

WinEdt Documentation

WinEdt PDF Sync

Yet Another PDF Viewer

ALEKSANDER SIMONIČ

Copyright © 2019–2026

Last revision: June 13, 2026

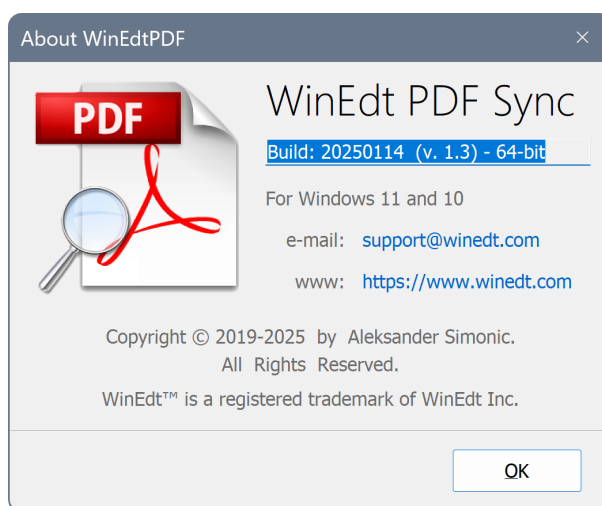
Contents

Introduction	iii
1 About WinEdt PDF Sync	1
1.1 WinEdtPDF License	1
1.2 PDFium Library License	2
1.3 An example of rendering Pgfplots graphics	3
1.4 SyncTeX: Forward and Inverse Search	4
2 Using WinEdt PDF Sync	5
2.1 Graphic Interfaces	6
2.2 Bookmarks	6
2.3 Keyboard Shortcuts	7
2.4 Printing	8
2.5 Recent Files List	9
2.6 Preferences Dialog	10
2.7 Themes	12
2.8 Toolbar	13
2.9 Interface	15
2.10 Resources consumption	16
2.11 Advanced: Command Line Switches	17

Introduction

Is the world itching for yet another PDF Viewer?

Not really, but...



WinEdt's default settings cannot assume that a suitable PDF viewer for working with TeX documents exists on the user's computer. Previewing problems start if the default app associated with pdf files is Adobe or Edge. Both applications lock the files they are previewing, and new users experience this as a WinEdt bug since they cannot recompile their TeX document until they release it from Adobe or Edge. WinEdt has a remedy to instruct Adobe to close the file before it launches a TeX compiler. But this has proven to be an unreliable fix as the DDE parameters change with new versions of Adobe.

Installing SumatraPDF solves this problem. However, many (would be) WinEdt users do not persist long enough to find this out, and they look for WinEdt alternatives to fix their pdf previewing woes. This issue needs a proper and reliable solution!

In the fall of 2019, I purchased *Foxit Quick PDF Library* to write a new PDF viewer that seamlessly integrates with WinEdt and provides the Forward and Inverse Search functionality based on SyncTeX technology which has been a standard for TeX compilers for over a decade. There were some snags along the way, and I had to purchase an even more expensive Quick PDF Library with Delphi sources to get around ambiguous or lacking documentation. After a while, PDFium open-source library replaced (now discontinued) Quick PDF. This transition considerably improved the rendering and printing performance. WinEdt PDF Sync is the result of these efforts (just in time to be included with the WinEdt 11 release).

WinEdtPDF is a *light-weight* pdf viewer on purpose. Fast performance, low resource consumption, and the ease-of-use were the guiding principles behind its design. I will continue to make improvements and implement essential functionality that is currently lacking (as required). However, I will not consider new features of interest to only a handful of users while compromising its design principles and primary function as a working pdf *viewer* for TeX documents. Keep this in mind before you send any feature requests for WinEdtPDF: sometimes less is more!

WinEdt can still use alternative PDF Viewers (just like before: see the Execution Modes Dialog). SumatraPDF sets the bar pretty high, and WinEdt PDF Sync does not match all its features. However, it does provide adequate functionality to be suitable as a working previewer for TeX documents.

Another reason for this project was that I wanted to create a multi-document app that does not rely on Windows MDI. WinEdtPDF is the result. It allows full screen and presentation view. Unfortunately, it would take plenty of work and testing to do the same with WinEdt. For over 25 years, WinEdt has evolved around MDI, and this reliance would be hard to change now. Eventually, WinEdt will drop MDI as it is no longer actively supported by MS.

Finally, with control over WinEdtPDF functionality, I will eventually change the format of WinEdt's documentation from HTML Help to PDF and use this app in place of Windows HTML Help (which is no longer supported by MS and will eventually disappear). The conversion will take considerable effort. Unless I receive some help with it, this project will have to wait for the next major release.

These are the reasons...

Now let's move on!

1 About WinEdt PDF Sync

WinEdtPDF comes bundled with WinEdt editor. It automatically installs and integrates with WinEdt. It will start when you successfully compile a TeX document to pdf in WinEdt (using Compile or PDFTeXify).

