User Manual

Foxit® Convert2PDF

Microsoft® Partner
Gold Independent Software Vendor (ISV)

TABLE OF CONTENTS

1 Introduction to Foxit Convert2PDF

Foxit Convert2PDF is an easy-to-use command line tool used to batch convert large volumes of images, text, HTML, Office files into high-quality PDF files, without requiring additional software installation except for Microsoft Office which is needed for Office documents conversion. Convert2PDF is an all-in-one PDF conversion solution, which gathers the features of image2pdf, text2pdf, html2pdf, and office2pdf. It supports various image formats like bmp, png, jpeg, jpx, gif, tiff (including the image with a single page or multiple pages); most popular Office document formats which include doc, docx, xls, xlsx, ppt and pptx; plain text files with the extension ".txt" and HTML files with the extension ".htm" or ".html".

1.1 Why Foxit Convert2PDF is your choice

Foxit is an Amazon-invested leading software provider of solutions for reading, editing, creating, organizing, and securing PDF documents. Foxit Convert2PDF is a command line tool to perform high volume PDF generation and processing. This tool helps IT organizations develop workflows to convert large different types of files into PDF files. It also provides libraries for software development groups to incorporate PDF conversion into their applications. Customers choose this product for the following reasons:

- **High performance** – Multi-threading support speeds up the PDF conversion based on today's server architectures.
- **Professional quality** – Professional-quality on PDF conversion.
- **Lightweight footprint** – Lower memory usage and faster installation.
- **Perfect message mechanism** – Gives more perfect message hints if users encounter problems when using the tool to offer better user experience.
- **Robust and stable** – Ensures smooth running of the application and enhancement of fault tolerant.
- **Easy to integrate** – Command line or application interfaces enable flexible and seamless integration with user's existing workflows.
- **Plug and Play** – Choose one or more of the specific modules that meet your needs.

Foxit offers 24/7 support for its products and are fully supported by the PDF industry’s largest development team of support engineers. Updates are released on a regular basis to improve user experience by adding new features and enhancements. Foxit Convert2PDF is the best solution for PDF batch conversion and integration of high performance features at a low cost!

1.2 Features

Foxit Convert2PDF provides the following features:

- Batch convert image, text, local HTML files, and Office files into PDF files.
- Convert a webpage (URL) into a PDF file.
- Finish faster with multi-threaded processing.
- Make the complex simple with sub-folder recursion processing.
- Support single-file processing and single-folder processing.
- Support image formats like bmp, png, jpeg, jpx, gif, and tiff.
- Support plain text files with the extension ".txt".
- Support html formats like htm and html.
- Support most popular Office document formats like doc, docx, xls, xlsx, ppt and pptx.
- Set page width and height for image, text, and HTML conversion.
- Set page margin for image, text, and HTML conversion.
- Support font style, font size, and font color for text conversion.
- Optionally support pagination wherever there is a page break for text conversion.
- Support cache address, timeout, and page rotation settings for HTML conversion.
- Set all the page contents to one single PDF page for HTML conversion.
- Use the cookie to convert a URL that requires login to PDF for HTML conversion.
- Apply JavaScript to webpages to handle some actions for HTML conversion.
- Optionally disable retaining hyperlinks in the generated PDF files for HTML conversion.
- Support password-protected Microsoft Office files.
- Provide scale options for Microsoft Excel conversion.
- Support wildcard characters in batch processing, e.g., *.jpg, *.html, *.txt, and *.doc.
- Offer simple-to-use API.

1.3 Common Use Case Scenarios

- Batch convert images into PDF files for better image management. For example, create electronic books for users to scan paper documents to image files, and then convert them into a single PDF file; make image albums for users to collect their photos and then convert them into a single PDF file.
- Batch convert Microsoft Office files into PDF files for better archiving.
- Convert a batch of local HTML files or a webpage (URL) into PDF file(s) for better printing or archiving. For some large HTML files or a webpage which contain(s) many contents, it is not convenient to print or archive them directly. In this case, users can convert them into PDF file(s) which are widely used in printing and document archiving.

- Batch convert plain text files into PDF files for better viewing or editing. It is not easy for users to view or edit plain text files, and in this case, users can convert them into PDF files for further processing.
- Batch convert image, text, HTML and Office files into PDF files at the same time.

1.4 System Requirements

Windows Platform：

- Windows Server 2012
- Windows Server 2008 R2
- Windows Server 2003
- Windows 10
- Windows 8.1
- Windows 7

Note *For the Office conversion, please make sure that you have already installed Microsoft Office 2007 SP2 or later, and the virtual printers. If Microsoft Office 2007 is not the SP2 version, please download the "Microsoft Save as PDF" plugin. For Microsoft Office 2016, please ensure that "Microsoft Print to PDF" has been set as the default Microsoft virtual printer, otherwise the Office conversion may not work and cause a crash.*

1.5 About This Manual

This manual aims at introducing how to use the command line of Foxit Convert2PDF tool. It is intended for the users who want to batch convert different types of files into PDF files.

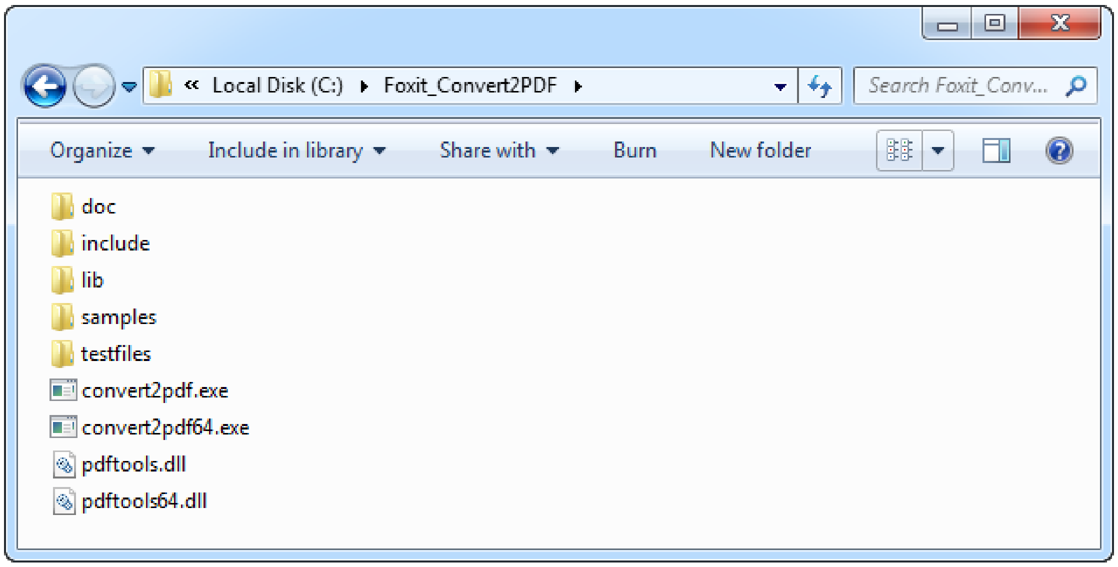
- Section 1: gives an introduction of Foxit Convert2PDF.
- Section 2: illustrates how to install and uninstall Foxit Convert2PDF.
- Section 3: describes basic usage of Foxit Convert2PDF.
- Section 4: shows how to work with API.
- Section 5: provides support information.

2 Installing and Uninstalling Foxit Convert2PDF

2.1 Installation

Installation of Foxit Convert2PDF tool is straightforward. You can download the trial release package, which is a zip file from Foxit website (https://www.foxitsoftware.com/products/pdf-toolkit/), and then extract the package to the desired location as shown in the following figure. In this manual, we rename the package "foxit_convert2pdf" and unzip it to the directory "C:\foxit_convert2pdf". The package contains:

doc:	user manual for Foxit Convert2PDF
include:	header file for Foxit Convert2PDF
lib:	third-part libraries and databases for Foxit Convert2PDF
samples:	some batch samples for Foxit Convert2PDF
testfiles:	testfiles for Foxit Convert2PDF

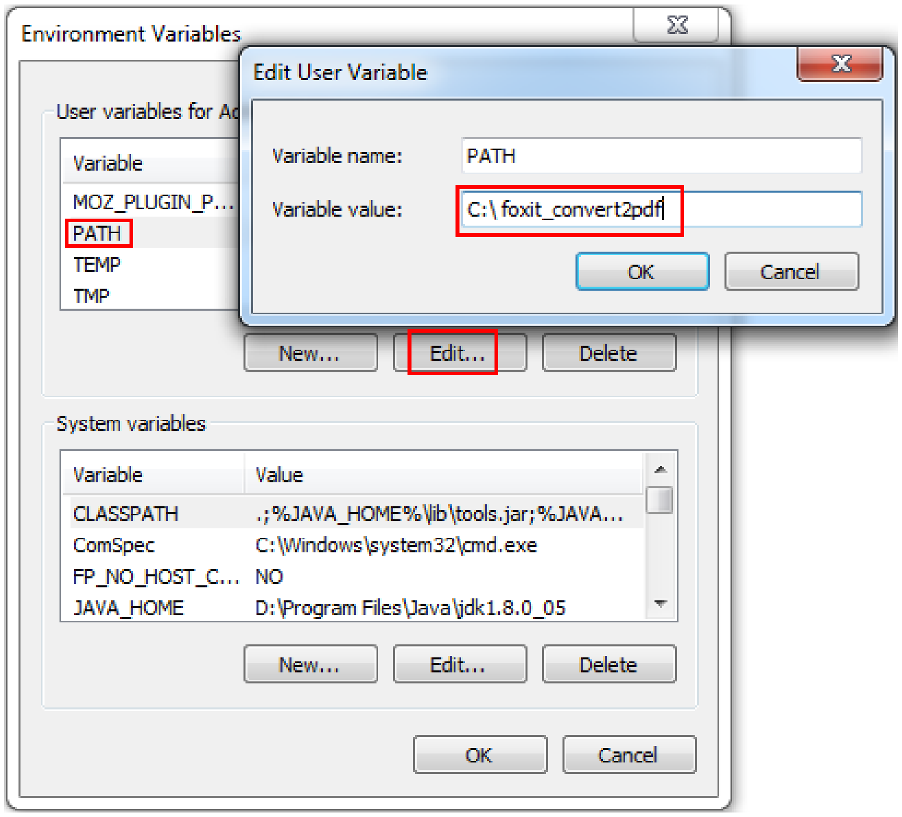


doc
include
lib
samples
testfiles
convert2pdf.exe
convert2pdf64.exe
pdftools.dll
pdftools64.dll

Foxit Convert2PDF package

After that, open a terminal session and navigate to the installation location to run your application. For the Office conversion, please make sure that you have already installed Microsoft Office 2007 SP2 or later, and the virtual printers. If Microsoft Office 2007 is not the SP2 version, please download the "Microsoft Save as PDF" plugin. For Microsoft Office 2016, please ensure that "Microsoft Print to PDF" has been set as the default Microsoft virtual printer, otherwise the Office conversion may not work and cause a crash.

Tips: You can set the installation path to Environment Variables, which allows you to use the commands in the terminal window directly without needing to change the directory to the installation location. Go to **Start**-> **Control Panel**-> **System**-> **Advanced system settings** -> **Advanced** -> **Environment Variables**, in the "User variables for Administrator" box, select **PATH** and **Edit**, and then add the installation path as shown in the following figure. If the environment variable "path" does not exist, create it by clicking **New**.



Variable
MOZ_PLUGIN_P...
PATH
TEMP
TMP

Variable name: PATH
Variable value: C:\ foxit_convert2pdf

Variable	Value
CLASSPATH	.;%JAVA_HOME%\lib\tools.jar;%JAVA...
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
JAVA_HOME	D:\Program Files\Java\jdk1.8.0_05

Set installation path to Environment Variables

2.2 Evaluation

Foxit Convert2PDF tool is distributed on a "try-before-you -buy" basis. Foxit allows users to download trial version to evaluate the features. You have a 30-day free trial, during which the pages of all generated PDF files will contain our watermark. After the evaluation period, customers that want to continue to use the product should purchase a licensed version from Foxit website at https://www.foxitsoftware.com/products/pdf-toolkit/.

