [path], [expand]
```

path	[optional] the path to show. If missing the current path is shown (moved into view).
expand	0: Collapse 1: Expand !: Toggle collapse/expand of a node missing: Do nothing about it

Remarks

- Showing here can just mean to scroll the node into view. If the node is already visible in the view then the command does nothing.
- Showing can also mean that the tree is expanded to the path if necessary and possible.

- The path is never selected (browsed) by this command.
- If the path does not exist the existing part of it (if any) is shown.

Examples

```
showintree "Desktop";
showintree "T:\Fotos\2015\";
showintree "paper:echo"; //yet unofficial but working
showintree <xypath>, 1; //expand
showintree , "!"; //toggle expansion of current node
```

showstatus

Controls whether certain commands show a statusbar message.

Syntax

```
showstatus [show=1]
```

show	1: [Default] Show 0: Don't show

Remarks

- The initial setting for any script is 1 (show).
- Commands that are affected: colorfilter(), tag, tagitems(), timestamp.

Example

```
showstatus 0; tag 4; echo "no status"; showstatus; tag 3; echo "status";     //tag the current item with and without status
```

skipundo

Skips adding actions to the Action Log.

Syntax

```
skipundo [skip=1]
```

Remarks

- While set to skip, nothing will be added to the Action Log while the script is running.
- An action that's not in the Action Log cannot be undone.
- It's automatically reset to "don't skip" when the script is done.
- If this setting is OFF then skipundo has no effect: Configuration | File Operations | Undo & Action Log | Log actions and enable undo/redo.

Examples

```
skipundo;   //skip Action Log from now on
skipundo 1; //skip Action Log from now on
skipundo 0; //stop skipping Action Log from now on
```

slog

Pops a dialog showing the current Status Log.

Syntax

```
slog [switches]
```

switches r: clear the log (remove current contents)

Remarks

- It's basically a shorthand for `text get("statuslog");` but with a little more information in the header.
- See also Status Log.

Example

```
slog; //show log
slog r; //clear log
```

sortby

Set the sort order for the file list.

Syntax

```
sortby [column], [order], [secondary_sort]
```

column Defines the column on which the sort will be done. Defaults to Name.
Tip: A partial match is used as a fall back if there is no full match for the "column" parameter, so to match the column "Modified" you could use either "m", "Mod", or "modified", for example.

order Defines the order:
[empty]: [Default] Uses the default order for that column.
a: Ascending.
d: Descending.
clk: This is the same behavior as when you click on a column: reverses the order if the sort was already made on that column, otherwise uses the default order.

secondary_sort
0: primary sort

1: secondary sort

Usage

Simply define which column, and optionally which order to use, call the command and the List gets sorted..

Examples

Sort by Name using default order (ascending):

`sortby;` or `sortby n;` or `sortby name;` or `sortby Name;` or `sortby Name, a;` or `sortby "Name", "a";`

Sort by Name descending:

```
sortby n, d;
```

Sort by Modified in descending order.

```
sortby m, d;
```

Secondary sort by size descending:

```
sortby "size", "d", 1;
```

sortbylist

Apply a custom sort order to the current list.

Syntax

```
sortbylist list, [separator=CRLF]
```

list	List of items (with or without path, case-insensitive). Can also contain wildcard patterns (using wildcards * and ?).
separator	[optional] Separates the items in the passed list. Defaults to CRLF (line feed).

Remarks

- You can pass the complete list but you don't have to. Just pass the items you want to have on top, in the desired order.
- The items are moved to the top one by one as if mouse-dragged via manual sorting, so everything between the old and the new position is moved one step down.

Examples

Move one file to the top:

```
sortbylist "test.jpg";
```

Move two files to the top:

```
sortbylist "test.jpg|thumbs.db", "|";
```

Move all filenames in the clipboard (be it as text (Ctrl+P) or as items (Ctrl+C)) to the top:

```
sortbylist <clp>;
```

Move the currently selected items to the top:

```
sortbylist <get selecteditemsnames>;
```

Examples using wildcard patterns

Move all XY* stuff to the top:

```
sortbylist "xy*";
```

Move all RAF files to the top, followed by all TIF files, followed by all JPG files:

```
sortbylist "*.raf|*.tif|*.jpg", "|";
```

You can freely mix patterns and full names:

```
sortbylist "test.jpg|*.txt|Pakon-7.jpg", "|";
```

sound

Plays a sound.

Syntax

```
sound [sound], [loop], [event]
```

sound	File or label; leave empty to stop any sound. *1: Use internal sound #1 (embedded in the executable). *2: Use internal sound #2, etc. Currently it goes up to *5.
loop	0: Don't loop. 1: Loop.
event	0: Sound is an audio file. All playable audio formats are supported (playable by Quartz.dll aka DirectX) . 1: Sound is an event label.

Examples

```
sound "F:\Fun\sounds\ted_stranglehold.wav";
sound <curitem>, 1; //loop the current WAV file
sound; // stop any playing sound
sound "SystemHand", , 1; //system error sound
```

Notes

Event labels can be found in your registry here:

HKEY_CURRENT_USER\AppEvents\EventLabels\<EVENTLABEL>

status

Show a message in the status bar.

Syntax

```
status text, [color RRGGBB], [icon=ready|progress|alert|stop]
```

text: message text.

color: [optional] message color; default = marked text color #1.

icon: [optional] choose among four icons; default = "ready".

Usage

The message may contain any sort of variables.

Examples

```
status "Done!";
```

Shows message "Done!" in the status bar.

```
status "It's <date hh:nn:ss>.";
```

Shows message "It's 11:19:40." in the status bar (if that's the time).

```
status "Whoops!", "FF0000", "alert";
```

Shows message "Whoops!" in red color with alert icon.

statusbartemplate

Sets or retrieves the status bar template.

Syntax

```
statusbartemplate([template], [use=0])
```

template	The template. If omitted then the current template is not changed.
use	-2 = Toggle use/don't use. -1 = Return current value of this setting. 0 = Don't use it now. 1 = Use it now.
return	The current template.

Remarks

- The "template" parameter corresponds to "Configuration | Colors and Styles | Templates | Status Bar".
- The "use" parameter corresponds to "Configuration | Colors and Styles | Templates | Status Bar | Use status bar template".

Examples

```
echo statusbartemplate(); //show current template

statusbartemplate('|s|<get lengthsselected a 1>'); //set template

echo statusbartemplate(""); //reset template; return the one before the reset

statusbartemplate(, 0); //don't use

statusbartemplate(, 1); //use

echo statusbartemplate(, -1); //show state of use flag

statusbartemplate('<prop #image.dimensions> (<prop #aspectratio>)', 1); //set template and ensure it's used:
```

step

Enables "Step Mode" for the current execution of scripts.

Syntax

```
step
```

Usage

Use the script command step to activate the Step Mode. In Step Mode, you will have the command and its parameters shown, any variables being resolved, and you are prompted whether to continue or cancel. This is highly recommended for debugging scripts.

Use the command unstep to disable Step Mode.

Remarks

(1) You can manually enable Step Mode using menu Scripting | Step Mode or the Step Mode toolbar button. When manually enabled, you will be prompted before each command is executed, and commands step and unstep will be ignored.

(2) Note that commands step and unstep only apply to the current script, while option "Step Mode" has a global scope and will apply to all scripts, even scripts called within other scripts.

(3) You can enable prompting on a per-command basis in a much simpler way, by using prefix ? before the command's name. When you want to be asked before one single command gets executed, this syntax is much simpler to use than to use both step and unstep before and after the given command. For example, in the following script you will only be prompted whether or not to execute "command3":

```
command1; command2; ?command3; command4;
```

Example

```
goto "Desktop"; sortby "created", d; sel 1; focus "L"; step; #200;
```

Goes to the Desktop folder, sorts the list by created date descending, selects the first item, sets focus to list, turns on Step Mode, executes command #200. What is #200? Step mode will tell you before it is executed! (Solution: it's menu Edit | Cut.)

strlen()

Return the length of a string.

Syntax

```
strlen(string)
```

Examples

```
echo strlen("abcdef");  // 6
echo strlen("<curitem>"); // length of current item
```

strpos()

Return position of first occurrence of a string.

Syntax

```
strpos(haystack, needle, [start], [matchcase])
```

haystack	[Required] String to search in.
needle	[Required] String to search for.
start	[Defaults to 0] Allows you to specify which position in haystack to start searching. The position returned is still relative to the beginning of haystack. Negative start positions are supported to begin searching from the start position leftwards.
matchcase	[Defaults to 0] Compare method. 0: A=a 1: A<>a
return	Position.

Examples

```
echo strpos("abcabc", "a");       // 0
echo strpos("abcabc", "d");       // -1 (not found)
echo strpos("abcAbc", "A");       // 0
echo strpos("abcAbc", "A", , 1);  // 3
echo strpos("abcAbc", "a", 1);    // 3
echo strpos("abcAbc", "a", 10);   // -1 (not found)
echo strpos("abcAbc", "");    // -1 (not found)
echo strpos("", "a");         // -1 (not found)
echo strpos("", "");          // -1 (not found)
echo strpos("abcab", "b",  0);    // 1
echo strpos("abcab", "b", -1);    // 4
echo strpos("abcab", "b", -2);    // 1
```

```
echo strpos("abcab", "b", -3);    // 1
echo strpos("abcab", "b", -4);    // 1
echo strpos("abcab", "b", -5);    // -1 (not found)
```

strrepeat()

Repeats a string.

Syntax

```
strrepeat(string, [count=0])
```

string	string to be repeated
[count]	number of repetitions; defaults to 0
return	repeated string

Example

```
echo strrepeat("abc", 2); //abcabc
```

strreverse()

Reverses a string.

Syntax

```
strreverse(string)
```

string	string to be reversed
return	reversed string

Example

```
echo strreverse("abc");     //"cba"
echo strreverse(<clp>);     //show reversed clipboard contents
copytext strreverse(<clp>); //reverse clipboard contents
```

sub

This command allows you to execute another script (subroutine) within the current script resource (file or script).

Syntax

```
sub label
```

label

[Required] The label of the script to execute.

Usage

While you can always execute any script within any file by using the command load with a label, the command sub allows you to execute scripts from within the same file in an easier way, as you don't need to specify the file name again.

Simply put the label of the script to execute. Once its execution is over, the current script will continue.

If a script, called using load or sub, has its execution aborted (user cancellation, error, etc) then any calling scripts will have their execution aborted as well.

Example

The following is inside a script file "date.xys" (located in app data path).

```
"Show date : date"
  msg "<date yyyy-mm-dd>";
"Show time : time"
  msg "<date hh:nn:ss>";
"Show Date && Time : datetime"
  sub "date";
  sub "time";
```

If you now execute the script `load "date.xys", "datetime"` then you will first see a message box showing the date, and then a message box showing the time.

substr()

Return part of a string.

Syntax

`substr(string, [start], [length])`

string	[Required] Input string.
start	Start of returned part; defaults to 0, first position is 0. If start is negative, the returned string will start at the start'th character from the end of string.
length	Length of returned part (number of characters). If missing then the whole string is returned (beginning from start). If length is negative, then that many characters will be omitted from the end of string (after the start position has been calculated when start is negative). If start denotes a position beyond this truncation, an empty string will be returned.
return	Part of the string.

Examples

```
echo substr("abcdef", 0, 3);   // "abc"
echo substr("abcdef", 1);      // "bcdef"
```

```
echo substr("abcdef", 1, 1);   // "b"
echo substr("abcdef", -1);     // "f"
echo substr("abcdef", -2);     // "ef"
echo substr("abcdef", -3, 1);  // "d"
echo substr("abcdef", 0, -1);  // "abcde"
echo substr("abcdef", 2, -1);  // "cde"
echo substr("abcdef", 4, -4);  // ""
echo substr("abcdef", -3, -1); // "de"
```

swapnames

Swaps the names of the two selected List items.

Syntax

```
swapnames [bases]
```

bases	0 = swap names 1 = swap bases

Usage

Simply call it and it swaps the names of the two selected List items.

In order to work, there must be exactly two items selected in List, no more, no less, otherwise an error message will be displayed.

Note that the identical command is available on menu File | File Special | Swap Names.

Examples

```
swapnames;    //swap names
swapnames 0;  //swap names
swapnames 1;  //swap bases
```

sync

Synchronizes two folders.

Syntax

```
sync source, target, [copy_items], [on_collision], [delete_items], [switches], [logfile], [filter], [on_failure=-1], [custom_flags=1]
```

source	Source folder.
target	Target folder. The contents of this folder will be made identical to those of the source folder (depending on the other arguments).
copy_items	

0 = Do not copy any items to target.

1 = Copy items to target.

If missing the current setting in the Sync Folders configuration dialog is used.

on_collision

-1 = Ask

0 = Overwrite if newer

1 = Overwrite if different size or date

2 = Overwrite

3 = Skip

4 = Overwrite if different contents (The contents are compared by comparing the SHA-256 hash of each file.)

If missing the current setting in the Sync Folders configuration dialog is used.

delete_items (bit field)

0 = Do not delete any items in target.

1 = Delete items in target.

2 = Delete to recycle bin (if possible).

4 = Prompt before delete.

If missing the current setting in the Sync Folders configuration dialog is used.

switches (lower case letters in any order)

b = Background operation allowed (needs global setting Configuration | File Operations | File Operations | Background Processing | Enable background processing enabled to actually happen).

c = Show Sync Folders configuration dialog before start, initialized to the settings as passed with this command. Any subsequent changes to the settings in the dialog only affect *this* job. They do *not* affect the settings of the GUI Sync Folders function.

k = Keep progress dialog open after job is completed. Note that the default (when no switches are passed) is now to NOT keep the dialog open.

n = The target folder will be silently created if it does not exist yet. Such a target folder is shown in blue color in the sync dialog.

p = Preview Mode: Nothing actually happens, you just get a detailed report of what would happen in real mode. Note: When you also pass switch "c" then this flag is ignored apart from the fact that the Preview button is pre-focused. The button you choose decides whether it's a preview or the real thing.

r = Reverse direction (target > source).

logfile Optionally pass the name (full path/file, native and environment variables allowed) of a log file to which the full report of the operation will be written when it's completed. If the file already exists then the report is appended to it. No log file is written in Preview

Mode.

filter List of patterns or full paths, separated by "|", used to include or exclude files or folders in/from the operation by name (see Using the Filter).

on_failure

-1 = [Default] Ask

0 = Continue

1 = Cancel

custom_flags (bit field)

0 = Copy junctions as folders.

1 = [Default] Skip junctions.

2 = Copy junctions as junctions (admin rights required).

4 = Exclude hidden files and folders (referring to the HIDDEN file attribute).

If items with the HIDDEN attribute are among the selected items (or among the items explicitly specified as source items) then they are NOT excluded.

Examples

```
sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 5:="c"; //show config

sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 5:="p"; //preview mode

sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 5:="k"; //keep progress open

//copy, overwrite if newer; perm delete, no prompt; preview; config:

sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 1, 0, 1, "cp";

sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 6:="<xydata>\Log\Sync_<date yyyy-mm-dd>.
txt"; //log

// creates a new target folder every new day:

sync "E:\XY\code\", "F:\bup\XY-code-<date yyyy>\<date yyyymmdd>\", 1, 0, 0, "ckn";

sync "T:\Test\Source", "T:\Test\Target", 5:="ck", 9:=1+4; //show config, keep open; skip
junctions + exclude hidden

sync "T:\Test\Source", "T:\Test\Target", 9:=4; //copy junctions as folders + exclude hidden

sync "T:\Test\Source", "T:\Test\Target"; //skip junctions
```

Using the Filter

The filter consists of a list of patterns or full paths, separated by "|", used to include or exclude files or folders in/from the operation by name.

Pattern syntax:

a) `+* vs -*`: Prefix a + to include, a - to exclude. The + is optional (you only must use it if an include pattern happens to start with a "-" character).

b) `*\ vs *`: Append a backslash for folders. Else the pattern is matched against files.

c) `*\?* vs *?*`: Have a backslash anywhere in the non-last position to match the pattern against the full path of an item. Else it is matched only against the item name without path.

Remarks:

- All matching is done against source items, not target items.
- Wildcards * and ? are supported. No wildcards are auto-added.
- An item is processed if it matches at least one of the include patterns AND none of the exclude patterns. Formula: Pass Filter = (+ OR + OR ...) AND NOT (- OR - OR ...)
- The order of patterns doesn't matter logic-wise, but performance-wise: The list is worked from left to right, so the patterns that are matched more likely should be positioned more to the left.
- The filter also affects deletions in target, sync caps, preserve all item dates.
- If a folder does not pass the filter, the whole branch is out.
- If only folders are filtered then all files pass (apart from those in excluded folders, of course).
- If only files are filtered then all folders pass.
- The filter list is shown at the bottom of the Sync config dialog and in the reports.

Examples for Using the Filter

```
// copy only *.jpg and *.png files (both are equivalent):
sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 7:="*.jpg|*.png";
sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 7:="+*.jpg|+*.png";
// copy only from subfolders called data (and from the top source folder)
sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 7:="data\";
// don't copy files that begin with "copy ":
sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 7:="-copy *";
// don't copy from path E:\Test\Sync\Source\Temp:
sync "E:\Test\Sync\Source", "E:\Test\Sync\Target", 7:="-E:\Test\Sync\Source\Temp\";
// this is used for one-click daily backups of the XY source code; no need to backup the appdata folder:
sync "E:\XY\code\", "F:\bup\XY-code-<date yyyy>\<date yyyymmdd>\", 1, 0, 0, "ckn", 7:="-appdata\";
```

syncselect()

Modifies the selection in one pane based on items in the other pane.

Syntax

```
syncselect([command], [justcalc], [ignoreextensions], [mode], [direction])
```

command — The syncselect command to be executed; the following commands are available (case-insensitive, A=a):
SelectMatches: [Default] Select all items that are also listed in the other pane.
SelectUniques: Select all items that are only listed in this pane.

SelectNewer: Select all matches with a more recent modified date.
SelectDifferent: Select all matches with a different size or modified date.
SelectUniquesAndNewer: Select all items that are unique or newer.
SelectUniquesAndDifferent: Select all items that are unique or different.
SelectSelected: Select all items that are selected in the other pane.

justcalc
0: [Default] Select the items in the active pane.
1: Do not select anything but only return what would have been selected.

ignoreextensions
0: [Default] Mind extensions when matching the names.
1: Ignore extensions.

mode
n: [Default] New selection (drop current selections).
a: Add the matches to any current selections.
r: Remove the matches from any current selections.

direction
0: [Default] Select items on active pane based on items on inactive pane.
1: Select items on inactive pane based on items on active pane.
2: Select items in both directions.

return Pipe(|)-separated list of selected item names.

Remark

This command corresponds to the Sync Select dialog in menu Panes. It uses the identical routines. With the command syncselect() you gain access to these routines via User-Defined Commands and Custom Toolbar Buttons.

Examples

```
syncselect(); //select matches on the active pane
syncselect( , , , , 1); //select matches on the inactive pane
syncselect("SelectNewer"); //select newer matches on the active pane
syncselect('SelectUniquesAndNewer', , , , 2); //"Select Unique and Newer" on both panes
text syncselect("selectuniques", 1); //show uniques in text box
```

tab()

Retrieve tab information, modify a tab, or open a new tab.

Syntax

```
tab([operation], [data], [index], [ID], [flags])
```

operation operation, identified by name

"close" = close tab

data:

0 = [default] no prompts

1 = prompt if the tab is a default, locked, or home zone tab

index: index of the tab to close (leftmost tab = 1); defaults to the current tab

return: index of the new current tab

"closeothers" = close all other tabs

data:

0 = [default] no prompts

1 = prompt if there are default, locked, or home zone tabs

index: unused

return: index of the current tab before the others were closed

"closeothersright" = close all tabs to the right of a reference tab

data:

0 = [default] no prompts

1 = prompt if there are default, locked, or home zone tabs

index: index of the reference tab; defaults to the current tab

return: index of the reference tab

"default" = set default tab

data:

[missing] = toggle default

0 = remove default

1 = set default

index: tab to set as default (if missing then current tab)

return: index of the affected tab

"get" = [default] return a value depending on data (first tab = 1)

data:

[empty or missing]: returns the index of the current tab (first tab = 1)

"c", "count": Tab count.

"caption": Caption of a tab (as displayed).

"data": (Unresolved) data, e.g. %temp%.

"filter": Returns the Visual Filter (if any) of a tab.

"flags": Returns various tab properties in a bit field (values combined by OR, which here is equivalent to being added).
1 = Locked Location

2 = Locked Home Zone
4 = Iconized
8 = live filtered

"home": Home path of a tab (if it has any).

"ID": Returns the ID of the tab referred to by index (or by ID). The ID is a unique and stable identifier for a tab that never changes as long as the tab lives. The smallest possible ID is 1.

"IDprev": On or after a tab switch it returns the ID of the previously selected tab. Initially (after startup, before any tab switch) the value defaults to 1. The value can be used e.g. in the Custom Event Actions "Before browsing a folder" or "Switch tabs".

"index": Returns the index of the tab referred to by ID (or by index). The smallest possible index is 1 and refers to the leftmost tab.

"livefilter": Returns the Live Filter (if any) of the tab referred to by index. Note that this is the last used live filter of that tab. This does not mean that the filter is currently active.

"mode": The list mode of the tab.
0 = Browse
1 = Find (Search Results)
2 = Drives
3 = Network
4 = Recycler
5 = Browse (Portable Device)
6 = Find (Portable Device)
7 = Drives (Portable Device)

"name": Name of a tab (as set by rename).

"textcolor": Custom textcolor of a tab (if any) (as set by textcolor).

"backcolor": Custom backcolor of a tab (if any) (as set by backcolor).

"path": (Resolved) path of a tab, without trailing backslash.

"term": The path including any Visual Filter or Quick Search (same as menu View | Tab | Copy Location Term).

index: Tab to get value from (if missing then current tab).

return: Value depending on data.

"filter" = apply a Visual Filter to a tab

data: the filter pattern (if empty then any filter is removed)

index: tab to filter from (if missing then current tab)

return: index of the affected tab

"move" = move a tab to a new position

data: The new position (first position is 1; defaults to 1). Position 0 (zero) is the last position. Positions smaller than 0 are calculated from the right end: -1 = pre-last position etc. Positions higher than the number of tabs are treated like zero: tab is moved to the last

position.

index: tab to move (if missing then current tab)

return: the new position

"new" = create new foreground tab

data: Location (if empty then current tab is cloned); also Quick Searches and Visual Filters are supported. You can pass a file path in the data argument: the new tab will open at the containing folder and the file will be focused and selected. Examples:

```
tab("new", "%computer%?tags:*");  //Quick Search
tab("new", "%desktop%|x*");  //Visual Filter
tab("new", "C:\Shortcuts\ProgramData.lnk");  //also LNK files can be passed (when the target is a folder)
```

index: Tab to clone from (if missing then current tab).

return: Index of the newly created tab.

"newb" = new background tab

data: Location (if empty then current tab is cloned). You can pass a file path in the data argument: the new tab will open at the containing folder and the file will be focused and selected.

index: Tab to clone from (if missing then current tab).

return: Index of the newly created tab.

"clone" = create new foreground clone of the current tab

```
tab("clone");
```

"cloneb" = create new background clone of the current tab

```
tab("cloneb");
```

"relocate" = relocate a tab (make it point to a new location)

data: new location

index: tab to relocate (if missing then current tab)

return: index of the affected tab

"rename" = rename a tab

data: new name (if empty then any name is removed)

index: tab to rename (if missing then current tab)

return: index of the affected tab

"textcolor", "backcolor" = sets the custom text/background color of a tab

data: color value in hexadecimal format (RRGGBB) (if empty then any custom color is removed)

index: tab to color (if missing then current tab)

return: index of the affected tab

"sethome" = sets a Home path to a tab

data: path to set

index: tab to set home to (if missing then current tab)

Note: If you set the home of a non-current tab then all relevant list settings of the *current* tab are used for the home definition of that other tab.

return: index of the affected tab

"iconize" = iconize a tab

data:

[missing] = toggle iconize

0 = de-iconize

1 = iconize

index: tab to iconize (if missing then current tab)

return: index of the affected tab

"lock" = lock a tab

data:

[missing] = toggle lock

0 = unlock

1 = lock

index: tab to lock (if missing then current tab)

return: index of the affected tab

"lockhomezone" = lock the home zone of a tab

data:

[missing] = toggle lock

0 = unlock

1 = lock

index: tab to lock (if missing then current tab)

return: index of the affected tab

Note: If a tab has no home you get an error message.

index — Refers to a tab by its position, first tab = 1
Alternatively pass the name (if you named the tab via "Rename Tab...") or the path.
Wildcards * and ? are supported. The matching is case-insensitive (A==a).
Defaults to the current tab. See also individual descriptions above under each operation.

ID — Unique and stable identifier for a tab that never changes as long as the tab lives. Does not change if tab is moved.
The smallest possible ID is 1.
If ID is passed then index is ignored.
If you pass an invalid ID an error message is shown.

flags	(bit field) 1: Suppress any error message.
return	Usually the index of the affected tab (see also individual descriptions above under each operation)

Notes

If index is negative then position is calculated from the right end (-1 points to the right-most tab). If index is invalid then the current tab's index is used. The function only affects tabs on the active pane.

Examples

```
text tab(); //ret index of current tab (first tab = 1)
text tab(, "c"); //ret tab count
text tab("get", "caption"); //caption of a tab (as displayed)
text tab("get", "caption", , 1); //caption of the tab with ID=1
text tab("get", "name"); //name of a tab (as set by rename)
text tab("get", "path"); //(resolved) path of a tab
text tab("get", "data"); //(unresolved) data, e.g. %temp%
text tab("get", "ID", "*23");   //get ID of first tab whose name ends with "23"
tab("close");       //close current tab (no prompts)
tab("close", 1);    //close current tab (prompts)
tab("close", , 2);  //close second tab (no prompts)
tab("close", 1, 2); //close second tab (prompts)
tab("close", , "Mickey");       //close first tab called "Mickey" (no prompt)
tab("close", , "Desktop\Desk"); //close first tab pointing to "Desktop\Desk" (no prompt)
tab("close", , "Search Results", , 1); //no error if "Search Results" doesn't exist
tab("closeothersright");        //close all tabs to the right of the current tab (no prompts)
tab("closeothersright", , 3);   //close all tabs to the right of the 3rd tab (no prompts)
tab("filter", "*.txt"); //apply filter "*.txt" to current tab
tab("filter"); //remove any filter from current tab
text tab("new"); //clones the current tab; ret index of new tab
tab("move", 1);     //move current tab to 1st position
tab("move");        //(same as above)
tab("move", 4);     //move current tab to 4th position
tab("move", 4, 2);  //move 2nd tab to 4th position
tab("move", 0);     //move current tab to last position
tab("move", -1);    //move current tab to pre-last position
tab("new", "C:\"); //new foreground tab at "C:\"; as above
tab("newb", "D:\"); //new background tab at "D:\"; as above
```

