callas software gmbh

pdfaPilot CLI

Manual

pdfaPilot CLI – Manual – Last modified: 15 September 2015

Content

Getting started

pdfaPilot CLI may be used to analyse files or to analyse and convert to PDF/A (if possible).

System requirements

The command line version of pdfaPilot is available for the following operating systems:

- Windows 2000/2003 Server/XP/Vista/7
- Mac OS X 10.6 or newer, Intel
- Linux Debian 5.0 (Lenny)
- Sun Solaris SPARC version 8 or newer
- Sun Solaris Intel version 10 or newer
- AIX 5.3 (oslevel 5300-07) or newer (call `oslevel -q` to check)

Spoken generally, pdfaPilot CLI should work with other Linux distributions as well, as long as there are system libraries installed that are compatibel to gcc-v3.4 or newer. The dependent libstdc++ is delivered with the pdfaPilot CLI.

You can easily test if pdfaPilot CLI is working on your system: Just type `pdfaPilot --help` in the terminal.

pdfaPilot CLI does also run on 64 bit systems if the required 32 bit compatibility packages are available.

Installing the software

Macintosh/Windows

To install the software start the pdfaPilot Server installer. The installation program will then take you through the necessary steps.

Linux/Solaris/AIX

Extract all files from the archive to a destination folder of your choice.

For automation purposes you should set the PATH variable to the path of the pdfaPilot CLI executable.

Additional information is provided in *<pdfaPilot CLI directory>/ReadMe.txt*

Activation

Before callas pdfaPilot CLI can be used, the software has to be activated.

Request an activation code

Open a terminal window and change to your pdfaPilot CLI installation directory. Type:

```
pdfaPilot --keycode <name> <company> <licenceCode>
```

Parameters

name	Name of licensee (e.g. ”Registered User”)
company	Name of company (e.g. ”User’s company”)

licencecode	Licence key obtained from the registration card To make a request for a trial version, please use the keword "trial" (for a pdfaPilot trial version) for this parameter The textual output of --keycode has to be send via e-mail to the e-mail address named in the text in order to receive an activation code from the registration server.

Activating pdfaPilot CLI

After having received the automatical reply e-mail to the activation request, save the attached licence file to the file system. Then use the following command:

```
pdfaPilot --activate <licence file>
```

Parameters

licence file	Full path to licence file

It is necessary to activate the received license file to get a permantent valid license file.

The license file received from the activation server must be activated within the timeframe listed in the license file.

In order to activate pdfaPilot CLI for all user accounts of one machine, save the activated license file next to the pdfToolbox binary. This file must be named "License.txt".
The activated license file will be stored in the user-preferences when the normal activation (command above) is used.
To create an activated license file at a custom location, just use the following command:

```
pdfaPilot --activate <licence file> -o=<path to result folder/
License.txt>
```

pdfaPilot CLI is searching for the license file at various folders:

- user-preferences-folder of actual user
- next to the pdfaPilot CLI binary
- cachefolder (if set)
- user-preferences-folder for all users (shared)

When using UNIX-based-systems the environment variable `CALLAS_SYSTEM_PREFERENCES` the path of the standard */usr/share/callas software/callas pdfaPilot CLI* can be changed:
`CALLAS_SYSTEM_PREFERENCES=tmp`
would result in the searchpath: */tmp/callas software/callas pdfaPilot CLI*
It is highly recommended to use the option --cachefolder instead.

Time-limited trial version

After requesting and entering a trial activation code, pdfaPilot CLI can be tested without any restrictions. When the evaluation period has expired, processing PDF files will no longer be possible until you request and enter a new activation code.

Activation using the Standalone application

Using Windows and MacOS, also the activation dialog of the Standalone Application can be used for requesting a activation as well as using a Key-code or just for a trial version. Also the activation itself can be done using that Interface.
All activations (for Desktop, Server as well as for the DeviceLink Addons) can be done using this dialog.

Deactivate pdfaPilot using the CLI

As the activation (and the resulting license file) is bound to the hardware. It is necessary to deactivate a license on one machine before a activation takes place on the new machine.

```
pdfaPilot --deactivate <activation code>
```

activation code	Unique identifier for each license

The respective license will be removed from the system

To complete the deactivation, the output of the command has to be sent manually to the activation server by e-mail.