At the moment only a 64-bit version of WinEdtPDF is available. If there is a strong demand for a 32-bit version I may consider making it available at a later date.

1.1 WinEdtPDF License

WinEdtPDF can be used free of charge with a registered (or trial) version of WinEdt 11. However, it is *not* a free software and it cannot be distributed or bundled with any other software without written permission from its author.

WinEdtPDF was compiled with Delphi 13 (Professional).

A binary library `PDFium.dll` is used to render pdf documents on the screen or printer's display context.

Future development and functionality of WinEdtPDF will be based on this library. The library was downloaded from GitHub. For security reasons, the library was signed with WinEdt's Code Signing Certificate. No other changes were made to PDFium.dll by the WinEdt Team. WinEdtPDF cannot run without this binary file in its install folder!

On the next page you can read the original *PDFium License and Disclaimer* as provided by its authors and contributors.

DISCLAIMER: WINEDT TEAM AND THE AUTHOR OF THIS PROGRAM ACCEPT NO RESPONSIBILITY FOR DAMAGES RESULTING FROM THE USE OF THIS SOFTWARE AND MAKE NO WARRANTY, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE. THIS SOFTWARE IS PROVIDED "AS IS", AND YOU, ITS USER, ASSUME ALL RISKS WHEN USING IT. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES HOWEVER CAUSED...

1.2 PDFium Library License

Please take a moment to read the *PDFium License Agreement and Disclaimer* about the terms of use and distribution of PDFium.dll library:

Copyright 2014 PDFium Authors. All rights reserved.

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

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- * Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

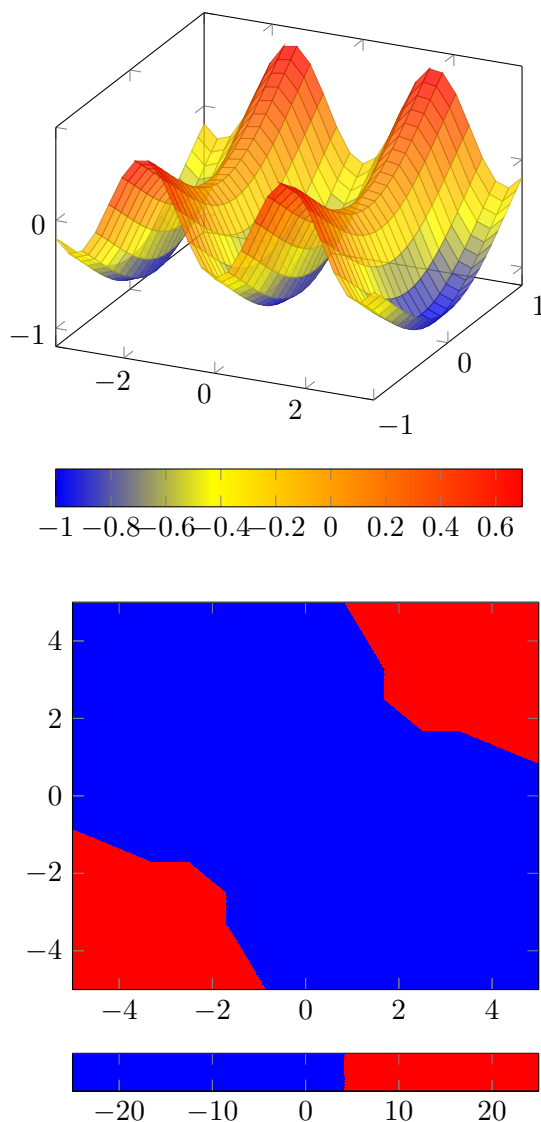
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE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

Copyright 2014 PDFium Authors. All rights reserved.

PDFium binaries are available on GitHub.

1.3 An example of rendering Pgfplots graphics

WinEdtPDF is not responsible for rendering images on the screen: this is done by the *PDFium* library. Here is an example of how such images are rendered:



The second picture did not look right with earlier versions of PDFium library. The problem was fixed in the new versions. Rendering issues with images are not a WinEdt-problem: the job is done by the PDFium library (which is a project in progress and it may fail with certain graphics inclusions in TeX documents)... But a native PDFium library is very fast!

1.4 SyncTeX: Forward and Inverse Search

As the name of this app suggests it is focused on previewing TeX documents and providing a link between the sources and pdf documents using SyncTeX files produced by TeX during the compilation. If you are not familiar with such functionality here is how it works:

- In WinEdt you have a button to initiate the *Forward Search* (**Shift+F8**). The previewer will highlight the corresponding text or object in a pdf file.
- Double-clicking on the text in the previewer starts the editor and highlights the corresponding line (or paragraph if you are using soft wrapping). This functionality is called the *Inverse Search*.

WinEdt's SyncTeX parser is custom-written. It was tested on different types of documents. It comes with no guarantees but I do not expect any problems here...

Note that SyncTeX technology has some limitations and its accuracy can be affected by certain packages or document classes. Before you report any problems with forward and inverse search you should first check out how SumatraPDF handles the situation. The editor is irrelevant here as all SyncTeX parsing is done by the previewer.