```
tab("rename", "Godzilla");  //name the current tab "Godzilla"
tab("rename"); //remove any name from the current tab
tab("iconize", 1); //iconizes the current tab
tab("iconize"); //de-iconizes the current tab
tab("iconize", 1, 1); //iconizes the first tab
tab("iconize", 1, -1); //iconizes the last tab
tab("lock");        //lock toggle current tab
tab("lock", , 0);   //lock toggle current tab
tab("lock", , 1);   //lock toggle first tab
tab("lock", , -1);  //lock toggle last tab
tab("sethome", '%temp%'); //set home %temp% to current tab
tab("sethome"); //remove any home from current tab
tab("sethome", '%temp%', 6); //set home %temp% to tab 6
tab("relocate", '%tmp%'); //relocate current tab to %tmp%
```

tabset()

Opens, saves, or renames a tabset.

Syntax

```
tabset([operation=open], [name], [pane=a])
```

operation — Operation, identified by name.

"browse" — Open/Load a stored tabset by browsing the file system.
name: Folder to start browsing in (defaults to <xydata>\Panes).

"new" — Start a virgin tabset.

"open" — [default] Open/Load a stored tabset from the complete list of available tabsets.
name: Folder to look in (defaults to <xydata>\Panes).

"load" — Open/Load a stored tabset by name.
name: Name or full path of the tabset to load.
If name is missing the default tabset is opened.

"rename" — Rename the current tabset.
If name is missing the name is prompted for.

"revert" — Revert the current tabset to its saved state.

"save" — Save the current tabset.

"saveas" — Save the current tabset under a new name.
name: New name or full path.

"savecopyas" — Save a copy of the current tabset under a new name.
name: New name or full path.

name	(if not otherwise noted above) Name of the tabset (folder relative to <xydata>\Panes) or full path. If name is missing it is prompted for.
pane	The affected pane; defaults to the current pane.
a	active [default]
i	inactive
1	1st pane
2	2nd pane
return	The full path to the current tabset.

Examples

Open tabsets listing in order to select one:

```
tabset();
```

Load the active pane's default tabset:

```
tabset("load");
```

Load "Bat" into the active pane:

```
tabset("load", "Bat");
```

Start a new tabset named "Sue":

```
tabset("new", "Sue");
```

Rename back pane tabset to "Jack":

```
tabset("rename", "Jack", "i");
```

Load tabset via absolute network path:

```
tabset("load", "\\Cooper\Shared\blah\");
```

Cloning Tabsets

The operations browse, load, and open support cloning, i.e. opening a tabset under a new name. This is achieved by using the "as:"-syntax: Original source name and clone name are separated by " as: ", e.g. "work_1 as: clone1" to open tabset "work1" under the name "clone1". Working with clones instead of originals protects your sources and offers you a "revert to saved" functionality simply by repeating the command that loaded the clone before. The main window offers the same functionality in menu Tabsets | Open As...

Examples for "load":

Open tabset "work_1" as "clone1":

```
tabset("load", "work_1 as: clone1");
```

Open current tabset as "clone1":

```
tabset("load", " as: clone1");
```

Open tabset "work_1" as clone (prompt for name of clone):

```
tabset("load", "work_1 as: ");
```

Open current tabset as clone (prompt for name of clone):

```
tabset("load", " as: ");
```

Examples for "open":

Select a tabset in E:\Tabsets and open as "clone1":

```
tabset("open", "E:\Tabsets as: clone1");
```

Select a tabset from the default tabsets folder and open as "clone1":

```
tabset("open", " as: clone1");
```

Select a tabset from the default tabsets folder and open as clone (prompt for name of clone):

```
tabset("open", " as: ");
```

tag

Tag item(s).

Syntax

```
tag [value], [itemlist], [type=0], [tagsmode=0]
```

value	Tag value, which (depending on type) can be label name or ID, tag(s), or comment, or Extra Tag. If missing: remove any tag of that type.
itemlist	CRLF- or |-separated list of items (full path) to tag; if empty then current list selections are tagged
type	0=Label [Default] 1=Tags 2=Comment 3, ex1 = Extra 1 4, ex2 = Extra 2 ... 18, ex16 = Extra 16
tagsmode	Supported for these types: *If type is 0 (Label):* 0=Set [Default] 1=Toggle 2=Unset *If type is 1 (Tags):* 0=Add [Default] 1=Set (Replace any existing tags with the new tags) 2=Remove

Examples

```
tag "Green"; //set label "Green" for all selected items
```

```
tag "Green", 3:=1; //toggle label "Green" on all selected items
```

```
tag "Green", 3:=2; //unset label "Green" on all selected items (keep other labels)

tag 3; //set label #3 to all selected items
tag 3, <curitem>; //set label #3 to current item
tag 0, <curitem>; //remove any label from current item
tag 3, "C:\foo.txt|D:\bar.txt"; //set label #3 to these items
tag; //remove all labels from all selected items

tag "cats,dogs", , 1; //add tags "cats" and "dogs" to all selected items
tag "cats,dogs", , 1, 1; //set tags "cats" and "dogs" to all selected items
tag "cats,dogs", , 1, 2; //remove tags "cats" and "dogs" from all selected items
tag , , 1, 2; //remove all tags from all selected items

tag "wow", , 2; //set comment "wow" to all selected items
tag , , 2; //remove all comments from all selected items

tag "wow", , "ex3"; // set "wow" to Extra Tag 3 of all selected items
```

Multi-line comments

You can as well define multi-line comments. The CRLF (carriage return line feed) sequence is internally stored as pilcrow character (¶). Also in the file list's Comment column, which has only a single line per item, pilcrows are used for CRLFs.

These script lines create identical 2-line comments:

```
tag "multi-line<crlf>comment.", , 2;
tag "multi-line¶comment.", , 2;
```

Self-Referential Variables

There is a set of variables that are resolved on a per-item basis as the batch of selected items is processed. These variables make reflexive (self-referential) batch tagging a piece of cake.

```
<item>          = The full path/name of the item.
<itembase>      = The base of the item.
<itemext>       = The extension of the item.
<itemname>      = The name of the item.
<itempath>      = The path of the item.
<itemprop ...>  = A property of the item (equivalent to  <prop ...>).
```

Remarks

- `<itemprop ...>` is channeled through the powerful `<prop ...>` variable for each item, so it can do a lot of interesting things.

Examples

```
tag <itemname>, , 1, 1; //set each selected item's Tags field to its name
tag <itemname>, , 2; //set each selected item's Comments field to its name
tag "Size of <itemname>: <itemprop #1>, <itemprop #hash.md5 n>", , 2; //set each selected item's Comments field to its name, size, and MD5 hash
```

tagcheck()

Checks and repairs the state of the current tags.

Syntax

```
tagcheck(flags, [include_removables])
```

flags	(bit field) 1: Count orphans 2: Remove orphans 4: Remove dupes 8: Correct capitalization
include_removables	1: Also scan items located on removable drives. Background: You normally don't want to throw items out of your database just because a particular USB drive isn't plugged in.
return	Count of removed, if none count of corrected, if none count of orphans.

Remark

This function does not change any data on disk, just the tags in memory.

Examples

```
echo tagcheck(1); //returns count of orphans
echo tagcheck(2); //remove orphans, returns count of removed orphans
echo tagcheck(4); //remove dupes, returns count of removed dupes
echo tagcheck(8); //correct capitalization, returns count corrected items
echo tagcheck(14, 1); //remove orphans and dupes, correct capitalization, include removables
                      //meaning of return value depends on what happened
```

tagexport()

Exports tags data to a new tags database.

Syntax

```
tagexport([db], [path], [storage], [flags])
```

db	Path of tags database to create and export to. If it exists it is overwritten without asking questions. Resolved relative to current path. Empty or missing: defaults to "<curpath>\XYplorerTag.dat".
path	Only export tags data under this path (this folder and all subfolders, i.e. the whole branch). Empty: Export all tags data (not limited to any path). Missing: Defaults to current path.
storage	Storage mode (see Storage): 0: [Default] Absolute paths 1: Relative to application drive 2: Relative to application 3: Relative to tags database 4: Use volume serials
flags	(bit field) 1: Remove exported items from main database (else they are kept). NOTE: Ignored if bit 2 is set! 2: Export only the dirty items, i.e. those items whose new or changed tags are not yet saved to disk. 4: Return the data instead of writing it to a database.
return	The new tags database.

Remarks

- tagexport() complements tagload(): You can outsource specific sections of your tags data to small local databases (tagexport) and load them on demand from the local databases (tagload). Gains:
  a) Slim down and speed up your main tags database.
  b) Allow easy backup-avec-tags by keeping the local databases together with the tagged items.
- The new tags database is *not* loaded into the app. It's just created on disk similar to a "Save Copy As" operation.
- Note that you currently cannot combine flags 1 and 2! If you pass value 3 then bit 1 is ignored.

Example

In a path with tagged items, say hundreds of tagged photos, you want to export the tags data of this path to a local database. This command will create "XYplorerTag.dat" in the path, fill it with the data, and remove the data from the main database (usually "<xydata>\tag.dat"):

```
tagexport(, , 3, 1);
```

To load the new database when you are in the path, do this:

```
tagload("XYplorerTag.dat");
```

Using flags 2 and 4:

```
tagexport("DirtyTags.dat", 3:=2); //export only the dirty items in the current path
text tagexport(3:=6);             //return only the dirty items in the current path
```

```
text tagexport( , "", , 6);       //return only the dirty items (from everywhere)
```

tagitems()

Tags items, and returns current tags of a particular item.

Syntax

```
tagitems([field], [value], [itemlist], [flags])
```

field	One of the following: lbl, label: [Default] Label Field. tags, tag: Tags Field. cmt, comment: Comment Field. 1-16: Extra Tag 1 - Extra Tag 16. ex1-ex16: Extra Tag 1 - Extra Tag 16. ex:Name: Extra Tag with Column Caption "Name". Name: Extra Tag with Column Caption "Name".
value	Tag value, which (depending on field) can be label name or ID, tag(s), comment, or data. Multiple tags can be separated by either , (comma) or CRLF (line feed). Empty string ('') is treated similar to "Remove Data". If omitted, field is retrieved but not set.
itemlist	CRLF- or |-separated list of items (full path) to tag. If CRLF is not present, then | is taken as separator (if present). If empty then current list selections are tagged.
flags	(bit field) 1: Return tags of all passed/selected items (else only the first/focused item is returned). The separator ... is | when no itemlist is passed. ... is | when itemlist is separated by |. ... is CRLF when itemlist is separated by CRLF. 2: Return the *name* of labels; else the *index* is returned.
return	The old value(s) for this field in the first passed item, or, if none passed, in the focused item.

Examples

```
tagitems(, 1);     //set Label of all selected items to 1
tagitems(, "Red"); //set Label of all selected items to "Red" (that Label has to exist, of course)
echo tagitems();   //retrieve Label of the focused item
```

```
tagitems(, "");    //remove any Labels of all selected items
tagitems("tags", "Rock,Disco");  //set Tags of all selected items to "Disco" and "Rock"
echo tagitems("tags");     //retrieve Tags of the focused item
tagitems("tags", "");  //remove any Tags of all selected items
tagitems("cmt", "Done."); //set Comment of all selected items to "Done."
tagitems("cmt", "Done.", "E:\Test\Test.txt"); //set Comment of "E:\Test\Test.txt" to "Done."
echo tagitems("cmt");     //retrieve Comment of the focused item
echo tagitems("cmt", , "E:\Test\Test.txt");     //retrieve Comment of "E:\Test\Test.txt"
tagitems("cmt", "");  //remove any Comments of all selected items
```

Setting multiple tags

```
tagitems("tags", "yes,no"); //sets tags "yes" and "no" to the selected item(s)
tagitems("tags", "yes<crlf>no"); //sets tags "yes" and "no" to the selected item(s)
```

Flags 1

```
text tagitems("tags", , , 1); //return the tags of all selected items, separated by |
text tagitems("tags",  , "E:\Test\Boy.jpg|E:\Test\Cow.jpg", 1); //return the tags of all selected items, separated by |
text tagitems("tags",  , "E:\Test\Boy.jpg<crlf>E:\Test\Cow.jpg", 1); //return the tags of all selected items, separated by CRLF
```

Flags 2

```
echo tagitems(3:=2);  //show label of the first-passed/focused item
```

taglist()

Sets a new tag list, or modifies the current one.

Syntax

```
taglist([tags], [flags=0])
```

tags	Comma-separated list of tags. Empty: keep current.
flags	0: Replace current tag list. 1: Add to current tag list. 2: Remove from current tag list.
return	The current (old) tag list.

Examples

```
text taglist(); // displays the current tag list
taglist("may,june,july");     // defines a new tag list
taglist("may,june,july", 1);  // adds to current tag list
```

```
taglist("may,june,july", 2);  // removes from current tag list
```

Usage

Using taglist() you can easily switch tag lists and make tagging new files a snap.

Remarks

- The "Tag List" is your tag cloud. It is there to let you select tags rather than type them again and again.
- In the Edit Tags dialog there is a checkbox "Add new tag(s) to tag list" which auto-maintains your tag cloud. New tags are always added to the top of the list.
- You can also manually manage the Tag List in List Management. This way you can easily load a new cloud to have a different set of tags ready for selection.
- Note that editing the Tag List does not change or remove any tags from your files or folders! The Tag List is only your interface to selecting/attaching tags.

tagload()

Loads a new tags database, or reloads the current one.

Syntax

```
tagload([db], [mode])
```

db	Tags database to load; full path/name or relative to current path. Empty: load default DB (usually "<xydata>\tag.dat"). Missing: reload current DB (same as menu Tags | Reload Tags Database).
mode	0: (re)load [Default] (fully update/replace the current data) 1: import (permanently merge into current DB) 2: include (temporarily merge into current DB in memory; not saved to disk) 3: import with header data (Labels, Extra Tags)
return	The full path of the loaded/reloaded/imported tags database.

Remarks

- This function allows you to quickly load an in-place tags database that has been stored "Relative to tags database" (see Storage). So you can have database and tagged items in the same location. Makes backing up tags a breeze.
- You can also specify the name of a new not-yet-existing database file. All current tags are removed from memory, and Labels are reset to the 7 default colors.
- Note that the loaded database is retained across sessions (INI key TagDatCustom). If you want to return to the default database you have to explicitly load it by tagload("");.
- The default DB is usually "<xydata>\tag.dat" unless another DB is specified in Admin.ini.

Remarks on modes import and include

- Mode 1 and 2: All header data (Labels, Extra Tags) are taken from the current DB; header data from the imported DB are totally discarded.
- Mode 0 and 3: All header data (Labels, Extra Tags) are taken from the imported DB. Extra column headers and Label colors are auto-updated.
- You can import a database while keeping your dirty tags (= unsaved changes) alive (see Examples for Multi-User Tagging below).
- Dupes are checked automatically on import/include. In case of a dupe the item of the current DB is kept, the one of the imported DB is discarded.
- Include: You can include as many DBs as you like.
- Include: Included DBs are fully searchable with XYplorer's search by tags functions. Not being saved to disk with the current/primary DB is the only difference to primary DBs.
- Include: If you modify the tags of an item that has been tagged via including a DB, this item is automatically added to the primary DB. Otherwise the modifications would never be saved to disk.
- Include: Once you manually edit the DB in memory via "Configuration | Tags | Options | Edit Tagged Items..." the included items lose their temporary nature and become a permanent part of the current/primary DB (and will be saved to disk in this DB).

Examples

```
tagload(); //reload current DB
```

```
tagload(""); //load default DB
```

```
tagload(<curitem>); //load the current file as new tags database
```

```
tagload("D:\Stuff\XYtagsPhoto2016.dat"); //load this DB
```

```
tagload("<xydata>\tagnew.dat"); //create a new DB (if tagnew.dat does not exist yet)
```

Examples for modes import and include

```
tagload("XYplorerTag.dat", 1); //import "XYplorerTag.dat" (in current path) (without header data)
```

```
tagload("XYplorerTag.dat", 3); //import "XYplorerTag.dat" (in current path) (with header data)
```

```
tagload(<curitem>, 1);  //import the selected tags database
```

Include:

```
tagload("D:\TagDBs\Photo2016Tag.dat", 2); //include "D:\TagDBs\Photo2016Tag.dat"
```

```
tagload(<focitem>, 2);  //include the focused tags database
```

Three (functionally identical) ways to get rid of included databases in memory:

```
tagload();  //reload primary database
```

```
tagload(<xytagdat>);  //load primary database
```

Click *Menu Tags | Reload Tags Database.*

Examples for Multi-User Tagging

Example 1: This line reloads the current tags database as an import (the import flag is set to 1). If any other process (i.e. other user in the network) changed that database the changes will be loaded into your XYplorer's memory, while leaving your unsaved ("dirty") tags untouched. In case of dupes you win: The imported data will not overwrite your dirty data:

```
tagload(, 1);
```

Example 2: This imports another database into the current database while leaving your unsaved ("dirty") tags untouched.

```
tagload("<xydata>\moretags.dat", 1);
```

tagsave()

Saves the current tags to a file.

Syntax

```
tagsave([file], [flags])
```

file	Full path to database file. If the path or file doesn't exist it will be created without asking questions. Defaults to the currently loaded database (<xytagdat>).
flags	(bit field) 1, 2, 4: reserved 8: SafeSave: This value is only useful in Multi-User Tagging. See SafeSave.
return	1 on success, else "" (nothing).

Examples

```
tagsave(); //save tags now (typically to "<xydata>\tag.dat")
tagsave(, 8); //save tags now, using the SafeSave strategy
tagsave("E:\Test\TagsDB\Test.dat"); //save the current tags to this file (take a snapshot)
```

text

Shows a multi-line text box, allowing to select and copy text to the clipboard.

Syntax

```
text text, [width=800], [height=400], [caption], [wrap], [linebreaker="<br>"]
```

text	The text to be shown in the window.
width	Width of window in pixels (minimum: 250; maximum: screen width; default: 800).
height	Height of window in pixels (minimum: 150; maximum: screen height; default: 600).
caption	Caption on window's title bar.
wrap	[optional]

w	wordwrap (show text wrapped)
linebreaker	[optional] Any character sequence to be replaced by a line break. Defaults to " ". Pass "" to prevent any replacing.

Usage

Simply call the command with whatever text you want to show. You can of course use any variables in your text, as well as put your text over multiple lines, using the line breaker <br> which will be converted to a CRLF (0D0A).

You can change the line breaker (<br>) by using command br first.

Remarks

Any NULL characters in the displayed text are replaced by spaces.

Example

```
text "XYplorer is running from <xypath><br>The application data are stored in <xydata>", 550, 150, "XYplorer Locations";
```

Shows a small text-window with the application's folder (path to running XYplorer.exe) and the application's data folder (where all settings are stored).

thumbscacherename()

Updates a renamed or moved path in the thumbnails cache.

Syntax

```
thumbscacherename pathold, pathnew
```

pathold	old path name (with or without trailing backslash)
pathnew	new path name (with or without trailing backslash)

Remarks

- When a folder is renamed or moved *within* XYplorer the thumbnails cache is updated automatically. However, when *another* application changes paths, this new command will come in handy to update your thumbnails cache and thereby avoid that all those thumbnails have to be created again.
- The Status Bar gives a feedback on the success of the command.

Examples

```
thumbscacherename "E:\Test\old", "E:\Test\new";
```

```
thumbscacherename "E:\Test\old\", "E:\Test\new\";
```

```
thumbscacherename "E:\Test\", "F:\Test\";
```

thumbsconf()

Set or retrieve various thumbnail related settings.

Syntax

```
thumbsconf([settings="ShowCaption,ZoomToFill,Style,Padding,Transparency,ShowIcon,
ShowDimensions,OverlayCaption,FolderThumbs,ZoomToFit"], [separator=","])
```

settings	Comma-separated list of the following values:	
	ShowCaption:	0 or 1
	ZoomToFill:	0 or 1
	Style:	0 - 4 ("Plain" - "Frame with Shadow")
	Padding:	0 - 7 (= pixels)
	Transparency:	0 = neutral, 1 = grid, 2 = white, 3 = black,! = toggle 0/1
	ShowIcon:	0 or 1 or ! (= toggle 0/1)
	ShowDimensions:	0 or 1 or ! (= toggle 0/1)
	OverlayCaption:	0 or 1 or ! (= toggle 0/1)
	FolderThumbs:	0 or 1 or ! (= toggle 0/1)
	ZoomToFit:	0 or 1 or ! (toggle 0/1)
	If numeric: Return this number of values (counted from left).	
separator	Separator between values (input and return), defaults to comma.	
return	The current settings (before setting new values).	

Remarks

- Note that "settings" is just one argument that's a list of values.
- Leave out any values that should not be touched.
- For Boolean values (0 or 1) you can pass "!" to mean "toggle" (just like with SetLayout).

Examples

```
echo thumbsconf();  //just return current settings, e.g.: 1,0,0,2,0,0,0,0,1

echo thumbsconf(4); //just return the first 4 current settings, e.g.: 1,0,0,2

thumbsconf("0,1,0,0"); //extreme gallery mode (just changes the first 4 settings, the others
are left untouched)

thumbsconf("1,0,4,2"); //pretty regular mode

thumbsconf("1,,0"); //just enable ShowCaption and set Style to "Plain"

thumbsconf("!"); //just toggle ShowCaption

thumbsconf(",!"); //just toggle ZoomToFill

thumbsconf(",,,,0"); //set neutral background

thumbsconf(",,,,1"); //set transparency grid

thumbsconf(",,,,!"); //toggle neutral background / transparency grid

thumbsconf(",,,,,!"); //toggle ShowIcon

thumbsconf(",,,,,,1"); //ShowDimensions on

thumbsconf(",,,,,,,!"); //toggle OverlayCaption

thumbsconf(",,,,,,,,!"); //toggle FolderThumbs
```

```
thumbsconf(",,,,,,,,!"); //toggle ZoomToFit
```

Example for a Custom Toolbar Button script (with pushed state):

```
"Toggle Wall of Pictures"
  $on = '0,1,0,0';
  $off = '1,0,0,2';
  thumbsconf((thumbsconf(4) == $on) ? $off : $on);
  ctbstate(thumbsconf(4) == $on);
```

timestamp

Change any of the three file dates.

Syntax

```
timestamp [type = "cma"], [date / source_item], [itemlist], [source_type = "*"], [shift]
```

type	c|m|a or any combinations in any order (cm, am, cma, ac...) Defaults to "cma".
date	Timestamp to apply. Must be in a format that the local system can understand. Set to "*" to use the item's own date (useful with cloning dates within files and with the shift argument). Set to the pseudo date "0" (zero) to set a file time to the lowest possible value. Can be an 8-byte little-endian hex value prefixed with 0x, e.g. 0x01DA002E35164B00. Note that this value potentially provides the full file time resolution of NTFS down to 100 nanoseconds. Stamping this value complements the "hex" functionality of SC filetime(). If missing: Defaults to the current time. If empty: Does not change date at all. Note that if you pass a date variable in single quotes it will be resolved per file. See examples below.
source_item	Item whose timestamps are applied to the target item(s). Note: The item must exist, otherwise the argument is taken to mean a date! Note: The timestamps are applied with full resolution (with milli, micro, nano etc seconds ... whatever is there).
itemlist	A CRLF- or |-separated list of items (full path) to timestamp. If empty or missing then the current list selections are time-stamped.
source_type	One of "*|c|m|a" describing which date value from the source to use. Defaults to "*" which uses the correct date for each corresponding value in the type parameter.
shift	A number and a unit, separated by a space. The following units are supported: y = years m = months

w = weeks
d = days
h = hours
n = minutes
s = seconds
ms = milliseconds
us = microseconds

Examples with explicit dates

Set all 3 dates of all selected List items to Now:

```
timestamp;
```

Set all 3 dates of all selected List items to 2008-12-31 00:00:00:

```
timestamp , "2008-12-31";
```

Set modified date of all selected List items to today, 12:00:00:

```
timestamp m, "<date yyyy-mm-dd> 12:00:00";
```

Set modified and created date of C:\Test.txt and C:\Test2.txt to today, 12:00:00:

```
timestamp mc, "<date yyyy-mm-dd> 12:00:00", "C:\Test.txt|C:\Test2.txt";
```

Set the modified date to the lowest possible value:

```
timestamp "m" , "0";
```

Using hex values:

```
timestamp "m", "0x01DA002E35164B00";  //= 2023-10-16 14:42:22 (no fractional seconds)
```

```
timestamp "m", "0x0000000000000001";  //= 1601-01-01 01:00:00.0000001 (hour may depend on your time zone)
```

FYI, in NTFS the lowest possible file time is 1601-01-01 00:00:00.0000001Z (Z for Zulu = UTC = Coordinated Universal Time).

For FAT (VFAT, FAT32, exFAT) drives (typically USB drives and flash cards) the earliest possible file time is 1980-01-01 00:00:00.0000000 (local time, not UTC!).

Passing a date variable in single quotes

When you pass a date variable in single quotes (so that it is not evaluated before it is passed to the timestamp function) the variable is evaluated per-file. See the difference:

Set modified and created date of all selected files to the EXIF date of the current file:

```
timestamp mc, "<dateexif>";
```

Set modified and created date of all selected files to the EXIF date of each of those files:

```
timestamp mc, '<dateexif>';
```

Set the created date of all selected files to the "Media Created" date of each of those files:

```
timestamp c, '<datemedia>';
```

UTC Support (Universal Time Stamp)

By appending "Z" (or "z") you can mark the time expression as UTC (Coordinated Universal Time). This way you can modify the UTC date of a file directly (as it is internally used by NTFS). For example, this will set the modified date to UTC 12:00 today (displayed as 14:00 in Germany with Daylight Saving Time in effect):

```
timestamp "m", "<date yyyy-mm-dd> 12:00Z";
```

That way you can guarantee the same universal timestamp regardless of any DST (Daylight Saving Time) offset in effect. Very nice for software developers with a global market.

The ISO 8601 formats for UTC and time offsets from UTC are supported. The following time expressions all refer to the same point in time:

```
2013-05-14 10:00:00+02:00
2013-05-14 08:00:00+00:00
2013-05-14 08:00:00Z
2013-05-14 03:00:00-05
2013-05-13 23:00:00-0900
2013-05-13 20:00:00-12:00
```

Examples with cloned dates between files

Set all three dates of all selected List items to match those of the focused item:

```
timestamp , <focitem>;
```

Set the modified and created dates of all selected List items to the focused item's accessed date:

```
timestamp 'mc', <focitem>, , 'a';
```

For all selected List items set their modified date to the focused item's modified date and their accessed date to the focused item's accessed date:

```
timestamp 'ma', <focitem>, , '*';
timestamp 'ma', <focitem>;          //functionally same as above
```

Set all 3 dates of all selected List items to match those of Windows Explorer:

```
timestamp 'cam', "C:\Windows\explorer.exe";
```

Set all 3 dates of all selected List items to match those of XYplorer:

```
timestamp 'cam', <xy>;
```

Examples with cloned dates within files

Set the created date of all selected list items to the modified date of each item:

```
timestamp 'c', '*', , 'm';
```

Set the modified date of all selected list items to the created date of each item:

```
timestamp 'm', '*', , 'c';
```

Examples with shift

Set all three dates of all selected List items to one day after now:

```
timestamp , , , , "1 d";
```

Set all three dates of all selected List items to 30 seconds after the focused item's dates:

```
timestamp , <focitem>, , , "30 s";
```

Shift the modified and created dates of all selected List items to 8 hours earlier:

```
timestamp "mc", "*", , , "-8 h";
```

toolbar()

Customizes the toolbar.

Syntax

```
toolbar([buttons], [size], [index_set])
```

buttons	[optional] Comma-separated list of button keys.
missing:	Keep current toolbar.
empty (""):	Reset toolbar to factory default buttons.
size	[optional] 0: small 1: large 2: extra large If missing: keep current setting.
index_set	[optional] Index of button set, one-based (first set = 1). If missing or 0, the current set is used.
return	The current (old) toolbar.

Examples

Displays the current toolbar button keys:

```
text toolbar();
```

Defines a mini toolbar:

```
toolbar("back,fore,up,refresh,exitnosave");
```

Use "-" for separators, and "|" to start a new row of buttons:

```
toolbar("{menus_all},|,back,fore,up,|,refresh,-,exitnosave");
```

Resets toolbar to factory default, and makes it small:

```
toolbar("", 0);
```

Change button size:

```
toolbar(, 0);  // switch to small buttons
toolbar(, 1);  // switch to large buttons
toolbar(, 2);  // switch to extra large buttons
```

Examples with index_set:

```
text toolbar(2:=3); //return button set #3
```

```
toolbar("tbs,{menus_all},|,hotlist,dice,dark", , 3); //set button set #3
toolbar(toolbar(), 2:=2);  //set button set #2 to current button set
toolbar(toolbar(2:=3));  //set current button set to button set #3
```

Remarks

With some simple scripting, you can write your own toolbar manager and have as many different custom toolbars as you want, selectable from a popup menu, for example.

Tip: Hold down CTRL while clicking Tools | Customize Toolbar... to see the button keys along with the buttons. Alternatively, hold down CTRL while you hover over a button in the toolbar to see the key in the tooltip.

topindex()

Sets or gets the top index of list or tree.

Syntax

```
topindex(index, [pane=a])
```

index	Index of the first row to be shown in the view port. The top row is 1. t: Scroll up to the top (alternative to "1"). b: Scroll down to the bottom. Missing: Don't change it.
pane	a: [default] active pane i: inactive pane 1: pane 1 2: pane 2 t: tree
return	The current/previous top index.

Remarks

The command does not change any selections or the focus.

Examples

```
echo topindex();      //return the current topindex of the active list
echo topindex(, i);   //return the current topindex of the inactive list
topindex(1);      //scroll the active list to the top
topindex(t);      //scroll the active list to the top
echo topindex(1); //scroll the active list to the top, return the previous index
topindex(1, i);   //scroll the inactive list to the top
topindex(b);      //scroll the active list to the bottom
topindex(b, i);   //scroll the inactive list to the bottom
```

```
topindex(1, t);   //scroll the tree to the top
topindex(t, t);   //scroll the tree to the top
topindex(b, t);   //scroll the tree to the bottom
topindex(100);    //scroll the active list to row 100
topindex(topindex() + 10);  //scroll the active list 10 rows down
topindex(topindex() - 10);  //scroll the active list 10 rows up
```

trayballoon

Displays a balloon notification in the system tray.

Syntax

```
trayballoon message, [title], [infoflags=1], [queueable=1]
```

message	The text to display in a balloon notification. If empty no balloon is shown.
title	The title of the balloon notification. Can be empty.
infoflags	0: No icon. 1: [default] Information icon. 2: Warning icon. 3: Error icon. 4: XYplorer icon. ? + 16: No sound (suppress any ding-dong). ? + 32: Make it a large icon (does not work with all values in all Windows versions).
queueable	0: Delete message from queue (and icon from tray). 1: [default] Keep message in queue (and icon in tray).

Notes

- The balloon disappears by itself. As of Vista notification display times are based on system accessibility settings.
- If queueable=1 the application icon will remain in the tray (or in the area hidden under the tray arrow, depending on your settings) after the balloon went away. There does not seem to be another way.
- In current Win10, messages are queued only when you wait for the one displayed to disappear AND the message text is different from the previous one.

Examples

```
trayballoon "Displays a balloon notification in the system tray.", "TrayBalloon";
```

```
trayballoon "Don't even think about it!", "Last Warning", 2;
trayballoon "Don't even think about it!", "Last Warning", 34; //BIG warning icon
```

trim()

Strips spaces (or other characters) from the beginning and/or end of a string.

Syntax

```
trim(string, [charlist=" "], [mode="LR"])
```

string	String to trim.
charlist	One or more characters to be stripped: single, e.g. "X" groups, e.g. "xyab" ranges, e.g. "a-m", "3-6" wildcards, e.g. "#" (digits) The default is " " (space).
mode	Where to trim: L: left R: right LR: both (default) (not case-sensitive, "l" and "r" work as well)
return	Trimmed string.

Notes

It is not possible to combine the closing bracket "]" with other characters in a group or range. As a single character it works.

When passing "*" or "?" as charlist they are automatically treated as verbatim characters, not as wildcards.

Examples

```
text trim("  d  ");               //"d"
text trim("  d  ", , "L");        //"d  "
text trim("  d  ", , "R");        //"  d"
text trim("  d  ", , "rl");       //"d"
text trim("aadaa", "a");          //"d"
text trim("abdab", "cba");        //"d"
text trim("123ab2ac654", "#");    //ab2ac
text trim("123ab2ac654", "3-5");  //123ab2ac6
text trim("]]123654]]", "]");     //123654
```

unset

Unsets an existing variable.

Syntax

```
unset variable(s)
```

variable — The name of the variable(s) to unset.

Usage

Simply put the name of the variable you want to unset, and it will be unset, or destroyed. This is not the same as setting its value to nothing. For example:

```
$a = "A"; msg "a=$a";  //a=A
$a = ""; msg "a=$a"    //a=
unset $a; msg "a=$a"   //a=$a
```

Supports up to 11 variables in one statement, separated by commas:

```
unset $a, $b, $c, $d, $e, $f, $g, $h, $i, $j, $k;
```

unstep

Disables Step Mode for the current execution of scripts.

Syntax

```
unstep
```

Usage

See step.

update

Updates this instance of XYplorer.

Syntax

```
update [flags]
```

flags — (bit field)
1: include beta version
2: run update also if current version is up-to-date
4: run app (as administrator)
8: run app, ask before (as administrator)
16: silent (no questions)
32: run app (unelevated)
64: run app, ask before (unelevated)
128: run app (with same elevation/privileges as the current instance)

Examples

```
update;   //update to the latest official version (if there is a new one)
update 1; //update to the latest beta version (if there is a new one)
update 3; //update to the latest beta version (no matter what)
update 1 + 4; //update to the latest beta version, and run it
update 1 + 8; //update to the latest beta version, and ask whether to run it
update 1 + 32;  //update to the latest beta version, auto-run it unelevated
update 1 + 64;  //update to the latest beta version, ask to run it unelevated
update 1 + 128; //update to the latest beta version, auto-run it
update 128;     //update to the latest official version, auto-run it
```

Remarks

- The latest installer is downloaded to the XYplorer Temp folder ("<xydata>\Temp"). After update this file is not auto-deleted. Either delete it yourself, or store it somewhere for later usage, or simply don't care. It's just about 4 MB. BTW, the downloaded installer has the full build number in the name.
- XYplorer will be automatically closed (else it could not be updated). Whether the settings are saved depends on your setting of Configuration | Startup & Exit | Save settings on exit.
- When the update is finished you have to manually restart XYplorer, unless you set flag 4 or 8. Reason: If it would be restarted from the installer it would inherit the elevated rights of the installer, something you should better control yourself.

urldecode()

Decodes URL-encoded string.

Syntax

```
urldecode(string, flags)
```

string	String to decode (max length is 2083 characters).
flags	(bit field) 0: (= bit 1 not set) URL-decode according to RFC 1738, except + => space. But it also converts %20 => space. 1: URL-decode according to RFC 1738 (%20 => space). It will NOT convert + => space. 2: URL-decode strings that were encoded by using the URL_ESCAPE_AS_UTF8 flag.

Remarks

"%20" is always converted to " " (space) with either setting.

According to MS docs the flag URL_UNESCAPE_AS_UTF8 is Win8 or later.

Examples

```
echo urldecode("%E4%20%F6%20%FC");    //ä ö ü
```

```
echo urldecode("%E4%20%F6%20%FC", 1); //ä ö ü !
echo urldecode("%E4+%F6+%FC");        //ä ö ü
echo urldecode("%E4+%F6+%FC", 1);     //ä+ö+ü !!!
text urldecode("%E5%8C%97%E4%BA%AC", 2);  //returns "Beijing" in Chinese characters
```

urlencode()

URL-encodes string.

Syntax

```
urlencode(string, flags)
```

string	String to encode (max length is 2083 characters).
flags	(bit field) 0: (= bit 1 not set) URL-encode according to RFC 1738, except space => +. 1: URL-encode according to RFC 1738 (space => %20) . 2: URL-encode all non-ASCII characters as their UTF-8 equivalents.

Remarks

It also encodes the percent character (% = %25).

Examples

```
echo urlencode("ä ö ü");    //%E4+%F6+%FC
echo urlencode("ä ö ü", 1); //%E4%20%F6%20%FC
text urlencode(<clp>, 2); //encode some non-ASCII characters in the clipboard
```

utf8decode()

Decodes UTF-8 encoded string.

Syntax

```
utf8decode(string, flags)
```

Example

```
echo(utf8decode("KÃ¶ln")); //Köln
```

utf8encode()

Encodes a string in UTF-8.

Syntax

```
utf8encode(string)
```

Example

```
echo(utf8encode("Köln")); //KÃ¶ln
```

varname()

Returns the name of a variable.

Syntax

```
varname(variable, [flags])
```

variable	Variable, e.g. $a or $a[0].
flags	(bit field) 0: As passed [Default]. 1: Strip any array index. 2: In calling scope.
return	The variable name, depending on the flags.

Remarks

Flags bit 2 falls back to the current name if there is no calling scope.

Example

```
// get the name of a variable in the calling scope
   $a[0] = "foo";
   // nothing special here; no caller, so varname() falls back to the current name
   echo varname($a[0], 2) . " = " . $a[0];  //$a = foo
   // go down to a user function
   godowntowork($a[0]);

function godowntowork($var) {
   echo '$var' . " = " . $var;  //$var = foo
   // here's the interesting part; varname() returns the original name in the calling
scope
   echo varname($var, 2) . " = " . $var;  //$a[0] = foo
   echo "Array name in caller: " . varname($var, 3); //$a
}
```

vartype()

Returns the state of a variable.

Syntax

```
vartype(variable)
```

variable	Variable, e.g. $a or $a[0].

return One of these states: array, empty, float, integer, novar, string, undefined.

Examples

```
$a = 0; echo vartype($a);          //integer
$a = 4; echo vartype($a);          //integer
$a = "4"; echo vartype($a);        //integer (!)
$a = 4.1; echo vartype($a);        //float
$a = 4.0; echo vartype($a);        //float (!)
$a = "b"; echo vartype($a);        //string
$a = "b"; echo vartype($b);        //undefined
$a = ""; echo vartype($a);         //empty
$a = array(); echo vartype($a);    //array
$a[9] = 1.1; echo vartype($a);     //array
$a[9] = 1.1; echo vartype($a[9]); //float
$a = "b"; echo vartype(a);         //novar
$a = "b"; $b = "b"; echo vartype($a . $b);  //novar
```

view()

Sets or gets the current list view.

Syntax

```
view([index], [switches])
```

index (optional)
0-9: Index of the view to switch to.
-1: Set the next view.
Missing: No switch happens.

switches (optional)
t: Toggle view with Details view.
o: Loop (if index = -1).
r: Reverse direction (if index = -1).

return Index of the current view (before any switch).

Remarks

These are the indices of the views (note that they do NOT correspond to the positions in the Views menu):

0 = Details
1 = Details with Thumbnails #1
2 = List
3 = Small Icons
4 = Thumbnails #1

5 = Thumbnails #2
6 = Thumbnails #3
7 = Large Icons
8 = Small Tiles
9 = Large Tiles

Examples

```
view(0);        //Details view
view(2);        //List view
view(2, "t");   //toggle between List view and Details view (run same line again to toggle)
view(0, "t");   //toggle between last view and Details view (run same line again to toggle)
view(-1);       //next view
view(-1, "r");  //previous view
view(-1, "o");  //loop through all views
view(-1, "ro"); //loop through all views, reverse direction
```

wait

Yields execution for a specified time so that the operating system can process other events.

Syntax

```
wait [msecs]
```

msecs	Number of milliseconds to wait. Allowed values: 0 to 2147483647 (= 0x7fffffff = ~ 25 days)

Examples

```
wait;       //minimum yield: let the OS do one thing
wait 2000;  //let the OS do its things for 2 seconds (2000 ms)
wait 60000; //wait a minute
wait 1000; echo "Hi!"; //show message after 1 second
```

wipe

Wipes the specified items.

Syntax

```
wipe [itemlist]
```

itemlist	CRLF- or |-separated list of items (full path) to delete. The separator or the items may be surrounded by any number of blanks. Defaults to the selected items in list.

Notes

- The Wipe operation is identical to the one in menu File | File Special | Wipe. The main advantage with SC wipe is that you can pass the items to wipe as an argument.
- For safety reasons there is no way to turn off the warning prompt before the operation starts.
- Just as a reminder: You can also wipe whole folders. All files found within are individually wiped. Then the folders are removed.

Examples

```
wipe; // wipe all selected items in the file list
wipe "E:\Test\1.txt|E:\Test\2.txt"; // wipe two particular files
```

writefile()

Writes data to a file.

Syntax

```
writefile(filename, data, [on_exist], [mode], [start=1], [numbytes=-1])
```

filename — Full path/name, or relative to the current path. Will be created if does not exist yet. Also the path of the file to be written is auto-created if necessary.

data — String data to write.

on_exist

o: [default] create new file/if existing: overwrite
a: create new file/if existing: append
n: create new file/if existing: do nothing
r: create new file/if existing: rename by appending the default suffix as defined in Configuration | Templates | Incremental Affix.

mode

t: [default] Text; auto-detects whether text can be written as ASCII or needs to be written as UNICODE.
ta:Text ASCII (1 byte per char); wide chars (upper Unicode) are represented by "?". Will convert UNICODE text to ANSI according to the current codepage!
tu: Text UNICODE (2 bytes per char).
Auto-puts UTF-16 LE BOM at file beginning (LE BOM = Little Endian Byte Order Mark: 0xFFFE)
b: Same as "ta", but kept for symmetry with SC readfile's "b" mode.
r: Raw bytes: writes the passed bytes without any conversion (same as "tu", but without auto-BOM).
Note that in literal text (quoted strings) each character has 2 bytes.
ru: Like "r" but converted from Unicode (2 bytes per character) down to 1 byte per character before the writing to file.
utf8: Converts data to UTF-8 before writing it to file. A UTF-8 BOM is not added.
utf8bom: Converts data to UTF-8 before writing it to file. A UTF-8 BOM is added.

start	Start writing at this byte position. Defaults to 1 which is the very beginning. Note: Writing here is always overwriting, not inserting!
numbytes	Write this number of bytes. Defaults to -1 which means: write everything passed in the data argument.
return	The following values or OR-ed bitwise: 0 = failed 1 = data written 2 = file existed 4|filename = file was renamed | new filename (only if on_exist is r)
Success codes depending on "on_exist" argument:	
	"n" AND file existed > 2
	"a"/"o" AND file existed > 3
	"a"/"o"/"n" AND file created > 1

Usage

WriteFile() is implemented as a function (instead of a statement) because the return value can be useful. You may, however, call functions without using a dummy variable. See examples below.

Examples

Creates a 4 byte file in current folder:

```
writefile("test_A.txt", "text");
```

Creates a 14 byte file (2 bytes BOM + 2 * 6) in current folder:

```
writefile("test_U.txt", "text ".chr(20000));
```

Creates a 6 byte file in current folder:

```
writefile("test_TA.txt", "text ".chr(20000), , "ta");
```

Creates a 14 byte file (2 bytes BOM + 2 * 6) in current folder:

```
writefile("test-TU.txt", "text ".chr(20000), , "tu");
```

Creates a 6 byte file in current folder:

```
writefile("test-B.txt", "text ".chr(20000), , "b");
```

Creates a BOM-less UTF-8 file:

```
writefile("test-UTF8.txt", "äöü", , "utf8");
```

Creates a UTF-8 file with BOM:

```
writefile("test-UTF8BOM.txt", "äöü", , "utf8bom");
```

Read and write the current file in "ru" mode; source and target will be 100% identical independently of your locale:

```
writefile("test-out.txt", readfile(, "ru"), , "ru");
```

Note, however, that using "r" modes is recommended for speed:

```
writefile("test-out.txt", readfile(, "r"), , "r");
```

Writes "ab" at pos 3 to the currently selected file:

```
writefile(, "abc", , , 3, 2);
```

Example with on_exist = "r"

For example, if "a.txt" exists in the current folder, a new file "a-01.txt" is created if the Incremental Affix is "-01":

```
text writefile("a.txt", "ab", "r");
```

The text window will show a return like this including the full filename. The "5" (1 OR 4) means "data written" and "file was renamed": `5|E:\Test\a\K\a-01.txt`

writepv

Writes all permanent variables (PVs) to a file.

Syntax

```
writepv [file]
```

file	The file to write to. Can be relative to the current path. If empty it defaults to "<xydata>\pv.dat".

Examples

```
writepv;  //write to "<xydata>\pv.dat"
```

Remarks

See readpv for reading files written with writepv.

zip_add()

Adds items to a Zip archive.

Syntax

```
zip_add(zipfile, itemlist, [separator="|"])
```

zipfile	Path/name of Zip archive. Can be relative to current path. If empty the name is prompted.
itemlist	List of items to add. If empty defaults to the selected items.
separator	Separator of items in itemlist.
return	Full path/name of Zip archive.

Examples

Add selected items to zip (name is prompted):

```
text zip_add();
```

Add selected items to "test.zip" (in current path):

```
text zip_add("test.zip");
```

Add selected items to "E:\test.zip":

```
text zip_add("E:\test.zip");
```

Add "E:\a.png" and "E:\b.png" to "E:\test.zip":

```
text zip_add("E:\test.zip", "E:\a.png|E:\b.png");
```

zip_extract()

Extracts items from a Zip archive.

Syntax

```
zip_extract(zipfile, [path], [itemlist], [separator="|"], [wait=1])
```

zipfile	Path/name of Zip archive. Can be relative to current path. If empty the current file is used.
path	Target path. If the path does not exist it is silently created. If missing the path is auto-named from the zip file (suffixed on collision) and created in the location of the zip file. If empty ("") the current path is used (= Extract Here).
itemlist	List of items to extract. Paths are to be given relative to the Zip. If empty all items are extracted.
separator	Separator of items in itemlist.
wait	0: Extract asynchronously. 1: [Default] Wait for extraction.
return	Target path.

Remarks

Also extracts *.rar and other WinRAR formats if WinRAR is installed on the system.

Also extracts *.7z and other 7-Zip formats if 7-Zip is installed on the system.

Examples

Extract the selected Zip archive to an auto-named folder in the current path, and return the full path of that folder:

```
text zip_extract();
```

Extract "test.zip" (in current path), don't return the target path:

```
zip_extract("test.zip");
```

Extract "test.zip" to "new\":

```
zip_extract("test.zip", "new");
```

Extract "test.zip" to "E:\new\":

```
zip_extract("test.zip", "E:\new\");
```

Extract "test.zip" (current path) to here:

```
zip_extract("test.zip", "");
```

Extract "E:\test.zip" to here:

```
zip_extract("E:\test.zip", "");
```

Examples for extracting particular items from an archive

Note that XP behaves differently from Vista and followers when items in subfolders are copied:

- XP: Source subfolders are auto-created in the target folder before items are copied (like XY's "Rich Copy").
- Vista and later: Items are all copied flat to the target path (with possible collisions).

Extract "XYplorer.exe" from "E:\Test\ZipTest\xyplorer_12.50_beta_noinstall.zip" to "D:\XY\":

```
text zip_extract("E:\Test\ZipTest\xyplorer_12.50_beta_noinstall.zip", "D:\XY\", "XYplorer.exe");
```

Extract Test\a\b.png from deep.zip in the current path. XP: target folder "Test\a\" will be created if necessary:

```
text zip_extract("deep.zip", "", "Test\a\b.png");
```

Extract two same-named files from one archive. No problem under XP; under Vista you get a collision prompt:

```
text zip_extract("deep.zip", "", "test\a\a.png|test\b\a.png");
```

Offer to extract items from the currently selected zip file:

```
zip_extract(, "", inputselect("Check Items to extract to current folder", replace(zip_list2(, "|", 0), "<tab>", ""), "|", 11,,,, "<curitem>"), "|");
```

zip_list2()

Lists the items in a Zip archive.

Note: A previous similar function, called zip_list(), is still supported but deprecated.

Syntax

```
zip_list2(zipfile, [separator="|"], [flags], [separatorProperties=<tab>)
```

zipfile Path/name of Zip archive.

Can be relative to current path.
If empty the current file is used.

separator — Separator of items returned.

flags — (bit field)
1 = List the names plus basic properties (type, size, and modified date).
Else: List the names only.
2 = Include folders.

separatorProperties — String that separates the properties (properties are only shown if flags bit 1 is set).
Defaults to <tab> (= TAB character).

return — Items in Zip.

Notes

The items are listed sorted by path/name.

Items in a subfolder are listed with the relative path.

The column order is: path, name, size, modified.

If present the services of UnRAR.exe and 7z.exe are used and all archive formats supported by them can be listed.

Examples

```
text zip_list2(, <crlf>);     //list selected archive
text zip_list2(, <crlf>, 1);  //list selected archive, show properties
text zip_list2(, <crlf>, 2);  //list selected archive, include subfolders
text zip_list2(, <crlf>, 3);  //list selected archive, include subfolders, show properties
text zip_list2(, <crlf>, 1, ";");  //list selected archive, show properties, separate by ";"
```

5.9 Scripting Command Get

get()

Returns various information.

Syntax

```
get(info, [param1], [param2], [param3])
```

info: [required] Sort of info (*not* case-sensitive, A==a).

param1: [optional] Depends on info.

param2: [optional] Depends on info.

param3: [optional] Format template.

return: Retrieved info .

Optional Format Template (param3)

You now can pass an optional format template in the 3rd argument after the property selector. The * is a place holder for the value (if missing the value is simply appended to the template). Must be wrapped in single or double quotes if a space is included:

```
echo <get BytesSelected a 1 'Bytes: *'>; //Bytes: 1,436,919
echo <get BytesSelected a 1 '* Bytes'>;  //1,436,919 Bytes
echo <get LengthsSelected a 4 'Duration [*]'>;  //Duration [01:00:52]
echo <get LengthsSelected a 4 'Duration: '>;  //Duration: 01:00:52
```

Possible Values for info, param1, and param2

Alias

```
"Alias", [AliasName]
```

Returns the value of the alias "AliasName". If the alias does not exist then AliasName itself is returned unchanged.

The value of the alias is returned without resolving any arguments within the alias. Example:

alias: `@Greet=::echo "Hello, <@1>! It's <@2 Dewey>!";`

```
echo get("alias", "Greet"); //::echo "Hello, <@1>! It's <@2 Dewey>!";
```

Bitness

```
"Bitness", [file]
```

Returns the bitness (target platform) of a binary file (typically EXE or DLL). One of these values is

returned:
16-bit
32-bit
64-bit
64-bit Itanium
(unknown)
Examples:

```
echo get("bitness");  //bitness of the selected file
echo get("bitness", "C:\Program Files\XYplorer\XYplorer.exe");  //64-bit
```

Box

`"box", [path]`

path: If missing the current tree path is used.
return: If boxed: the background color in RRGGBB; Else: nothing
Use it to find out whether a tree folder is boxed and what's the background color.
In dark mode it still returns the color it would have in light mode. That's the value you can use for SC box.
It does not return a color for a node within a box, just for the boxed node itself.

BytesSelected

`"BytesSelected", [pane=a], [flags]`

Returns the sum of bytes of all selected items of any of the two panes. If folder sizes are shown then they are included in the byte count.
pane:
a = [default] active pane
i = inactive pane
1 = pane 1
2 = pane 2
flags: (bit field)
1 = Show thousand separators. On combination with flag 2: Show exact size as well.
2 = Friendly format.
4 = Size on disk.
Example:

```
status <get 'BytesSelected' '' 7>; //friendly size on disk with exact size
```

BytesTotal

`"BytesTotal", [pane=a]`

Returns the sum of bytes of all items of any of the two panes. Syntax see "BytesSelected" above.

Char

`"char", [codepoint][+codepoint][+codepoint]...`

Returns the character (or string of characters) defined by the codepoint(s). Each codepoint is passed as 4-digit hex value, so all 16-bit Unicode characters can be returned here if you know their number. Examples:

```
echo <get char 0041>; //A
echo <get char 0041>.<get char 0042>; //AB
echo <get char 0041+0042>; //AB
```

There is a shorthand <U+[codepoint]>:

```
echo <U+0041>; //A
echo <U+0041>.<U+0042>; //AB
echo <U+0041+0042>; //AB
```

1-digit, 2-digit, 3-digit hex values work as well:

```
echo <U+41+42+43>; //ABC
```

Also mixed digit counts is okay:

```
echo hexdump(<U+9+09+009+0009>, , "r"); //09 09 09 09
```

CmdLine

`"CmdLine", [token]`

Returns the app's command line, or a specific token of it. Examples:

```
echo get("CmdLine");    //returns the command line (without application)
echo get("CmdLine", 0); //returns the application as called
echo get("CmdLine", 1); //returns the 1st argument
echo get("CmdLine", 2); //returns the 2nd argument, etc.
```

CmdLineUser

`"CmdLineUser", [token]`

Returns the user field of the command line (passed via command line switch /user), or a specific token of it.

#CommandID

`#CommandID, [ReturnOnTrue = 1], [ReturnOnFalse = 0]`

Here you can retrieve the current state of a number of toggles with a command ID, and control what is returned on what state. Currently supported are the following Command IDs:

#178 = Floating Preview
#311 = Branch View (True when the active list is in Branch View)
#312 = Show Sort Headers In All Views
#315 = Touchscreen Mode
#316 = Dark Mode

#333 = Sort Folders Apart

#388 = Hide Protected Operating System Files

#391 to #399, #418, #419, #431, #466 to #468, #488 (all toggles in the "Tools | Customize Tree" submenu)

#401 to #417 (all toggles in the "Tools | Customize List" submenu)

#471 = Enable Folder View Settings

#480 = Auto-Refresh

#481 = Auto-Refresh is suspended or off

#489 = Mini Tree

#491 = Show Floppy Drives

#492 = Show Hidden Drives

#493 = Show Hidden Files and Folders

#494 = Show System Files and Folders

#495 = Show Junctions

#496 = Show Folders in List

#497 = Ghost Filter

#548 = Skip Custom Columns

#660 = Show Address Bar

#661 = Show Toolbar

#662 = Show Tab Bar

#663 = Show Navigation Panels

#664 = Show Catalog

#665 = Show Info Panel

#668 = Show Tree

#669 = Show Status Bar Buttons

#670 = Show Status Bar

#675 = Show Live Filter Box

#685 = Wide Info Panel

#686 = Catalog First

#689 = Allow Multiple Button Rows

#800 = Dual Pane

#801 = Horizontal Panes

#813 = Sync Scroll

#814 = Sync Browse

#1061 = Miscellaneous | General Functions | Show/Hide Main Menu

For example, this returns "DP" when Dual Pane is enabled, and "SP" when it's not:

```
text get("#800", "DP", "SP");
```

And this returns "1" or "0":

```
text get("#800");
```

Examples

```
$a = get("CountSelected"); msg $a;
```

```
$a = get("SelectedItemsPathNames", "|"); msg $a;
```

```
msg get("selecteditemsnames", chr(10));
```

```
text get("tree", <crlf>); //one path per line
```

```
echo get("bytesselected", i);  //inactive pane's sel. bytes
text get("selecteditemsnames", , "i"); //inactive pane's sel. items, one per line
```

CopiedData

`"CopiedData", [token]`

Returns the last data copied to this window via WM_COPYDATA. The returned data depends on the value of the "token" argument:

0: hWndSender|dwData|Data
1: hWndSender
2: dwData
3: Data

For example, this 2-line script will retrieve the value of <curitem> in another instance of XYplorer (721424), and then display it in a message box in this XYplorer (<hwnd>):

```
copydata 721424, "::copydata <hwnd>, " . '<curitem>' . ";", 1; echo <get copieddata 3>;
```

CountIcons

`"CountIcons", [file]`

Retrieve the number of embedded icons in a file.

file: Full path to the file. If missing the currently focused file is used.
return: The number of embedded icons in the file.

The following file type are scanned for embedded icons: .exe.dll.cpl.ocx.scr.icl.bpl.wlx.wfx.wcx.wdx.acm

Additionally it can count the number of images in an ICO and CUR resource (*.ico, *.cur).

Example for a Custom Column of type Template (the quotes are necessary to handle items with spaces in the name):

```
<get counticons "<cc_item>">
```

CountItems

`"CountItems", [pane=a]`

Count of items in List.

CountSelected

`"CountSelected", [pane=a], [flags]`

Count of selected items in List.

pane: see above at `"BytesSelected"`

flags: (bit field)

1 = Show thousand separators.

CurItem

`"curitem", [part], [flags]` and `"curitemprev", [part], [flags]`

Returns the current/previous item on whatever pane. Both take optional parameters path, file, base, and ext.

flags: (bit field)

1: Mind folders when returning "base" or "ext".

Note: get(curitem) differs from the native variable <curitem> in that it will return the previously selected and focused item if the currently focused item is not selected.

Examples:

```
echo get("curitem",);           //E:\Test\b\c\Jane.txt
echo get("curitem", "drive");   //"E:" or "\\VEGA\Users"
echo get("curitem", "path");    //E:\Test\b\c
echo get("curitem", "file");    //Jane.txt
echo get("curitem", "base");    //Jane
echo get("curitem", "ext");     //txt (original case extension)
echo get("curitem", "lext");    //txt (lower case extension)
echo get("curitem", "base");    //treats file and folders the same
echo get("curitem", "base", 1); //treats folders differently (folders have no
extensions)
```

CursorPos

`"cursorpos", param1`

Returns the current mouse cursor position on screen.

Syntax: get("cursorpos", param1)

param1:

x,X: X position

y,Y: Y position

[else]: Separator between X and Y position

If empty it defaults to ";"

return: Mouse cursor position on screen.

Examples:

```
echo get("cursorpos");      //425;237
echo get("cursorpos", ","); //425,237
echo get("cursorpos", ", "); //425, 237
echo get("cursorpos", "X"); //425
echo get("cursorpos", "y"); //237
```

Darkness

`"darkness"`

Returns the value of "Configuration | Colors and Styles | Highlights & Dark Mode | Dark mode |

Level of darkness (0 is darkest)".

DarkContrast

`"darkcontrast"`

Returns the value of "Configuration | Colors and Styles | Highlights & Dark Mode | Dark mode | Text contrast".

DefaultTab

`"defaulttab", [pane=a]`

Returns the index of the default tab; "1" for the first tab; "" if no default tab exists.

Dimensions

`"dimensions", [file], [template="%w% x %h%"]`

Returns various basic image properties.

file: The file. Defaults to the currently focused file.

template: Format of the returned string. Supports the following variables:

%w% = width

%h% = height

%r% = aspect ratio term, e.g. "1:2"

%rq% = aspect ratio quotient, e.g. "0.5"

%a% = area raw, e.g. "1048576"

%af% = area formatted or friendly, e.g. "1,048,576"

Defaults to "%w% x %h%".

return: Dimensions, area, and/or aspect ratio, depending on the template.

Examples, using the <get ...> syntax:

```
text <get dimensions>; //1200 x 1600
text <get dimensions  "%w% x %h% [%r%]">; //1200 x 1600 [3:4]
text <get dimensions  "ratio=%rq%">; //ratio=0.75
echo <get dimensions  %a%>; //1920000
echo <get dimensions  "%af% square pixels">; //1,920,000 square pixels
```

DriveLetter

`"driveletter", [drivename], [path]`

Returns the first drive letter matching the given drivename (aka volume label). An optional path parameter can be used to format the returned string.

drivename: Drive name (defaults to the current path).

path: Any path with a drive letter, or a "?" in the place of the drive letter. If the function finds a matching drive letter then it will replace the drive letter in [path] with it and return the new path.

return: The drive letter, or the path with the drive letter. If no matching drive letter is found nothing is returned.

Example (drive F:\ is named "Fun"):

```
echo <get driveletter "Fun" "C:\Windows">; //F:\Windows
```

DriveName

```
"drivename", [path], [flags]
```

Returns the name of the drive (aka volume label) where path is on.

path: Path (defaults to the current path).

flags:

0 = return drive name without drive letter

1 = return the drive letter as well (formatted according to system settings found in registry)

Drives

```
"drives", [type], [separator="|"]
```

Returns a list of drives.

The "type" parameter corresponds to the Windows drive types enumeration:

```
DRIVE_UNKNOWN       = 0
DRIVE_NO_ROOT_DIR   = 1
DRIVE_REMOVABLE     = 2
DRIVE_FIXED         = 3
DRIVE_REMOTE        = 4
DRIVE_CDROM         = 5
DRIVE_RAMDISK       = 6
```

If you omit the "type" parameter then all existing drives are returned.

Examples:

```
text get("drives"); //A:\|C:\|D:\|E:\|F:\|H:\|T:\|X:\|Y:\|Z:\
text get("drives", 0);  //
text get("drives", 1);  //Z:\
text get("drives", 2);  //A:\
text get("drives", 3);  //C:\|D:\|E:\|F:\|T:\
text get("drives", 4);  //X:\|Y:\
text get("drives", 5);  //H:\
text get("drives", 6);  //
```

Drop

```
"drop", [separator=CRLF]
```

Returns the items dropped onto a script in Catalog or onto a script file.

See Drop on a Script File.

Exif

`"exif", TagID, [file="<curitem>"]`

Returns the raw EXIF (Exchangeable image file format) value found in the file by the Exif tag ID. Examples:

```
echo get("exif", 33434); //ExposureTime (Exposure time, given in seconds), e.g.: "10/1250"
echo get("exif", 0x829A); //ExposureTime (Exposure time, given in seconds), e.g.: "10/1250"
```

Usually TagID is numeric but the function knows two special cases where it can be a string used to retrieve formatted GPS data from photo files (JPEG):

```
echo get("exif", "gps");    //coords, eg: 51°10'41.8"N 1°49'35.9"W
echo get("exif", "gpsdec"); //decimal, eg: 51.178278, -1.826639
```

Both formats work with Google Maps when appended to https://www.google.com/maps/place/

FileSystem

`"filesystem", [path]`

Returns the file system (NTFS, FAT32 ...) for the current or the specified path. Examples using the analog variable <get ...>:

```
echo <get FileSystem>;
echo <get FileSystem J:>;
echo <get FileSystem "J:\rocky 3.jpg">;
```

Find_Contents

`"find_contents", [(unused)]`

Returns the contents of the Contents field of the last search in the current pane in this session.

Find_QueryParsed

`"find_queryparsed", [(unused)]`

Returns the parsed search query of the most recent search in the current pane (it's stored for each pane separately), in a human-friendly format.

Note that you get this value only *after* the search has started. Works for both, Find Files and Quick Search.

Find_QueryParsed_Last

`"find_queryparsed_last", [(unused)]`

Like `"find_queryparsed"` but includes searches via SC quicksearch().

FocusedControl

`"FocusedControl", [(unused)]`

"A" for AddressBar, "T" for Tree, "C" for Catalog, "L" for List, "LFB" for Live Filter Box, "X" for other.

FocusedPos

`"FocusedPos", [pane=a]`

Position of focused item from top (top = 1) in List.

For the optional "pane" argument see "BytesSelected" above.

FreeSpace

`"FreeSpace", [item], [flags]`

Lets you retrieve the free space on a drive.

item: Full path to a drive, or item on a drive. If missing the currently focused item is used. If item is not a drive itself then the item's drive is used for the calculation.

flags: 0: Flexible format (unit depends on size, e.g. 77.58 GB, 1.17 MB, 120.24 KB, 123 bytes)

1: Return the exact bytes (no formatting).

2: Return the exact bytes with thousand separators.

4: Flexible format + formatted exact bytes.

8: Append the percentage relative to the total capacity.

return: The number of free bytes on the drive, depending on flags formatted or raw.

Note: Mount points are honored.

Examples:

```
echo get("freespace");
echo get("freespace", "E:\Test");
echo get("freespace", , 1); //return raw bytes
echo <get freespace '' 4>;  //12.05 GB (12,939,067,392 bytes)
```

GVF

`"gvf"`

Returns the current Global Visual Filter.

Instance

`"Instance", [on_first=1], [on_nonfirst=2]`

Tells you whether the current instance is the first instance or not. The latter can be the case if you opened multiple instances of XYplorer. Note that even if you later close the first instance of two, the remaining instance will continue to report that it is non-first.
[on_first]: String to return if current instance is the first instance. Defaults to "1".
[on_nonfirst]: String to return if current instance is not the first instance. Defaults to "2".
Example: `echo get("instance", "First", "Second");`

Item

`"Item", [pane=a], [pos]`

Returns the focused item of any of the two panes.
pane: For the "pane" argument see "BytesSelected" above.
pos: Pass the position of the item in the list (the first position is "1"). If missing the focused item is returned.
Example: `echo get("Item", i);  //inactive pane's focused item`

ItemsNames

`"ItemsNames", [separator=CRLF], [pane=a]`

All items (name, no path) in List, separated by param1.

ItemsPathNames

`"ItemsPathNames", [separator=CRLF], [pane=a]`

All items (full path) in List, separated by param1.

ItemsPathNamesSlashed

`"ItemsPathNamesSlashed", [separator=CRLF], [pane=a]`

All items (full path) in List, separated by param1. Identical to "ItemsPathNames" but returned folders come with a trailing backslash.

LengthsSelected

`"LengthsSelected", [pane=a], [flags]`

Returns the sum for the durations of all selected media files.
pane:
a = [default] active pane; i = inactive pane; 1 = pane 1; 2 = pane 2
flags: (bit field)
1 = Show milliseconds.

2 = Return value only if the Length column is visible (i.e. if it can be pulled from that column).
4 = Use the shell property "Length" (otherwise use XYplorer's native and more exact special property "audio.length").
return: Sum of durations in format [hh:]nn:ss[.fff].

List_CopyTo

`"list_copyto", [separator=CRLF]`

List of current MRU items used in Copy To / Move To / Backup To.

List_HiliteFolder

`"list_hilitefolder", [separator=CRLF], [flags]`

[separator]: String to place between items. Defaults to CRLF (line feed).
flags: Defaults to "fc".
f = Return folders.
c = Return colors.
fc = Return folders and colors.
return: List of highlighted tree folders with their colors in RRGGBB (depending on flags). Format is identical to that used in the INI file.
This example loads a Mini Tree with all highlighted folders:

```
loadtree get("list_hilitefolder", "|", "f");
```

List_History

`"list_history", [separator=CRLF], [flags], [tab_index]`

[separator]: String to place between items. Defaults to CRLF (line feed).
flags: Defaults to "fc".
f = Return folders.
c = Return colors.
fc = Return folders and colors.
tab_index: Tab index (first tab = 0) for a tab-specific history. If missing then the global history of the pane is returned.
return: List of historical locations (identical to menu Go | History), ordered in backward chronology from latest to earliest. The current position in history is ignored.
Examples:

```
text get("list_history");       //global history of the current pane
text get("list_history", , 1);  //history of the 2nd tab of the current pane
```

List_RecentLocations

`"list_recentlocations", [separator=CRLF]`

List of recent locations (identical to menu Go | Recent Locations).

List_RecentlyOpenedFiles

`"list_recentlyopenedfiles", [separator=CRLF]`

List of recently opened files (as used in menu File | Open...). Example:

```
open inputselect("Open any of the Recently Opened Files", get
("list_recentlyopenedfiles"), <crlf>, 5);
```

ListOfCommands

`"listofcommands", [flags], [separator=CRLF]`

Returns the list of all commands.

flags: (bit field)

1 = Prepend Command IDs, separated by a TAB.

2 = Append Keyboard Shortcuts, separated by a TAB.

4 = Include accelerators.

return: The list of all commands.

LiveFilter

`"livefilter", [pane=a]`

Returns the currently active live filter (or nothing if none).

For the optional "pane" argument see "BytesSelected" above.

LoadTimes

`"LoadTimes", [threshold=10]`

Returns the load times as string.

threshold: Threshold in milliseconds to filter the load times listing. The default is 10 ms, i.e. only events that took at least 10 ms are listed.

MenuCaption

`"MenuCaption", command_ID, flags]`

Returns the command's menu caption in the currently loaded language.

command_ID: Command ID.

flags: (bit field)

1 = Append the currently assigned KS, separated by a TAB.

2 = Strip accelerators.

4 = Show error message if command_ID does not exist.

return: The menu caption.

Network

`"network", [method=0]`

Returns the network items (items listed directly under the Network node).

method:

0: [Default] Shell. Works everywhere, but is not the fastest.

1: NetServerEnum. Works fast if it works.

2: WNet. Works great on Win8 and earlier, but needs SMB1 and hence fails on Win10 and later.

Example: `text get("network", 1);  //NetServerEnum`

Pane

`"Pane", [(unused)]`

Returns "1" if the first pane is active, "2" if the second pane is active.

Path

`"Path", [pane=a]`

Returns the path of any of the two panes, without the trailing slash.

For the optional "pane" argument see "BytesSelected" above.

Example: `echo get("path", i);  //inactive pane's path`

Pick

`"pick", [filter=-1], [folder]`

Return a subset of files (no folders) from a given folder (not recursive, i.e. not including subfolders).

filter: if numeric: Number of files to pick.

-1 [Default]: pick all files.

Append .m, .c, or .a to the number to pick the latest files by Modified, Created, or Accessed date.

Append .n to the number to pick the top files by Name.

Append .x to the number to pick the top files without any active sorting.

Append r to the above sort switch reverse the order (pick the oldest/bottom).

if textual: Simple filename pattern (case-insensitive: A==a).

Automatically surrounded by asterisks if there are no wildcards.

A list of patterns separated by | or ; is okay, too.

folder: Folder to pick files from. Final backslash is optional.

Defaults to the current tree folder.

return: Set of files (no folders), one per line, each with full path.

Remarks

This command has been specially tailored for the use in Virtual Folders.

There are related parameters that can return folders as well:

pick: pick all files
pickfiles: pick all files (synonym of pick)
pickdirs: pick all folders
pickall: pick all files and folders

There are related parameters that can pick items from the current list instead of the current folder. This makes picking work in Search Results, Branch View, Visual or Live Filtered lists, or Virtual Folders (one can pick from another!):

pick_list: pick all files from the current list
pickfiles_list: pick all files (synonym of pick_list) from the current list
pickdirs_list: pick all folders from the current list
pickall_list: pick all files and folders from the current list

Examples

```
text <get pick>;                //pick all files
text <get pick 8>;              //pick any 8 files
text <get pick 8.m>;            //pick the latest 8 files
text <get pick 8.mr>;           //pick the oldest 8 files
text <get pick *.txt>;          //pick all TXT files
text <get pick 7/*.txt>;        //pick any 7 TXT files
text <get pick 7.m/*.txt>;      //pick the latest 7 TXT files
text <get pick {:Image}>;       //pick all image files
text <get pick 3/{:Image}>;     //pick any 3 image files
text <get pick 3.m/{:Image}>; //pick the latest 3 image files
text <get pick 4>;                      //pick 4 from the current tree folder
text <get pick 12 "E:\Test\Text">;    //pick 12 from E:\Test\Text
```

The folder can be soft:

```
text <get pick 12 %SystemRoot%>;   //pick 12 files from C:\Windows
text <get pick 12 <xyscripts>>;    //pick 12 files from XYplorer's scripts path
text <get pick 12 paper:Pictures>; //pick 12 files from Paper Folder "Pictures"
```

Examples using the short form without "get" and within a Virtual Folder definition

```
vi:<pickall 12.m>     //the newest 12 items (files and folders) from the current folder
vi:<pick_list 12.n>   //the top 12 items from the current list (after sorting the list
by name)
vi:<pick_list 12.x>   //the top 12 items from the current list (no sorting)
vi:<pick_list 12.xr>  //the bottom 12 items from the current list (no sorting)
vi:<pick_list>;       //all items from the current list, one per line
vi:<pick_list a*>;    //all items starting with a* from the current list, one per line
```

```
vi:<pick_list 12.m>   //the newest 12 items from the current list
```

PreviewHandler

`"PreviewHandler", [extension], [separator=CRLF]`

Retrieve the GUID, Standard value, and DisplayName of the default preview handler associated with a given extension.

extension: Any registered file extension. Defaults to the extension of the currently selected list item.

separator: Separator between GUID, Standard, and DisplayName in the returned string.

return: GUID,separator,Standard,separator,DisplayName. Nothing if no preview handler is found.

Example: `text get("previewhandler", "pdf");`

Omit the extension parameter to learn about the currently active preview handler while a preview is being shown.

Properties

`"properties", [flags=14], [item]`

Retrieve all available properties of an item.

- flags: (bit field)
  - 1: Show empty properties as well.
  - 2: Show XY Special Properties.
  - 4: Show Shell Properties.
  - 8: Show System Properties (see Notes!).
  - Defaults to 14 (2+4+8).
- item: Full path to an item.
  - If missing the currently focused item is used.
- return: The properties in form of a block of text.

Notes:

- Will take one or more seconds to return.
- For the System Properties to work you need a helper file that contains all names of the properties. This file must be called "SystemProperties.txt" and it has be located in the scripts folder (<xyscripts>). Download one here: https://www.xyplorer.com/download/SystemProperties.txt

Examples:

```
text <get properties>; //show all available non-empty properties of the focused item
text <get properties 1>; //show all available properties of the focused item
text <get properties 2>; //show the non-empty XY Special Properties
text <get properties 3>; //show all XY Special Properties
```

PropertyIndex

`"PropertyIndex", PropertyName`

- PropertyName: Name of the property. It's identical to the column header in Details view, and has to be in the system locale (e.g. in German in a German Windows).
- return: The numerical index of that property.

Note: The return varies from Windows to Windows, and from system to system.

Usage: Make scripts using extended properties (SC property, SC report with "prop:") portable and working under different Windows versions.

Examples:

```
text get("propertyindex", "besitzer");    // -> might return "8" in a German XP
text get("propertyindex", "orientation"); // -> might return "254" in a English Win7
```

RegCmd

`"regcmd", extension`

Retrieves the default verb and command line (the one triggered by dbl-click) for any registered file type from the registry.

Example: `text get("regcmd", "mp3");`

RowCountView

`"RowCountView"`

Returns the number of fully visible rows in the current viewport of the list. Note that the value can be larger than the number of items in the list, namely in small lists without visible scrollbars. Half-visible rows are not counted.

RS

`"rs", [length=8]`

`[characters="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789"]`

Returns a random string.

- length: Length (character count) of the returned string. Defaults to 8. You can also give a min and a max value separated by "-" for a random length, e.g. "8-12". There is an arbitrary maximum length of 32768, just to protect you from yourself.
- characters: Characters from which the returned string is built. Defaults to "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789".

Examples:

```
echo get("rs");
echo get("rs", "4-12", "abc");
```

SaveOnExit

`"saveonexit"`

Returns the value of "Configuration | General | Startup & Exit | Save Settings | Save settings on exit" (0 = unticked, 1 = ticked).

Screen

`"Screen", [width|height|dpi]`

Returns the screen width or height in pixels, or DPI (dots per inch).

Returns "width|height|dpi" when the 2nd argument is omitted.

SelectedItemsNames

`"SelectedItemsNames", [separator=CRLF], [pane=a]`

All selected items (name, no path) in List, separated by param1.

Alternative short form: `"sin"`

Note that before v9.40.0007 the separator was actually a delimiter (i.e. it was returned after each item, not between the items).

SelectedItemsPathNames

`"SelectedItemsPathNames", [separator=CRLF], [pane=a]`

All selected items (full path) in List, separated by param1.

Alternative short form: `"sip"`

See note above.

SelectedItemsPathNamesSlashed

`"SelectedItemsPathNamesSlashed", [separator=CRLF], [pane=a]`

Identical to "SelectedItemsPathNames" but returned folders come with a trailing backslash.

Alternative short form: `"sips"`

SelExt

`"SelExt", [separator=";"]`

Returns a list of all currently selected file extensions in the common format "*.aaa;*.bbb;*.ccc".

For files without extension the pattern "*." is returned.

Selected folders are ignored. If only folders are selected then nothing is returned.

Shift

`"Shift"`

Returns the held modifier keys:
0 = No keys held
1 = Shift
2 = Ctrl
3 = Ctrl + Shift
4 = Alt
5 = Shift + Alt
6 = Ctrl + Alt
7 = Ctrl + Shift + Alt

Sort

`"Sort", [pane=a], [dimension=1]`

Returns the sorted column name and sort direction for the primary or secondary sort order.
pane: a = [default] active pane; i = inactive pane; 1 = pane 1; 2 = pane 2
dimension: 1 (default) or 2 for primary or secondary sort order.
Examples for returns:
"Name,a" //Name, ascending
"Name,d" //Name, descending
"Modified,d" //Modified, descending
"Random" //Random Order
"Unsorted" //Unsorted

SpecialFolder

`"SpecialFolder", [CSIDL]`

[CSIDL]: Numerical constant predefined by the OS. Defaults to 0 (= Desktop folder). You can easily find those values in the web.
Returns the full path to the special folder.

SpotPatterns

`"SpotPatterns"`

Returns the last applied Type Ahead Find or Spot patterns in a |-separated list. BTW, these patterns are not remembered across sessions.

Status

`"Status", [section]`

Returns the current contents of the Status Bar.
The optional "section" argument can be:
1,2,3: Index of the section, the first section being number 1.

help: Last displayed help text (shell context menu hovering).
If missing all numbered sections are returned.

Stepping

`"stepping"`

Returns whether the script is currently in step mode. 0 = step mode OFF, 1 = step mode ON.
Note that the return is not identical to the current setting of "Scripting | Step Mode". This is only one factor that affects whether you are actually in step mode.

Tabs

`"Tabs", [separator="|"], [pane=a]`

Returns the paths of all tabs in the current pane, from left to right, separated by separator.
For the optional "pane" argument see "BytesSelected" above.

Tabs_sf

`"Tabs_sf", [separator="|"], [pane=a]`

Same as "Tabs" above but the path of the selected tab returned first.
For the optional "pane" argument see "BytesSelected" above.

TargetItems

`"TargetItems", [separator=CRLF]`

Full paths of all target items of the last file operation (copy, move, backup, create new, add to paper folder).

TextMetrics

`"TextMetrics", text`

text: text string to measure
return: - Width and height of text in pixels if it would be printed to the file list with the current font settings.
- Current font settings of the file list.
The Width entry additionally compares the results of three different measurement methods. They should be identical when the font is well constructed. The third result is from the method used for the file list.
Example: `echo get("textmetrics", "abc");`

ThumbnailProvider

`"ThumbnailProvider", extension, separator=CRLF`

Like "PreviewHandler" above, only that it retrieves information on the thumbnail provider.

Thumbs_Cache

`"thumbs_cache"`

Full path of the currently used thumbnails cache file if any, e.g. "E:\XYThumbs\00aa5de8b596e9448481100523751227.dbits".

Tree

`"Tree", [separator="|"]`

List of paths present in the current tree, separated by separator. The current path is the first in the list.

Usage: Mainly useful for creating the pathlist argument for the loadtree command.

Trigger

`"Trigger", [mode="mousebtn"]`

mousebtn: Returns the mouse button that triggered a script of a Custom Toolbar Button.

Returned values:

0 = None = 0

1 = LeftClick = 1

2 = RightClick = 2

4 = MiddleClick = 4 (planned)

16 = Drop (OR-ed with the button values if triggered by drag-and-drop)

ctbindex: Returns the user button index that triggered the script.

context: Returns the context in which the current script was called. Returned values:

0 = manually started

1 = Custom Column

2 = User Button

4 = UDC (User-Defined Command)

8 = CEA (Custom Event Action)

callshift: Returns combined shift buttons value at the moment the current script was called.

The Shift keys have the following values that can be combined in a bit field:

1 = Shift

2 = Ctrl

4 = Alt

menushift: Returns combined shift buttons value at the moment the last popup menu item was clicked.

The Shift keys have the following values that can be combined in a bit field:

1 = Shift

2 = Ctrl

4 = Alt (rather theoretical since Alt will kill any popup menus)

Notes on mousebtn and ctbindex:

- The <get trigger> variable is cleared after the script is processed.
- Currently implemented only for Custom Toolbar Buttons.

Notes on callshift and menushift:

- The value is never reset within the running session, so you can retrieve it even hours after the last menu was clicked.
- Example: `echo "<get trigger menushift>"; // e.g. 3 = Ctrl+Shift was held down`

UsedSpace

`"UsedSpace", [item], [flags]`

Lets you retrieve the used space on a drive.

Syntax analog to "freespace" (see above).

UserName

`"Username", [(unused)]`

Returns the name of the user associated with the current thread.

UserRole

`"Userrole", [(unused)]`

Returns the user's membership (or his security role in the current process) in one of the following security groups: Admin, User, Guest, Power User. If any other groups apply then "Other" is returned.

UTCoffset

`"UTCoffset", [(unused)]`

The offset in minutes of the local time from the UTC (Coordinated Universal Time).

VisualFilter

`"VisualFilter", [pane=a]`

Returns the currently active visual filter (or nothing if none).

For the optional "pane" argument see "BytesSelected" above.

View

`"View", [pane=a]`

Returns the index of the current list view.

For the optional "pane" argument see "BytesSelected" above.

Possible returns and their meanings:

0 = Details

1 = Details with Thumbnails#1

2 = List

3 = Small Icons

4 = Thumbnails #1

5 = Thumbnails #2

6 = Thumbnails #3

7 = Large Icons

8 = Small Tiles

9 = Large Tiles

XYcopy_Pending

`"XYcopy_Pending", [(unused)]`

The number of file operations to be handled in the background (by XYcopy) that are not yet completed.

5.10 Startup Settings

Using Startup.ini

Here is a way to control the Application Data Path (ADP) used by XYplorer and thus avoid any issues with UAC (User Account Control). You can set ADP to any location you like by manually editing the file "Startup.ini" in the application path.

The format of Startup.ini is that of INI files, and it has only one section and one key. Here's an example with a hard-coded application data path on drive E:

```
[Appdata]
Path=E:\XYplorer\appdata
```

Here's an example with a soft-coded application data path using an environment variable. It is resolved depending on the current user account:

```
[Appdata]
Path=%appdata%\XYplorer64
```

You may as well state a path relative to the application path (where XYplorer.exe is located), for example:

```
[Appdata]
Path=appdata
```

The following files and folders are written to and read from ADP:

```
Catalogs <DIR>
  catalog.dat
FindTemplates <DIR>
NewItems <DIR>
Panes <DIR>
Scripts <DIR>
Thumbnails <DIR>      (configurable)
action.dat
fvs.dat
ks.dat
lastini.dat
servers.ini
tag.dat
udc.dat
XYplorer.ini (and other, alternative INIs)
```

Naturally, if you change the ADP you will have to move your current application data to the new path if you want to continue working in the same state as you left the application before changing the ADP.

Tip: If you still get an UAC prompt at startup then look on the "Compatibility" tab of the properties for

the xyplorer.exe file and/or the shortcut you're using to start XYplorer. If the "Run this program as an administrator" box is checked, then uncheck it to avoid the UAC prompt.

Tip: You can set the ADP also using a command line switch! See Command Line Switches.

Redirecting Startup.ini

There is a way to read the startup settings from a file other than Startup.ini. Add this section to the beginning of Startup.ini (the path is an example):

```
[Redirect]
Path=\\ANDROMEDA_CORP\XYplorer\StartupCentral.ini
```

If this key is found, and if the file exists and can be read, the rest of this Startup.ini file is ignored and all startup settings are read from the new file. This allows managing a multi-user setup from one central location.

5.11 Variables

Index

XYplorer Native Variables
Simple Variables
Date Variables
Variable <curver>
Variables <selitems ...> and <allitems ...>
Variable <get ...>
Variable <pad ...>
Variable <perm ...>
Variable <prop ...>
Variable <file filename>
Environment Variables

XYplorer Native Variables

These variables that can be used in User-Defined Commands, Scripting, Batch Rename, and other contexts. Note that all of these variables are read-only, i.e. you cannot deliberately alter their values.

Simple Variables

```
<curpath>    real current path (unslashed)
          e.g. C:\Dokumente und Einstellungen\Donald\Desktop
               C:\Dokumente und Einstellungen\Donald\Desktop\New Stuff
               C:
<curpath_s>  special current path (if applicable, else = <curpath>)
          e.g. Desktop
               Desktop\New Stuff
               C:
<curpath_dos> current path in DOS format (8.3)
<curpathprev> previous path of current pane, unslashed
<curtab>     location term of the current tab, e.g. %USERPROFILE%
<curfolder>  name of the current folder (without path)
<curitem>    current (selected & focused) list item (full path)
<curitemprev> previous current (selected & focused) list item (full path)
```

<curitem_dos> current item in DOS format (8.3)

<curitempath> current (selected & focused) list item's full path

<curbase> base-name of the current list item

<curext> extension of the current list item

<curname> name (base.ext) of the current list item

<cursize> file size in bytes of the current list item

<curver> version of the current list item (so-called "fixed file version" format, eg "23.60.0.5")

<curvers> version of the current list item (so-called "string file version" format, eg "23.60.0005")

<curlen> length in characters of the current list item

<thispath> path of the active pane in the way it is current displayed, i.e. instead of a real path it can also be a special path like "Documents\Songs"

<otherpath> path of the inactive pane in the way it is current displayed, i.e. instead of a real path it can also be a special path like "Documents\Songs"

<focitem> focused item in tree (if tree has focus) or list (else);

note that, contrary to <curitem>, <focitem> will also

return the focused item if it is not selected

<selitem> the first selected list item, no matter which item is focused

<selitems> space-separated list of all currently selected list items in quotes

... see below for options!

<allitems> space-separated list of all currently listed items in quotes

... see below for options!

<datem yyyymmdd_hhnnss> modified date/time of the current list item

<date yyyymmdd_hhnnss> date/time now

... see below for more on date variables!

<xy> XYplorer executable path and name

<xyexe> XYplorer executable name

<xypath> XYplorer application path

<xyparent> parent of XYplorer application path

<xydata> XYplorer data path

<xycatalogs> XYplorer catalogs path

<xythumbs> XYplorer thumbnails cache path

<xyicons> XYplorer icons path

<xypaper> XYplorer Paper Folders path

<xypane> XYplorer active pane data path

<xypane1> pane 1 data path

<xypane2> pane 2 data path

<xytagdat> XYplorer current tags database path/file

<xytagdatfile> XYplorer tags database file (without path)

<xytagdatdef> XYplorer default tags database path/file

<xynewitems> XYplorer new items path

<xyscripts> XYplorer scripts path

<xyini> current XYplorer INI file (without path)

<xyac> full path of the actually used Admin.ini file

<xydrive> XYplorer drive (unslashed), e.g. C: or \\Server\Share

<xyver> XYplorer version

<xysource> "VB" or "TB" depending on the language in which the source code was written.

<xybitness> "32-bit" or "64-bit" depending on the bitness of the executable (XYplorer.exe)

<xybitness n> "32" or "64" depending on the bitness of the executable (XYplorer.exe)

<path 1> current path of pane 1

<path 2> current path of pane 2

<path a> current path of active pane

<path i> current path of inactive pane

Scripting only

<clipboard> Current clipboard contents; if files are on the clipboard then their path/names are returned as text, one per line.

<clp> Shorthand for <clipboard>.

<drop> A synonym for <get drop>.

<dropsel> It's a combination of <drop> and <selitems <crlf>>. If there is a drop, it resolves to the dropped items, otherwise

it resolves to the currently selected list items. Useful for scripted bookmark buttons.

<taggeditem> Resolves to the item (full path) that is being tagged via clicking a column cell.

<taggedcolumn> Resolves to the column caption that is being tagged via clicking a column cell.

<hwnd> Handle of the main application window.

Scripting with Custom File Associations enabled only (previously called Portable File Associations)

<pfaitem> The item (full path) the opening application or script in a Custom File Association is associated with.

<pfaitems> The items (full path) that are being opened via a Custom File Association.

Constants

<crlf> Carriage Return Line Feed (0x0D0A)

<cr> Carriage Return (0x0D)

<lf> Line Feed (0x0A)

<space> Space character (0x20)

<tab> Tab character (0x09)

Note that <crlf>, <cr>, <lf>, <space>, and <tab> take an optional count parameter for repetition. Example:

```
::echo "Catch<crlf 22>22";
```

User-Defined Command "Open With" only

<items> space-separated list of the currently selected list items in quotes

<item1> <item2> the first two of all selected items

```
<base>      base (= base name only, no path and no extension) of currently processed item
            (only meaningful when calling multiple instances)
            You can use it like this:
              "winzip32" -e <items> "<base>"
            This will extract all currently selected archives, each into a
            folder auto-named to the base-name of the respective archive.
<title>     title (= base and extension) of currently processed item
            (only meaningful when calling multiple instances)
            You can use it like this:
              "winzip32" -a "<title>.zip" <items>
            This will add all currently selected items, each to an archive
            auto-named to the title of the respective item.
```

Date Variables

Without any format argument, the format used will be the general system date format:

```
<date>      current date
<datem>     modified date of the current list item
<datec>     created date of the current list item
<datea>     accessed date of the current list item
<dateexif>  EXIF date of the current list item (JPG and RAW formats)
<datemedia> Media Created date of the current list item (audio and video formats)
```

Use a format argument to specify a user-defined format:

```
<date yyyy>
<datem yyyy-mm-dd>
<datec yyyy-mm-dd hh:nn:ss>
<datea yymmdd_hh_nn_ss>
```

Use argument "List" to apply the current file list date format:

```
<date List>, <datem List> ...
```

Date Shifting: Pass an optional extra argument to shift the date to the past or the future by a certain number of time units. Supported units are y = years, m = months, w = weeks, d = days, h = hours, n = minutes, s = seconds:

```
<datem +8h yyyymmdd_hhnnss>     //a file's modified date + 8 hours
<datec -8d yyyymmdd>            //a file's created date - 8 days

<date -70000d yyyy-mm-dd>       //70000 days before now
<date +2y yyyy-mm-dd>           //2 years from now
<date +7s yyyy-mm-dd hh:nn:ss>  //7 seconds from now
```

Special Date Arguments:

```
<date ISOWeek> will resolve to "2008-W25-5".
```

Variable <curver>

You can add a textual context to <curver> that only will be returned when an actual file version is present. This allows for better formatting control. The general syntax is <curver|text*text>, where * is the place holder for the actual version number. If you omit * then text is by default prefixed to the version info proper.

Examples, each resolved for two items, one with version info, and one without version info:

(1) `echo "<curname><curver|, v>";`

XYplorer.exe, v24.0.0.706

History.txt

(2) `echo "<curname><curver| [ver *]>";`

XYplorer.exe [ver 24.0.0.706]

History.txt

(3) `echo "<curname> <curver>";`

XYplorer.exe 24.0.0.706

History.txt

Variables <selitems ...> and <allitems ...>

The native variable <selitems...> returns the currently selected list items with full path. You can pass a separator (one or more characters; the space character is not possible here). If a separator is passed then the individual items are not quoted. Examples:

```
text "<selitems>";          //items are quoted, separated by space
text "<selitems ;>";        //items are separated by ;
text "<selitems |>";        //items are separated by |
text "<selitems //>";       //items are separated by //
text "<selitems <crlf>>";   //items are separated by CRLF
```

The native variable <allitems...> works exactly like <selitems>, but returns all list items, not only the selected ones.

Variable <get ...>

The native variable <get ...> corresponds to the scripting function get().

Syntax: `<get info [param1]>`

Examples:

```
text "<get specialfolder 9>"; //returns sendto folder
```

Quoted Parameters:

Note that <get ...> supports quoted parameters in some contexts. This is useful when passing parameters containing spaces (because the space character also serves to separate the parameters in this context). The parameters themselves are unquoted after parsing. For example, this returns "Dual Pane" when Dual Pane is enabled, and "Single Pane" when it's not:

```
text <get #800 "Dual Pane" "Single Pane">;
```

You can even pass embedded quotes by doubling them:

```
text <get #800 """Dual Pane""" "Single Pane">;
```

Tip: There is an special shorthand for <get alias [name]>: <@[name]>. So instead of <get alias icons> you can do <@icons>.

Variable <pad ...>

The native variable <pad ...> can be used to pad a string with a number of characters at a specified position until the string reaches a specified length.

Syntax: <pad length [char] [switches]>

length	Final length of the padded string, or of the padded number within the string (depends on the switches).
char	Character to pad with. If missing it defaults to " " (space). Note that strings longer than one character are supported, although it's hard to see a good use for this.
switches	b = only work on the base of a filename (exclude the extension) n = pad the first (left-most) number in the string nr = pad the last (right-most) number in the string

Examples:

```
echo "ab<pad 12>cd";          //ab        cd          --> pad char defaults to space
echo "ab<pad 12 ->cd";        //ab--------cd
echo "<pad 8 0>98.mkv";       //0098.mkv
echo "<pad 8 0>64.mpeg";      //064.mpeg
echo "<pad 4 0 b>98.mkv";     //0098.mkv
echo "<pad 4 0 b>64.mpeg";    //0064.mpeg
echo "64<pad 4 0 b>.mpeg";    //6400.mpeg
```

Examples for number padding:

```
echo "<pad 4 0 n>Text 1.";      //Text 0001.
echo "<pad 4 0 n>Text 12.";     //Text 0012.
echo "<pad 4 0 n>Text 123.";    //Text 0123.
echo "<pad 4 0 n>Text 12 34.";  //Text 0012 34.
echo "<pad 4 0 nr>Text 12 34."; //Text 12 0034.
```

Examples of use in Batch Rename (note that the asterisk is necessary, though its position is irrelevant):

1) Pattern: <pad 4 0 n>* = (only the left-most number is padded)

Old Name:	New Name:
image_1.jpg	image_0001.jpg
image_23.jpg	image_0023.jpg
image_345.jpg	image_0345.jpg
image_4567.jpg	image_4567.jpg

2) Pattern: <pad 4 0 nr>* = (only the right-most number is padded)

Old Name:	New Name:
24h-image_1.jpg	24h-image_0001.jpg
24h-image_23.jpg	24h-image_0023.jpg
24h-image_345.jpg	24h-image_0345.jpg
24h-image_4567.jpg	24h-image_4567.jpg

3) Pattern: <pad 4 0 bnr>* = (only the right-most number in the base is padded)

Old Name:	New Name:
1.FC 1 vs FC Porto.mp4	1.FC 0001 vs FC Porto.mp4
1.FC 20 vs FC Porto.mp4	1.FC 0020 vs FC Porto.mp4
1.FC 300 vs FC Porto.mp4	1.FC 0300 vs FC Porto.mp4

Variable <perm ...>

Permanent variables can be accessed from anywhere in the app using the `<perm ...>` meta variable. For example:

```
perm $xy = "XYplorer"; echo "Perm test: <perm xy>.";  //Perm test: XYplorer.
```

Once the permanent variable is defined you can always access it wherever XY native variables are supported, e.g. as path in the address bar:

```
C:\Users\%user%\AppData\Roaming\<perm xy>\
```

If a permanent variable does not exist, `<perm ...>` is not resolved:

```
echo "Perm test: <perm grrg>."; //Perm test: <perm grrg>.
```

So here you have user defined global variables that persist across sessions.

Tip: You can review and change your permanent variables at any time using the "Scripting | Permanent Variables" command.

Variable <prop ...>

The native variable `<prop ...>` corresponds to the scripting function property(). It returns any available Shell Property value of the current item. See property() for details and caveats.

Syntax:

```
<prop property_index_or_name [format_template]>
```

Example scripts:

Retrieve Size, Type, and Version of the current item (under XP):

```
echo "<prop #1>, <prop #2>, <prop fileversion>";
```

Suffix " n" to the property index return the (locale aware) name of the property:

```
echo "<prop #1 n>, <prop #2 n>, <prop fileversion>";
```

Usage with Rename

Append the file version to the name:

```
rename b, '*-<prop fileversion>', p;
```

Append width and height to the name (images only; some videos work as well):

```
rename b, '*-<prop ImageX>x<prop ImageY>', p;
```

Append camera model to the name (photos only, of course):

```
rename b, '*-<prop CameraModel>', p;
```

Usage in Status Bar Template

To have the camera model of the current file on the status bar, use this template in Configuration | Colors and Styles | Templates | Status Bar:

`<curname>, Camera: <prop CameraModel>` --> Pacific.jpg, Camera: NIKON D80
`<curname><prop , Camera:CameraModel>` --> (cooler version: only adds ", Camera: [...]" when a value is returned)

Alternatively you can pass the "n" (for "name") switch to return the original (locale aware) name of that property:

`<prop #hash.md5 n>` --> MD5: 9e156af7feda895cebc70d42256f819a

For numbered shell properties it works just the same:

`<prop Boss:#10>` --> Boss: Donald
`<prop #10 n>` --> Owner: Donald

Optional Field Prefix

You can optionally prefix a name of your choice (it can even have spaces) to the property selector, separated by ":". It will be prefixed to the result (only if there is a result, that's the cool part) separated by colon-space (": "):

`<prop Size:#image.dimensions>` --> Size: 123 x 456

```
<prop The Size:#image.dimensions> --> The Size: 123 x 456
```

Using the Format Template

You can pass an optional format template in the argument after the property selector. The * is a place holder for the value (if missing the value is simply appended to the template). The template must be wrapped in single or double quotes if a space is included:

```
echo <prop #image.dimensions 'Size: * px'>; //Size: 1024 x 460 px

echo <prop #image.dimensions "Size: * px">; //Size: 1024 x 460 px

echo <prop #image.dimensions "Size: ">; //Size: 1024 x 460
```

Using the Variant <propt ...>

There is an alternative form `<propt ...>` which will retrieve the properties from the link target if there is any. For example, if "rocky-3.jpg.lnk" is selected:

```
echo <prop #image.dimensions>;  //returns nothing (LNK is not an image)

echo <propt #image.dimensions>;  //returns "770 x 513"
```

If "rocky-3.jpg" is selected then both return the same ("770 x 513").

Tip 1: Note that the <prop ...> variable is also supported by Rename Special (regular UI, not only via scripting).

Tip 2: Also supported are XYplorer-specific named properties like #JunctionTarget, #Tags, or #Comment. See property() for details.

Tip 3: Also supported is the asterisk syntax `<prop *width>` as described in property().

Variable <file filename>

The variable <file filename> is resolved to the contents of the file specified by the "filename" argument (resolved relative to the Scripts path). If the file is not found or cannot be opened, the variable is kept unresolved.

This script, for example, shows the contents of the file exclude.txt:

```
text <file exclude.txt>;
```

This Quick Search line uses the <file filename> variable for defining the excluded paths (BTW: /excl supports CRLF as separator, but take care not to mix separators in your switch):