2.3 About License

Foxit Convert2PDF provides three types of licenses for customers to choose.

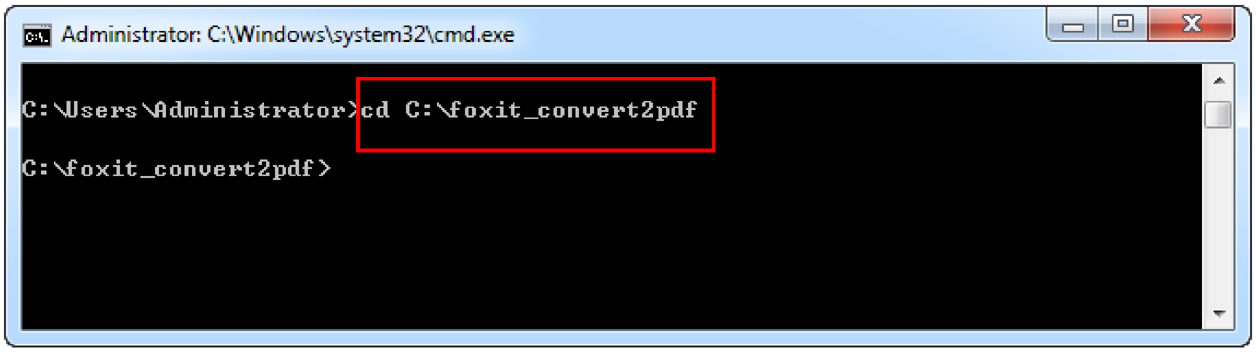
- **Desktop License**- It is only available on desktop systems, which is perfect for personal or small business use. Each license is good for one user on one machine.
- **Server License**- It is available on servers, with 8 CPUs, which is good for small-to medium-sized businesses that need higher performance on a single server. If your server has more than 8 CPUs, please contact Foxit sales team to purchase Enterprise License.
- **Enterprise License**- It is intended for large companies that need to process a large number of PDF documents on multiple high-performance servers. Enterprise License also includes features for integrating the Foxit Convert2PDF into your own applications. Please contact Foxit sales team to purchase Enterprise License.

2.4 Registration

When you get the activation code for the Convert2PDF tool, please use the argument "**-register <code> <licensee>**" to activate it in the Command Prompt window. The following two steps show how to open a Command Prompt window based on Windows 7 system.

a. Click on the **Start** menu.
b. Type "***cmd***" in the *Search programs and files* box and press **Enter**.

In the opening Command Prompt window, type "***cd C:\foxit_convert2pdf***" to navigate to the installation location as follows.

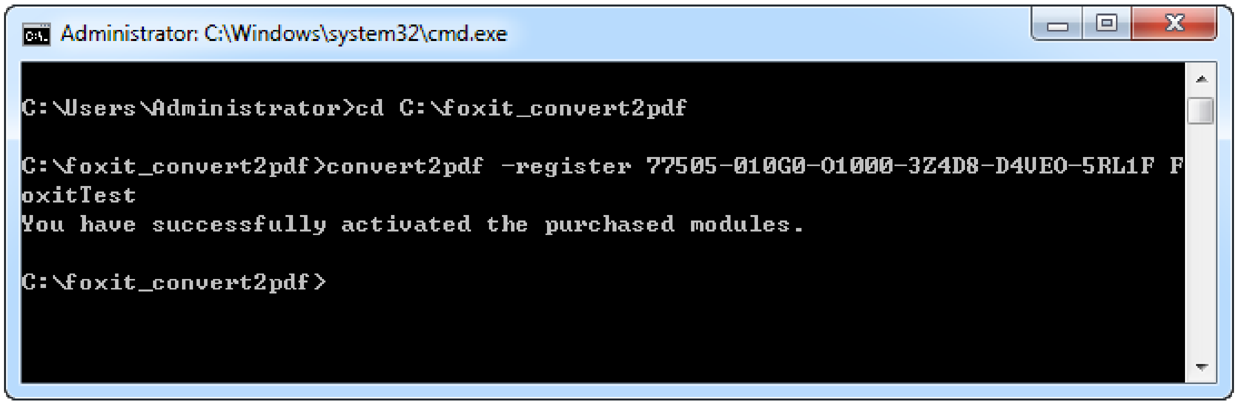


```
C:\Users\Administrator>cd C:\foxit_convert2pdf

C:\foxit_convert2pdf>
```

Navigate to the installation location

Assume you get an activation code (*77505-010G0-O1000-3Z4D8-D4VEO-5RL1F*), you may type "***convert2pdf -register 77505-010G0-O1000-3Z4D8-D4VEO-5RL1F FoxitTest***" in the Command Prompt window as shown below.



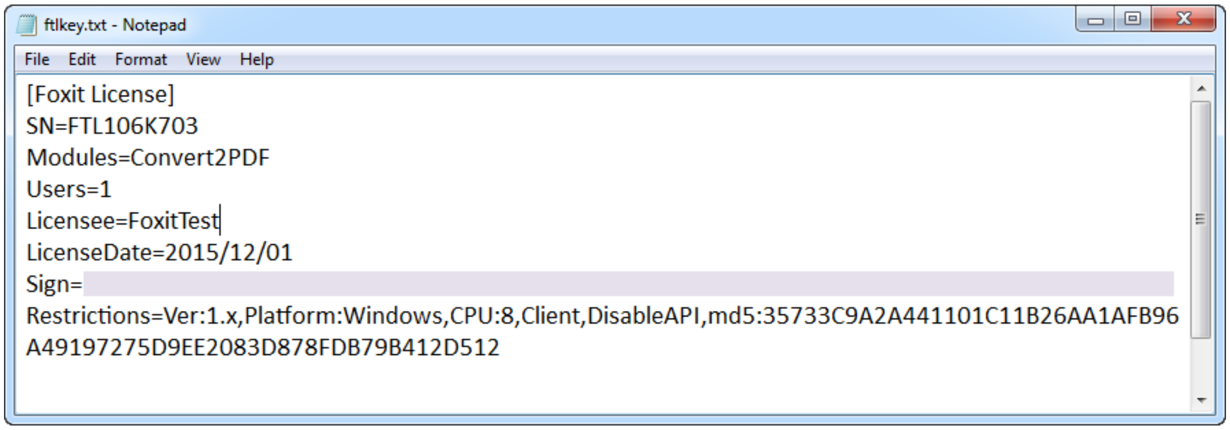
```
C:\Users\Administrator>cd C:\foxit_convert2pdf

C:\foxit_convert2pdf>convert2pdf -register 77505-010G0-01000-3Z4D8-D4VEO-5RL1F F
oxitTest
You have successfully activated the purchased modules.

C:\foxit_convert2pdf>
```

Register Foxit Convert2PDF tool

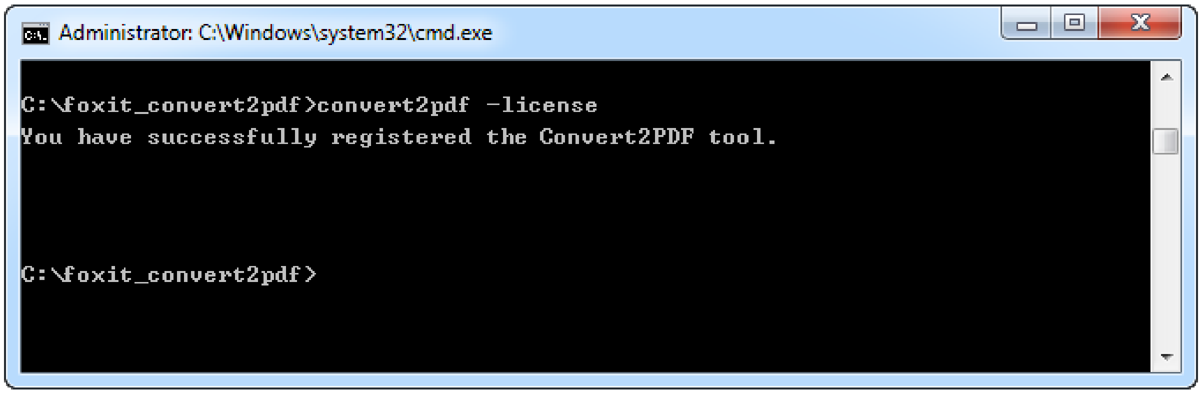
Here, "*FoxitTest*" is the licensee name you input. After activation, a key file named "**ftlkey.txt**" will be generated in the installed path with the contents shown in the following figure.



```
[Foxit License]
SN=FTL106K703
Modules=Convert2PDF
Users=1
Licensee=FoxitTest
LicenseDate=2015/12/01
Sign=
Restrictions=Ver:1.x,Platform:Windows,CPU:8,Client,DisableAPI,md5:35733C9A2A441101C11B26AA1AFB96
A49197275D9EE2083D878FDB79B412D512
```

The content of the generated key file

Then you can run "***-convert2pdf -license***" in the Command Prompt window to check the license agreement as follows.



```
C:\foxit_convert2pdf>convert2pdf -license
You have successfully registered the Convert2PDF tool.



C:\foxit_convert2pdf>
```

Check the license agreement

2.5 Uninstallation

If you want to uninstall the Foxit Convert2PDF tool, all you need to do is to delete the installation folder.

3 Command line usage

3.1 Basic Syntax

The basic command line syntax for Convert2PDF is:

```
convert2pdf <-i <srcfile/srcfolder>> <-o <destfile/destfolder>> [-modetype <mode type>]
            [-width <PDF width>] [-height <PDF height>] [-margin <left [top right bottom]>]
            [-font <fontname>] [-fs <fontsize>] [-fontcolor <R> <G> <B>] [-breakpage]
            [-cache <cache folder>] [-timeout <load timeout>] [-singlepage] [-nolink]
            [-rotate <0/90/180/270>] [-checklazyload] [-cookies <cookie file>] [-js <JavaScript file>]
            [-scale <page scale>] [-op <password>]
            [-r [recursion]] [-t <thread>] [-log <logfile>] [-l <level>]
convert2pdf -register <code> <licensee>
convert2pdf -license
convert2pdf -version/-v
convert2pdf -help/-h
convert2pdf -copyright
```

Note:

- <> required
- [ ] optional
- / mutually exclusive
- A space is needed between the command line argument and the value

Only the *<-i <srcfile/srcfolder>>* and *<-o <destfile/destfolder>>* arguments are actually required. All others are optional, which are available for controlling the conversion as desired. The arguments could be given in any order. Full details on each are explained in the following section.

3.2 Command Line Summary

Note For some arguments whose values are strings, users can choose whether to add *quotation marks (" ") to the strings. In the manual we have added notes where adding quotation marks (" ") is required.*

Option	Parameter	Description
-i	*<-i <string>>* *e.g.* *-i c:\input\1.png* *-i c:\input\2.txt* *-i c:\input\3.html* *-i c:\input\4.docx* *-i c:\input* *-i "c:\input\*.pptx"*	Specifies the input file to be converted. ▪ The input string can be the name of a single file (image, text, HTML, or Office file) or a folder. ▪ The file name can contain the wildcard character (*). For example, use *.txt to include all text files in a given folder. **Note** Wildcard character (*.*) is currently not supported.
-o	*<-o <string>>* *e.g.* *-o d:\output\1.pdf* *-o d:\output*	Specifies the path of the output PDF file or folder. ▪ If the input is a single image, text, HTML or Office file, the output should be a single PDF file, (*e.g. -o d:\output\1.pdf*). ▪ If the input is a folder, the output should be a folder, (*e.g. -o d:\output*). **Note** ▪ The specified output path must already exist. ▪ The output can be a PDF file when the input are images whose names only contain wildcard characters, such as, -i "c:\input\*.jpg" -o d:\output\output.pdf.
-modetype	*[-modetype <string>]* *-modetype <mode type for conversion>* *e.g.* *-modetype "1"* *-modetype "2"* *-modetype "3"* *-modetype "4"* *--modetype "1,2,4"* *-modetype "2,4"* *-modetype "1,2,3,4"*	Specifies the file types to be converted. ● **-modetype "1"**: converts image to PDF. ● **-modetype "2"**: converts Office to PDF. ● **-modetype "3"**: converts text to PDF. ● **-modetype "4"**: converts HTML to PDF. ● **-modetype "1,2"**: converts image and Office to PDF. … **Note** The <string> should be entered in the format "1,4", "1,2,3,4" without any spaces.