In WinEdtPDF you start the Document Properties Dialog (in the File Menu **Ctrl+F3** or the last button in the toolbar). It will display the properties of the currently active pdf document. At the bottom you will find the SyncTeX file status and data size:

...

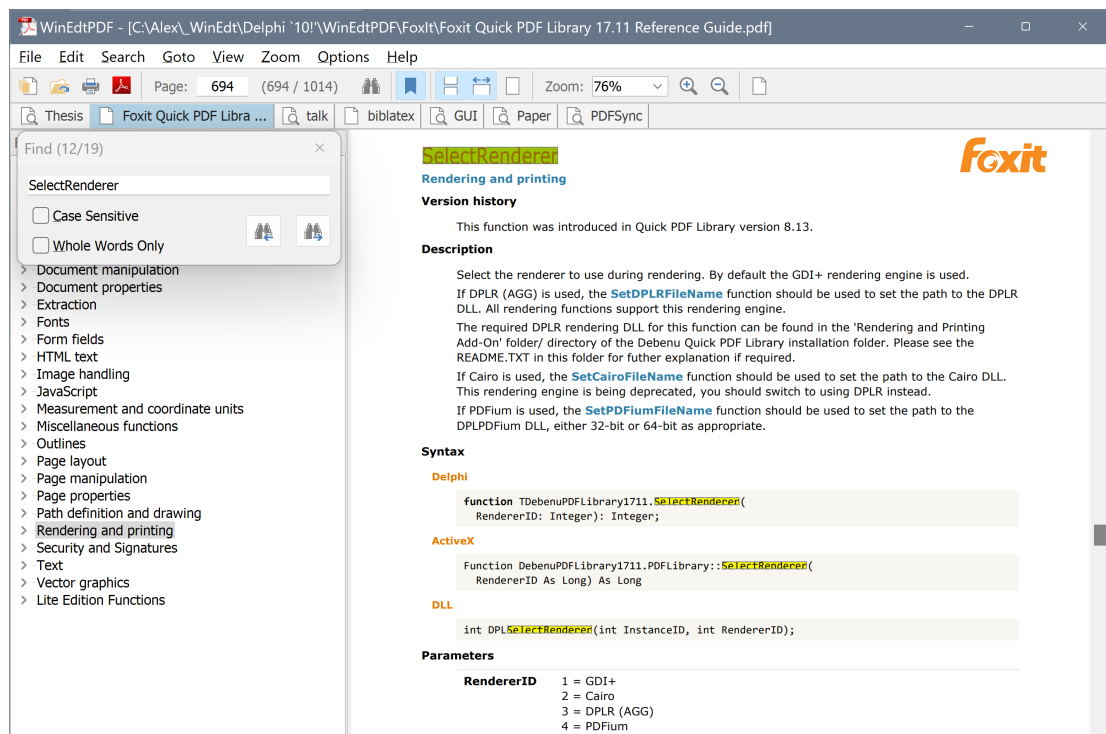
```
SyncTeX:   Loaded (OK)
Data Size: 506.3 KB (518,491 bytes)
GZ Size:   87.1 KB (89,259 bytes)
Created:   2022-10-15 17:00:56
Modified:  2022-10-15 17:44:01
```

If (plain `.synctex` or compressed `.synctex.gz`) SyncTeX file does not exist next to the `.pdf` source you cannot use the Forward or Inverse Search functionality. By default, WinEdt compilations macros take care of SyncTeX creation by passing the appropriate `--synctex` switch to TeX compilers that create the *SyncTeX* data.

2 Using WinEdt PDF Sync

WinEdtPDF interfaces were designed to be intuitive and easy to use without consulting any documentation. It has a reasonably small menu where you can find most of the available actions with their shortcuts. Eventually, you will memorize a few shortcuts for frequently used actions. Until then you can always use the menu to refresh the memory.

Taking a quick look at the menu and toolbar is a good idea. This way you will learn how it is organized and where to go if you want to perform a certain action or change the document view.



2.1 Graphic Interfaces

A small toolbar provides an interface to frequently used actions and it also displays the currently viewed page label and zoom properties.

Document tabs become (by default) visible when there is more than one document opened. The icon at the beginning of the tab reflects the document status (such as locked, modified, with or without SyncTeX, etc...). For the active document, this icon is duplicated at the end of the toolbar in case you hide document tabs or use a single document interface. This icon can also be used to close the tab and the pdf document associated with it. Tabs can be rearranged with drag-and-drop and they also respond to right clicks with a context popup menu.

Status line is (by default) not visible and there is no need to change this as the toolbar reflects all the info available in its panels. It was only useful during the early stages of testing and debugging.

Hint: *The Options Menu contains items that can be checked or unchecked to show or hide any of these controls (including scrollbars in the preview form). They are automatically hidden in the presentation view.*

2.2 Bookmarks

The bookmarks form is displayed on the left side and its tree view reflects the pdf document outline and structure (when available). Bookmarks can be shown or hidden from the View menu ([F4](#) shortcut).