```
?*.jpg /excl=<file exclude.txt>
```

Environment Variables

To see a list of all valid environment variables and their values click menu Help | Environment Variables.

Some of them are XYplorer only:

%rapidaccess%	"Rapid Access" special folder (listing the Rapid Access Folders)
%computer%	"Computer" special folder (listing the logical drives)
%desktop%	"Desktop" special folder
%net%	"My Network Places" special folder
%recycler%	"Recycle Bin" special folder
%personal%	"My Documents" special folder
%desktopreal%	"Desktop" real path (this user)
%personalreal%	"My Documents" real path
%winsysdir%	Windows system directory
%winsysnative%	"System32" path on 32-bit Windows, "SysNative" path on 64-bit Windows
%commonappdata%	All Users application data folder
%commondesktop%	All Users desktop folder
%winver%	Current Windows version number (e.g. 5.1)
%osbitness%	OS bitness (e.g. 64 on a 64-bit Windows)

Note that with environment variables case is ignored: %desktop% = %DESKTOP%.

Note that all paths are returned without backslash.

5.12 Tweaks

What's a Tweak?

A "tweak" is a manual change in XYplorer's INI file (usually XYplorer.ini). It's easy to do once you have a text editor and an idea how to do it.

To do it right, you have to know that XYplorer reads the INI at startup and writes it at Exit (unless you unticked "Save Settings on Exit"). So, if you open the INI from a running XYplorer, do your manual editing, save and close the INI, and then close and restart XYplorer in order to read the tweaked INI, it might not work because when you closed XYplorer it has probably overwritten the INI with the old values stored in memory! So the trick is to avoid this. Here's how you can it:

Tweaking the INI file from running XYplorer

(1) Menu File | Save Settings (unless you don't care about the current state)

(2) Menu Tools | Open Configuration File... (will open the INI in your default editor)

(3) In the editor: Do your editing (look up the tweak key and edit its value), save the tweaked INI

(4) Menu File | Restart without Saving (this is the important part)

Done. Now XYplorer should have read the tweaked values.

Why tweak the INI?

Because a couple of XYplorer settings don't have a GUI (they are not featured in the Configuration dialog or elsewhere in the interface). Those settings are called "Tweaks".

Part VI

Registration

6 Registration

6.1 How to purchase a license

XYplorer Is Not Freeware

You may evaluate XYplorer for free for a maximum period of 30 days. If you wish to continue using it after 30 days, you have to register and purchase a license.

The XYplorer License

LICENSE GRANT. As an individual person, you need one license to use XYplorer. This license grants you a non-exclusive and non-transferable right to use XYplorer on any number and kind of devices. As an employer, you need a license for each employee using XYplorer. Floating or concurrent license usage is expressly prohibited.

There are three types of licenses:

(1) The "Lifetime License" entitles one person to use the full version of XYplorer on any number of devices for an unlimited time. The license includes free version upgrades forever.

(2) The "Standard License" entitles one person to use the full version of XYplorer on any number of devices for an unlimited time. The license includes free version upgrades for one year.

(3) The "Rookie License" entitles one person to use the full version of XYplorer on any number of devices for an unlimited time. The license doesn't include free version upgrades.

Volume Discount Prices

There are volume discount prices, site licenses, and corporate licenses. See https://www.xyplorer.com for further details and prices.

Part VII

Support

7 Support

7.1 Contact

Support contact

Address

Cologne Code Company
Alteburger Str. 7
50678 Cologne, Germany

Email

support@xyplorer.com

Homepage

Find the latest version and information here. The site is updated frequently.

https://www.xyplorer.com/

User Forum

Feel free to join the XYplorer User Forum:

https://www.xyplorer.com/xyfc/

Check for upgrades once in a while...

Due to lots of user feedback, XYplorer is developing at a rapid step. If you are a registered user, you might check in once in a while to get your free upgrade (depending on your license).

Part VII

Legal Stuff

8 Legal Stuff

8.1 Disclaimer

DISCLAIMER OF WARRANTY

This program is distributed as is. The author (Donald Lessau) makes no warranty of any kind, expressed or implied, including but not limited to warranties of merchant ability or fitness for a particular purpose, with respect to this software and documentation. In no event shall the author be liable for any damages, including lost profits, lost savings, or any other incidental or consequential damages arising out of the use of or the inability to use this program.

By using the above mentioned program you acknowledge that you have read this disclaimer of warranty, understand it, and agree to be bound by its terms and conditions. You also agree that the limited warranty is the complete and exclusive statement of agreement between the parties and supersede all proposals or prior agreements, oral or written, and any other communications between the parties relating to the subject matter of the limited warranty.

Donald Lessau, Cologne Code Company

8.2 Copyright

XYplorer

Copyright © 1997-2025 Cologne Code Company. All Rights Reserved.

Trademark Information

Products

XYplorer™ is trademarked 2005 by Cologne Code Company.

TrackerV3™ is trademarked 1997 by Cologne Code Company.

Windows® is a registered trademark of Microsoft Corporation.

Features

Image Columns™ is trademarked 2023 by Cologne Code Company e.K.

Joker Tabs™ is trademarked 2023 by Cologne Code Company e.K.

Managed Tree™ is trademarked 2022 by Cologne Code Company e.K.

Recent Location Pins™ is trademarked 2019 by Cologne Code Company e.K.

Mouse Up Show Down™ is trademarked 2018 by Cologne Code Company e.K.

Drag Status Box™ is trademarked 2018 by Cologne Code Company e.K.

Ghost Filter™ is trademarked 2015 by Cologne Code Company e.K.

Paper Folder™ is trademarked 2014 by Cologne Code Company e.K.

Click and Search™ is trademarked 2014 by Cologne Code Company e.K.

Tree Path Tracing™ and Tree Path Tracking™ are trademarked 2012 by Cologne Code Company e.K.

Droppable User Buttons™ is trademarked 2011 by Cologne Code Company e.K.

Mini Tree™ and MiniTree™ are trademarked 2008 by Cologne Code Company e.K.

Portable Openwith Menu™ is trademarked 2008 by Cologne Code Company e.K.

Portable File Associations™ is trademarked 2006 by Cologne Code Company e.K.

Details with Thumbnails™ is trademarked 2006 by Cologne Code Company e.K.

Mouse Down Blow Up™ is trademarked 2004 by Cologne Code Company e.K.

Rich File Operations™, Rich Copy™, and Rich Move™ are trademarked 2000 by Cologne Code Company e.K.

All other trademarks mentioned in this help file are the property of their respective owners.

Index

- # -

(List Column) 339

- / -

/exp 505
/feed 505
/flg 506
/fresh 506
/hwnd 506
/ini 507
/new 507
/path 507
/readonly 508
/restore 508
/script 508
/select 509
/title 509
/user 509
/win 510
/win (extended syntax) 510

- [-

[AccessControl] Section (Admin Settings) 500
[Paths] Section (Admin Settings) 498
[Settings] Section (Admin Settings) 496
[Tags] Section (Admin Settings) 499

- _ -

_ to Spaces 144

- < -

<cc_item> and other <cc_ ...> variables (Custom Columns) 351

- 2 -

2D Border (Preview Border Style) 98

- 3 -

3D Border (Preview Border Style) 98

- 4 -

4 Key Navigation (Address Bar) 296

- 6 -

6 Key Navigation (Address Bar) 295

- A -

A* A*.aaa 144
Aaa Aa.aaa 144
A-B repeat 388
About XYplorer 229
Accelerators 125
Access Control 500
AccessControl (Admin Settings) 496
Accessed (List Column) 339
Accidental drop (Undo) 486
Acronyms 13
Action (Custom Event Actions) 36
Action Log 488
Action Log... 165
Action on click 365
Activate/deactivate fonts on the fly 385
Adaptive colors 51
Add (Previewed Formats) 102
Add Column 359
Add Folder to Toolbar 32, 472
Add Last Tags 198
Add new items at the end of the list 17
Add new tags to tag list 284
Add Tags 284
Add Tags by List... 198
Add Tags... 198
Add to {Zipname Item} 474
Add to {Zipname Parent} 474
Add to Zip Archive... 474
Add to Zip... 142
Adding categories and items to the Catalog 365
Adding to / removing from the current selections 165
Address Bar 292
Address Bar (Auto-Complete Path Names) 42
Address Bar and Toolbar Stacked 226

Address Bar Icon Context Menu 292
Address Bar searches 298
Adjust to OS light/dark mode at startup 46
Admin Settings (Enterprise Edition Only) 496
Advanced Options... (Floating Preview) 431
Affix current date to copy 79
Affix current date to existing 79
Affix last modified date to copy 79
Affix last modified date to existing 79
Age Circles 221
Age Search 406
Aliases 300
Aliases with arguments 300
Aliases... 191
Align to bottom 104
All Free Shortcuts... 469
All List Rows 142
All Menus (Group) 307
All White Space (Glider) 332
Allow Button Set Switching (Toolbar) 304
Allow Custom Keyboard Shortcuts in Preview (Floating Preview) 431
Allow dragging from a background window 32
Allow dragging items by the thumbnail 110
Allow ESC to Close Preview 430
Allow move on rename 23
Allow Multiple Button Rows 226, 227
Allow multiple instances 46
Allow only certain locations (Access Control) 500
Allow panning 110
Allow repeated characters 97
Allow Zombies 448
Allow zombies in the Mini Tree 42
Allowed number of entries in the action log 87
Allowed number of items per logged action 87
AllowedDirs (AccessControl) 496
Also auto-select tabs in the inactive pane 115
Also on Full Row Select 65
Alternate Groups 51
Always autosize the Size column 53
Always keep 1st pane visible 119
Always show folder sizes 17
Always Show Path Column 448
Always Sort Search Results This Way 89
Animated GIFs 386
Append 165
Append to existing file 74
Application Data Folder 11
Application Data Path 10, 11, 510, 850
Applied filters (Find Files tab) 394
Applied Settings 459
Apply (Configuration) 16
Apply (Tags) 392
Apply Changed Tags Automatically 392
Apply color filters to the List 57
Apply color filters to the Tree 57
Apply colors (Tabs) 50
Apply Default Folder View 187
Apply Highlighting (Customize Tree) 214
Apply Last Label 198
Apply list styles globally 53
Apply Previous Folder View 187
Apply tagging to all selected items 65
Apply text colors to the Name column only 57
Apply this Folder View Also To... 187
Apply to all controls (generic icons) 29
Apply to Files Only (Visual Filters and Live Filter Box) 95
Apply zoom (Mouse Down Blow Up) 110
Archive Contents Preview (Hover Box) 346
Archiving file find results 482
Archiving files 484
Arrangement (in menu Window) 226
ASCII file options (Raw View) 391
Ask 79
Ask before merging folders 79
Ask before overwriting read-only files 79
Aspect ratio of images 98
Assign Keyboard Shortcut (UDC) 201
Assume that mapped network drives are available 39
Assume that servers are available 39
Asynchronous copy 75
Attr (List Column) 339
Attributes 340
Attributes Search 406
Audio File Preview 388
Audio preview 110
Auto-Cleaning 23
Auto-Complete 292, 295
Auto-Complete Path Names 42
Auto-complete recently used items 29, 42
Auto-create any missing folders 119
Auto-Elevation (External Copy Handlers) 84
Autofit the width of the Name column 53
Auto-increment filenames on collision 60
Automatic Redundancy Removal 482
Automatic Rich File Operations 482
Automatically Apply Default Folder View 187
Auto-optimize tree 17
Autoplay 98
Auto-Refresh 29, 92, 180
Auto-refresh tags 65
Auto-replace invalid characters 23

Auto-rotate preview 98
Auto-rotate thumbnails 104
Auto-Save Changes 459
Auto-save tabsets on switch 115
Auto-Scroll to End (Floating Preview) 431
Auto-Scroll to End (Raw View) 391
Auto-select first item 17
Auto-select first match (Live Filter Box) 94
Auto-select matching items 119
Auto-select tabs on drag-over 115
Autosize Buttons (Toolbar) 304
Autosize Columns (Customize List) 214
Autosize columns maximum width 53
Autosize Name column minimum width 53
Autosize Name column right margin 53
Auto-Sync Select 209
Availability Status 341

- B -

Back to English 122, 229
Background Jobs Button 378
Background Jobs Dialog 378
Background Jobs... 75, 378
Background Processing 75
Backup 484
Backup a Catalog Category 165
Backup Application Data Folder... 162
Backup Here 476
Backup Operations 77
Backup settings on save 46
Backup To 154
Backup To... (Dialog) 165
Bars (Size Column) 219
Basic info to CSV 427
Batch files for search jobs 439
Batch Move 145
Batch Rename... 144
Beep 125, 469
Best Match Algorithm 459
Beta channel 46
Binary file options (Raw View) 391
Binary Sort (Sort method) 23
Bitmap Fonts Preview 385
Bitness (File Info Tips) 70
Bold 325
Bookmark Buttons 310
Bookmark Toolbar Buttons 310
Boolean Logic (Color Filters) 270
Border style options 98
Borders 51
Both Edges (Glider) 332
Boxed Branch 195
Branch View 180, 441
Branch View Configuration 92
Breadcrumb Bars 325
Breadcrumb... 191
Browse for Network Server... 214
Browsing Tabs 319
But only in network locations 29
Button Captions Font (Toolbar) 304
Button Push Effect (Toolbar) 304
Button Sets 317
Button Size 304
Button Text Font (Toolbar) 304
Buttons In Catalog 365
Buttons position 115
Buy a license 864
Bypassed Tree 328
Bytes (file size display options) 214

- C -

Cache folder sizes 17, 19
Cache network servers 39
Cache path 104
Cache Search Results 89
Cache specific icons 29
Cache thumbnails on disk 104
Caches (Submenu)
  Create Missing Thumbnails 190
  Refresh Icons 190
  Refresh Selected Thumbnails 190
  Refresh Thumbnails 190
  Update New Items Menu 190
Calculate folder size by clicking the donut 219
Calculate Folder Sizes 180, 219
Cancel 79
Cancel (Configuration) 16
Cancel Job 378
Caption lines 104
Catalog 365
Catalog as Toolbar 365
Catalog backups 165
Catalog First 226
Categories (Previewed Formats) 102
CD Audio 388
CD-ROM drives (Items in Tree and List) 21
Cell Context Menu 337
Centered 110
CFA 512
Change file dates 381

Changing file attributes 382
Character to replace invalid characters in dropped messages 60
Check beforehand whether there is enough space 77
Check existence of subfolders in tree 17
Check for language updates at startup 46
Check for Subfolders 325
Check for Updates 229
Check for updates at startup 46
Check Network Locations for Content-Based Folder Icons 463
Check Sleeping Drives for Content-Based Folder Icons 463
Checkbox Selection (Customize List) 214
Chevrons 325
Circles (Date Column) 220
Circles (Size Column) 219
Classic directory dump 74, 427
Clear Clipboard 165
Clear Folder Size Cache Everywhere 19
Clear Folder Size Cache Here 19
Clear Folder Size Cache... 219
Clear history on exit 46
Clear tabs on exit 46
Click and Search 373
Click and Search: Tags 365
Click and Search:Labels 365
Click and Tag: Labels 365
Click and Tag: Tags 365
Click to set colors 50
Clicking the Status Bar 376
Clipboard Markers 53
Clipboard Viewer 165
Clone (UDC) 201
Close Dialog and Trigger Command 469
Close Preview by ESC 430
Close this Pane 209
Closing the Preview 430
Closing the tabs 319
Cloud Availability Status 341
Code pages in text preview 384
Color Coding 266
Color Filter Types 273
Color Filters 266
Color Filters (Configuration) 57
Color Filters by a list of file attributes 275
Color Filters by file age 277
Color Filters by file attributes 274
Color Filters by file date 276
Color Filters by file properties 279
Color Filters by file size 275
Color Filters by filename length 279
Color Filters by folder name 274
Color Filters by name 273
Color Filters by tags 281
Color Filters Overview 266
Color Histogram (Floating Preview) 431
Color tint (0 is neutral) 51
Color-code empty folders 279
Colored lines 53
Colored Menu 325
Colors 50
Column Drag 335
Column Drag (Customize List) 214
Column headers 428
Column Layouts 309
Column Sets 309
Column Width by Keyboard 339
Column-Click Tagging 65
Columns 53
Columns (Submenu)
- Add Column 182
- Autosize Columns Now 182
- Grow Name Column 182
- Line Number Column Width 182
- Load Column Layout... 182
- Save Column Layout As... 182
- Set Line Number Column Width 182
- Show All Columns 182
- Show Columns... 182
- Shrink Name Column 182
- Skip Custom Columns 182

Columns can be hidden 335
Columns can be reordered by dragging 335
Columns in the File List 339
Column-wise Selection Filter 176
Combined Searches 373
Command IDs (Create Listing) 229
Command IDs on Menu 229
Command Line Interpreter 60
Command Line Switches 505
Command Prompt 472
Comment (List Column) 339
Comment Search 407
Comment... 198
Comments 65, 284, 392
Comments (Quick Search) 247
Comments as Captions (Click and Search) 373
Comments in New Items 174
Comments shared in a network 291
Common (Favorites menu) 195
Compact File Info 142
Compare (submenu) 165

Compare Current File on Both Panes 165
Compare Current File with File in Clipboard 165
Compare Current File with Previous File 165
Compare directory trees 422
Compression ratio 386
Configuration 16
  Color Filters 57
  Colors 50
  Controls & More 42
  Custom Columns 70
  Custom Event Actions 36
  Dual Pane 119
  Features 121
  File Info Tips & Hover Box 70
  File Operations 75
  Find Files & Flat View 89, 93
  Fonts 59
  Menus, Mouse, Usability 32
  More Colors 51
  Mouse Down Blow Up 110
  Preview 98
  Previewed Formats 102
  Refresh, Icons, History 29
  Report & Data 74
  Safety Belts, Network 39
  Shell Integration 120
  Sort and Rename 23
  Startup & Exit 46
  Styles 53
  Tabs 115
  Tags 65
  Templates 60
  Thumbnails 104
  Tree and List 17
  Undo & Action Log 87
Configuration of Custom Columns 351
Configuration... 214
Configure Custom Column dialog 351
Configure Extra Column [#]... 287
Configure Thumbnails 180
Confirm copy and move operations 39
Confirm copying tags 65
Confirm delete operations 39
Confirm drag and drop 39
Contact 866
Content Dupes 422
Content Search 411, 412
Content-Based Folder Icons 463
Contents (Properties tab) 380
Contents and Index (Help menu) 229
Context menus (List) 342
Context menus (Tree) 328
Continue 79
Continue All 79
Control Panel... 214
Control Selector (Color Filters) 267
Controls 365, 376, 380
Convert to ASCII 144, 380
Copy 165
Copy All Items in Category 365
Copy as folders (Junctions) 79
Copy as junctions (Junctions) 79
Copy Cheat Sheet 469
Copy Containing Folder(s) 426
Copy Data 337
Copy File Age 337
Copy File Dates 337
Copy File Name 337
Copy File Size 337
Copy folder structure 476
Copy Handler 75
Copy Here 156, 476
Copy Here As... 156
Copy Here to New Subfolder... 156
Copy Here with Current Date 156
Copy Here with Increment 156
Copy Here with Last Modified Date 156
Copy Item 292
Copy items from source to target (Sync Folders) 363
Copy items with their folder structure 482
Copy items with their source location structure 482
Copy Junction Target Name 342
Copy Name 325, 380
Copy Name with Path 380
Copy Original (Floating Preview) 431
Copy Path 292, 325
Copy Path (Custom Context Menu Commands (List)) 472
Copy Path (Custom Context Menu Commands (Tree)) 472
Copy paths to the clipboard with a trailing slash 42
Copy Preview (Floating Preview) 431
Copy Real Path 292, 325
Copy selected items here (Glider) 332
Copy Shortcut Target Item 342
Copy Shortcut Target Name 342
Copy Special Path 325
Copy Special Path (Custom Context Menu Commands) 472
Copy Status to Clipboard (Status Bar) 376
Copy Tab to Other Pane 209
Copy tags on backup and sync operations 65
Copy tags on copy operations 65

Copy To 154
Copy to Other Pane 209
Copy To... (Dialog) 165
Copy tree 476
Copy/Move Here As... 476
Copy/Move Here to New Subfolder 476
Copy/Move Here With Current Date 476
Copy/Move Here With Last Modified Date 476
Copy/Move Here With Number 476
Copy/Move Here with Path... 476
Copy/Move items with their folder structure 482
Copy/Move to Other Pane 472
Copying files
comparative overview 480
Copying from the Catalog 365
Copying Icons 142, 381
Copying lines (Raw View) 391
Copyright 869
Corporate License 864
Create all thumbnails at once 104
Create all thumbnails now 104
Create Backup.log 476
Create Branch(es) Here 478
Create Folder(s) Here 478
Create log file 79
Create Missing Thumbnails 104
Create New Subfolder Here 472
Create Shortcut to this Configuration... 162
Create Shortcut(s) 142
Created (List Column) 339
Creating Reports 427
CSV Field Separator 74
CSV Files 427
Ctrl+Wheel scrolls through the list views 32
Cumulative undo 486
Cumulative undo/redo 87
Current date 428
Current Folder (Report tab) 427
Current List (Report tab) 428
Current List Columns (Catalog Context Menu) 365
Current Mini Tree (Catalog Context Menu) 365
Custom column definitions 70
Custom Columns 351
Custom Columns (Configuration) 70
Custom Context Menu Commands 472
Custom Context Menu Commands (List) 472
Custom Context Menu Commands (Tree) 472
Custom Copy Blacklist 77, 79
Custom Copy Operations 77
Custom Event Sounds 42
Custom File Associations 512
Custom File Icons 463
Custom items in context menu 32
Custom Menu 325
Custom Move 77
Custom Popup Menu 523
Custom Status Bar Info 64
Custom tab icons 320
Custom Toolbar Buttons 311
Customize Age Graphics 221
Customize File Associations 512
Customize File Associations... 214
Customize File Icons 463
Customize File Icons... 214
Customize Keyboard Shortcuts 469
Customize Keyboard Shortcuts... 214
Customize List 214
Customize Toolbar... 214
Customize Tree 214
Customizing the Hamburger 325
Customizing the Toolbar 304
Cut 165
Cut and Add (Batch Rename) 144
Cycle Background Color (Floating Preview) 431
Cycle first day of the week 417
Cycle tabs in recently used order 115
Cycle Transparency Background (Floating Preview) 431
Cyclic Navigation (Floating Preview) 431

- D -

Dark Mode 180
Database Check... 65
Date affix 60
Date Dupes 422
Date Format 220
Date format in action labels 87
Date Picker 417
Date Search 406
Date Variables 165
<dateexif> 855
<datemedia> 855
Date Shifting 855
Date variables in New Items 174
Date/time as file name suffix 74
Day Groups (Sorting) 339
DBCS encoding 384
Debugging scripts 207
Default (Customize Keyboard Shortcuts) 469
Default action on drag and drop to different drive 120
Default action on drag and drop to same drive 120
Default branch view type 92
Default Event Sounds 42

Default File Manager 120
Default Folder View 459
Default name to "[Current folder].txt" 74
Default Profile Path (Admin Settings) 498
Default Tab 319
Default to repeat action on collisions 77
Default to tree-like sort order 92
Define this Folder View as Default 187
Delay before a dragged-over tab is auto-selected 115
Delay before filter is applied (Live Filter Box) 94
Delete 142
Delete (No Recycle Bin) 142
Delete (Skip Locked) 142
Delete (Wipe) 142
Delete File (Floating Preview) 431
Delete items in target that have no matches in source (Sync Folders) 363
Delete Junction 342
Delete Long 142
Delete on key up 39
Delete to recycle bin 87
Delete to recycle bin (if possible) (Sync Folders) 363
Dependencies 10
Depth-Limited Search 237
Deselect All 165
Desktop folder (Items in Tree and List) 21
Desktop location 328
desktop.ini 335
Details 180
Details with Thumbnails 180
Details... 378
Detect portable devices 29
Different Extension (Dupes) 422
Digital Cameras 492
Digital Signatures 10
Dimmed icons 53
Directional formatting codes protection 39
Directory Print 427
Dirty Tags (Color Filters) 281
Disable certain features (Access Control) 500
Disable Check for Updates 496
DisabledFeatures (AccessControl) 496
Disallow certain file operations (Access Control) 500
Disallow certain locations (Access Control) 500
Disallow delete by key in folder tree 39
Disallow left-dragging from file list 39
Disallow left-dragging from folder tree 39
DisallowedDirs (AccessControl) 496
DisallowedDirsExcept (AccessControl) 496
DisallowedOperations (AccessControl) 496
Disclaimer 868
Disconnect Mapped Network Drive... 214
Display Hash Values 142
Display Modes 338
Display options for Raw View 391
Display Tabs as spaces 98
Do this also for the next collisions 77, 79
Document Preview 385
Documentation of whole directories 427
Documents folder (Items in Tree and List) 21
Don't save history 46
Don't save tabs 46
Donut (Size Column) 219
DOS 8.3 format 142
DOS box through Address bar 300
DOS-Box 472
Double Size (Floating Preview) 431
Down 191
Downloads folder (Items in Tree and List) 21
Drag and Drop 33, 120
Drag and drop confirmation 39
Drag and drop context menu 476
Drag and Drop within file list 335
Drag Status Box 39
Dragging from a background window 32
Draw background colors as rounded rectangles 57
Draw background colors as wide as the column 57
Draw background colors in distinctive shapes 57
Draw hidden icons ghosted 29
Draw selected list icons dimmed 29
Drawing Bars 354
Drawing Circles 353
Drive Bars and Drive Buttons 307
Drives... 191
Drop Menu on Hover 325
Drop on a Script File 537
Drop on Custom Toolbar Buttons 311
Drop on Zip 479
Drop Stacks 452
Drop Text To File 479
Dropdown Button on the Left 292
Droppable User Buttons 316
Dropped messages 60
Dropping onto the catalog 365
Dropping onto the Scripts 365
Dual Locations 195
Dual Pane 119, 191, 209
Dual startup path 46
Dual Startup Path (Command Line Switch) 511
Dupe Removal (Virtual Folders) 454
Dupes 422

Duplicate (Submenu) 156
Duplicate a file 476
Duplicate File Finder 422
Duplicate files with folder tree structure 482
Duplicate Folders 422
Duplicate Images 422
Duplicate Item (Catalog Context Menu) 365

- E -

Edit (Previewed Formats) 102
Edit Accessed Date... 338
Edit Box Keyboard Shortcuts 133
Edit Clipboard and Paste (Toolbar) 309
Edit Clipboard... 165
Edit Created Date... 338
Edit Filter... (Visual Filters) 94
Edit Folder View 187
Edit Folder View dialog 459
Edit Item Names... 144
Edit menu 165
Edit Modified Date... 338
Edit Orphans (All Drives)... 65
Edit Orphans (Fixed Drives Only)... 65
Edit Script... (Glider) 332
Edit Search... 93
Edit Tagged Items... 65
Edit Tags by List... 198
Edit Tags... 198
Edit User Button 311
Edit User Code Pages 384
Email (author) 866
Embedded Icons 381
Empty Folder 472
Empty Folder [folders only] 472
Empty Folder Icons 463
Empty List Message 336
Empty Paper Folder 186
Empty Recycle Bin... 214
Enable background processing 75
Enable blow ups on file icons as well 110
Enable color filters 57
Enable extended pattern matching 89, 93
Enable extended pattern matching (Live Filter Box) 94
Enable extended pattern matching (Visual Filters) 94
Enable Folder View Settings 187
Enable navigation keys (Live Filter Box) 94
Enable Server Mappings 385
Enable server mappings (Web Preview) 98
Enable smart Boolean query parsing 89
Enable surround selection 42
Enable type ahead find 97
Enable zoom by Ctrl+mouse wheel 59
Encoding 74
Encoding (File Info Tips) 70
End of List (Customize Toolbar) 304
Enlarge Preview tab 384
Enterprise Edition 496
Environment Variables 229, 860
Environment variables usage 303
Environments (= Tabsets) 323
ESC key 125
Even on exit without saving 87
Event (Custom Event Actions) 36
Event Sounds 42
Exact matches in Visual Filters 250
Exact Matching (Color Filters) 270
Examples for implicit path matching 243
Examples for Quick Searches 248
Exclude file extension from initial selection 23
Exclude hidden files and folders (Backup, Custom Copy) 79
Exclude hidden files and folders (Sync Folders) 363
Executing DOS commands 300
Existence Check (Virtual Folders) 454
Exit 164
Exit without Saving 164
Expand in tree 46
Expand tree nodes on browse 17
Expand tree nodes on drag-over 17
Expand tree nodes on single click 17
Expansion 472
Explicit Save Only 448