Option	Parameter	Description
-width	*[-width <points>]* *e.g.* *-width 612*	Sets the page width of the output PDF file in points. **Note** ▪ The **-width** argument is valid only for image/text/HTML conversion. ▪ For **image conversion**, the default value is the width of input image. The **-width** and **-height** options should be used together with a set value greater than 0. ▪ For **text conversion**, the default value is 595 points with the allowable range of 8-14400 points. ▪ For **HTML conversion**, the default value is 842 points with the allowable range of 16-14400 points.
-height	*[-height <points>]* *e.g.* *-height 792*	Sets the page height of the output PDF file in points. **Note** ▪ The **-height** argument is valid only for image/text/HTML conversion. ▪ For **image conversion**, the default value is the height of input image. The **-width** and **-height** options should be used together with a set value greater than 0. ▪ For **text conversion**, the default value is 842 points with the allowable range of 8-14400 points. ▪ For **HTML conversion**, the default value is 595 points with the allowable range of 16-14400 points.
-margin	*[-margin <points [points points points]>]* *-margin <left [top right bottom]>* *e.g.* *-margin 20* *-margin 10 20*	Sets size of margin for each PDF page in points. Allowable values: 0-size of page in points; in addition, the sum of the left and right values should be less than the width of the page, and the sum of the top and bottom values should be less than the height of the page. ***-margin left top right bottom*** **-margin 20**: sets the left margin to 20 points.

Option	Parameter	Description
	-margin 10 20 0 *-margin 10 20 0 40*	**-margin 10 20**: sets the left margin to 10 points and the top margin to 20 points. **-margin 10 20 0**: sets the left margin to 10 points, the top margin to 20 points, and the right margin to 0 points. **-margin 10 20 0 40**: sets the left margin to 10 points, the top margin to 20 points, the right margin to 0 points, and the bottom margin to 40 points. **Note** ▪ The **-margin** argument is valid only for image/text/HTML conversion. ▪ For **image conversion**, the default value for each margin is 0. ▪ For **text conversion**, the default margin values are 60 72 60 72. ▪ For **HTML conversion**, the default margin values are 10 10 10 10.
-font	*[-font <string>]* *-font <fontname>* *e.g.* *-font "Calibri"* *-font "Helvetica"* *-font "Arial"* *…*	Sets the font style for text conversion. **Note** ▪ The **-font** argument is valid only for text conversion. ▪ The font style you set should be installed on local environment, otherwise the default font style will be used.
-fs	*[-fs <point(pt)>]* *-fs <fontsize>* *e.g.* *-fs 8* *-fs 11* *-fs 72*	Sets the font size for text conversion. Default value: 9pt. Allowable range: 8-72pt. **Note** The **-fs** argument is valid only for text conversion.

Option	Parameter	Description
-fontcolor	*[-fontcolor <integer> <integer> <integer>]* *-fontcolor <R> <G> <B>* *e.g.* *-fontcolor 0 0 0* *-fontcolor 255 255 0* *-fontcolor 255 255 255* ...	Sets the font color for text conversion. By default, the font color of the output PDF is black. Allowable range of the values for each RGB component is 0-255. **Note** The **-fontcolor** argument is valid only for text conversion.
-breakpage	*[-breakpage]* *e.g.* *-breakpage*	If this flag is set, the output PDF file will be paginated wherever there is a page break. **Note** A page break in text files is encoded with the Form Feed (new page) ASCII character (12 (0xC in hexadecimal)). The **-breakpage** argument is valid only for text conversion.
-cache	*[-cache <string>]* *-cache < cache folder>* *e.g.* *-cache "d:\resources"*	Sets cache address to store HTML page resources temporarily. The page resources of the converted webpage will be downloaded and stored to this cache address first, and then will be deleted after conversion. **Note** ▪ The **-cache** argument is valid only for HTML conversion. ▪ If it is not set, a folder named "cache" will be generated in the installation folder.
-timeout	*[-timeout <integer>]* *-timeout <load timeout>* *e.g.* *-timeout 30*	Sets timeout in seconds to load webpages. Default value is 120s. The webpages will not continue to be loaded when the time is used up. **Note** The **-timeout** argument is valid only for HTML conversion, and should be set to a value greater than 15. If users set a value less than 15, the timeout value will be set to 15.
-singlepage	*[-singlepage]* *e.g.* *-singlepage*	Sets all the page contents to one single PDF page. **Note** The **-singlepage** argument is valid only for HTML conversion.

Option	Parameter	Description
-nolink	*[-nolink]* *e.g.* *-nolink*	Converts the input to PDF files with no link annotations retained. **Note** The **-nolink** argument is valid only for HTML conversion.
-rotate	*[-rotate <0/90/180/270>]* *e.g.* *-rotate 0* *-rotate 90* *-rotate 180* *-rotate 270*	Sets page rotation for output PDF files. The value should be 0, 90, 180 or 270. The default value is 0. **Note** The **-rotate** argument is valid only for HTML conversion.
-checklazyload	*[-checklazyload]* *e.g.* *-checklazyload*	Improves the conversion quality if the webpages include lazy loading elements or if the network/hardware performance is not good enough. The tool will spend at least 5 seconds waiting for loading the web elements before starting conversion. **Note** The **-checklazyload** argument is valid only for HTML conversion and it is useful in the following two situations: ▪ Some special long webpages use lazy loading design pattern to make the page load faster and reduce server load. ▪ The performance of the network or the hardware you use is not good enough.
-cookies	*[-cookies <string>]* *e.g.* *-cookies cookie.txt*	Specifies the path of the cookies file which stores the authorization information of a URL that you want to convert.
-js	*[-js <string>]* *e.g.* *-js login.js*	Specifies the path of the JavaScript file which stores some JS scripts that will be applied to webpages to handle some actions, such as login simulation, closing popup window, scrolling to obtain the asynchronous data and more.

Option	Parameter	Description
-scale	*[-scale<integer>]* *e.g.* *-scale 0* *-scale 1* *-scale 2* *-scale 3*	Specifies a conversion mode for Microsoft Excel files. The default is 1. • **-scale 0**: No scaling. Convert sheets at their actual size. • **-scale 1**: fits all columns on one page. Scale every sheet so that it is one page wide. • **-scale 2**: fits all rows on one page. Scale every sheet so that it is one page high. • **-scale 3**: fits sheet on one page. Scale every sheet so that it fits on one page. **Note** This argument is supported on the version higher than Microsoft Office 2007 and it is valid only for Microsoft Excel files.
-op	*[-op<string>]* *e.g.* *-op welcome* *-op 123456*	Specifies the open password for the input file. Not required if the input file is not password protected. **Note** ▪ The **-op** argument is valid only for Office conversion. ▪ The output PDF file will not retain the open password from the input file.
-r	*[-r [integer]]* *e.g.* *-r* *-r 0* *-r 1* *-r 2* …	Specifies the number of layers to recurse when the input is a folder. • **-r 0 <-r>**: searches the full folders. • **-r 1**: searches only the current folder. • **-r 2**: searches the current folder and its sub-folders … **Note** ▪ If no integer value is specified, or if the integer value is 0, then full folders will be searched. By default, the number of layers to recurse is 1, which means that only the current folder will be searched and not sub-folders.

Option	Parameter	Description
		▪ The input folder or Office file will be skipped if it is secured and the messages will be displayed.
-t	*[-t <integer>]* *e.g.* *-t 1* *-t 2* …	Specifies the number of CPU threads to use. The default value is 1.
-log	*[-log <string>]* *e.g.* *-log d:\a.log*	Writes log information into a logfile at the specified existing path.
-l	*[-l <integer>]* *e.g.* *-l 1* *-l 2* *-l 3* *-l 4*	Sets the log level. The default is 4. ● **-l 1**: logs messages only concerning program crashes. ● **-l 2**: logs failure messages concerning the errors caused during execution or those returned from underlying libraries, as well as those for level 1. ● **-l 3**: logs warning messages concerning the PDF files that are overwritten, as well as those for level 2. ● **-l 4**: logs informational messages, as well as those for level 3. **Note** The argument (**-l**) is valid only when (**-log**) is used.
-register	*[-register <String> <String>]* *-register <code> <licensee>*	Registers the command line tool. ▪ **<code>**: the activation code from Foxit. ▪ **<licensee>**: the Licensee name designated by the users.
-help/-h	*[-help]/[-h]* *e.g.* *-help* *-h*	Prints the usage information.
-version/-v	*[-version]/[-v]* *e.g.*	Prints the version information.

Option	Parameter	Description
	-version *-v*	
-license	*[-license]* *e.g.* *-license*	Prints the license agreement.
-copyright	*[-copyright]* *e.g.* *-copyright*	Prints the copyright information.

3.3 Basic Usage

3.3.1 Input and Output

a) Input (-i)

- The input file should be a single file (image, text, HTML, or Office file), a URL, or a folder. Users are not able to input multiple files or folders, as well as a mixture composed of folders and files. For example:

```
-i c:\input\1.jpg                    (a single image file)
-i c:\input\2.txt                    (a single text file)
-i c:\input\3.html                   (a single HTML file)
-i c:\input\4.docx                   (a single Office file)
-i "www.foxitsoftware.com"           (a URL)
-i c:\input                          (a single folder)
```

- It supports relative paths if the input file is in the current working folder. Users can input just the name of the file or folder, instead of an absolute path. For example:

```
-i test\1.jpg                        ("test" folder is in the current working folder)
-i test\2.txt
-i test\3.html
-i test\4.docx
-i test
```

- It also supports wildcard characters, which are used to process multiple files. For example:

```
-i "c:\input\*.png"                  (Only convert PNG files under "c:\input" folder)
```

-i "c:\input\*.txt"	(Only convert TEXT files under "c:\input" folder)
-i "test*.html"	(Only convert HTML files under "test" folder)
-i "test*.docx"	(Only convert DOCX files under "test" folder)
-i "c:\input\*.bmp,*.html"	(Only convert BMP and HTML files under "c:\input" folder)
-i "test*.jpg,*.txt,*.doc"	(Only convert JPG, TEXT and DOC files under "test" folder)

Note *When using wildcard characters in the input files, it is recommended to enclose the input files with quotation marks (" "). In this manual, we add (" ") whenever the input files contain wildcard characters.*

b) Output (-o)

- If the input is a single file, you should specify the output path of a PDF file. If the input is a single folder or a file whose name only contains wildcard characters, you can only specify the output folder. (Note: The output can also be a PDF file when the input are images whose names only contains wildcard characters, such as, -i "c:\input\*.jpg" -o d:\output\output.pdf). For example:

-o d:\output\output.pdf	(a single PDF file)
-o d:\output	(a single folder)

Note *The specified output path must already exist.*

- The output also supports relative paths if the specified output location is in the current working folder. Users can input just the name of the output PDF file or folder, instead of an absolute path. For example:

-o output\output.pdf	("output" folder is in the current working folder)
-o output	("output" folder is in the current working folder)

Usage Examples

1) Convert a single image file to a PDF file:

```
convert2pdf -i c:\input\image_1.jpg -o d:\output\image_1.pdf
convert2pdf -i test\image_2.tif -o output\image_2.pdf
```

2) Convert a single text file to a PDF file:

```
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf
convert2pdf -i test\text_2.txt -o output\text_2.pdf
```

3) Convert a single HTML file or a webpage to a PDF file:

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf
convert2pdf -i test\html_2.html -o output\html_2.pdf
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf
```

4) Convert a single Office file to a PDF file:

```
convert2pdf -i c:\input\doc_1.doc -o d:\output\doc_1.pdf
convert2pdf -i test\pptx_1.txt -o output\pptx_1.pdf
```

5) Convert the files (image/text/HTML/Office files) in a folder to PDF files:

```
convert2pdf -i c:\input -o d:\output
convert2pdf -i test -o output
```

6) Convert only PNG files in a folder to PDF files:

```
convert2pdf -i "c:\input\*.png" -o d:\output
convert2pdf -i "test\*.png" -o output.pdf
```

7) Convert only BMP, DOC and TEXT files in a folder to PDF files:

```
convert2pdf -i "c:\input\*.bmp,*.doc,*.txt" -o d:\output
convert2pdf -i "test\*.bmp,*.doc,*.txt" -o output
```

3.3.2 File Types Selection for Conversion

- The optional argument (**-modetype**) is used to specify the file types to be converted. If this argument is not set or if it is set to "1,2,3,4", all the files (image, text, HTML, and Office) will be converted. For more details about this argument, please refer to section 3.2 "Command Line Summary".

Usage Example

1) Convert image files to PDF files: (-modetype "1")

```
convert2pdf -i c:\input -o d:\output -modetype "1"
convert2pdf -i test -o output -modetype "1"
```

2) Convert text and HTML files to PDF files: (-modetype "3,4")

```
convert2pdf -i c:\input -o d:\output -modetype "3,4"
convert2pdf -i test -o output -modetype "3,4"
```

3) Convert Office, text and HTML files to PDF files: (-modetype "2,3,4")

```
convert2pdf -i c:\input -o d:\output -modetype "2,3,4"
convert2pdf -i test -o output -modetype "2,3,4"
```

3.3.3 Settings for Image, Text, and HTML Conversion

a) Page size setting (-width, -height)

- The optional arguments (**-width**) and (**-height**) are used to set the page width and height for the output PDF file in points. They are valid only for image, text, and HTML conversion. For image conversion, the default value is the width and height of the input image; for text conversion, the default width and height are 595 and 842 points respectively with the allowable range of 8-14400 points; for HTML conversion, the default width and height are 842 and 595 points respectively with the allowable range of 16-14400 points.

  Note *For image conversion, the -width and -height options should be used together with a set value greater than 0.*

Usage Example

1) Set the page width and height to 400 and 300 for the output PDF file (-width 400 -height 300)

```
convert2pdf -i c:\input\image_1.jpg -o d:\output\image_1.pdf -width 400 -height 300
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -width 400 -height 300
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -width 400 -height 300
convert2pdf -i c:\input -o d:\output -width 400 -height 300
convert2pdf -i test -o output -width 400 -height 300
convert2pdf -i "c:\input\*.png" -o d:\output -width 400 -height 300
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -width 400 -height 300
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -width 400 -height 300
```

b) Margin setting (-margin)

- The optional argument (-margin) is used to set size of margin for each PDF page in points. It is valid only for image, text, and HTML conversion. For image conversion, the default value for each margin is 0; for text conversion, the default values are 60 72 60 72; and for HTML conversion, the default values are 10 10 10 10. For more details about this argument, please refer to section 3.2 "Command Line Summary".

Note *The sum of the left and right values should be less than the width of the page, and the sum of the top and bottom values should be less than the height of the page.*

Usage Example

1) Set the left margin to 20 points (-margin 20)

```
convert2pdf -i c:\input\image_1.jpg -o d:\output\image_1.pdf -margin 20
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -margin 20
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -margin 20
convert2pdf -i c:\input -o d:\output -margin 20
convert2pdf -i test -o output -margin 20
convert2pdf -i "c:\input\*.png" -o d:\output -margin 20
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -margin 20
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -margin 20
```

2) Set the left margin to 20 points, and the top margin to 10 points (-margin 20 10)

```
convert2pdf -i c:\input\image_1.jpg -o d:\output\image_1.pdf -margin 20 10
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -margin 20 10
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -margin 20 10
convert2pdf -i c:\input -o d:\output -margin 20 10
convert2pdf -i test -o output -margin 20 10
convert2pdf -i "c:\input\*.png" -o d:\output -margin 20 10
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -margin 20 10
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -margin 20 10
```

3) Set the left margin to 10 points, the top margin to 10 points and the right margin to 30 points (-margin 10 10 30)

```
convert2pdf -i c:\input\image_1.jpg -o d:\output\image_1.pdf -margin 10 10 30
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -margin 10 10 30
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -margin 10 10 30
convert2pdf -i c:\input -o d:\output -margin 10 10 30
convert2pdf -i test -o output -margin 10 10 30
convert2pdf -i "c:\input\*.png" -o d:\output -margin 10 10 30
```

```
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -margin 10 10 30
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -margin 10 10 30
```

4) Set the left margin to 10 points, the top margin to 10 points, the right margin to 30 points and the bottom margin to 20 points (-margin 10 10 30 20)

```
convert2pdf -i c:\input\image_1.jpg -o d:\output\image_1.pdf -margin 10 10 30 20
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -margin 10 10 30 20
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -margin 10 10 30 20
convert2pdf -i c:\input -o d:\output -margin 10 10 30 20
convert2pdf -i test -o output -margin 10 10 30 20
convert2pdf -i "c:\input\*.png" -o d:\output -margin 10 10 30 20
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -margin 10 10 30 20
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -margin 10 10 30 20
```

3.3.4 Settings for Text Conversion

a) Text font (-font)

- The optional arguments (**-font**) is used to set the font style for text conversion. It is valid only for text conversion.

  Note *The font style you set should be installed on a local environment, otherwise the default font style will be used.*

Usage Example

1) Set the font style to "Calibri" for text conversion (-font "Calibri")

```
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -font "Calibri"
convert2pdf -i test\text_2.txt -o output\text_2.pdf -font "Calibri"
convert2pdf -i c:\input -o d:\output -font "Calibri"
convert2pdf -i test -o output -font "Calibri"
convert2pdf -i "c:\input\*.txt" -o d:\output -font "Calibri"
```

b) Text font size (-fs)

- The optional argument (**-fs**) is used to set the font size for text conversion. It is valid only for text conversion. The default value is set to 9pt, and the allowable range is from 8 to 72pt.

Usage Example

1) Set the font size to 12pt for text conversion (-fs 12)

```
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -fs 12
convert2pdf -i test\text_2.txt -o output\text_2.pdf -fs 12
convert2pdf -i c:\input -o d:\output -fs 12
convert2pdf -i test -o output -fs 12
convert2pdf -i "c:\input\*.txt" -o d:\output -fs 12
```

c) Text font color (-fontcolor)

- The optional argument (**-fontcolor**) is used to set the font color for text conversion. It is valid only for text conversion. By default, the font color is black. The allowable range of the values for each RGB component is from 0 to 255.

Usage Example

1) Set the font color to blue (-fontcolor 0 0 255)

```
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -fontcolor 0 0 255
convert2pdf -i test\text_2.txt -o output\text_2.pdf -fontcolor 0 0 255
convert2pdf -i c:\input -o d:\output -fontcolor 0 0 255
convert2pdf -i test -o output -fontcolor 0 0 255
convert2pdf -i "c:\input\*.txt" -o d:\output -fontcolor 0 0 255
```

d) Support pagination (-breakpage)

- The optional argument (**-breakpage**) indicates the output PDF file will be paginated wherever there is a page break.

  Note *A page break in text files is encoded with the Form Feed (new page) ASCII character (12 (0xC in hexadecimal)).*

Usage Example

1) Support pagination wherever there is a page break (-breakpage)

```
convert2pdf -i c:\input\text_1.txt -o d:\output\text_1.pdf -breakpage
convert2pdf -i test\text_2.txt -o output\text_2.pdf -breakpage
```

```
convert2pdf -i c:\input -o d:\output -breakpage
convert2pdf -i test -o output -breakpage
convert2pdf -i "c:\input\*.txt" -o d:\output -breakpage
```

3.3.5 Settings for HTML Conversion

a) Cache address (-cache)

- The optional argument (**-cache**) is used to set cache address to store HTML page resources temporarily. The page resources of the converted webpage will be downloaded and stored to this cache address first, and then will be deleted after conversion. If this argument is not set, a folder named "cache" will be generated in the installation folder. It is valid only for HTML conversion.

Usage Example

1) Set cache address to "*d:\resources*" (-cache "d:\resources")

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -cache "d:\resources"
convert2pdf -i test\html_2.html -o output\html_2.pdf -cache "d:\resources"
convert2pdf -i c:\input -o d:\output -cache "d:\resources"
convert2pdf -i test -o output -cache "d:\resources"
convert2pdf -i "c:\input\*.html" -o d:\output -cache "d:\resources"
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -cache "d:\resources"
```

b) Timeout for loading (-timeout)

- The optional argument (**-timeout**) is used to set timeout in seconds to load webpages. The webpages will not continue to be loaded when the time is used up. The default value is 120s, and the timeout value will be set to 15 if users set a value less than 15. This argument is valid only for HTML conversion.

Usage Example

1) Set the timeout to "200s" to load webpages (-timeout 200)

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -timeout 200
convert2pdf -i test\html_2.html -o output\html_2.pdf -timeout 200
convert2pdf -i c:\input -o d:\output -timeout 200
convert2pdf -i test -o output -timeout 200
convert2pdf -i "c:\input\*.html" -o d:\output -timeout 200
```

```
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -timeout 200
```

c) Single page (-singlepage)

- The optional argument (**-singlepage**) is used to set all the page contents to one single PDF page. It is valid only for HTML conversion.

Usage Example

1) Set all the page contents to one single PDF page (-singlepage)

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -singlepage
convert2pdf -i test\html_2.html -o output\html_2.pdf -singlepage
convert2pdf -i c:\input -o d:\output -singlepage
convert2pdf -i test -o output -singlepage
convert2pdf -i "c:\input\*.html" -o d:\output -singlepage
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -singlepage
```

d) Disable retaining hyperlinks (-nolink)

- The optional argument (**-nolink**) is used to convert the input to PDF files with no link annotations retained. If users set this argument, no action will be triggered when they click the links in the output PDF file.

Usage Example

1) Convert the input to PDF files with no link annotations (-nolink)

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -nolink
convert2pdf -i test\html_2.html -o output\html_2.pdf -nolink
convert2pdf -i c:\input -o d:\output -nolink
convert2pdf -i test -o output -nolink
convert2pdf -i "c:\input\*.html" -o d:\output -nolink
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -nolink
```

e) Page rotation (-rotate)

- The optional argument (**-rotate**) is used to set page rotation for the output PDF files. The setting value should be 0, 90, 180 or 270 and the default value is 0. This argument is valid only for HTML conversion.

Usage Example

1) Set page rotation to 90 degree (-rotate 90)

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -rotate 90
convert2pdf -i test\html_2.html -o output\html_2.pdf -rotate 90
convert2pdf -i c:\input -o d:\output -rotate 90
convert2pdf -i test -o output -rotate 90
convert2pdf -i "c:\input\*.html" -o d:\output -rotate 90
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -rotate 90
```

f) Check lazy load (-checklazyload)

➢ The optional argument (**-checklazyload**) is used to improve the conversion quality if the webpages include lazy loading elements or if the network/hardware performance is not good enough. This argument is valid only for HTML conversion.

Note *The **-checklazyload** argument is useful in the following two situations:*

- *Some special long webpages use lazy loading design pattern to make the page load faster and reduce server load, therefore some web elements are designed to be delayed loading, which will affect the conversion quality.*
- *If the network or hardware performance is not good enough, the web elements loading will be influenced, which will also affect the conversion quality.*

If this argument is set, the tool will spend at least 5 seconds waiting for loading the web elements before starting conversion, which can help improve the conversion quality.