It has a context menu that allows you to expand or collapse all branches and enable or disable the tracking (automatic selection of the branch reflecting the current position in the document). Double-clicking on a branch displays the page (or a more precise destination when available) associated with it.

When the Bookmark form becomes active the navigation shortcuts are processed by the tree view control rather than the pdf viewer. Press [Tab](#) or [Esc](#) key to activate the view form and switch back to pdf navigation.

Hint: In TeX documents the outline is usually generated by the `hyperref` package. The preamble should contain something like:

```
\usepackage{hyperref}
```

2.3 Keyboard Shortcuts

Document navigation is done with standard shortcuts such as: [Page Up](#), [Page Down](#), [Home](#), [End](#), [Left](#), [Right](#), [Up](#), [Down](#), ...

The [Enter](#) key can be used to enter custom zoom or page destination from the toolbar. [Enter](#) and [Shift+Enter](#) shortcuts also perform the search (next and previous, respectively) actions while the Search Dialog is visible.

The [Esc](#) key can be used to cancel the full screen or presentation view (when the menu and toolbar are not visible). This key will also remove any selection, close the Search Dialog, etc...

[Tab](#) and [Shift+Tab](#) key switch the input focus between the Bookmarks and the Preview form.

As expected, plus and minus keys can be used to zoom in or out.

If you manually change the Page or Zoom value through the edit control in the toolbar you must press [Enter](#) to apply your change or [Esc](#) to cancel it. The associated edit control does not automatically update its contents while it is focused because it expects user input. This is not a *bug*!

WinEdtPDF responds to [m](#) key to enter the "measuring" mode and display cursor screen -> page coordinates in a popup edit control. Pressing [m](#) repeatedly toggles between units that are used for coordinates (pt, mm, or in). [ESC](#) key should be used to cancel this functionality and hide the popup.

When WinEdtPDF is in measuring mode (initiated with [m](#) key) [Ctrl+Click](#) will reset the origin to the current (cross) cursor position. The same can be done by pressing [o](#) while [Shift+o](#) restores the default left-top origin. This makes it easy to measure relative horizontal or vertical distances. The default origin is in effect when you initiate this function.

The rest of the shortcuts can be seen in the menu and there is no point listing them here...

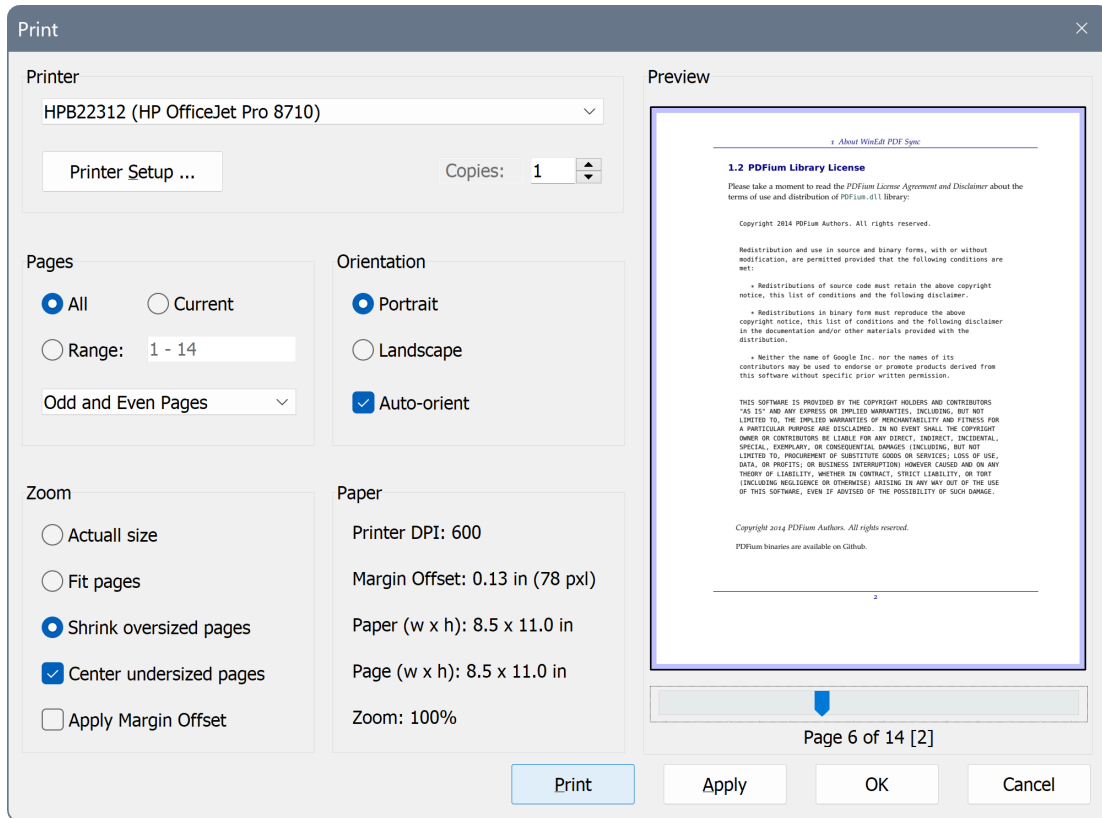
The custom mouse interface allows scrolling and zooming (when [Ctrl](#) key is down) with the scroll wheel (if available).