The activation code for all license are listed using the status command:

```
pdfaPilot --status
```

Deactivation using the Standalone application

Similar to the deactivation using the CLI, the Desktop on Windows and MacOS can be used for deactivation.
The selected license will be removed from the system as well and the necessary e-mail to the activation server will be sent automatically.

Displaying program information

Display program version

```
pdfaPilot --version
```

will display the currently used version of pdfaPilot CLI.

Display usage information

```
pdfaPilot --help
```

will give you a complete overview about all available commands for processing.

```
pdfaPilot --help <command>
```

will give you an overview about all available options for the command.

Display status

```
pdfaPilot --status
```

will inform you about the current license state as well as the possible return and reason codes (see *"Results"*).

Hints and troubleshooting

Ensure sufficient free disk space

To ensure stable processing, it is recommended to have at least 4 times of the input file size of processed files available for intermediate file system storage (e.g. /tmp on Unix and similar on other systems).

Avoid stopping workflows on Windows

On Windows, you can prevent your workflow from stopping in case of a pdfToolbox CLI crash by setting the following registry entry:

```
HKEY_LOCAL_MACHINE\
SYSTEM\
CurrentControlSet\
Control\
Windows\
ErrorMode
```

If ErrorMode is set to "2", crash dialogs will be suppressed. For further details, see: http://support.microsoft.com/kb/128642/en-us?fr=1

Limiting the maximum memory used by pdfToolbox

Using Linux, you can limit the amount of memory used by a single process by an additional parameter:

```
--maxmemory=<max. memory in MB>
```

Processing will stop and result in an error if memory is exceeded.

Performance enhancement

If you want to enhance the performance of your pdfaPilot CLI processes, please keep in mind the following rules:

- For embedding missing fonts your system font folder will be scanned unless defined otherwise (see *"Font Embedding"*). pdfaPilot will create a font cache to improve the performance time, but still it might be useful to remove fonts that are not needed from this directory.
- If you are using any font embedding fixups, your system font folder will be scanned unless defined otherwise in the fixup configuration. A font cache will be created to improve the performance time, but still it might be useful to remove fonts that are not needed from this directory.
- Creation of XML or PDF reports takes less time than the XSLT option (see *"Report types"*).
- Creation of reports takes additional time – even if a profile contains only fixups, an analysis will be executed for gathering report information.

Optimization of needed installation space

To reduce the pace needed by the installation of pdfsPilot, it is possible to delete some subfolders of the CLI component (in subfolder /cli) if their respective functions are not needed in the individual use case.

To avoid processing errors or unexpected behaviour of pdfaPilot any modification should be done well-considered.

etc/Actions	If no Arrange action is used

etc/APDFL	If no font embedding or PDF/A conversion is used (or if font situation is clear)
etc/Backgrounds	If no layer/image mask report is used
etc/Certify	If no preflight certification is used
etc/ColorConversion	If no color conversion is used
etc/HtmlConverter	If no PDF report based on HTML template is used
etc/Inventory	If no inventory report is used
etc/MailConverter	If no e-mails are processed
etc/PDFOfficeTool	If no Office-files are processed
etc/PDFPSTool	If no PostScript-files are processed
etc/Reports	If no PDF/A-HTML Report or ZUGFeRD is used
etc/TPex	If no tagged PDF to HTML/EPUB export is used
etc/UnpackTool	If no archive files are processed
etc/Visualizer	If no Comparison is used

Get in touch

If some necessary information is not provided by this manual or if there are any questions or feedback please contact the product management by using the "Contact Support" form on *www.callassoftware.com*.
You can also send an e-mail to *support@callassoftware.com*.

If you file a bug report please make sure your inquiry contains the following information:

- operating system
- pdfaPilot version (call `pdfaPilot --version`)
- command line call
- original PDF (please delete unnecessary pages to avoid long file transfers), used profiles or configuration files
- converted PDF (if available)

You can also visit the support section on *www.callassoftware.com* to get answers to common questions or find a reseller near you. The latter might be useful if you want to send a support request that is neither in English nor German.