Usage Example

1) Improve the conversion quality if the webpages include lazy loading elements or if the network/hardware performance is not good enough (-checklazyload)

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -rotate 90
convert2pdf -i test\html_2.html -o output\html_2.pdf -rotate 90
convert2pdf -i c:\input -o d:\output -rotate 90
convert2pdf -i test -o output -rotate 90
convert2pdf -i "c:\input\*.html" -o d:\output -rotate 90
convert2pdf -i "www.foxitsoftware.com" -o output –checklazyload
```

g) Cookie File (-cookies)

- The optional argument (**-cookies**) is used to specify the path of the cookies file which stores the authorization information of a URL that you want to convert.

  The **-cookies** argument is useful for the online webpages (except for some financial websites) that require authorization information, such as username, user password, and some other verification code which are required when you sign in the website. To convert such webpages, users should sign in the website first, export the cookie file from the webpage, and then do the conversion with the exported cookie file.

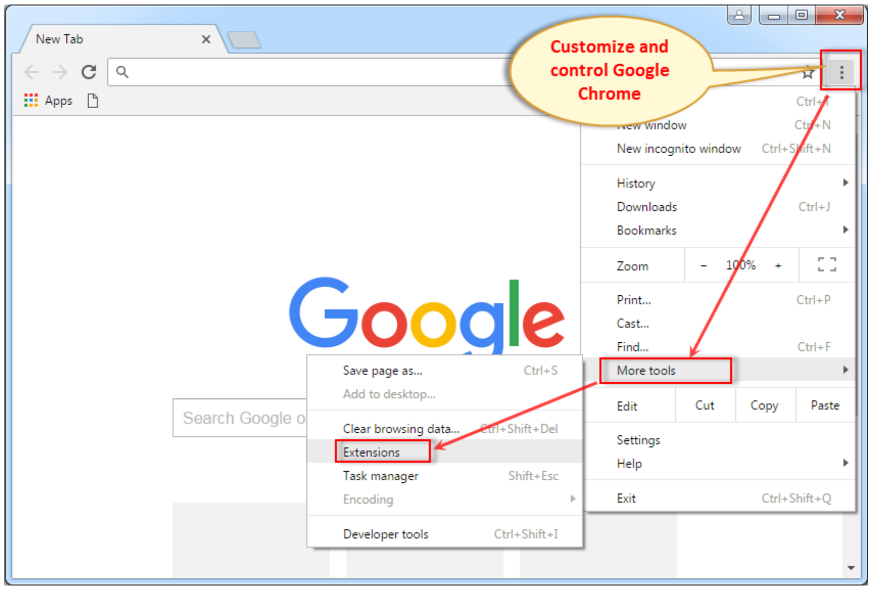
  For example, for an e-book store, one user wants to convert the favorite page into PDF. It is evident that the user should sign in the e-book store first, then he/she can browse his/her favorite. That means the favorite page requires authorization information, so the user should export the cookie file from the favorite page, and then convert the page to PDF using the "-cookies" argument with the exported cookie file. Otherwise, maybe only the login page will be converted into PDF.

  How to export the cookie file from a login webpage?

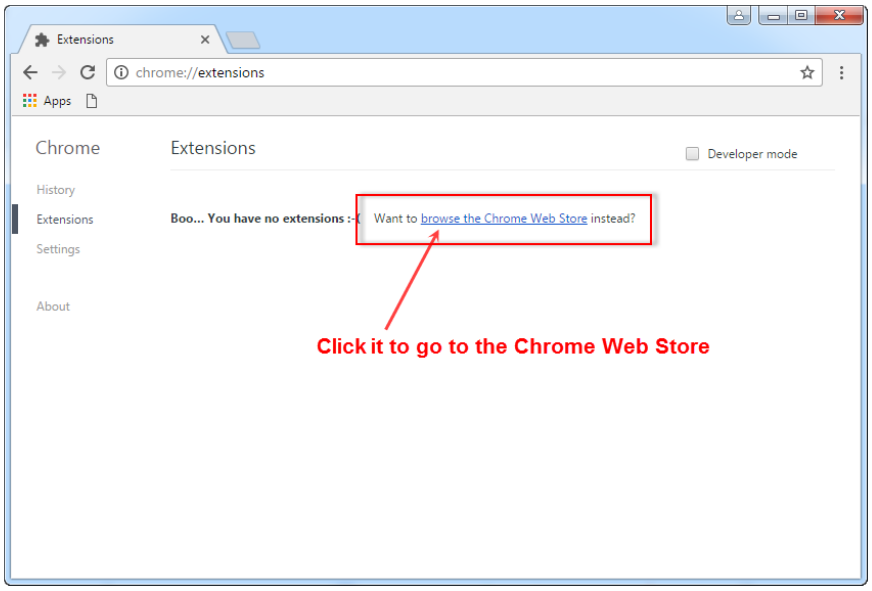
  Most browsers have already provided a plugin called "Edit This Cookie". It is a cookie manager that allows you to add, delete, edit, search, protect and block cookies. In this manual, it takes Chrome browser as an example to show you how to install the plugin and how to export the cookie file from a login webpage.

  To install the plugin, follow the steps below:

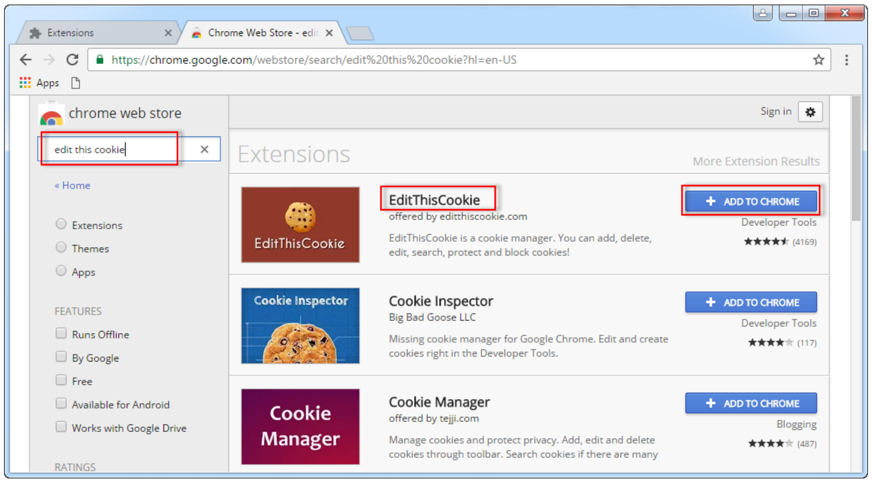
  a) Open Google Chrome browser, click the ⋮ (Customize and control Google Chrome) button, and then select **More tools -> Extensions** as shown in the following figure.



b) Click on **browse the Chrome Web Store** to go to the extensions store as follows.


Extensions
chrome://extensions
Apps
Chrome
Extensions
Developer mode
History
Extensions
Settings
About
Boo... You have no extensions :-(
Want to browse the Chrome Web Store instead?
Click it to go to the Chrome Web Store

c) In the store, type the "edit this cookie" to search for the plugin, and then add it to Chrome as shown in the following figure.

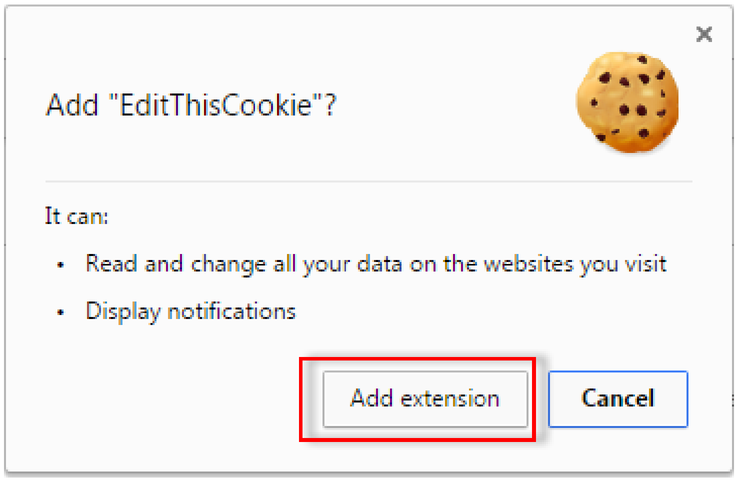


EditThisCookie
offered by editthiscookie.com
EditThisCookie is a cookie manager. You can add, delete, edit, search, protect and block cookies!
Developer Tools (4169)

Cookie Inspector
Big Bad Goose LLC
Missing cookie manager for Google Chrome. Edit and create cookies right in the Developer Tools.
Developer Tools (117)

Cookie Manager
offered by tejji.com
Manage cookies and protect privacy. Add, edit and delete cookies through toolbar. Search cookies if there are many
Blogging (487)

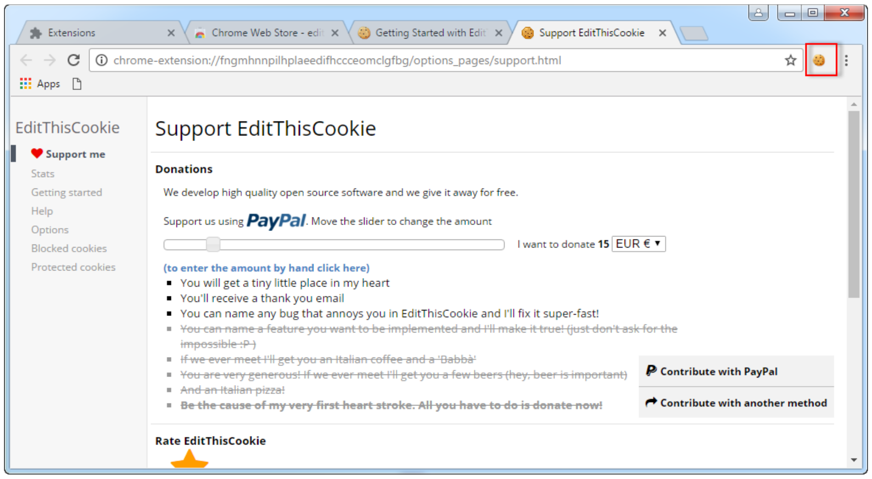
d) A window will popup asking you whether to add this extension. Click **Add extension**.


Add "EditThisCookie"?

It can:

- Read and change all your data on the websites you visit
- Display notifications

e) After adding, the plugin will appear on the top toolbar as shown in the following figure.



Support EditThisCookie

Donations

We develop high quality open source software and we give it away for free.

Support us using PayPal. Move the slider to change the amount

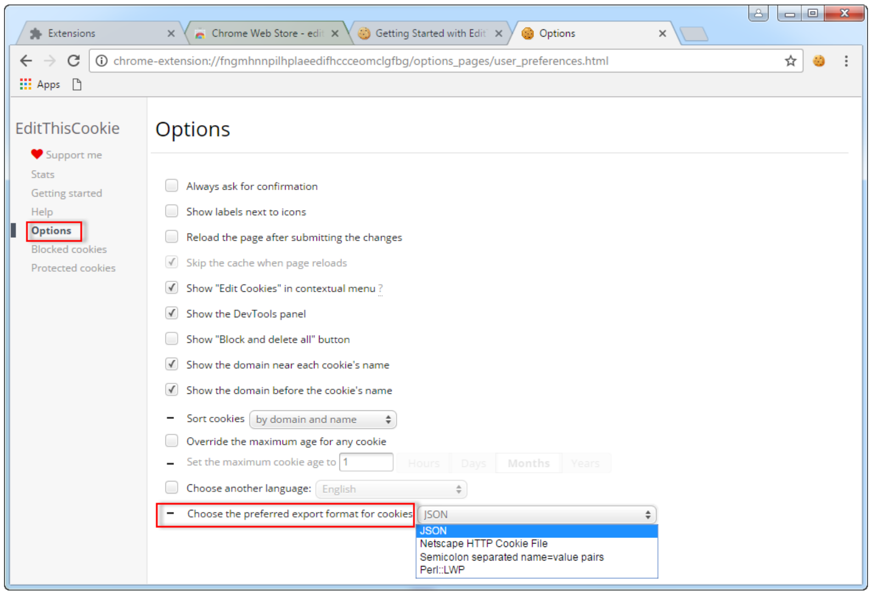
I want to donate **15** EUR €

(to enter the amount by hand click here)

- You will get a tiny little place in my heart
- You'll receive a thank you email
- You can name any bug that annoys you in EditThisCookie and I'll fix it super-fast!
- ~~You can name a feature you want to be implemented and I'll make it true! (just don't ask for the impossible :P)~~
- ~~If we ever meet I'll get you an italian coffee and a 'Babbà'~~
- ~~You are very generous! If we ever meet I'll get you a few beers (hey, beer is important)~~
- ~~And an italian pizza!~~
- ~~**Be the cause of my very first heart stroke. All you have to do is donate now!**~~

Rate EditThisCookie

***Note** Foxit Convert2PDF currently supports three types of export format for cookies: **JSON**, **Semicolon separated name=value pairs** and **Perl::LWP**. The default and recommended format is JSON. If you want to change the format, click on **Options**, find **choose the preferred export format for cookies**, and then choose the format you wish as shown in the following figure.*



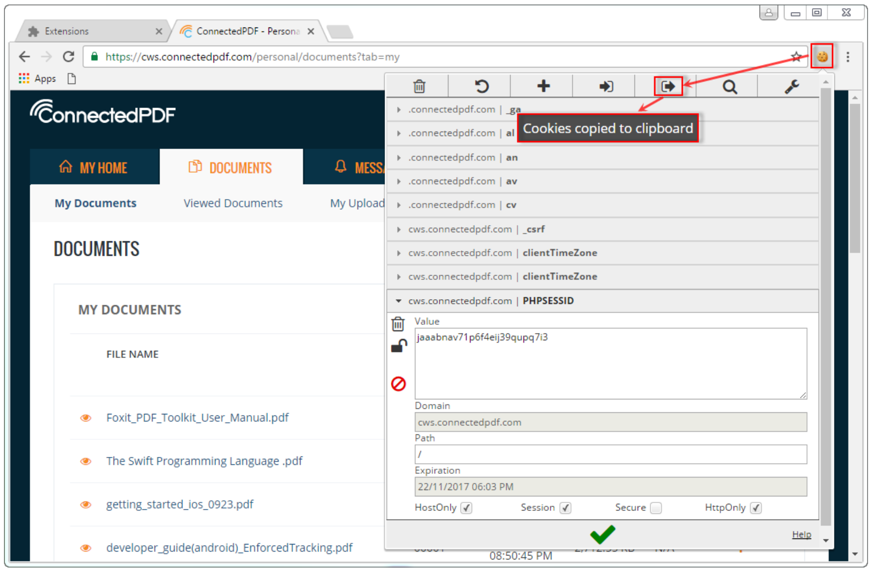
Options

- [ ] Always ask for confirmation
- [ ] Show labels next to icons
- [ ] Reload the page after submitting the changes
- [x] Skip the cache when page reloads
- [x] Show "Edit Cookies" in contextual menu ?
- [x] Show the DevTools panel
- [ ] Show "Block and delete all" button
- [x] Show the domain near each cookie's name
- [x] Show the domain before the cookie's name
- Sort cookies: by domain and name
- [ ] Override the maximum age for any cookie
- Set the maximum cookie age to 1 Hours Days Months Years
- [ ] Choose another language: English
- Choose the preferred export format for cookies: JSON
  - [✓ JSON]
  - Netscape HTTP Cookie File
  - Semicolon separated name=value pairs
  - Perl::LWP

***To export the cookie file from a login webpage, follow the steps below**:*

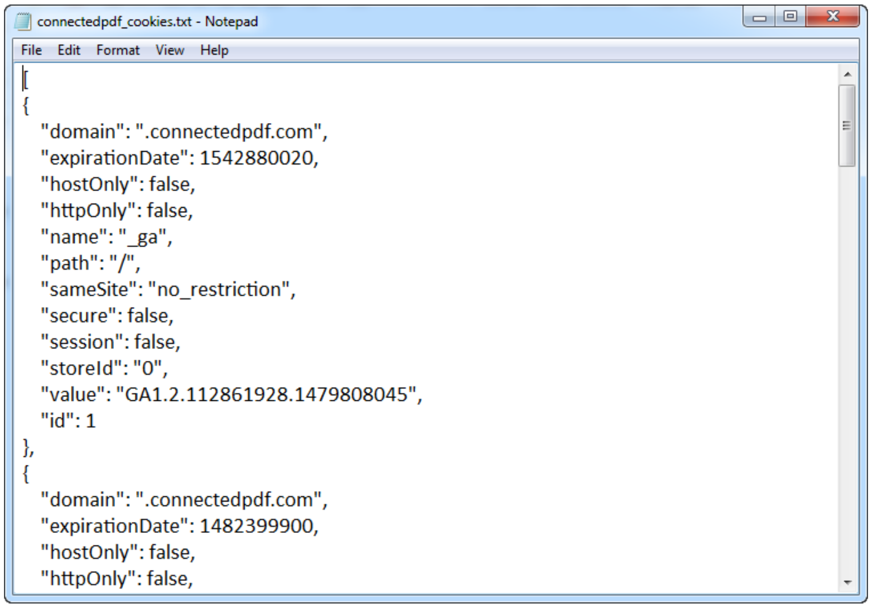
a) Open the webpage you want to convert, making sure the webpage requires authorization information. In this manual, it takes Foxit ConnectedPDF personal document page as an example: https://cws.connectedpdf.com/personal/documents?tab=my (I have already signed in this website).

b) Click the plug in the top toolbar, and then choose the export button as shown in the following figure. After clicking the export button, it will prompt that cookies copied to clipboard.


Extensions
ConnectedPDF - Person
https://cws.connectedpdf.com/personal/documents?tab=my
Apps
ConnectedPDF
MY HOME
DOCUMENTS
MESS
My Documents
Viewed Documents
My Upload
DOCUMENTS
MY DOCUMENTS
FILE NAME
Foxit_PDF_Toolkit_User_Manual.pdf
The Swift Programming Language .pdf
getting_started_ios_0923.pdf
developer_guide(android)_EnforcedTracking.pdf
Cookies copied to clipboard
.connectedpdf.com | _ga
.connectedpdf.com | al
.connectedpdf.com | an
.connectedpdf.com | av
.connectedpdf.com | cv
cws.connectedpdf.com | _csrf
cws.connectedpdf.com | clientTimeZone
cws.connectedpdf.com | clientTimeZone
cws.connectedpdf.com | PHPSESSID
Value
jaaabnav71p6f4eij39qupq7i3
Domain
cws.connectedpdf.com
Path
/
Expiration
22/11/2017 06:03 PM
HostOnly
Session
Secure
HttpOnly
Help
08:50:45 PM

c) Create a txt file and paste the cookies into the txt file. For example, in the installation folder, create a txt file called connectedpdf_cookies.txt, and then paste the exported cookies into it.

Some contents of the cookies are displayed in the following figure with JSON format. Now, the cookies for the Foxit ConnectedPDF personal document page have been exported successfully.