WinEdtPDF also responds to the gestures when a touch screen is enabled. Panning, zooming, and rotating can be performed with touchscreen gestures (when available).

2.4 Printing

A binary library `PDFium.dll` is used to render pdf documents directly on the printer's display context. This makes printing on Windows much faster and more efficient than using intermediary bitmaps. The size of printed output is about the same (or even slightly smaller) as that produced by Adobe Acrobat.

Print Dialog (with preview) can be used to adjust and preview most common printing settings:



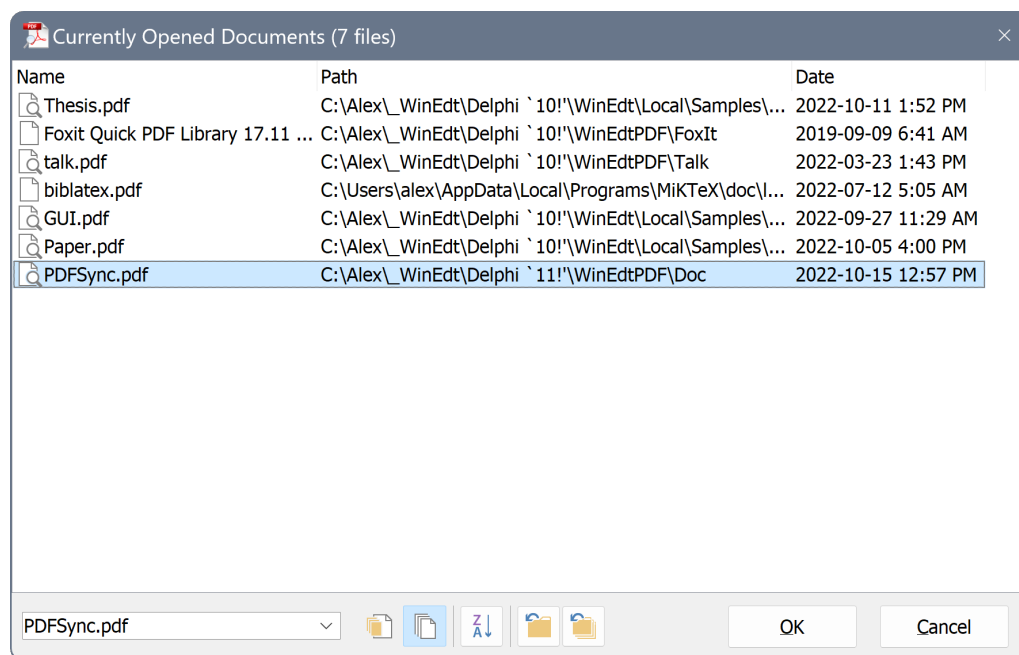
Hint: Should these options prove insufficient to meet advanced requirements, print from Adobe Acrobat. WinEdtPDF provides a toolbar button interface to an alternative viewer (Adobe Acrobat or another pdf viewer of your choice).

2.5 Recent Files List

WinEdtPDF can maintain a list of up to 999 recently opened files with their positions and properties (view, rotation, zoom...). The list can be maintained through a special dialog (**F2** shortcut) or the first toolbar button. There you can make multiple selections to maintain the list or bulk-open (or close) selected documents.

The maximal number of documents in the list (0..999) can be entered in the Preferences Dialog. By default, the last 99 documents are maintained.

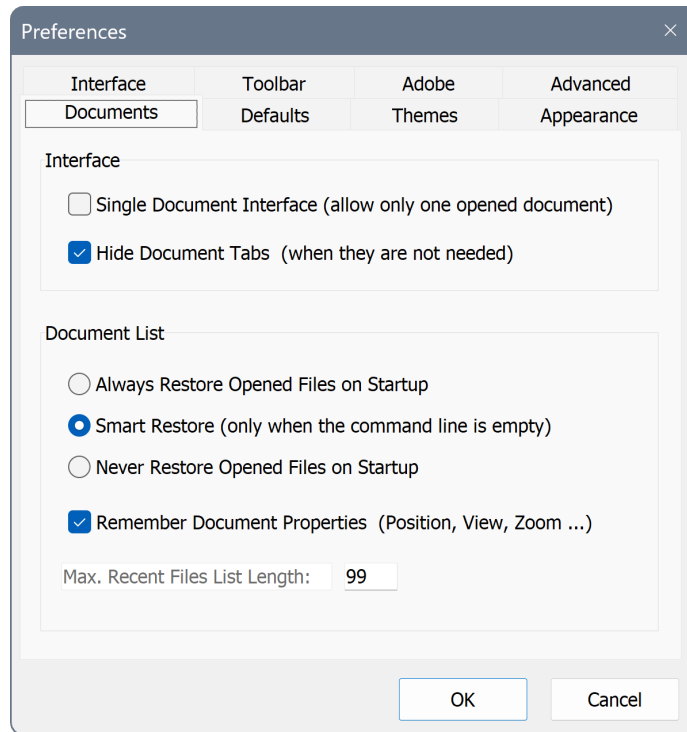
The Document List Dialog has two modes of operation: it can also display the list of currently opened documents. The buttons at the bottom of the dialog or **Ctrl+Tab** shortcut can be used to switch between the two lists. This may come in handy for users that have a lot of documents opened or for those that do not want to use the document tabs.