Processing

Input files from Office applications

pdfaPilot CLI is able to convert common file formats from Office applications directly to PDF/A. For more information and a list of supported applications and files have a look at:
http://www.callassoftware.com/goto/apl_ENU_topdf

Office file conversion is currently not supported on Solaris and AIX systems.

OpenOffice

```
--topdf_forceopenOffice
```

When defined, Microsoft Office files are processed with OpenOffice.

Create PDF for print

```
--topdf_print
```

The PDF will be created with image resolution sufficient for printing, thus leading to larger files.

Excel-Sheets without removing white space

```
--topdf_useexcelpagelayout
```

Use Excel page layout, white space will not be removed.

Special handling for Excel-Sheets

```
--topdf_parameter=[ShowHiddenColumns|ShrinkToFit]
```

Special parameters to achieve some special layouts for Excel files.

Parameters

ShowHiddenColumns	Show columns which are not visible due to small width or other settings.
ShrinkToFit	Shrinks the content of a cell so that the content fits inside.

When `--topdf_useexcelpagelayout` is used, this parameter will not be respected.

Logging of dialogs in defined log file

```
--topdf_guiactionslog=<path>
```

Parameters

path	Path to folder or logfile.

All dialogs occuring during processing the office file will be logged within this file.

See internet page (listed above) for further information about handling of dialogs from office applications.

General options

Usually pdfaPilot CLI is started with:

```
pdfaPilot <PDF file>
```

Input file

May be one or a number of input files (PDF or Office files) to be analysed and converted.

If an input file spec is pointing to an existing folder, all files inside this folder are processed:

Process folders recursively

```
--recursive
```

If the file spec for the input file is pointing to an existing folder all PDF files inside the folder on all levels are processed

Empty the font cache

```
--emptyfontcache
```

Removes all font files from the font cache folder of pdfaPilot CLI.

Incremental saving

```
--incremental
```

Allows to modify the input file, only writing the changes to the original PDF. This can increase the speed significantly since pdfaPilot CLI does not need to create a new copy of the file.

PDF structure and font optimization

```
--nooptimization
```

The internal PDF structure and fonts are not optimized when saving the PDF file.PDF structure and font optimization

Analyze only

```
-a --analyze
```

The input file is not being converted but is analyzed whether it is PDF/A compliant.

Analyze only certain pagerange

```
-p --pagerange=<firstpage>[-<lastpage>]
```

Only applied when analyzing not when converting PDF files.

When converting non-PDF documents the page range of the original document can be specified.

Parameters

firstpage	Page where analysis should start

lastpage	optional, page where analysis should end

Example:

```
--pagerange=5-33
```

Setting the cache folder

```
--cachefolder=<path>
```

Sets the cache folder path. This is set by default to:

Windows:	*C:\Documents and Settings\<user>\Application data\callas software\callas pdfaPilot CLI*
Macintosh:	*/Users/<user>/Library/Preferences/callas software/callas pdfaPilot CLI*
Unix:	*<home directory as defined in /etc/passwd>/.callas software/callas pdfaPilot CLI*

This option is mandatory when running the CLI as a user without a home directory.

Parameters

path	absolute path to custom cache folder

PDF/A specific options

PDF/A Compliancy level

```
--level=<level>
```

You can define which PDF/A level you need (default is 1b).

Parameters

level	3b, 3u, 3a, 2b, 2u, 2a, 1b or 1a

Deactivate removal of non-compliant metadata

```
--noxmpremoval
```

Normally, XMP Metadata which is not compliant with PDF/A is removed during conversion. This switch prevents the removal.

Force Conversion to PDF/A

Due to several reasons a regular conversion may not result in a valid PDF/A document. To ensure conversion 3 additional steps can be performed after a normal conversion if a file can not be converted to PDF/A within this first step. The order they will be performed is as listed below. After each step the resulting file is checked for compatibility with the choosen standard. Each step is optional.

```
--forceconversion_reconvert
```

Performs a re-conversion of the PDF via PostScript.

```
--forceconversion_pagestoimages
```

Convert pages with problems into images, while converting the text is transmuted into invisible text, which is correctly positioned to keep the text available for marking and copying.

```
--forceconversion_doctoimages
```

Convert all pages into images, while converting the text is transmuted into invisible text, which is correctly positioned to keep the text available for marking and copying.

```
--forceconversion_resolution=<resolution in ppi>
```

Image resolution in ppi used for the rendered content. (Default = 100 ppi).