```
[
{
   "domain": ".connectedpdf.com",
   "expirationDate": 1542880020,
   "hostOnly": false,
   "httpOnly": false,
   "name": "_ga",
   "path": "/",
   "sameSite": "no_restriction",
   "secure": false,
   "session": false,
   "storeId": "0",
   "value": "GA1.2.112861928.1479808045",
   "id": 1
},
{
   "domain": ".connectedpdf.com",
   "expirationDate": 1482399900,
   "hostOnly": false,
   "httpOnly": false,
```

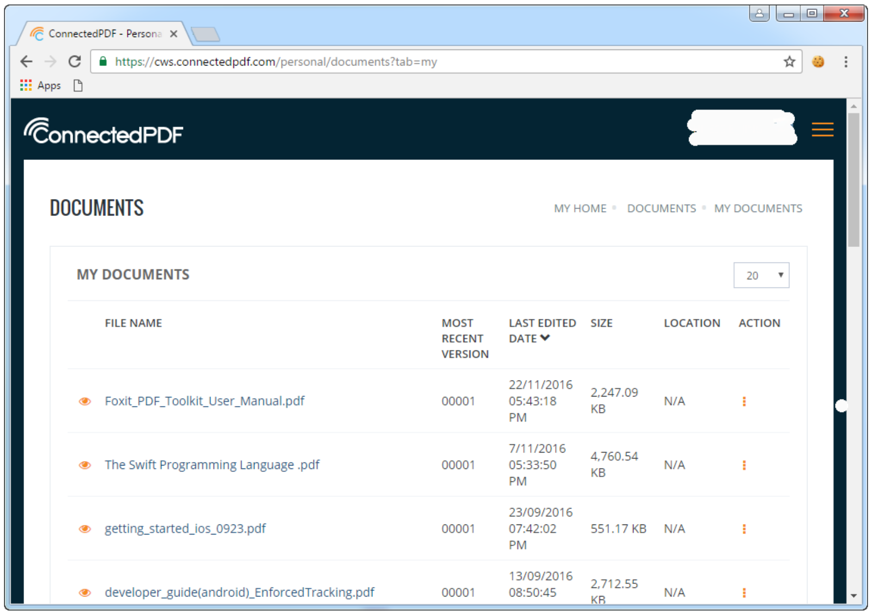
Usage Example

1) Convert a URL that requires login into PDF file (-cookies connectedpdf_cookies.txt)

convert2pdf -i "https://cws.connectedpdf.com/personal/documents?tab=my" -o d:\output -cookies connectedpdf_cookies.txt

Conversion Result

The Foxit ConnectedPDF personal document page that I want to convert: (only capture a part of the page)



[redacted]

DOCUMENTS

MY HOME • DOCUMENTS • MY DOCUMENTS

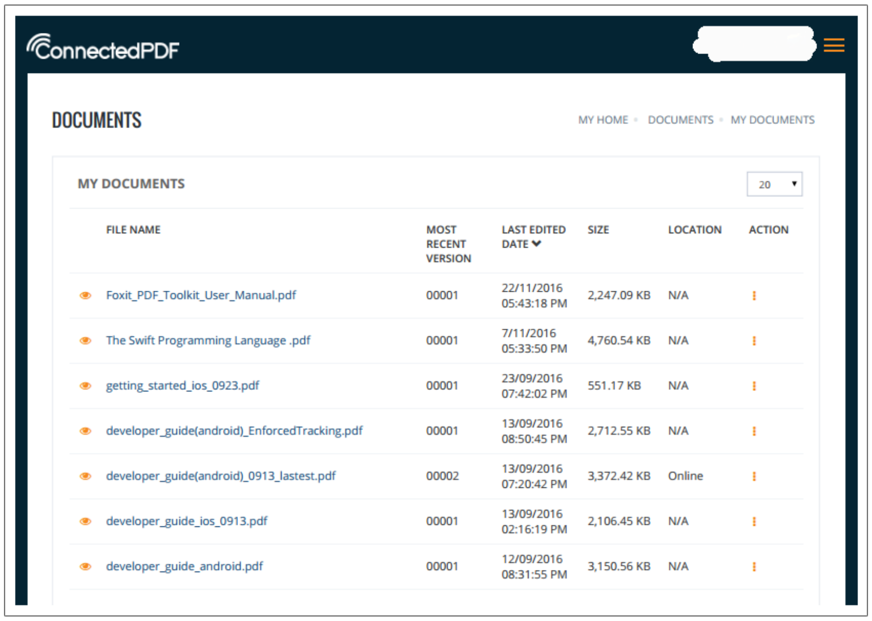
MY DOCUMENTS

20

	FILE NAME	MOST RECENT VERSION	LAST EDITED DATE	SIZE	LOCATION	ACTION
	Foxit_PDF_Toolkit_User_Manual.pdf	00001	22/11/2016 05:43:18 PM	2,247.09 KB	N/A	⋮
	The Swift Programming Language .pdf	00001	7/11/2016 05:33:50 PM	4,760.54 KB	N/A	⋮
	getting_started_ios_0923.pdf	00001	23/09/2016 07:42:02 PM	551.17 KB	N/A	⋮
	developer_guide(android)_EnforcedTracking.pdf	00001	13/09/2016 08:50:45	2,712.55 KB	N/A	⋮

Convert the above URL into PDF file with "-cookies" argument: (only capture the first page of the PDF file)

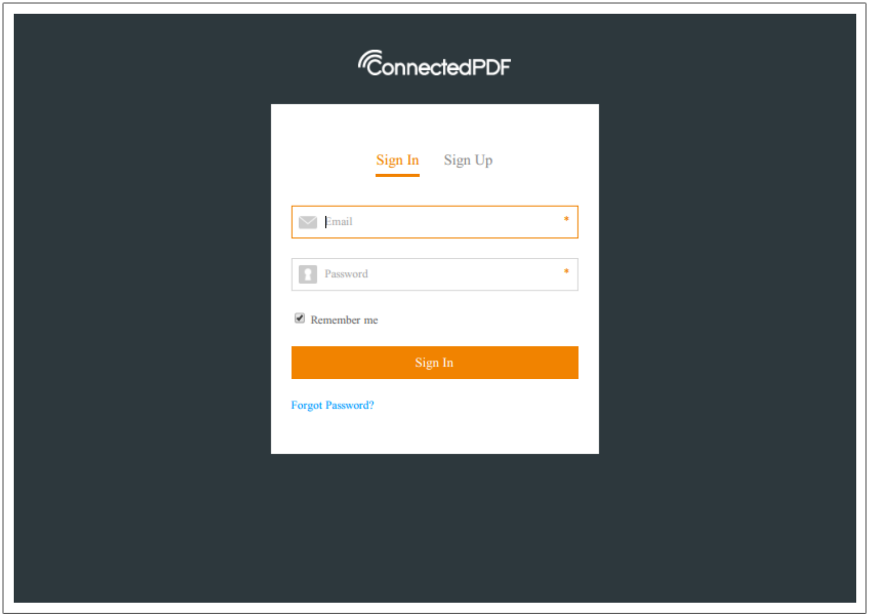
convert2pdf -i "https://cws.connectedpdf.com/personal/documents?tab=my" -o d:\output -cookies connectedpdf_cookies.txt



	FILE NAME	MOST RECENT VERSION	LAST EDITED DATE	SIZE	LOCATION	ACTION
👁	Foxit_PDF_Toolkit_User_Manual.pdf	00001	22/11/2016 05:43:18 PM	2,247.09 KB	N/A	⋮
👁	The Swift Programming Language .pdf	00001	7/11/2016 05:33:50 PM	4,760.54 KB	N/A	⋮
👁	getting_started_ios_0923.pdf	00001	23/09/2016 07:42:02 PM	551.17 KB	N/A	⋮
👁	developer_guide(android)_EnforcedTracking.pdf	00001	13/09/2016 08:50:45 PM	2,712.55 KB	N/A	⋮
👁	developer_guide(android)_0913_lastest.pdf	00002	13/09/2016 07:20:42 PM	3,372.42 KB	Online	⋮
👁	developer_guide_ios_0913.pdf	00001	13/09/2016 02:16:19 PM	2,106.45 KB	N/A	⋮
👁	developer_guide_android.pdf	00001	12/09/2016 08:31:55 PM	3,150.56 KB	N/A	⋮

Convert the above URL into PDF file without "-cookies" argument:

```
convert2pdf -i "https://cws.connectedpdf.com/personal/documents?tab=my" -o d:\output
```



h) JavaScript File (-js)

➢ The optional argument (**-js**) is used to specify the path of the JavaScript file which stores some JS scripts that will be applied to webpages to handle some actions, such as login simulation, closing popup window, scrolling to obtain the asynchronous data and more.

Note

- *It supports native DOM/BOM API, and currently can only support ECMAScript 5.1.*
- *For login simulation, it is only valid for the websites that are not related to finance and do not need verification code.*

Usage Example

1) Apply JavaScript to webpages (-js login.js)

```
convert2pdf -i c:\input\html_1.html -o d:\output\html_1.pdf -js login.js
convert2pdf -i test\html_2.html -o output\html_2.pdf -js login.js
convert2pdf -i c:\input -o d:\output -js login.js
convert2pdf -i test -o output -js login.js
convert2pdf -i "c:\input\*.html" -o d:\output -js login.js
convert2pdf -i "www.foxitsoftware.com" -o d:\output\foxit.pdf -js login.js
```

In the "samples\javascript" folder of the installation path, Foxit Convert2PDF provides several JavaScript samples which includes the actions about login simulation, closing popup window and scrolling to obtain the asynchronous data. You can refer to them to write the JavaScript that you want.

3.3.6 Settings for Office Conversion

a) Scale (-scale)

- The optional argument (**-scale**) is used to specify a conversion mode for Microsoft Excel files. For more details about this argument, please refer to the section 3.2 "Command Line Summary".

  Note *This argument is supported on the version higher than Microsoft Office 2007 and it is valid only for Microsoft Excel files conversion.*

Usage Example

1) Convert sheets at their actual size (-scale 0)

```
convert2pdf -i c:\input\1.xlsx -o d:\output\1.pdf -scale 0
convert2pdf -i c:\input -o d:\output -scale 0
convert2pdf -i test -o output -scale 0
convert2pdf -i "c:\input\*.xls" -o d:\output -scale 0
```

2) Fit all columns on one page (-scale 1)

```
convert2pdf -i c:\input\1.xlsx -o d:\output\1.pdf -scale 1
convert2pdf -i c:\input -o d:\output -scale 1
convert2pdf -i test -o output -scale 1
convert2pdf -i "c:\input\*.xls" -o d:\output -scale 1
```

3) Fit all rows on one page (-scale 2)

```
convert2pdf -i c:\input\1.xlsx -o d:\output\1.pdf -scale 2
convert2pdf -i c:\input -o d:\output -scale 2
convert2pdf -i test -o output -scale 2
convert2pdf -i "c:\input\*.xls" -o d:\output -scale 2
```

4) Fit sheet on one page (-scale 3)

```
convert2pdf -i c:\input\1.xlsx -o d:\output\1.pdf -scale 3
convert2pdf -i c:\input -o d:\output -scale 3
convert2pdf -i test -o output -scale 3
convert2pdf -i "c:\input\*.xls" -o d:\output -scale 3
```

b) Open password (-op)

- The optional argument (**-op**) indicates the open password for a password-protected input file. It is not required if the input file is not password protected. This argument is valid only for Office conversion.

  Note *The output PDF file will not retain the open password from the input file.*

Usage Example

1) Specify the open password for a password-protected input Office file (-op 123)

```
convert2pdf -i c:\input\1.docx -o d:\output\1.pdf -op 123
convert2pdf -i test\2.pptx -o output\2.pdf -op 123
```

2) Specify the open password for all input Office files that have been protected with the same password (-op 123)

```
convert2pdf -i c:\input -o d:\output -op 123
convert2pdf -i test -o output -op 123
```

Note *It only supports typing one value for the argument (**-op**). Only files with the same open password can be processed together and files with different open password need to be processed separately.*

3.3.7 Recursion Depth of Sub-folders

- The optional argument (**-r**) indicates the recursion depth of sub-folders, which is valid only when the input is a folder or a path that includes wildcard character like "c:\input\*.doc". By default, the recursion depth is 1, so the sub-folders will not be processed. For more details about this argument, please refer to section 3.2 "Command Line Summary".

Usage Examples

1) Search the full folders (-r or -r 0)

```
convert2pdf -i test -o output -r
convert2pdf -i c:\input -o d:\output -r
convert2pdf -i c:\input\*.jpg -o d:\output -r
convert2pdf -i c:\input\*.txt -o d:\output -r
convert2pdf -i c:\input\*.html -o d:\output -r
convert2pdf -i c:\input\*.docx -o d:\output -r
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -r

convert2pdf -i test -o output -r 0
convert2pdf -i c:\input -o d:\output -r 0
convert2pdf -i c:\input\*.jpg -o d:\output -r 0
convert2pdf -i c:\input\*.txt -o d:\output -r 0
convert2pdf -i c:\input\*.html -o d:\output -r 0
convert2pdf -i c:\input\*.docx -o d:\output -r 0
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -r 0
```

2) Search only the current folder (-r 1)

```
convert2pdf -i test -o output -r 1
convert2pdf -i c:\input -o d:\output -r 1
convert2pdf -i c:\input\*.jpg -o d:\output -r 1
convert2pdf -i c:\input\*.txt -o d:\output -r 1
convert2pdf -i c:\input\*.html -o d:\output -r 1
convert2pdf -i c:\input\*.docx -o d:\output -r 1
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -r 1
```

Note *If you don't use this argument, the current folder will be searched by default. For example:*

```
convert2pdf -i test -o output
convert2pdf -i c:\input -o d:\output
convert2pdf -i c:\input\*.jpg -o d:\output
convert2pdf -i c:\input\*.txt -o d:\output
convert2pdf -i c:\input\*.html -o d:\output
convert2pdf -i c:\input\*.docx -o d:\output
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output
```

3) Search the current folder and its sub-folders (-r 2)

```
convert2pdf -i test -o output -r 2
convert2pdf -i c:\input -o d:\output -r 2
convert2pdf -i c:\input\*.jpg -o d:\output -r 2
convert2pdf -i c:\input\*.txt -o d:\output -r 2
convert2pdf -i c:\input\*.html -o d:\output -r 2
convert2pdf -i c:\input\*.docx -o d:\output -r 2
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -r 2
```

3.3.8 Multi-thread Support

- The optional argument (**-t**) indicates the number of threads that are used to speed up batch programming by making full use of the CPU. By default, the number of threads is 1.

  Note *It is recommended that you set the value of the number according to your computer's CPU configuration.*

Usage Example

1) Set the number of threads to 3 (-t 3)

```
convert2pdf -i test -o output -t 3
convert2pdf -i c:\input -o d:\output -t 3
convert2pdf -i c:\input\*.jpg -o d:\output -t 3
convert2pdf -i c:\input\*.txt -o d:\output -t 3
convert2pdf -i c:\input\*.html -o d:\output -t 3
convert2pdf -i c:\input\*.docx -o d:\output -t 3
convert2pdf -i "c:\input\*.jpg,*.txt,*.html" -o d:\output -t 3
```

3.3.9 Other Optional Arguments

a) Log file (-log<logfile> -l<log level>)

- The optional argument (**-log**) indicates the path of logfile, where the log message is placed. The argument (**-l**) indicates the log level. It is valid only when (**-log**) is used. By default, the log level is 4. For more details about this argument, please refer to section 3.2 "Command Line Summary".

Usage Example

1) Save the log file to "d:\output\conversion.log" and set the log level to 3 (-log d:\output\conversion.log -l 3)

```
convert2pdf -i c:\input -o d:\output -log d:\output\conversion.log -l 3
```

b) Register information (-register <code> <licensee>)

- The optional argument (**-register**) is used to register the command line tool. The **<code>** is the activation code from Foxit and the **<licensee>** is the licensee name designated by the users.

Usage Example

1) Register the convert2pdf tool with the code "77505-010G0-O1000-3Z4D8-D4VEO-5RL1F" and the licensee "Foxit" (-register 77505-010G0-O1000-3Z4D8-D4VEO-5RL1F Foxit)

```
convert2pdf -register 77505-010G0-O1000-3Z4D8-D4VEO-5RL1F Foxit
```

c) License agreement (-license)

- The optional argument (**-license**) is used to print the license agreement.

Usage Example

1) Print the license agreement (-license)

```
convert2pdf -license
```

d) Version information (-version/-v)

- The optional argument (-version/-v) is used to print the version information.

Usage Example

1) Print the version information (-version/-v)

```
convert2pdf -version
convert2pdf -v
```

e) Help information (-help/-h)

- The optional argument (**-help/-h**) is used to print the usage information.

Usage Example

1) Print the usage information (-help/-h)

```
convert2pdf -help
convert2pdf -h
```

f) Copyright information (-copyright)

- The optional argument (**-copyright**) is used to print the copyright information.

Usage Example

1) Print the copyright information (-copyright)

```
convert2pdf -copyright
```

4 Working with API

Foxit Convert2PDF provides another way for users who want to perform PDF manipulation through API.

Note *To integrate the Foxit Convert2PDF into your own applications with API, please contact Foxit sales team to purchase Enterprise License.*

Foxit Convert2PDF tool offers a simple-to-use API. Four functions are required for developers who want to integrate the Foxit Convert2PDF into their own applications.

First, initialize Foxit Convert2PDF library and check the license.

```
int FXT_InitLibrary(const wchar_t* key, int screenFlag);
```

The parameter "**key**" is the path of the license file ("**ftlkey.txt**", generated in the installed path after registering the tool using the activation code purchased from Foxit.).The parameter "**screenFlag**" controls whether to print the output information to screen. If the value is 1, print the output information to screen, not vice versa.

Second, call the function of Convert2PDF tool.

```
int FXT_Convert2PDFRun(const wchar_t* commandline, FXT_CallbackFun callback, void* userData = 0);
```

The parameter "**commandline**" is a command string which is exactly the same as the general syntax used for the command line application (e.g. "-i c:\input -o d:\output"). The parameter "**callback**" is the callback function provided for users to do some special processing. The parameter "**userData**" is a pointer used to transfer user data.

Third, declare the callback function.