*I decided against appending the truncated version of the recently opened files list to the end of the File menu. The list view is a superior interface to navigate between a large number of files: it can sort them according to name, path, date, or index. One can also type a partial name to access a file buried in the middle of a long list. **F2** to the rescue!*

2.6 Preferences Dialog

Few users will need to change any settings since the defaults have been carefully prepared with an average user in mind. If you need to change the default document properties or interface you can do so from the Preferences Dialog (the first item in the Options Menu).



On the first page you can enable the Single Document Interface if you choose to allow only one document (without document tabs) to be opened at any time. You can also specify how the document list will be maintained and used.

The second page allows you to specify the initial properties that are applied to a document when it is opened for the first time. They can be changed on a per-document basis while the document is previewed.

Most of the available options are self-explanatory. For example, you can specify a list of predefined zoom values that can be appended to the Zoom Menu (if you so wish).

If you find the white background hard on your eyes you can specify a custom color range in the Appearance Page of Preferences Dialog. The RGB color components are blended to the specified values (the darkest and brightest admissible values). Replacing the white color #FFFFFF with a (solarized) value like #F9E4C6 reduces the blue color background component making it easier on the eyes. The custom range effect can be combined with any rendering mode. It can be amplified by specifying a darker color for the White RGB substitute...

You can adjust some common Document Tab properties. If you are not infested in Themes and Dark Mode rendering you can hide the Themes menu.

In the Adobe page of the Preferences Dialog you can define an external PDF viewer that can be conveniently started from WinEdtPDF. By default Adobe is used based on the Windows Registry info:

```
%HKCR\Software\Adobe\Acrobat\Exe
```

WinEdtPDF is a new app and it may fail to properly display or print certain documents or lack advanced functionality that may be required by some documents. You can start Adobe to perform missing actions there and then close it before you can recompile. If on the other hand, the problem persists in Adobe then you know that this is not a WinEdt-related issue...

In the Advanced page you can see the currently used configuration file (this is where your custom settings and file list is stored). This may be of interest to advanced users that run different instances of the program using the command line switches.

It is recommended to enable the [Use Threads](#) option. This will make loading faster since SyncTeX files are loaded in a background thread.

Disabling the option [Load PDF Documents into memory](#) saves some memory but it also makes WinEdtPDF more vulnerable during compilation and it prevents pdf files from being deleted until they are closed. If you are working with large pdf files (200+MB) you may consider disabling this option and see if this improves the performance.

Finally, you can specify or view the command line that will be used to launch the editor and initiate the *Inverse Search*. For example, the following specification works with any version of WinEdt:

```
"WinEdt.exe" "[Open(|%f|);SelPar(%l,8);]"
```

The *Inverse Search* command is automatically reset when WinEdt starts the viewer and (for most users) there is no need to modify it manually.

2.7 Themes

WinEdtPDF supports dark mode rendering. This is done with post-processing of the rendered PDF contents on the pixel level and it does not affect the actual content of the PDF file. The options menu allows you to switch between the default and the dark theme (F7 shortcut). The details can be adjusted in the Themes page of the Preferences Dialog or through the items in the (optional) Themes menu.

Although the inverted rendering mode requires extra image processing, it has been heavily optimized and it does not have any noticeable effect on the performance. The custom (non-linear) algorithm to invert the brightness has the most CPU-intense effect.

The (optional) Themes menu can be used to adjust the particulars of the selected theme (dark or light) on the go.

Examples below illustrate the difference between Inverted Brightness or RGB values:

LastRenderError

Miscellaneous functions, Rendering and printing

Version history

This function was introduced in Quick PDF Library version 7.13.

Description

Returns the exception information in cases where the renderer encountered an error.



LastRenderError

Miscellaneous functions, Rendering and printing

Version history

This function was introduced in Quick PDF Library version 7.13.

Description

Returns the exception information in cases where the renderer encountered an error.



LastRenderError

Miscellaneous functions, Rendering and printing

Version history

This function was introduced in Quick PDF Library version 7.13.

Description

Returns the exception information in cases where the renderer encountered an error.



2.8 Toolbar

Users seldom agree when it comes to the default toolbar. To address this issue the toolbar is now customizable through the Preferences Dialog. The default definition contains all available buttons, labels, and edit controls. Most are commented out as the space is limited (and sometimes less is more).