Font folders

If a font is not embedded and an embedding is required by a PDF/A-conversion or a profile, pdfaPilot CLI will search the system font directories in order to find the needed font file, which are:

Windows	• C:\Windows\Fonts
Macintosh	• /Users/<user>/Library/Fonts
	• /Library/Fonts
	• /System/Library/Fonts
Linux, Solaris Sparc, Solaris x86, AIX	• /usr/lib/X11/fonts
	• /usr/local/X11R6/lib/X11/fonts
	• /usr/share/fonts
	• /<user home>/.fonts

Additionally the font folder installed together with pdfaPilot CLI will be searched. This folder lies next to the executable in *"<callas pdfaPilot CLI directory>|etc|APDFL|Resource|Font"*.

ICC-profiles folders

The following folders are searched for required ICC-profiles, unless they are already contained in the .kfpx-profile already.
These folders lies next to the executable in:

- *"<callas pdfaPilot CLI directory>|etc|ICC profiles"*
- *"<callas pdfaPilot CLI directory>|etc|APDFL|Resource|Color|Profiles"*

Some system folders for colors are searched addtionally:

MacOS:	\Library\Application Support\Adobe\Color
Windows:	\Windows\system32\spool\drivers\color

Set a processing timeout

```
--timeout=<seconds>
```

Sets the maximum processing time in seconds. If the process exceeds this duration, the execution process will be killed and the processing will result in an error.

File content options

Deactivate transparency flattening

```
--notransparencyflattening
```

This switches off both flattening of any contained transparency and setting the blend color space to sRGB.

Add XMP metadata

```
--addxmp=<path>
```

The XMP metadata is merged into any existing XMP metadata.

Parameters

path	Path to an XMP file

Add bookmarks

```
--addbookmarks=<path>
```

The bookmarks are embedded into the PDF file.

Parameters

path	Path to an XML file containing bookmarks

Set the OutputIntent

```
-i --OutputIntent=<path>
```

Path to a PDF file with an OutputIntent – forces use of this OutputIntent.

Parameters

path	Path to an OutputIntent

Define ICC profiles

CMYK

```
--defaultprofile_cmyk=<path>
```

The given profile is embedded as the default profile making device dependent CMYK page objects device independent.

Parameters

path	Path to a CMYK ICC profile

RGB

```
--defaultprofile_rgb=<path>
```

The given profile is embedded as the default profile making device dependent RGB page objects device independent.

Parameters

path	Path to an RGB ICC profile

Gray

```
--defaultprofile_gray=<path>
```

The given profile is embedded as the default profile making device dependent Gray page objects device independent.

Parameters

path	Path to a Gray ICC profile

Font Embedding

If a font is not embedded pdfaPilot CLI will search the system's font directories in order to find the needed font file, which are:

Windows	• C:\Windows\Fonts
Macintosh	• /Users/<user>/Library/Fonts
	• /Library/Fonts
	• /System/Library/Fonts
Linux, Solaris Sparc, Solaris x86, AIX	• /usr/lib/X11/fonts
	• /usr/local/X11R6/lib/X11/fonts
	• /usr/share/fonts
	• /<user home>/.fonts

Additionally the font folder installed together with pdfaPilot CLI will be searched. This folder lies next to the executable in *"<callas pdfaPilot CLI directory>|etc|APDFL|Resource|Font"*.

Define an additional font folder

```
--fontfolder=<path>
```

Additional folder to look up fonts for embedding.

Parameters

path	Path to config file

Example:

```
--fontfolder="C:\AdditionalFonts"
```

You can force pdfaPilot CLI to only scan the folder defined by `--fontfolder` (and not search the system's font folders) by using the option `--fontonly`.

Substitute fonts

```
--substitute[=<path>]
```

Font substitution can be used when the original fonts used in a PDF file are not available. By default, the font substitution file pdfa.cfg stored in *"etc/FontSubstitution/"* provides the basis for substitution.