Export Category (Catalog) 365
Export Local Tags 198
Ext (List Column) 339
Extended compatibility for clipboard and drag and drop 120
Extended File Info 142
Extended info to CSV 427
External Copy Handlers 84
Extra Column Types 287
Extra Columns 287
Extra fields (File Info Tips) 70
Extra Large Buttons (Toolbar) 304
Extra Large Icons (Touchscreen Mode Button) 308
Extra Tags 287
Extract Here 142, 474, 476
Extract text (Raw View) 391
Extract to {Folder} 474
Extract to Other Pane 474
Extracting Icons 381

- F -

FastCopy Integration 84
Favorite Files 195
Favorite Folders... (Move/Copy/Backup To) 154
Favorite Live Filters 263
Favorites 195
Favorites menu 195
Favorites menu by right-clicking Tree white space 328
Features 121
File associations 512
File Attributes 340
File attributes (Color Filters) 274
File Comments 284
File Find Tips 426
File formats supported by Audio/Video preview 388
File Labels 284
File List (Shell context menu) 32
File menu 142
File operation progress modeless 75, 86
file protocol 302, 365
File Special (Submenu) 142
File Tags 284
File version 383
Filename affixes 60
Filename templates 60
Files scanned 376
Files without extensions (Previewed Formats) 102
File-specific Metadata 383
Filter (Auto-Complete Path Names) 42
Filter Syntax (Visual Filters) 250
Filter-As-You-Type 263, 264, 336
Filter-As-You-Type right in the file list 263
Filtered Branch 441
Find as you type 97, 336
Find Bar 343
Find by Tag List... 198
Find By Tags 284
Find by Tags... 198
Find duplicate Files 422
Find Duplicate Images 422
Find Files 165, 393
Find Files by Age 406
Find Files by Attributes 406
Find Files by Columns 410
Find Files by Contained Characters 412
Find Files by Contents 411
Find Files by Custom Columns 351
Find Files by Date 406
Find Files by Extra Tags 287
Find Files by Meta Properties
  Canonical properties 414
  Meta Properties Search 414
  Properties Search 414
  Windows canonical properties 414
Find Files by Multi Field Search
  Boolean RegExp 413
  Field Type Inheritance 413
  Multi Field Search 413
Find Files by Properties 410
Find Files by Size 405
Find Files by Size Column 410
Find Files by Soft Columns 359
Find Files by Tags (Labels, Tags, Comments) 407
Find Files by templates 439
Find Files by the Length of their Name 415
Find Files commands in List context menu 32
Find Files Location
  All Drives Search (Find Files) 401
  Auto-Sync (Find Files) 401
  Edit Locations (Find Files) 401
  Follow folder links (Find Files) 401
  Include subfolders (Find Files) 401
  Maximum depth (Find Files) 401
  Multiple Location Search 401
  Selected Locations (Find Files) 401
Find Files Location (Auto-Complete Path Names) 42
Find Files Name
  Boolean Search 395
  Comparison Operators 404
  Find by Type 400
  Generic File Types (Search) 400
  Loose Boolean Match 395
  Mode 395
  Name 395
  RegExp Search 395
  Standard Search 395
  Type Filter 400
Find Files Tab "Attributes"
  Not Content Indexed 418
Find Files Tab "Contents"
  Advanced Hex Search: Position 419
  Advanced Hex Search: Wildcards 419
  ASCII mode 419
  Binary search 419
  Case Sensitive 419
  Content search 419
  Grep 419
  Hex mode 419
  Hex Search Advanced 419
  IFilters 419

Find Files Tab "Contents"
Invert 419
It's a Hex String 419
Match Case 419
Metadata 419
RegExp 419
String search 419
Whole Words 419
Wildcards 419
Find Files Tab "Date"
Between ... and/add 416
Date Range popup menu 416
Date stamp "(unknown)" 416
File Date/Time 416
From start of unit 416
In the last 416
Like (Date Range popup menu) 416
Universal Time Search 416
UTC Support 416
Find Files Tab "Dupes"
Duplicate File Finder 422
Find duplicate Files 422
Find Files Tab "Excluded"
Add Current (Excluded folders) 425
Find Files Tab "Name & Location"
Boolean Logic 395
Exact Match 395
Ignore Diacritics 395
Invert (Inverted File Search) 395
Location 395
Loose Boolean Match (LBM) 395
Loose match 395
Match Case (Case-sensitive File Search) 395
Mode 395
Multiple search patterns 395
Path 395
Prefix :for Boolean 395
Prefix > for RegExp 395
Regular Expressions 395
Simple multiple pattern search 395
Whole Words 395
Wildcards 395
Find Files Tab "Size"
Search for folders as well 416
Find Files Tab "Tags"
Filter "Comment" 418
Filter "Labels" 418
Filter "Tags" 418
Finding Empty Fields 418
Search everywhere 418
Select Labels... 418
Select Tags... 418
Find Files via Address Bar 298
Find filters 394
Find hidden (Find Files) 399
Find Images Similar to Clipboard Image 424
Find Now 165, 394
Find Similar Images 424
Finding Tabs 319
Fine Zooming (Floating Preview) 431
Fire click on mousedown (Custom Toolbar Buttons) 311
Fit All (Floating Preview) 431
Fit Height (Floating Preview) 431
Fit popup to screen 110
Fit popup width only 110
Fit Width (Floating Preview) 431
Fit width only 110
Fixed File Version 383
Fixed Mini Tree (Access Control) 500
Flat View 441
Flatten Folder 472
Flatten Folder [folders only] 472
Flexible (Size Format) 219
Flexible tab width 115
Flipped (Floating Preview) 431
Floating Preview 142, 431
Floating Preview Keys 131
Floppy drives (Items in Tree and List) 21
Focus rectangle 51
Folder contents preview 113
Folder Contents Preview (Hover Box) 346
Folder Size Caching 19
Folder thumbnails 104
Folder to apply the settings to 459
Folder Tree (Shell context menu) 32
Folder Tree turns grey 441
Folder View Settings 459
Folder View Settings (Submenu) 187
Folder View Settings with Visual Filters 459
Folders only 32
Folder-Specific View Settings 459
Follow junctions 89, 93
Font File Preview 385
Font Size of the Status Bar 376
Font size selection
Font File Preview 385
Fonts 59
For all copy operations 77
For all move operations 77
For cross-volume moves only 77
For executables as well (File Info Tips) 70
For junctions as well (File Info Tips and Hover Box) 72

Format of a Paper Folder 448
Format(Custom Columns) 351
Freeware 864
Freeze Info and Preview 430
Freezing the Preview 430
Fresh Instance 505
Full name column select 32
Full Name Length 70
Full Row Select (Customize List) 214
Full Row Select (Customize Tree) 214
Full Screen (Floating Preview) 431
Full screen preview 386
Full Screen Preview (Floating Preview) 431
Full Screen Preview Keys 131
Fully Collapse 472
Fully Collapse Drive 472
Fully Expand 472
Functions in Scripting 528
Further Remarks on Backup and Custom Copy 79
Fuzzy Favorites 195
Fuzzy Search 237
FVS (Folder View Settings) 459

- G -

General Remarks on Custom Columns 351
Generic File Types 260
Generic icons 29
Ghost Filter 446
Glider 332
Global Power Filters 262
Global Visual Filters 261
Glossary 13
Go Here in Other Pane 209
Go menu 191
Go Now 191, 308
Go to 191
Go to Application Data Folder 191
Go to Application Folder 191
Go to Dragged Item 476
Go to Drive Root 307
Go to Focused Item 426
Go to Focused Item in New Tab 426
Go to Junction Target 342
Go to Last Target 191
Go to Line 191
Go to New Items Folder 174
Go to Original Location 490
Go to Other Location 209
Go to Previous Item in List 191
Go to Previous Location 191
Go to Real Location 490
Go to Recent Path on Drive 307
Go to Scripts Folder 207
Go to Shortcut Target 342
Go to Tabset Folder 323
Going home also restores the list layout 115
Grayscale (Floating Preview) 431
Grid in List (Customize List) 214
Grid Lines 51
Grid style 51
Grouped Zebra Stripes 51

- H -

Hamburger 325, 523
Hard Link (create one) 165
Hash Values 142
Heavy Font 391
Help (Configuration) 16
Help menu 229
Help on Keyboard Shortcuts 229
Help on Scripting Commands 229
Hidden drives (Items in Tree and List) 21
Hidden files and folders (Items in Tree and List) 21
HiddenItems (AccessControl) 496
Hide certain items (Access Control) 500
Hide Column 359
Hide completed and skipped 378
Hide Current Folder (Mini Tree) 186
Hide Extensions (Customize List) 214
Hide extensions from rename edit box 23
Hide Folder from Tree 330, 472
Hide shell extensions from shell context menu 32
Hide Siblings (Mini Tree) 186
Hide Siblings from Tree 472
Hide Unchanged Items 144
High quality image resampling 98
High Quality Image Resampling (Floating Preview) 431
High Similarity 424
High Speed, Fast, Crisp, Smooth (Thumbnails) 104
Highlight Focused Item (Customize List) 214
Highlight hovered items 32
Highlight matches 97
Highlight matches (Live Filter Box) 94
Highlight Selected Rows (Customize List) 214
Highlight Sorted Column (Customize List) 214
Highlight strings in the file list 343
Highlighted Folder 195
Highlighted Groups 51
History 191, 866
History per Tab 29

History retains selections 29
History retains sort order 29
History without duplicates 29
Hold Ctrl to invert the above selection 32
Hold Ctrl to show cell context menu (Cell Context Menu) 32
Home Tabs 319
Home Zone Tabs 184
Homepage 866
Honor relative paths 119
Horizontal Panes 209
Horizontal splitter 384
Horizontal Swipe Toggles the Preview Pane 377
Hotlist... 191
Hover BlackList 73
Hover Box 346
Hover Box Background 346
Hover Box Default Size Reset 346
Hover Box Icon 346
Hover Box Icon Display 346
Hover Box Keyboard Shortcuts 134
Hover Box Properties and Keyboard Tricks 349
Hover Box Scrolling/Scaling 346
Hover Box Size 346
Hover Box Sort Order 346
Hover Box Status 346
Hover Box Status Display 346
Hover Box Word Wrap 346
Hover Zone (Glider) 332
How to close the Hover Box 346
How to Edit the Interface Language 122
How to open the Hover Box 346
How to order the software 864
How to quote strings in Scripting 531
How to Select a Language 122
How to tweak the INI file 862
HTML Preview 385
http protocol 302, 365

- I -

Icon overlays 29
Icons 142, 381
Icons for the Selected Tree Folder 463
Icons in the Action Log 488
ID3v1.1 tags 70
ID3v1.1-tag editing 390
Ignore Articles When Sorting (Customize List) 214
Ignore Diacritics 57, 97, 250
Ignore diacritics (Find Files) 399
Ignore diacritics (Visual Filters and Live Filter Box) 95
Ignore Extension (Dupes) 422
Ignore extensions 209
Ignore numbers (Dupes) 422
Image (To Clipboard) 142
Image Columns 287
Image Columns (Extra Columns) 289
Image Dupes 422
Image File Preview 386
Image Hash 424
Import Catalog (Catalog) 365
Import Local Tags 198
In list (Folder contents preview) 113
In network locations as well (Always show folder sizes) 17
In network locations as well (File Info Tips and Hover Box) 72
In network locations as well (icon overlays) 29
In network locations as well (subfolders in tree) 17
In tree (Folder contents preview) 113
In Tree as well (File Info Tips) 70
In tree as well (icon overlays) 29
Include basic item data 74
Include beta versions (on check for updates) 46
Include files 74
Include local disks 104
Include most-recently-used lists on save 46
Include network locations (Auto-refresh) 29
Include removable drives (Auto-refresh) 29
Include removable media and network locations (thumbnails cache) 104
Include search results (thumbnails cache) 104
Include virtual folders (Auto-refresh) 29
Including Catalogs 365
Increment on collision 144
Incremental affix 60
Indent (Tree) 125
Index (List Column) 339
Info Panel 380
Info Panel Toggle 378
Information Bar (Quick Search & Find Files) 89
Initial delay milliseconds (File Info Tips and Hover Box) 72
Initial Delay... (Glider) 332
In-place tooltips 70
Insert as New Category Here 365
Install Package 10
Installation 10
Installing XYplorer 10
Installing XYplorer in Companies 496
Instant Color Filters 281
Instant Search 223, 263
Interface Colors 50

Interface Language 122
Interface Translation Tool 229
Interface Translation Tool (ITT) 122
Internal Viewer 383
Intra-Volume Moves vs Cross-Volume Moves 77
Introduction 9
Invert (Find Files) 399
Invert (Floating Preview) 431
Invert Selection 165
Item Base(s) 142
Item Name(s) 142
Item Path(s) 142
Item Path/Name(s) 142
Item Short Path/Name(s) 142
Item UNC Path/Name(s) 142

- J -

Jump and Spot 343
Jump to an Item 343
Jump to Setting... (Configuration) 16
Jump... 469
Junction (create one) 165
Junction (in context menu) 342
Junction Target 70
Junctions (Items in Tree and List) 21

- K -

Keep current item in view after sorting 23
Keep folders on top 23
Keep Particular Characters... 144
Keep playing when info panel is hidden 98
Keep progress dialog open 79
Keeping the folder structure when copying files 482
Keyboard Shortcuts 125
Keyboard Shortcuts (Create Listing) 229
Keyboard Shortcuts on Menu 229

- L -

Label captions and colors 65
Label Clouds 373
Label Search 407
Label style 65
Labels 284, 392
Labels (Submenu) 198
Labels, Tags, Comments 284
Large Buttons (Toolbar) 304
Large Font 391
Large Icons 180
Large Icons (Touchscreen Mode Button) 308
Large Tiles 180
Last Size/Minimize Info Panel 469
Laws of Backup 484
Left Edge (Glider) 332
Len (Full Path) 70
Len (List Column) 339
Let folders pass all filters 92
Level of darkness (0 is darkest) 51
Level-indent 92
Level-indent width in pixels 92
Lic.ini 496
License Key 229
Lighter text in details columns 53
Limiting the selections to files or folders 165
Line feed on oversized filenames 74
Line numbers 335
Line Spacing 53
Line spacing in Tree and List 125
Links folder (Items in Tree and List) 21
List 335
List All Commands... 229
List Centered 226
List Left 226
List Management 214
List navigation right from the Live Filter Box 263
List Right 226
Listing URLs in Virtual Folders 458
Live Filter Box 214, 223, 263
Live Filter Box Configuration 94
Live Filter Box in Status Bar 226
Live Filter Box in Status Bar (Status Bar context menu) 376
Live Filter Box Right-Click Menu 263
Live Filter via Cell Context Menu 337
Live Filtering 336
Load Configuration... 162
Load Favorite Mini Tree 186
Load Layout... 226
Load Previous Mini Tree 186
Load Script File... 207
Load Selected Script File 207
Load Tags Database... 198
Localization 122
Location 373
Lock Expansion State (Customize Tree) 214
Lock Expansion State (Lock Tree) 328
Lock Home Zone 184
Lock Scroll Position (Customize Tree) 214
Lock Tree 215
Lock Tree (Customize Tree) 214
Lock Zoom (Floating Preview) 431

Lock Zoom Position (Floating Preview) 431
Locked Tabs 319
Log actions and enable undo/redo 87
Log clipboard contents and enable restore 87
Log to default location 79
Logarithmic Zooming (Floating Preview) 431
Long filenames (> 260 characters) 42
Loop 98, 110
Loose matches in Visual Filters 250
Loupe 110
Low Similarity 424
Luminance Histogram (Floating Preview) 431

- M -

Make Default 459
Make Mini Tree From Folders 365
Make selected tab bold 115
Manage Commands 201
Manage Favorite Files... 195
Manage Favorite Folders... 195
Manage Folder Views 187
Managed Tree (Access Control) 500
Manual Sorting 218
Map Network Drive... 214
Mark 472
Mark Favorites 195
Mark Favorites (Bold) (Customize Tree) 214
Mark Files in Clipboard as 'Copied' 165
Mark Files in Clipboard as 'Cut' 165
Mark intermediate nodes (Tree Path Tracing) 51
Match All 284
Match anywhere 97
Match at beginning 97
Match breadcrumb bar with custom colored tab 50
Match case (Find Files) 399
Match case (Visual Filters and Live Filter Box) 95
Match color with breadcrumb bar (Tree Path Tracing) 51
Match color with tree path tracing (Recent Location Pins) 51
Match selected tab with breadcrumb bar (Tabs) 50
Matching (Type ahead find) 336
Maximize/Minimize Info Panel 469
Maximum length of generated filenames (0 = unlimited) 60
Maximum number of items cached 89
Maximum number of pins (Recent Location Pins) 51
Maximum number of tabs 115
Media Control Buttons 388
Medium Similarity 424
Menu Buttons (Toolbar) 307
Menu Command Default Shortcuts 125
Menu Commands 142, 165, 191, 195, 201, 214, 229, 476, 482
Meta tab 383
Metadata 383
Metadata (menu File) 142
Metadata layouts 383
MIDI 388
Mini Tree 186, 330
Mini Tree (Submenu) 186
Mini Tree from Current Tabs 186
Mini Tree from Here 186
Mini Tree from Recent Locations 186
Minimize to tray 46
Minimize to tray on X close 46
Minimum / Maximum Name column width 53
Minimum / Maximum tab width in pixels 115
Mirror Browse 209
Mirror tree box color in list 53
Miscellaneous (Customize Keyboard Shortcuts) 469
Mixed Columns 351
Mixed sort on date columns 23
Mixed sort on path columns 23
Mixed sort on tag columns 23
Mobile Hover Box 346
Modeless dialog (Quick File View) 98
Modes of Display 338
Modified (List Column) 339
Modify file dates 381
Modifying the height of the tab bars 319
Mouse Click Navigation (Address Bar) 297
Mouse Down Blow Up 110, 388
Thumbnails 335
Mouse Down Blow Up (Floating Preview) 431
Mouse Down Blow Up for Audio Files 113
Mouse Down Blow Up Hex View 110
Mouse Down Blow Up of Animated GIFs 110
Mouse Down Blow Up of Audio Files 110
Mouse Down Blow Up of Text 110
Mouse Down Blow Up Zoomed 110
Mouse Down Blow Up: Shrink to Fit (Floating Preview) 431
Mouse Down on Thumbnails and Icons 110
Mouse Tricks 125, 136
Mouse Up on Folder Icons 110, 113
Mouse Up Show Down 113
Move Here 476
Move Here to New Subfolder... 156
Move last used item to top 42
Move on Rename 23, 145
Move on rename (Batch Rename) 144

Move on rename (Search and Replace) 144
Move selected items here (Glider) 332
Move Tab to Other Pane 209
Move To 154
Move to Other Pane 209
Move To... (Dialog) 165
Move Up 32, 472
Move/Copy/Backup To 472
Movement 110
Moving the Hover Box (aka Mobile Hover Box) 346
Moving the tabs 319
MP3 388
MP3 Preview 388
Mp3 Special (Submenu) 144
Multi Branch View 441
Multi branch view lists top folders 92
Multi-column matching (Visual Filters and Live Filter Box) 95
Multi-column Sort 339
Multi-level Undo/Redo 486
Multiline Paste 23
Multilingual Support 122
Multimedia Preview 388
Multiple Location Search 402
Multiple Locations (Quick Search) 298
Multiple Locations Advanced (Quick Search) 298
Multiple parallel instances 46
Multiple Select (Customize List) 214
Multi-RowToolbar 305
Multi-selections in tabs 319
Multi-Tabbed Browsing 319
Multi-User License 864
Multi-User Tagging 291
Multi-User Tagging (Admin Settings) 499
Must-Match-Patterns 260
My Network Places 338

- N -

Name (List Column) 339
Name Dupes 422
Name Search Bar 93
Name(s), Bytes, Modified[, Version] 142
Name(s), Bytes, Modified[, Version], Path 142
Name(s), Bytes, Modified[, Version], Path, MD5 142
Named Drives 321
Narrow (Glider) 332
Narrow Tree 214
Native context menu 32
Native Drag and Drop Context Menu 32, 33, 476
Natural Sort (Sort method) 23
Navigate All Files (Floating Preview) 431
Navigate by Category (Floating Preview) 431
Navigate by Click (Floating Preview) 431
Navigate by Extension (Floating Preview) 431
Navigation commands in List context menu 32
Network 39, 338
Network folder (Items in Tree and List) 21
Network Places On Demand 338
Neutral, Grid, White, Black (Transparency background) 98
New 165
New Files... 165
New Folder (...) 165
New Folders... 165
New Items Menu 174
New Path... 165
New Row(Customize Toolbar) 304
New Shortcut... 165
New tab path 115
New Text File (...) 165
NewItems 174
No Border (Preview Border Style) 98
No Graphics (Date Column) 220
No Graphics (Size Column) 219
No network browsing at startup 46
No progress dialog on duplications 77
No progress dialog on intra-volume moves 77
No-Extension (Visual Filters) 250
No-Install Package 10
Non-existing paths are created on the fly 165
Non-sequential undo 486
Non-sequential undo/redo 87
NTFS junction point (create one) 165
Nuke 308
Number of Button Sets (Toolbar) 304
Number of Caption Lines (Toolbar) 304
Number of files scanned 376

- O -

Office Preview 385
OK (Configuration) 16
On autosize disregard the column headers 53
On both panes 209
On closing the current tab 115
On Delete Remove Items from Paper Folder 448
On failures 79
On KeyUp (UDC) 201
On left mouse down 110
On left mouse up (Folder contents preview) 113
On left-click (Custom Toolbar Buttons) 311
On middle mouse down 110

On middle-click (Custom Toolbar Buttons) 311
On name collisions 79
On name collisions (Sync Folders) 363
On right mouse down 110
On right mouse up (Folder contents preview) 113
On right-click (Custom Toolbar Buttons) 311
On sorting keep tagged items on top 65
On the icon only 32
One-click backup of distributed sources 165
OneDrive folder (Items in Tree and List) 21
OneDrive Status 341
Online Support (submenu) 229
Only while the shift key is held down (File Info Tips) 70
Only while the shift key is held down (Hover Box) 346
Open Catalog to Include 365
Open command line start path in new tab 46
Open Command Prompt Here 472
Open Configuration File... 214
Open Containing Folder in New Background Tab 426
Open DOS-Box here 472
Open favorite files directly 42
Open Files from Address Bar 292
Open Focused Item 154
Open Folders in Tabs 365
Open in New Background Tab 472
Open in New Background Tab [folders only] 472
Open in New Tab 472
Open in New Tab [folders only] 472
Open in Other Pane 472
Open in Other Pane [folders only] 472
Open Items by First 'Open with...' Match 512
Open new instance always 46
Open new tab 115
Open Recycle Bin... 214
Open Selected Item(s) 154
Open Throw Away Clone 142
Open with Arguments... 154
Open With... 154, 472
Open... 154
Opening folders and files using the Catalog 365
Opens with (File Info Tips) 70
Open-With Panels from the Toolbar 316
Operator 373
Optimize Tree 472
Options (Manage User-Defined Commands) 201
Options Button (Customize Toolbar) 304
Order the software 864
Original Size (Floating Preview) 431
Other (CSV Field Separator) 74
Other Shortcuts 129
Output File Options 74
Output header format 428
Overall Spacing 53
Overflow Dropdown 304, 306
Overlay caption 104
Overlay icons 29
Overlong Names 43
Overwrite always 79
Overwrite button 79
Overwrite if different contents 79
Overwrite if different size or date 79
Overwrite if newer 79

- P -

Pad blanks 428
Pad filenames 857
Padding 104
Panel 380
Panes Menu 209
Panning (Floating Preview) 431
Paper Folders 448
Paper Folders (Submenu) 186
Parallel processing 75
Partial Matching in the Address Bars 300
Paste 165
Paste (Backup) 165
Paste (Copy) 165
Paste (Move) 165
Paste and Find 97
Paste and Go 97, 325
Paste and Search 325
Paste As Hard Link(s) 165
Paste As Junction(s) 165
Paste As Shortcut(s) 165
Paste As Symbolic Link(s) 165
Paste Extracted 165
Paste Folder Structure 165
Paste here (Glider) 332
Paste Here As... 165
Paste Here to New Subfolder... 165
Paste Here with Path... 165
Paste Image Into New [EXT] File 165
Paste into this Zip 474
Paste Special (submenu) 165
Paste Text As Item(s) 165
Paste Text Into New File 165
Paste to selected list folder 42
Paste Zipped 165
Path (Find Files) 399

Path (Find settings) 428
Path (List Column) 339
Pattern Switches (Quick Search) 298
Pause queue 378
PDF Preview 385
Perceptual Image Hash 424
Permanent Custom Sort Order 218, 448, 452
Permanent Custom Sort Order per Folder (Folder View Settings) 459
Permanent Favorites 223
Permanent startup path 46
Permanent Variables 207
Persist across folders 92
Persist quick search across folders 89
Persist Visual Filters Across Folders 250
Persist visual filters across folders (Visual Filters) 94
Persistent live filters (Live Filter Box) 94
PFA 512
Photo Data 74
Play a sound on certain events 42
Play also when info panel is hidden 98
Play only the first seconds 98
Playback Speed 388
Point to select 32
POM (Portable Openwith Menu) 518
Pop up image in original size on mouse down (Mouse Down Blow Up) 110
Popup by tag columns right-click 65
Portability of Folder View Settings 459
Portable Devices 492
Portable devices (Items in Tree and List) 21
Portable File Associations 512
Portable File Icons 463
Portable Homes 319
Portable Openwith Menu 518
Portable paths 321
Portable tabs 321
Portable Tags 65, 68, 284
Power Filters 250, 261
Prefer matches at beginning 97
Prefixes (Quick Search) 247
Prefix-Overwrite and Suffix-Overwrite (Batch Rename) 144
Preselect name 23
Preserve all item dates 79
Preserve custom colors (Tabs) 50
Preserve permissions on move operation 75, 86
Preserve the folder structure when copying files 482
Preserving file dates on copy 484
Preview 383
Preview All (Rename Special) 144
Preview all Rename Special operations 23
Preview as Thumbnail 102
Preview as thumbnail (Video Preview) 98
Preview Button (Sync Folders) 363
Preview delay 98
Preview of installed or uninstalled fonts 385
Preview Pane 437
Preview Pane to the Left 226
preview supported file formats 390
Preview tab 383
Previewed Formats 102
Prices 864
Print Directories 427
Private File Associations 512
Progress Bar 388
Progress bar (Audio/Video preview) 388
Progress status 376
Prompt before delete 87
Prompt before delete (Sync Folders) 363
Prompt before undo/redo 87
Prompt on closing a locked tab 115
Properties (menu File) 142
Properties tab 380
Protected operating system files (Items in Tree and List) 21
Protocols 302
Purchasing a license 864

- Q -

Quality 104
Queue file operations 75
Queued File Operations 378
Quick Audio Preview 113
Quick File View 142
Quick Find Files 249
Quick Search 236
Quick Search Comments 247
Quick Search Dialog 249
Quick Search Examples 248
Quick Search Help Switch 237
Quick Search Switches 237
Quick Search via Address Bar 298
Quick Search via Cell Context Menu 337
Quick Search... 165
Quick Select 344
Quick Visual Filters 299
Quotes in Scripting 531
Quoting strings in Scripting 531

- R -

Rapid Access 233
Rapid Access Folders 233
Raster Fonts preview 385
Ratcliff/Obershelp 237
Rating Stars 287
Raw Recycle Bins 491
Raw View 391
Raw View tab 391
Read-Only Instance 505
Read-only tabsets 323
ReadonlyTagsDB (AccessControl) 496
RealAudio 388
Recent File Operations... 165
Recent Location Pins 51
Recent Location Pins (Customize Tree) 214
Recent Locations... 191
Recent Locations... (Move/Copy/Backup To) 154
Recently Used Catalogs 365
Reconnect All Mapped Network Drives 214
Reconnect mapped network drives at startup 46
Recreate source folder structure 86
Recycle Bin 490
Recycle Bin folder (Items in Tree and List) 21
Recycle Bin Stats... 214
Recycled Name 490
Redirect typing to Live Filter Box 97
Redirecting Admin.ini 496
Redirecting Startup.ini 850
Redo 165, 486
Referencing a Paper Folder 448
Refine a previous search 394
Refresh 180
Refresh All Icons (Catalog) 365
Refresh Cell by Click 351
Refresh Column 359
Refresh Current Folder 180
Refresh during file operations 29
Refresh Folder 180
Refresh Folder Sizes 219
Refresh List 180
Refresh Tree 180
RegExp Rename... 144
Registration 864
Regular Expressions in Visual Filters 250
Relative Paths 165
Reload All Included Catalogs 365
Reload Tags Database 198
Relocate Tab 184
Relocating tabs 321
Remember Last Input 165
Remember list settings per tab 53
Remember permanent variables 29
Remember relative position 110
Remember Selected Code Page 384
Remember state of tree 17
Remember the logged actions between sessions 87
Remember tree scroll position per tab 115
Remove (Previewed Formats) 102
Remove all Shortcuts... 469
Remove All Tags 198
Remove All Tags... 65
Remove Column 359
Remove Diacritics 144
Remove Filter (Visual Filters) 94
Remove Folder View 187
Remove Items from Database (Tags) 198
Remove read-only attribute 79
Remove Search 93
Remove Selected Items 186
Remove Tags 284
Remove Tags by List... 198
Removing XYplorer 12
Rename 142
Rename Column... 359
Rename File (Floating Preview) 431
Rename folders on collision 79
Rename Next Item 142
Rename On Slow Double-Click (Customize List) 214
Rename On Slow Double-Click (Customize Tree) 214
Rename Preview 144
Rename Special 144, 472
Renaming folders (press F2) 328
Renaming Items in Tree and List 23
Renaming Tabs 319
Repeat Filter (Visual Filters) 94
Repeat Last Quick Search 165
Repeat Last Search 165
Repeat Search 93
Replace Windows Explorer as default file manager 120
Replace with File in Clipboard 32
Replace with File in Clipboard [files only] 472
Report tab 427
Reporting Extra Columns 287
Reset All Shortcuts To Defaults... 469