The most obvious choice is whether or not you want the items related to the search in the toolbar or do you prefer a dialog to initiate the search. The option [Place Search in Toolbar](#) can be used to make the choice. The search items are placed at the end of the toolbar and they are automatically removed if there is not enough space for them in which case the Search Dialog is restored for this functionality.

Making any other item visible or invisible is pretty straightforward. Remove or insert a comment ";" in front of its definition. Separators can be specified by a line consisting of a single character "|". They get automatically removed when there is nothing to separate. Simple cut and paste can be used to rearrange the order of buttons and controls.

Buttons in the toolbar inherit their image and functionality from the associated menu item. A few up/down buttons that are associated with the checked menu items need an extra image specification. For example:

```
9 ShowBookmarks "Bookmark"  
|  
9 CountiniousView "ContView"  
9 ZoomFitWidth "FitWidth"  
9 ZoomFitPage "FitPage"
```

Labels and edit controls can be optionally followed by a string that determines their width. Furthermore, the caption for labels (such as Page, Zoom, or Find) can be replaced by a custom string, making it possible to translate them into a different language:

```
9 PageLabel " Page: " "Page: "  
9 PageGoto Selector " 9999 "
```

Each item (other than separator) starts with its priority: 0..9. If the toolbar becomes too small to fit all enabled items those with the priority 0..8 are removed until all remaining items are visible. Items with priority 9 are *always* visible (but they can be clipped if the toolbar is too small). How you will use this feature depends on the way that you use the previewer. If you frequently resize it to place it next to the editor you probably want a smaller toolbar without the search. If, on the other hand, you use the previewer in the maximized mode you may consider adding a few extra buttons (such as [Back](#) and [Forward](#)).

The visibility of items associated with Adobe and Search is linked with the checkbox properties in the Preferences Dialog. That's why their definition contains a "filter" "A" or "S" immediately after their priority (lowercase filter indicates unchecked option):

```
; =====
9 FileList
9 Open
|
9 DocumentProperties
8 OpenFolder
7 Print
6A OpenAdobe
;5 SendMail
;5 Reload
;5 Save
;5 Close
;5 CloseAll
|
...

; Search Items (if there is enough room)
; -----
1s Find
1S FindOpt
;0S FindLabel " Find: " "Find: "
1S FindString "A long String to Find ... "
1S FindNext
1S FindPrev
1S FoundCount " 9999/9999  "
; =====
```

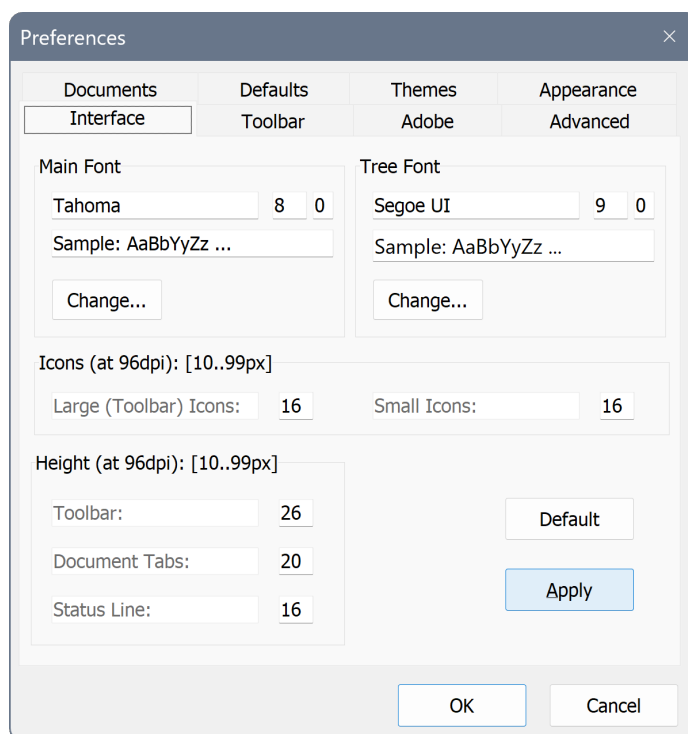
The icon for [Document Properties](#) button changes depending on the presence or absence of the syntex file. It also indicates locked document status while a document is compiled or previewed in an external viewer such as Adobe. This provides useful visual feedback.

In the Toolbar Page of the Preferences Dialog you can press the [Apply](#) button to immediately apply the changes to the toolbar. Should you manage to mess up with your customizations you can also restore the default toolbar definition (and try again).

Hint: *If you have spent a lot of time customizing the toolbar to your needs you should consider copying and pasting its definition into a text file for future use and safekeeping!*

2.9 Interface

The options on this page allow you to tweak the GUI. The preferred font, icons, and control sizes may depend on your display size and dpi.



You can adjust the main font that is used in the toolbar, document tabs, bookmark caption, and status line. A different font can be used for the bookmark tree control.

The font size is specified in points (1pt=1/72in) independently of your screen resolution. Common values for GUI fonts are in the range: [8..12]. It is also possible to manually specify extra pixels next to the font size. On displays with a resolution larger than 72dpi, this makes it possible to adjust the font height beyond the precision of the size. For example, on 144dpi display 1pt=2px.