Alternatively, you can also enter a custom path to a fontsubstitution file, e.g.:

```
--substitute=C:\fontsubstitution.cfg
```

If you want to switch off font substitution completely, just hand over an invalid value, e.g.:

```
--substitute=no
```

Parameters

path	Optional, path to config file

The following notations are allowed in the *fontsubstitution.cfg*:

SubstituteAll

Each font can be substituted by every other font of the entry.

```
SubstituteAll<tab>fontname<tab>fontname<tab>fontname...
```

SubstituteFirst

Only the first font of the entry can be substituted by the following fonts.

```
SubstituteFirst<tab>font to be substituted<tag>fontname
<tab>...
```

Creating file packages

Some PDF standards allows the embedding of PDF- and also non-PDF-files into another PDF file. Sometime these filepackages are also called collections. Using pdfaPilot CLI it is possible to create such file packages from a complete folder or to define different ways how a file which shall be embedded is handled.
In general a file package is created with `--collection` This will create an index document, which lists all embedded files from the given folder. Also an existing folder structure will be respected

```
--collection <folder>
```

In general a file package is created with `--collection` This will create an index document, which lists all embedded files.

```
--collection <file> [<file>]
```

Settings for file embedding

```
--collection [--embedinto=[target],<file>] [--embedfile=
[target,[relationship],<file>] [--embedwithlink=
[area,<file>]
```

--embedinto

It is possible to use own templates or normal PDF for embedding files. The standard for the file where other files will be embedded can be defined using the conversion target (see below). If no file is defined, an index file is created.
Instead of using a destination file, also a folder containing a HTML-based Template with an index.html can be used as destination. The overview page of all embedded files will be created with the layout defined within this template then.

Parameters

target	A3b, A3u, A3a, A2b, A2u, A2a, A1b, A1a or No (Default)

--embedfile

Also for files to embed a conversion target can be defined using the conversion target. For PDF/A-3 standards also a relationship entry for each embedded file can be set.

Parameters

target	A3b, A3u, A3a, A2b, A2u, A2a, A1b, A1a, PDF or No (Default)

Using the target "No", no conversion to PDF is done. (Only available for embedded files.)

relationship	Source, Data, Alternative, Supplement, Unspecified (Default)

--embedwithlink

Alternatively, files can be embedded with defining an area in the containing document, where a link to the contained file is created. No conversion will take place with the file to embed.

Parameters

area	X1,X2,Y1,Y2[pt, in, cm, mm]

Defines a rectangular area, based on the lower left corner of the page, where a link to the embedded file is inserted. Default unit is pt.

The "--embedwithlink"-function is not supported when using a HTML-based template for the creation of the overview page.

Example:

```
--collection --embedinto=A3b,<PDF file> --embedfile=A3b,
Alternative,<file> --embedfile=A2b,Source,<Office file>
--embedfile=No,Data,<file>
```

```
--collection --embedinto=A3b,<path to HTML template
folder> <folder with files to create a collection from>
-o=<path to result PDF>
```

```
--collection --embedwithlink=10,10,100,100,<file> --emb
edwithlink=10mm,100mm,100mm,200mm,<file>
```

Converting e-mail files to PDF

Some common e-mail file formats (MSG, EML, EMLX) are supported for conversion to PDF.
Some PDF standards do not allow embedding files (like PDF/A-1) but other allows the embedding of PDF- (PDF/A-2) and also non-PDF-files (PDF/A-3) into another PDF file. To match these different targets, using pdfaPilot CLI it is possible to define variety of different settings for handling attachments of e-mails.
In general a PDF is created with `--emailtopdf`
This will create a PDF containing all attachments, those who can be converted as PDF (like Office files or images), all other in ther native format.

```
--emailtopdf <e-mail file>
```

Settings for e-mail conversion

```
--emailtopdf [--level=<level>] [--attachments=<pa
rameter>,<parameter>,...] [--onerror=<parameter>]
[--embedsource] [--noembed=<file type>,<file
type>,...] [--noconvert=<file type>,<file type>,...]
--template=<path to folder> <e-mail file>
```

--level

You can define the required PDF/A level (default is no PDF/A conversion).