Reset Colors 50
Reset Columns 70
Reset Filters 394

Reset List 180
Reset Tree 180
Reset Unused Shortcuts To Defaults... 469
Reset Zoom 59
Resizing the window 119
Resolve cache path from current folder 104
Resolve junctions 42
Resolve Links (Raw View) 391
Resolve Shortcuts before Matching 512
Resort list immediately after rename 23
Respond to file system notifications 29
Restart without Saving 164
Restore Application Data Folder... 162
Restore Folder View 187
Restore Previous Clipboard 165
Restore Selection 165
Retry 79
Reuse existing tabs when changing the location 115
Revealing Toolbar Image Keys 304, 306
Revert to Saved 323
Rich Copy (Copy items with their folder structure) 482
Rich Copy Here 476
Rich File Operations (Copy/Move items with their folder structure) 482
Rich Move (Move items with their folder structure) 482
Rich Move Here 476
Rich Paste 482
Right Edge (Glider) 332
Rocker-click gesture 337
Rotate Left (Floating Preview) 431
Rotate Right (Floating Preview) 431
Rounded (Labels) 65
Row height in Tree and List 125
Run Script (Floating Preview) 431
Run Script Again 207
Run script here (Glider) 332
Run Script... 207

- S -

Safe overwrite 79
SafeSave (Multi-User Tagging) 291
Safety Belts 39
Same Extension (Dupes) 422
Save All Settings 162
Save Catalog 162
Save changes to disk immediately 46
Save Configuration 162
Save Configuration As... 162
Save Copy of Configuration As... 162
Save Folder View 187
Save Folder View Settings 162
Save Keyboard Shortcuts 162
Save Layout As... 226
Save Servers 162
Save Settings 142
Save settings on exit 46
Save Tags 162
Save User-Defined Commands 162
Scale Font (Touchscreen Mode Button) 308
Scale Toolbar (Touchscreen Mode Button) 308
Scaling the Hover Box 346
Scope (Keyboard Shortcuts) 469
Scope (Shell Integration) 120
Scope Suffix (Color Filters) 268
Scope Suffixes (Visual Filters) 250
Script (Custom Event Actions) 36
Scripted Columns 351, 356
Scripted Extra Columns (Extra Columns) 289
Scripting 528
  Arrays 546
  Binary Numbers 551
  Boolean Constants 554
  Boolean Operators 553
  Command Prefixes (Scripting) 531
  Commands and Functions (Scripting) 531
  Comments 542
  Comparisons 552
  Compound Assignment Operators 556
  Concatenation 530
  Conditional Logic 555, 556
  Control Structures 555, 556
  Dereference Operator 566
  Equal-operator (=) 543
  Error Messaging 532
  Exponentiation (^) 549
  For Loops 559
  Foreach Loops 559
  Foreach Loops with Arrays 547
  Fractional numbers 549
  Functions 568
  General Command Syntax 530
  Global Variables 545
  Heredoc Syntax 564
  Hex Numbers 551
  Hiding scripts inside a script file 538
  Icons, States, and Levels 538
  If/Elseif/Else Blocks 556
  Include 570
  Include Automatically 571
  Include Statement 570

Scripting 528
Include_once 571
Include_once Statement 571
Increment Syntax (++/--) 543
Initialize and Terminate 534
Integer Division (\) 549
Interpolation 543
Labels 538
Libraries 536
Lifetime of Variables 545
Like (Likel) operator 552
Local Variables 545
Loops in Scripting 547, 558, 559
Math Operators (+.*/) 549
Modulo (%) 549
Multi-line Scripts 534
Multi-Scripts 534
Nested Expressions 549
Nowdoc Syntax 564
Numbered Arguments 530
Operator Precedence 567
Permanent Variables 545
Quick Scripting 532
Relative Levels (Multi-Scripts) 538
Remote Control 567
Risk Classes 532
Scope of Variables 545
Scripting and User-Defined Commands 534
Scripting by Numbers 532
Show User Functions (Step Dialog) 532
Show Variables (Step Dialog) 532
Step Dialog 532
Step Mode: Stepping Through Scripts 532
Switch Statements 561
Ternary Conditionals 555
The Goto-Shorthand 538
Unary Boolean Operators 553
Unary Math Operators (+-) 549
User Functions 568
User-Defined Functions 568
Using Quotes in Scripting 531
Variable Scope 543
Variables in Captions 538
Variables in Scripts 543
Variables... (Step Dialog) 532
While Loops 558
XYplorer Script Files (*.xys) 536
XYS files 536
Scripting Commands 572
abs() 572
age() 573
ageclasses() 574
aid 574
air 575
appicon 575
applyfolderview() 575
asc() 576
assert 577
attrstamp() 578
automaxcolumn 579
autosizecolumns 580
backupto 580
base64decode() 580
base64encode() 581
beep 582
box 582
br 583
break 583
button 584
buttonset 585
catalogexecute 585
catalogload 586
cataloglocation() 587
catalogreport() 588
cea() 588
ceil() 590
ces() 590
charview 592
chr() 593
colorfilter() 593
columnlayout() 594
compare() 594
conf() 595
confirm() 597
continue 598
controlatpos() 598
controlposition() 600
copier() 600
copy 601
copyas 602
copydata 603
copyitem 604
copytext 605
copyto 606
count() 606
ctbicon() 607
ctbname() 607
ctbstate() 608
dark 609
datediff() 610
datepicker() 611
dectohex() 611
delete 612
dlog 613

Scripting Commands 572
download 614
echo 615
editconf 615
end 617
eval() 617
exists() 618
exit 619
explode() 620
extlist() 620
extracttext() 621
extratag() 621
Fast Folder Delete using RMDIR 746
favs() 622
filesequal() 623
filesize() 624
filetime() 624
filetype() 625
filter 625
flattenfolder() 626
floor() 627
focus 628
folderreport() 629
foldersize() 631
font() 632
format() 633
formatbytes() 634
formatdate() 635
formatlist() 637
fresh 639
freshhere 640
get() 640, 827
getkey() 640
getpathcomponent() 641
getsectionlist() 643
gettoken() 644
gettokenindex() 645
ghost() 646
global 647
goto 648
gpc() 641
hash() 649
hashlist 650
hex() 611
hexdump() 651
hextodec() 652
highlight 652
hotkeyshowapp() 653
html() 654
htmlencode() 654
id3tag() 655
id3v2tag() 656
implode() 656
incr 657
indexatpos() 657
input() 658
inputfile() 659
inputfolder() 660
inputselect() 661
interfacecolors 664
internetflags 666
isset() 667
isunicode() 667
itematpos() 668
lax() 669
listfolder() 669
listpane() 671
llog 672
load 672
loadlayout() 675
loadsearch 676
loadsettings 677
loadtree 678
logon 679
makecoffee 679
md5() 679
moveto 680
mru() 686
msg 686
new() 688
newwindow 692
now() 692
obfuscate 693
open 694
openwith 695
outputfile() 696
paperfolder() 696
pasteto 698
patchimage 698
pathreal 699
pathspecial 700
pathvirtual 700
perm 701
popupcontextmenu 702
popupmainmenu 703
popupmenu() 703
popupnativecontextmenu 706
popupnested() 707
posatpos() 708
preview 708
previewcheck 710
property() 710
quickfileview 716
quicksearch() 717

Scripting Commands 572
quote() 719
raf() 720
rand() 720
readfile() 721
readonly 723
readonlyhere 723
readpv 723
readurl() 724
readurlutf8() 725
recase() 725
refresh 727
refreshlist 727
regexmatches() 728
regexreplace() 728
releaseglobals 729
rename 729
renameitem() 732
replace() 734
replacelist() 735
report() 736
resolvepath() 742
return 744
rotate 744
round() 746
rtfm 746
run 746
runq 749
runret() 750
savesettings 750
savethumb() 751
searchtemplate() 754
sel 755
selectitems 757
selectthumbs() 758
self() 759
selfilter 760
seltab 762
set 763
setcolumns() 764
seticons() 767
setkey 767
setlayout() 769
setthumb 769
setting 771
settingp 773
shellopen 774
shellprops 774
showhash 775
showintree 775
showstatus 776
skipundo 776
slog 777
sortby 777
sortbylist 778
sound 779
status 779
statusbartemplate 780
step 781
strlen() 782
strpos() 782
strrepeat() 783
strreverse() 783
sub 783
substr() 784
swapnames 785
sync 785
syncselect() 788
tab() 789
tabset() 795
tag 797
tagcheck() 799
tagexport() 799
tagitems() 801
taglist() 802
tagload() 803
tagsave() 805
text 805
thumbscacherename() 806
thumbsconf() 806
ThumbsOverlayContent 595
ThumbsOverlaySpecs 595
timestamp 808
toolbar() 811
topindex() 812
trayballoon 813
trim() 814
unset 815
unstep 815
update 815
urldecode() 816
urlencode() 817
utf8decode() 817
utf8encode() 817
varname() 818
vartype() 818
view() 819
wait 820
wipe 820
writefile() 821
writepv 823
zip_add() 823
zip_extract() 824
zip_list2() 825

Scripting Commands Reference 572
Scripting Custom Event Actions 36
Scripting Menu 207
Scroll horizontally by wheel 32
Scroll margin 32
Scroll selected folder to the top 17
Scroll subfolders into view 17
Scroll through the list views by wheel 32
Scroll to top after resorting 23
Scrollable Toolbar 304
Scrolled Toolbar Wraps 304
Scrolling the Hover Box 346
Scrolling the Toolbar 304
Seamless wave looping 98
Search and Replace... 144
Search Box 223
Search by filename 236
Search Everywhere 284
Search Here 284
Search In List 403
Search in Listed Items 394
Search in Location 394
Search in Selected Items 394
Search Information Bar 89
Search items by Tags 284
Search only among Tagged Items 407
Search parameters 393
Search Pattern Prefixes 247
Search Pattern Switches 237
Search Results Caching 89
Search Results Context Menu 426
Search results inherit current columns 89, 93
Search results sort order 89
Search Results Tab 319
Search Results to Tab 319
Search Templates
  Load Excluded Folders 439
  Load saved results 439
  Load search location 439
  Load Template 439
  Run search at once 439
  Save search results 439
  Save to Template... 439
Search Templates... 165
Search This Branch 284
Searches via Address Bar 298
Secondary Sorting 339
Secure Delete 142
Select (submenu) 165
Select All 165
Select All Files 165
Select all on focus by key 42
Select all on focus by mouse 42
Select all on item change 42
Select By Selected Type(s) 165
Select Color... (Glider) 332
Select Context... (Hover Box) 346
Select copy handler 84
Select Custom Column... 359
Select Extra Fields... (File Info Tips) 70
Select files by type 261
Select Item Types... (Hover Box) 346
Select Items... 165
Select Language... 122, 229
Select last used subfolder 17
Select list items on mouse hover 42
Select Local Language File... 229
Select match on drop down 42
Select next item after delete and move 17
Select parent of deleted folder 17
Select parent of moved folder 17
Select Property... 359
Select Special Property... 359
Select Standard Fields... (File Info Tips) 70
Selected files only 428
Selected List Item(s) (Catalog Context Menu) 365
Selected List Rows 142
Selection by Path Components 292
Selection Filter via Cell Context Menu 337
Selection Filter... 176
Selection Filters 176
Selection Stats 165
Selections 51
Selections in focused controls (XYplorer Style only) 51
Selections in non-focused controls 51
Semi-transparent grid color 53
Separator (Customize Toolbar) 304
Separators in toolbar 304
Serial Rename 142
Serial rename with Up and Down keys 23
Server Mappings 385
Servers outside primary workgroup 338
Session Title 505
Set archive attribute on folder rename 23
Set as Favorite Mini Tree... 186
Set Box Color 195
Set Created Date to Exif 142
Set Extension... 144
Set Highlight Color 195
Set Home 319
Set Modified Date to Current 142
Set Modified Date to Exif 142
Set Tags 284

Setting the Application Data Path (Command Line Switch) 510
Setting the Startup Path (Command Line Switch) 511
Settings Special (Submenu) 142
Shade inactive pane 119
Shadow (Preview Border Style) 98
Shared Tags 291
Shell context menu 32, 120, 479
Shell Context Menu... 292
Shift+Wheel scrolls horizontally 32
Short File Names 142
Short paths (8.3 style) in the Address Bar 292
Shortcut (create one) 165
Shortcut files 335
Shortcut Keys 125
Shortcut Target (in context menu) 342
Shorten Filenames (Batch Rename) 144
Shorthand trick for Toolbar buttons 314
Show Address Bar 226
Show Age (Date Format) 220
Show Age Graphics in Date Column 221
Show Age maximum hours 53
Show All Items In Branch 165
Show audio info and tags (File Info Tips) 70
Show Breadcrumb Bar 226
Show Button Captions 226
Show Button Captions (Toolbar) 304
Show cached folder sizes only 19
Show cached thumbnails only 104
Show caption 104
Show Catalog 226
Show Character Table 380
Show Columns... 359
Show custom file icons 29
Show dimensions of original 104
Show Downward Paths 325
Show drag status box 39
Show Drive Labels 325
Show embedded icons on Properties tab 29
Show Exact Bytes in Status Bar (Status Bar context menu) 376
Show Expansion Icons (Customize Tree) 214
Show file icon on thumbnail 104
Show File Info Tips 70
Show film strip overlay on video thumbnails 104
Show filter information in list (Visual Filters) 94
Show filter information in tab headers (Visual Filters) 94
Show filter warning in list 250
Show Folder Row Colors 219
Show folder size by clicking the donut 219
Show folder size on Properties tab 17
Show Folder Sizes 219
Show Folder Sizes (Customize List) 214
Show folder thumbnails 104
Show Full Paths in 'Open With...' Menu 512
Show Grid (Customize List) 214
Show Histogram (Floating Preview) 431
Show Hover Box 346
Show Icon 325
Show icon overlays 29
Show icons 115
Show Icons (Catalog) 365
Show Icons (Customize Tree) 214
Show implicit secondary sort order arrow 23
Show Info Panel 226
Show Information Bar in the List 448
Show item count with folder sizes 17
Show Items (Submenu)
- Edit Ghost Filter... 189
- Ghost Filter 189
- Hide Protected Operating System Files 189
- Set Global Visual Filter 189
- Show CD-ROM Drives 189
- Show Custom File Icons 189
- Show Floppy Drives 189
- Show Folders in List 189
- Show Hidden Drives 189
- Show Hidden Files and Folders 189
- Show Junctions 189
- Show System Files and Folders 189
- Toggle Global Visual Filter 189

Show last actions in toolbar button menu 87
Show Length without Path (Len column) 339
Show line numbers 335
Show Line Numbers (Customize List) 214
Show Lines (Customize Tree) 214
Show Live Filter Box 226
Show localized folder names 17
Show Menu Button 325
Show Menu Icons (Manage User-Defined Commands) 201
Show message when list is empty 42
Show Milliseconds (Date Format) 220
Show name length while renaming 23
Show Navigation Buttons 325
Show Navigation Panels 226
Show 'New Tab' button 115
Show options in menu 87
Show Photo Data 180
Show Photo Data (Floating Preview) 431
Show photo data in the Hover Box 74
Show photo data in the Large Tiles view 74

Show Preview Pane 226
Show progress dialog 79
Show quick search results in current tab 89, 93
Show Real Path 292
Show relative path in Path column 89, 93
Show Script Button (Floating Preview) 431
Show search information in list 89, 93
Show search results in 89, 93
Show Section Colors 216
Show Section Colors (Customize Tree) 214
Show shared folder overlays 29
Show shortcut overlays 29
Show Size on Disk 219
Show sort headers in all views 23
Show splash screen while loading 46
Show Status Bar 226
Show Status Bar (Floating Preview) 431
Show Status Bar Buttons 226
Show Status Bar Message (Manage User-Defined Commands) 201
Show Status Log 377
Show summary report 79
Show Tab Bar 226
Show 'Tab List' button 115
Show Tag Bar (Floating Preview) 431
Show tags in file list 65
Show these fields (File Info Tips) 70
Show thumbnails for non-images 104
Show thumbnails for RAW files 104
Show thumbnails in tiles views 104
Show Times in UTC (Date Format) 220
Show tips for clipped tree and list items 70
Show Toolbar 226
Show tooltips 32
Show Trailing Slash 292
Show Transparency grid 104
Show Tree 226
Show verbatim tooltips 32
Show version information in the Status Bar 42
Show Weekday (Date Format) 220
Show X close buttons on tabs 115
Showing the preview image in original size 388
Shrink to fit 110
Silent uninstall 12
Similar Images Finder 424
Simple Variables 852
Single File Search 246
Single License 864
Single Pane 209
Single step undo/redo 87
Single-click to open an item 32
Site License 864
Size (List Column) 339
Size Dupes 422
Size Format 219
Size on disk (Properties tab) 380
Size Search 405
Skip 79
Skip (Junctions) 79
Skip [...] milliseconds (Video Preview) 98
Skip and Skip All 79
Skip button 79
Skip calculation of free disk space for mapped network drives 39
Skip invisible subfolders (Find Files) 89
Skip Job 378
Skip single spaces 97
Skip verification on local hard disks 79
Skipping Locked File (Nuke) 308
Slashes 325
Small Audio Preview 437
Small Buttons (Toolbar) 304
Small Font 391
Small Icons 180
Small Tiles 180
Smart Dropdown Buttons 314
Smart Phones 492
Smart Right-Click 337, 338
Smart Section Sizing (Status Bar context menu) 376
Snap Next to Mouse (Glider) 332
Snap to Edge (Glider) 332
Snap to First Button (Glider) 332
Snap to Main Window (Floating Preview) 431
Snapshot 388
Soft Columns 359
Soft labels 65
Solid Lines (Customize Tree) 214
Sort By (Submenu)
  Include Folders (Random Order) 181
  Previous Order 181
  Random Order 181
  Reverse Order 181
  Selected Items to Top 181
  Show Sort Headers in All Views 181
  Sort Again 181
  Sort Folders Apart 181
  Unsorted 181
Sort Category (Catalog Context Menu) 365
Sort date columns descending by default 23
Sort filenames by base 23
Sort folders always ascending 23
Sort folders apart 23
Sort method 23

Sort order dirty 335
Sort size columns descending by default 23
Sorted by (Folder contents preview) 113
Sounds played on an event 42
Spaces to _ 144
Special Properties 712
Special System Folders 195
Special Toolbar Buttons 308
Specifying Menu Icons (UDC) 201
Spot an Item 343
Spot and Jump 343
Standard Menu 325
Start Job Now 378
Start Parameters 505
Startup maximized 46
Startup minimized 46
Startup pane 46
Startup Path 511
Startup window state 46
Startup with a Script 505
Startup.ini 10, 850
Status Bar 376
Status Bar (Templates) 60
Status Bar Buttons 378
Status Bar Font Size 376
Status Bar Templates 64
Status Column 341
Status Log 377
Stay up 110
Step Mode 207
Sticky checkbox selection 53
Sticky Section 223
Sticky Selection (Customize List) 214
Sticky Selection Deluxe 65
Stop (Find Files tab) 394
Stop running processes by ESC key 125
Storage 65, 68
String File Version 383
Style 104
Style Switches (Color Filters) 269
Subitem separator 428
Sub-Loop (Audio/Video Preview) 388
Suffix increment to copy 79
Suffix increment to existing 79
Sunday is the first day of the week 42
Support overlong filenames 42
Support volume labels in paths 42
Supported protocols (Catalog) 365
Suppress delete confirmation dialog 86
Surround selection 42
Suspend Auto-Refresh 180
Swap Locations 209
Swap Names 142
Swiping the Status Bar 377
Switch Button Sets 317
Switch Catalogs on the fly 365
Switch languages on the fly 122
Switch sets of tabs 365
Switch Toolbar Button Set 226
Switches (Color Filters) 268
Switches (Quick Search) 237
Symbolic Link (create one) 165
Sync Browse 209
Sync caps (Sync Folders) 363
Sync Folders 363
Sync Folders... 209
Sync Scroll 209
Sync Select... 209
Sync Sort 209
Synchronize tree with search location 89, 93
Syntax Checking 207
System Default (Text preview) 384
System files and folders (Items in Tree and List) 21
System list separator (CSV Field Separator) 74
System requirements 10

- T -

Tab (CSV Field Separator) 74
Tab (in menu View)
- Background Color... (Tab) 183
- Clone Tab 183
- Close Other Tabs 183
- Close Other Unlocked Tabs 183
- Close Tab 183
- Close Tabs to the Right 183
- Copy Location Term 183
- Default Tab 183
- Filter By Selection(s) 183
- Go Home 183
- Iconized Tab 183
- Lock Location 183
- New Tab 183
- Open New Tab 183
- Portable Tabs 183
- Relocate Tab... 183
- Rename Tab... 183
- Restore Last Closed Tab 183
- Select Color... (Tab) 183
- Set Home 183
- Set Visual Filter 183
- Tab History... 183
- Text Color... (Tab) 183

Tab (in menu View)
Toggle Live Filter 183
Toggle Visual Filter 183
Tab captions 115
Tab key 119
Tab Key Navigation (Address Bar) 296
Tab position 319
Tab sets 323
Tab switching by mouse wheel 319
Tabbed Browsing 319
Table width 74
Tablets 492
Tablist... 191
Tabs 115, 319
Tabs (in menu Panes) 209
Tabs are drop targets 319
Tabs with a Home 184
Tabs with Home 319
Tabs... (Move/Copy/Backup To) 154
Tabsets 323
Tabsets can revert after saving settings 115
Tag (List Column) 339
Tag Clouds 373
Tag List 284
tag.dat 284
Tags 65, 284, 392
Tags (Submenu) 198
Tags Search 407
Tags tab 392
Target (Shortcuts only) 70
Team Tags 291
Template for dropped messages 60
Template for filenames 60
Template for Status Bar 60
Template for Title Bar 60
Templates 439
TeraCopy Integration 84
Text (To Clipboard) 142
Text contrast 51
Text preview 384
Textual Sort (Sort method) 23
The Cell Context Menu 335
The Configuration Dialog for Backup and Custom Copy operations 79
The Progress Dialog 79
Thumbnails 104, 335
Thumbnails #1/#2/#3 180
Thumbnails cache 104
Thumbs 335
Time Format 220
Timestamp 142
Time-stamping 338, 381
Title Bar (Templates) 60
Title Case 152
To Clipboard 472
To Clipboard (Report tab) 427
To Clipboard menu 142
To File (Report tab) 427
To Popup (Report tab) 427
To Recycler (Nuke) 308
To the icon only 32
Toggle Active Pane 209
Toggle Boxed Branch 195
Toggle Dual Favorite Folder 195
Toggle Favorite File 195
Toggle Favorite Folder 195
Toggle Favorite Live Filter 263
Toggle Highlighted Folder 195
Toggle On Same Filter 250
Toggle on same filter (Visual Filters) 94
Toggle on same query 92
Toggle Paper Folder 186
Toggle Quick Search 165
Toggle Selection 165
Toggle tags by column click 65
Toggle Zoom (Floating Preview) 431
Tolerance (Dupes) 422
Toolbar 304
Toolbar First 226
Toolbar Style 304
Toolbar Zoom 304
Tools Menu 214
Tools Special 214
Tooltip zoom (%) 32
Top 191
Top-align if Vertically Cropped (Floating Preview) 431
TortoiseSVN icon overlays 29
Total Similarity 424
Touc hModified Date 338
Touch 338
Touch Accessed Date 338
Touch Created Date 338
Touch file dates 381
Touch Screen 110
Touchscreen Mode 180
Touchscreen Mode (Toolbar Button) 308
Trademark Information 869
Translation of XYplorer into your language 122
Translucent selection box 53
Transparency background 98
Transparency grid 98
Treat hyphens and apostrophes like normal characters 23

Treat portable devices as read-only 39, 492
Tree 328
Tree and Catalog Stacked 226
Tree Expansion Icon, right-click menu 328
Tree Node Crumbs 328, 330
Tree Path Tracing 51
Tree Path Tracing (Customize Tree) 214
Tree plus/minus symbol, right-click menu 328
Tree Section Colors 216
Tree structure 74
Tree Structure (Report) 427
Tree turns grey 441
Tree white space, right-click menu 328
Tree-Like Sort Order 441
Tree-Shaped History 330
Trial Version Info... 229
Triangles 325
Trigger (Custom Columns) 351
TrueType preview 385
Truncate filenames in the middle 53
Truncate Paths from Beginning (Path column) 339
Try Script... 207
Turn off delete confirmation 75
Tweaks 862
Type (List Column) 339
Type ahead find 97, 336
Type Stats and Filter...
  Select files by type 185
  Sort by Count (Type Stats) 185
  Sort by Extension (Type Stats) 185
Type-1 Preview 385

- U -

UAC (User Account Control) 850
UDC (User-Defined Commands) 201
Underline selected rows 53
Undo 165, 486
Unicode to UTF-8 144
Uninstall 12
Universal Time Stamp 381
Unload All Included Catalogs 365
Unlock Trial Version... 229
Up 191
Update Category 365
Update Registration Details... 229
Update Tag List 198
Upgrades 866
Uppercase the first character 144
UrlEscape (' ' to '%20'...) 144
UrlUnescape ('%20' to ' '...) 144
Usability 32
Use (Selections in focused controls) 51
Use (Selections in non-focused controls) 51
Use (Thumbnails View Background) 104
Use Cache on Calculate Folder Sizes 219
Use category for searching 373
Use custom command line interpreter 60
Use Custom Copy 77
Use dialog to rename single items 23
Use empty cell defaults 53
Use generic icons for super-fast browsing 29
Use localized search and filter patterns (Cell Context Menu) 32
Use native handling in the preview pane 98
Use sorted column 97
Use space character for Boolean AND (Visual Filters and Live Filter Box) 95
Use standard shell drag and drop 120
Use standard shell file info tips 70
Use Status Bar for Messages (Access Control) 500
Use status bar template 60, 64
Use Status Bar Template (Status Bar context menu) 376
Use whole screen 110
User Account Control 850
User Button Backgrounds 311
User Button Icons 311
User Button Labels 311
User Buttons 311
User Buttons as Drop Targets 316
User Buttons as Open-With Panels 316
User Buttons as Open-With Panels (Toolbar) 317
User folder (Items in Tree and List) 21
User Forum 866
User menu 201
User-defined code pages 384
User-Defined Commands (UDC) 201
User-Defined Functions 568
User-defined popup menu 523
User-Defined Preview Handlers 102
Using Power Filters to Select Files 261
Using Property-based color-coding in the Tree (Color Filters) 279
Using quotes in Scripting 531
Using Startup.ini 850
Using Variables in Virtual Folders 456
UTC Support 381
UTF-8 auto-detection 98
UTF-8 to Unicode 144

- V -

Variable <allitems ...> 856
Variable <curver> 856
Variable <file filename> 860
Variable <get ...> 856
Variable <pad ...> 857
Variable <perm ...> 858
Variable <prop ...> 859
Variable <propt ...> 859
Variable <selitems ...> 856
Variables 852
Variables (Quick Search) 247
Variables in Visual Filters 250
Various Information 229
Verbatim switch 407
Verification 79
Version in the Status Bar 376
Version Information 70
Version tab 383
Vertical grid lines in details view 53
Vertical Panes 209
Vertical Popup Toolbars 315
Vertical Swipe Toggles the Info Panel 377
Video bytes 386
Video File Preview 388
View menu 180
Viewer 383
Views (Submenu) 180
Virtual Folders 453
Virtual Folders as Filters 458
Virtual Folders, Integration 455
Virtual Folders, Properties 455
Virtual Folders, Switches 454
Virtual Folders, Syntax 453
Visible time in milliseconds (File Info Tips and Hover Box) 72
Visual Filter Bar (Visual Filters) 94
Visual Filter Syntax 251
Visual Filter Syntax: AND 251
Visual Filter Syntax: Captions 253
Visual Filter Syntax: Comments 253
Visual Filter Syntax: No Extension 252
Visual Filter Syntax: NOT 252
Visual Filter Syntax: OR 251
Visual Filter Syntax: Special Operators 252
Visual Filter Syntax: Wildcards 251
Visual Filters 250
Visual Filters by Attributes, Size, Date, Age, Length, and Properties 250
Visual Filters by Columns 250, 259
Visual Filters by Custom Columns 250, 259
Visual Filters by File Types 250
Visual Filters by Particular Properties 255
Visual Filters by Properties 260
Visual Filters by Selectors 255
Visual Filters by Tags 250, 257
Visual Filters Configuration 94
Visual Filters via Address Bar 299
Visual Filters: Matching 253
Visual Filters: Usage 254
Visual Filters: Variables 254
Visual style (Tabs) 115
Visually Similar Images 424
Volume Discount Prices 864

- W -

Web Preview 385
Web Site 866
Wheel scroll lines 32
When hovering over the filename (File Info Tips) 70
When hovering over the filename (Hover Box) 346
When hovering over the icon (File Info Tips) 70
When hovering over the icon (Hover Box) 346
White Border (Floating Preview) 431
Whole words (Find Files) 399
Wide (Glider) 332
Wide Info Panel 226
Wide Tab Bar 226
Width of the Live Filter Box 263
Width of trace in pixels (Tree Path Tracing) 51
Wildcards in Copy/Move Here As... 156
Wildcards in Tags 407
Wind Backward and Forward 388
Window Menu 226
Windows Canonical Properties 712
Windows Theme Style 51
Windows Themes Style (Tabs) 115
Wipe 142
Wiping Beyond Recovery (Nuke) 308
With border 110
With Confirmation (Nuke) 308
Word Preview 385
Word Wrap (Text Preview) 384
Working with many Catalogs 365
Workspaces (= Tabsets) 323
Wrap-around list 17

- X -

X close buttons on each tab 319

XYcopy 75
XYplorer in shell context menu 120
XYplorer is default file manager 120
XYplorer Native Variables 852
XYplorer Style 51
XYplorer Style (Archaic) (Tabs) 115
XYplorer Style (Rounded) 51
XYplorer Style (Rounded) (Tabs) 115
XYplorer Style (Tabs) 115
XYThumbs.txt 104

- Z -

Zebra Stripes: Alternate Rows 51
Zebra Striping in List (Customize List) 214
Zeropad numbers 857
Zip 472
Zip Folders 472
Zip Here 476
Zip View 474
Zoom by Wheel (Floating Preview) 431
Zoom font size 125
Zoom In (Floating Preview) 431
Zoom Out (Floating Preview) 431
Zoom smaller originals to fit preview area 98
Zoom to Fill (Floating Preview) 431
Zoom to fill (Thumbnails) 104
Zoom to Fit (Floating Preview) 431
Zoom to fit (Thumbnails) 104
Zooming the Toolbar 304, 306