```
typedef wchar_t*(*FXT_CallbackFun)(void* userData, int mode, wchar_t* msg, bool* isStop)
```

The parameter "**useData**" is a pointer of user data. The parameter "**mode**" specifies the output mode of information including CALLBACK_PDFTOOL_RUN_ERROR, CALLBACK_PDFTOOL_PARAM_ERROR, CALLBACK_PDFTOOL_MSG and CALLBACK_PDFTOOL_PASSWORD. The parameter "**msg**" is the output message when calling the callback function. The parameter "**isStop**" controls whether to stop the current application. If the value is true, stop the current application, otherwise, continue running the application.

Last, release and destroy Foxit Conver2PDF library.

```
void FXT_DestroyLibrary();
```

For more details about the API, please refer to the header file "fxpdftools_convert.h" in the "include" folder in the installation path.

The following are some examples on how to work with Convert2PDF API.

4.1 Working with Convert2PDF API

The following is the simplest application that can be built using Convert2PDF API:

```
// Using C/C++
void main(int argc, char* argv[])
{
	int ret = FXT_InitLibrary(L"license_key", 0);
	FXT_Convert2PDFRun(L"-i d:\\input -o d:\\output", myCallBack, NULL);

	FXT_DestroyLibrary();
}
```

This application converts all the files (image, text, HTML, and Office files) in the "d:\input" folder into PDF files under the "d:\output" folder.

The command string (`"-i d:\\input -o d:\\output"`) of the above application is set by users directly in advance. Users also can get the command string through command line. Building a command line application using Convert2PDF API is as simple as the following:

```
// Using C/C++
void main(int argc, char* argv[])
{
	int ret = FXT_InitLibrary(L"license_key", 0);

	std::wstring strCommandline = GetCommandLineW();
	std::wstring::size_type pos = strCommandline.find(L".exe");
	strCommandline = strCommandline.substr (pos+6);

	FXT_Convert2PDFRun(strCommandline.c_str(), myCallBack, NULL);

	FXT_DestroyLibrary();
}
```

It is also possible to build the command string dynamically (e.g. based on the user or a dynamic input) as illustrated in the following code snippet:

```
void main(int argc, char* argv[])
{
	int ret = FXT_InitLibrary(L"license_key", 0);

	std::wstring input_folder = L"d:\\samples\\input";//or an/a image/text/HTML/Office file
	std::wstring output_folder = L"d:\\samples\\output";//or a PDF file: output.pdf...
	std::wstring open_password = L"secret"; //open password for Office files
```

```
    std::wstring mode_type = L"2,3,4"; //Convert text, HTML and Office files into PDF files
    std::wstring log_file = L"d:\\samples\\output\\convert2pdf.log";


    // page size of the output PDF file.
    int width = 500, height = 500;

    // page margin of the output PDF file.
    int margin_top = 20, margin_bottom = 20, margin_left = 20, margin_right = 20;

    // font style for text conversion.
    std::wstring font = L"Calibri";

    // font size for text conversion.
    int fs = 11;

    // font color for text conversion.
    int fontcolor = 255 255 0;

    // support pagination wherever there is a page break for text conversion.
    bool breakpage = true;

    // cache address of HTML page resources.
    std::wstring cache = L"d:\resources";

    // timeout to load a webpage.
    int timeout = 10;

    // set all the page contents to one single PDF page.
    bool singlepage = true;

    // disable retaining hyperlinks in PDF files from HTML files or URL.
    bool nolink = true;

    // page rotation for the output PDF file.
    int rotate = 90;

    // improves the conversion quality if the webpages include lazy loading elements or if
    // the network/hardware performance is not good enough.
    bool checklazyload = true;

    // specifies the cookies file for a URL.
    std::wstring cookies = L"d:\cookies.txt";

    // specifies the JavaScript file for HTML files or URL.
    std::wstring js = L"d:\login.js";

    // specify the conversion mode for Microsoft Excel files.
    int scale = 3;

    // recursion depth of search sub-folders. 0: search all of the full folders.
    int depth = 0;

    // log level.
    int log_level = 4;

    // --------------------------------------------------
    // Given the above settings build a command string.
    std::wstring strCommandline = L"";
```

```
        if(!input_folder.empty())
                strCommandline.append(L"-i ").append(input_folder).append(L" ");

        if(!output_folder.empty())
                strCommandline.append(L"-o " ).append(output_folder).append(L" ");

        if(!mode_type.empty())
                strCommandline.append(L"-modetype ").append(mode_type).append(L" ");

        if(!font.empty())
                strCommandline.append(L"-font " ).append(font).append(L" ");

        if(!cache.empty())
                strCommandline.append(L"-cache ").append(cache).append(L" ");

        if(!cookies.empty())
                strCommandline.append(L"-cookies ").append(cookies).append(L" ");

        if(!js.empty())
                strCommandline.append(L"-js ").append(js).append(L" ");

        if(!open_password.empty())
                strCommandline.append(L"-op ").append(open_password).append(L" ");

        if(!log_file.empty())
                strCommandline.append(L"-log ").append(log_file).append(L" ");

        const int MAX_LEGHT = 128;
        const int DATA_RADIX = 10;
        wchar_t temp[MAX_LEGHT] = {0};

        if (log_level > 0)
        {
                _itow(log_level,temp,DATA_RADIX);
                strCommandline.append(L"-l ").append(temp).append(L" ");
        }

        if (depth >= 0)
        {
                wmemset(temp, 0 , MAX_LEGHT);
                _itow(depth, temp ,DATA_RADIX);
                strCommandline.append(L"-depth ").append(temp).append(L" ");
        }

        if (width > 0 && height > 0)
        {
                wmemset(temp, 0 , MAX_LEGHT);
                _itow(width, temp ,DATA_RADIX);
                strCommandline.append(L"-width ").append(temp).append(L" ");

                wmemset(temp, 0 , MAX_LEGHT);
                _itow(height, temp ,DATA_RADIX);
                strCommandline.append(L"-height ").append(temp).append(L" ");
        }

        if (margin_top >= 0 || margin_right >= 0 || margin_bottom >= 0 || margin_left >= 0)
        {
                bool flag = false;
                if (margin_left >= 0)
                {
                        wmemset(temp, 0 , MAX_LEGHT);
                        _itow(margin_left, temp ,DATA_RADIX);
                        strCommandline.append(L"-margin ").append(temp).append(L" ");
                        if(margin_top >= 0)
                        {
                                wmemset(temp, 0 , MAX_LEGHT);
                                _itow(margin_top, temp ,DATA_RADIX);
                                strCommandline.append(temp).append(L" ");
                                if(margin_right >= 0)
                                {
                                        wmemset(temp, 0 , MAX_LEGHT);
                                        _itow(margin_right, temp ,DATA_RADIX);
                                        strCommandline.append(temp).append(L" ");
                                        if(margin_bottom >= 0)
                                        {
                                                wmemset(temp, 0 , MAX_LEGHT);
                                                _itow(margin_bottom, temp ,DATA_RADIX);
                                                strCommandline.append(temp).append(L" ");
                                        }
                                }
                        }
                }
        }

        if (fs>= 0)
        {
                wmemset(temp, 0, MAX_LEGHT);
                _itow(fs, temp ,DATA_RADIX);
                strCommandline.append(L"-fs ").append(temp).append(L" ");
        }

        if (R >= 0 && G >= 0 && B >= 0)
        {
                wmemset(temp, 0, MAX_LEGHT);
                _itow(fontcolor, temp ,DATA_RADIX);
                strCommandline.append(L"-fontcolor ").append(temp).append(L" ");
        }

        if(breakpage) strCommandline.append(L"-breakpage").append(L" ");

        if (timeout>= 0)
        {
                wmemset(temp, 0, MAX_LEGHT);
                _itow(timeout, temp ,DATA_RADIX);
                strCommandline.append(L"-timeout ").append(temp).append(L" ");
        }

        if(singlepage)  strCommandline.append(L"-singlepage").append(L" ");

        if(nolink)      strCommandline.append(L"-nolink").append(L" ");

        if (rotate>= 0)
        {
                wmemset(temp, 0, MAX_LEGHT);
                _itow(rotate, temp, DATA_RADIX);
                strCommandline.append(L"-rotate ").append(temp).append(L" ");
        }

        if(checklazyload) strCommandline.append(L"-checklazyload").append(L" ");
        if(scale >= 0)
        {
                wmemset(temp, 0, MAX_LEGHT);
                _itow(scale, temp, DATA_RADIX);
                strCommandline.append(L"-scale ").append(temp).append(L" ");

        }

        FXT_Convert2PDFRun(strCommandline.c_str(), myCallBack, NULL);

        FXT_DestroyLibrary();
}
```

4.2 Reporting Progress Messages and Errors

To find out if Convert2PDF processing was successful, the application can query the status code returned by FXT_Convert2PDFRun().

For example,

```
int ret = FXT_Convert2PDFRun(...);
if (ret == FXT_ERROR_SUCCESS) {
        // No errors...
}
else if (ret == FXT_ERROR_LICENSE) {
        // Invalid license...
}
else if (ret == FXT_ERROR_PARAM) {
        // Invalid param
}
else if (ret == FXT_ERROR_LIBRARY) {
        // Failed to initialize Foxit PDF SDK Library
}
else if (ret == FXT_ERROR_ERROR) {
        // Failed to convert the input file to PDF file
}
else {
        // Other error
}
```

A non-zero value indicates that an error was encountered. You can find the list for all error code in the "include/fxpdftools_convert.h" header.

For more detailed error and message reporting, you can pass a pointer to the custom callback function in the second parameter of FXT_Convert2PDFRun(). The last parameter in FXT_Convert2PDFRun() is a pointer to custom data that you may want to pass to the callback function.

The following is a sample callback function:

```
// Using C/C++
Wchar_t* MyCallback(void* userData, int mode, wchar_t* msg, bool* isStop) {
	if (mode == CALLBACK_PDFTOOL_RUN_ERROR) {
		wcout << L"Error: " << msg << endl;
	}
	else if (mode == CALLBACK_PDFTOOL_PARAM_ERROR) {
		wcout << msg;
	}
	else if (mode == CALLBACK_PDFTOOL_MSG) {
		wcout << msg;
	}
	else if (mode == CALLBACK_PDFTOOL_PASSWORD) {
		static wstring password;
		wcin >> password;
		return (wchar_t*)password.c_str();
	}
	return 0;
}
```

5 Support

5.1 Reporting Problem

Should you encounter any technical questions or bug issues when using Foxit Convert2PDF command line tool, please submit the problem report to Foxit support team at http://www.foxitsoftware.com/support/. In order to better help you solve the problem, please provide the following information:

- Contact details
- Product name and its version
- Your Operating System
- Detailed description of the problem
- Any other related information, such as error screenshot

Note

- *In the unfortunate event that Foxit Convert2PDF tool should crash, Foxit Convert2PDF will generate two files named "CRASHLOG.TXT" and "CRASH.DMP" under the current execution directory. The "CRASHLOG.TXT" file will record the detailed information of the conversion module and the system that are used at the time the crash occurred. To help our engineering team track down the problem and provide a solution, please submit the two files to Foxit support team. Thank you for your cooperation.*
- *For HTML conversion, if you have installed security software locally and access the webpages embedded with advertising JavaScript, the main program "conver2pdf.exe" (or "conver2pdf64.exe") and the background program"fxhtml2pdf.exe" may be deleted by the security software. In order to use HTML conversion properly, please add the following lists to the security software's white list, where "[installPath]" is the installation path of Foxit Convert2PDF.*

  [installPath]\convert2pdf.exe
  [installPath]\convert2pdf64.exe
  [installPath]\lib\html2pdf\x64\fxhtml2pdf.exe
  [installPath]\lib\html2pdf\x86\fxhtml2pdf.exe

5.2 Contact Information

You can contact Foxit directly, please use the contact information as follows:

Foxit Support:

- http://www.foxit.com/kb.html

Sales Contact:

- Phone: 1-866-680-3668

Support & General Contact:

- Phone: 1-866-MYFOXIT or 1-866-693-6948