Parameters

level	A3b, A3u, A3a, A2b, A2u, A2a, A1b, A1a or No (default: no)

--attachments

Depending on the defined parameter, attachments will not be embedded or embedded in their original file format, converted to PDF or as pages. The parameters can be combined comma separated.

Parameters

IGNORE	Attachments will be ignored and not converted or attached
ORIGINAL	Attachments will be embedded in their original file format
PDF	Attachments will be converted to PDF (if possible) (Default)
PAGES	Attachments will be converted (if possible) and attached as additional pages to the e-mail content

--onerror

If an attachment can not be converted to PDF, this setting allows the definition of handling such e-mail files.

Parameters

ABORT	Aborts processing if an attachment can not be converted to PDF
SKIP	Attachment will be skipped and reported on the CLI.
FALLBACK	Attachment will be embedded in original file format, if it can not be converted to PDF. If a PDF/A-level has been selected, PDF/A-3 with the respective sublevel will be used. (Default)

--embedsource

The e-mail-file itself will also be embedded in the resulting PDF.

--noembed

RegEx expression for file type extensions for attachments, which will not be embedded or converted to PDF although is attached to the e-mail file. Samples for valid RegEx (RegEx is case-sensitive, "(?i)" will make it case-insensitive.

--noembed=".*.vir"	matches extension "vir" in small letters
--noembed=".*.(VIR|EXE)"	matches the listed extensions in big letters
--noembed="(?i).*.VIR"	matches all possible variations of "VIR"

--noconvert

RegEx expression for file type extensions for attachments, which will not be embedded or converted to PDF although is attached to the e-mail file. See --noembed for RegEx samples.

--template

It is possible to create own HTML-Template and Cascading Style Sheets (CSS) to adjust the visual appearance of the created PDF.

Creating HTML from tagged PDF

Converts a tagged PDF file into HTML, which can be opened in a browser to check the tagging strukture or viewing the content in various styles.

```
--createhtml <PDF file>
```

It is possible to create own Cascading Style Sheets (CSS) to adjust the visual appearance of the created EPUB.

```
--stylefolder=<path to folder>
```

By adjusting the `<tag-and-attrib-processing.cfg>` in the `/config` folder of the predefined templates, it is possible to suppress tags or attributes or their content as well as replacing tags or attributes by other tags or attributes.

Creating EPUB from tagged PDF

Converts a tagged PDF file into EPUB, which is the standard format for eBooks.

```
--createepub <PDF file>
```

Defining the EPUB level will result in the respective standard conformance.

```
--level=[2|3]
```

When one or more tags are defined, new chapters will be created at these contents.

```
--chapter=[TAG,][TAG,]...
```

It is possible to create own Cascading Style Sheets (CSS) to adjust the visual appearance of the created EPUB.

```
--stylefolder=<path to folder>
```

The visual appearance of the created EPUB files can easily adjusted by modifying a CSS-file. Predefined templates are located in the folder ../var/Actions/TaggedPDF/ExportEPUB. By changing the values in the CSS, the resulting EPUB can be adopted to individual needs.
By adjusting the `<tag-and-attrib-processing.cfg>` in the `/config` folder of the predefined templates, it is possible to suppress tags or attributes or their content as well as replacing tags or attributes by other tags or attributes.

Creating ZUGFeRD invoices

The German e-invoicing standard ZUGFeRD (Central User Guide Forum electronic Invoice Germany) defines a document, which contains an invoice in a human readable (PDF/A-3) and a machine readable format (XML). By using the PDF/A-3 standard the created file satisfies long-term archiving regulations.
With pdfaPilot it is possible to embed such an ZUGFeRD-XML file (created by an electronic billing or accounting system) into a PDF, setting the required ZUGFeRD-XMP metadata and making the PDF a valid PDF/A-3 in one run.