You can also specify the size of large and small icons that are used in the toolbar or document tabs and main menu, respectively. The images are automatically adjusted to your display dpi.

Finally, if you opt for a larger font or icons you will probably also want to increase the height of the toolbar, document tabs, and status line...

2.10 Resources consumption

This is a warning to users that have a habit of having many dozens of documents opened in WinEdt. PDF files and SyncTeX can use a lot of memory resources. While I was testing I came upon a 2000+ pages sample:

```
Path: C:\WinEdt 11\WinEdtPDF\Test
File: Huge File.pdf
```

```
Title: N/A
Author: N/A
```

```
Subject: N/A
Keywords: N/A
```

```
PDF Producer: MiKTeX pdfTeX-1.40.24
PDF Creator: LaTeX with hyperref
PDF Version: 1.5
```

```
Pages: 2350
Page Size: 8.5 x 11.0 in (216 x 279 mm, 612 x 792 pt)
```

```
Size: 11.2 MB (11,797,192 bytes)
Created: 2022-01-08 16:35:21
Modified: 2022-01-08 16:35:21
```

```
SyncTeX: Pending (OK)
Data Size: 101.8 MB (106,831,076 bytes)
GZ Size: 23.2 MB (24,342,123 bytes)
Created: 2022-01-08 17:35:21
Modified: 2022-01-08 17:36:20
```

Note the size of SyncTeX Data! This example convinced me to modify the SyncTeX parser and make it work directly with UTF8 strings (rather than using Windows native UTF16 wide strings which would be more convenient but would also double the memory consumption to over 200MB)!

Furthermore, SyncTeX files of such magnitude are no longer loaded with the document until they are needed for forward or inverse search and they are released from the memory 10 minutes after their last use (good riddance!).

You live you learn...

2.11 Advanced: Command Line Switches

WinEdtPDF accepts a set of startup switches optionally followed by the filename and switches for forward and inverse search. The startup switches are used to determine if a new instance of the program will be started or if the parameters will be passed to the existing instance.

Startup Switches:

=====

```
-V (Virgin Start:
    do not reload any documents)
-R (Reload documents regardless of the settings
    in the Preferences Dialog)
-N (Start a new instance:
    even if one with the same caption is already running)
-B (Pass the parameters to the existing instance in the background
    without activating or focusing it)
-C="Caption"
-I="Icon File.ico"
-L="Local Configuration File"
-D="Local Debug File"
```

-I, -L, AND -D can use %B, %b, %H, or %h in path specification

Advanced WinEdt users will be familiar with the logic since WinEdt already uses similar switches. These switches can be followed by an (optional) inverse search command definition, an additional switch for forward search or destination, and a pdf filename.

Inverse Search Command Definition:

=====

```
--inverse-search <action>
    Inverse Search Command Definition from WinEdt:
    '--inverse-search "||%B\WinEdt.exe|| -C=||%!W|| '+>
    '||[Open(||%%f|);SelPar(%!1,8);]||"
```

Destination Switches:

=====

```
--page-index <num>
--page-label <string>
--named-dest <string>
--forward-search "%f" %!l (file and line number)
```

Only one destination switch can be specified for each file. In case of multiple specifications the last one prevails.

The command line for WinEdtPDF should be specified as follows:

```
WinEdtPDF.exe [startup switches] [--inverse-search ...]  
[ [destination switch] "source.pdf"] ...
```

- The startup switches are *not* passed to the existing instance of WinEdtPDF and they should be always specified at the beginning of the command line.
- More than one source file (preceded by an optional destination switch) can be specified in the command line.

For example, WinEdt's Forward Search command for WinEdtPDF is defined as the following macro:

```
Run( '%$(! "PDF-View"); '+>  
'--inverse-search "||%B\WinEdt.exe|| -C=||%!W|| '+>  
'||[Open(|%%f|);SelPar(%%l,8);]||" '+>  
'--forward-search "%f" %!l '+>  
'"%P\%N.pdf"' );
```

The switch `--inverse-search` defines the inverse search command that will be executed by WinEdtPDF when it initiates the inverse search. WinEdtPDF substitutes variables `%f` and `%l` in the inverse search definition by the TeX source filename and line number matching the current PDF position (based on the SyncTeX data generated by TeX during the compilation). The string `||` is replaced with a double-quote which is required when parameters include spaces. An almost identical command is used for SumatraPDF except that a different convention (`\`) is used to *escape* and pass double quotes inside the command line.

Named destinations in TeX documents can be inserted into the pdf file with `\hypertarget` command defined in the `hyperref` package. For example:

```
\hypertarget{My_Target}{}Targeted text....  
...  
\hyperlink{My_Target}{Goto My Target}
```

Document Properties Dialog in WinEdtPDF has an option to list all *named destinations* in a document. This can come handy when debugging your TeX sources if hyper-links in compiled pdf document don't work as expected...