Create a ZUGFeRD invoice using an invoice PDF and an invoice XML:

```
--zugferd --create=<ZUGFeRD XML file> <PDF file>
```

Adding a GiroCode during creation with a ZUGFeRD invoice using an invoice PDF and an invoice XML (GiroCode data will be determined from XML file):

```
--zugferd --create=<ZUGFeRD XML file> --girocode=<lower
left x>,<lower left y>,<size>,<unit>,<pagenumber> <PDF
file>
```

Parameters for GiroCode:

lower left x	Position from the lower left on x axis
lower left y	Position from the lower left on y axis
size	Size of resulting GiroCode (always quadratic)
unit	Optional unit for coordiantes and size: mm, inch, pt (default: mm)
pagenumber	Page, where the GiroCode will be positioned

Validate a PDF and the containing XML against the ZUGFeRD specification:

```
--zugferd <PDF file>
```

Extract the embedded XML from a ZUGFeRD PDF file:

```
--zugferd --extract <PDF file>
```

Create several reports from a ZUGFeRD invoice:

```
--zugferd --export=[<option>,<option>,...]
[--resolution=<resolution in ppi>] [--imgformat=<image
format>] <PDF file>
```

Options for ZUGFeRD reports:

NOPDFIMAGE	No images from the PDF pages
NOXML	XML invoice is not extracted
NOHTML	No HTML view of the XML
NOHTMLSYNOPSIS	No synopsis of PDF images and XML

NOVALIDATIONREPORT	No ZUGFeRD validation report

Addtional parameters for the export of PDFIMAGES:

RESOLUTION IN PPI	optional, resolution in ppi or width x height in pixel, e.g. 1024x800 (default: 72)
IMGFORMAT	optional, JPEG, PNG, TIFF, PDF (default: JPEG)

Report creation

For each pdfaPilot CLI run several reports may be generated by inserting `--report` or `-r` switches into the pdfaPilot call. Following all options and reporttypes are listed. The options of `--report` are treated case insensitive and have to be separated by commas.

```
--report=[<type>,][<trigger>,][<options>,]<PATH=path>
```

Parameters

type	Optional, see "*Report types and their options*"
trigger	Optional, see "*Report triggers*"
options	Optional, see "*Report types and their options*"
path	See "*Report path*"

Report types and their options

HTML (default)

A html file is created. The format may be modified by combining this option with other options. It can be opened by any webbrowser.

Additional options for HTML reports

All following switches may be simultaneously used in order to add different types of additional content.

NOICONS	Create a report without any images
NOCORRECTIONS	Do not log corrections
NODETAILS	Suppress details for the occurences
OPENRESULT	All entries in the report are opened in Initial view (closed by default if JavaScript is enabled)

Set the path for referenced objects

```
--linkpath
```

HTML reports are generated without exporting referenced objects. This option requires a path (URL) to a folder where the referenced objects reside (Default: folder "\etc\reporttemplate" in pdfaPilot CLI directory)

Example:

```
--linkpath="file:\\\Programme\pdfaPilot\etc\
reporttemplate"
```

Customize your report

If you create a HTML report, you can completely customize it by adapting the CSS-File to your needs. You can even exchange the pictures.

You can find all material used for the HTML reports within the folder *"|etc| reporttemplate"* in your pdfaPilot CLI directory.

XML

XML file which is intended to be processed with software (parsers). The scheme of the XML report structure can be found within the xsd-file stored in *"|var|XMLV2 report schema|"*.

MHT

A mht file is created. MHT is a data format created by Microsoft which offers the possibility to include all resources (like style sheets, images and JavaScripts) to be included in one file. This file type can only be read by Internet Explorer.

TEMPLATE

```
TEMPLATE=<path>
```

path	Path to template folder (or respective index.html directly) PDF overview reports are created based on a style defined in a HTML/CSS template. A predefined template can be found in the Server/CLI path: `../var/Reports/Templates`

Report triggers

ALWAYS	Always create a report (default)
IFNOPDFA	Create a report only if the file cannot be converted to PDF/A
IFPDFA	Create a report only if the file can be converted to PDF/A

Report path

```
--report=<Report type>,<Report options>,PATH=<Path>
```

Full path of report file – the report path must not contain any commas.

`PATH=` must always be the last element.

Example:

```
--report=HTML,NOICONS,OPENRESULT,PATH=C:\Sample.html
```

Set the report language

```
-l --language=<language>
```

Defines the language in which the report is generated. Default language is English (en). The language is specified by using a two digit abbreviation. The following values can be used in the standard configuration:

Parameters

EN	English
DE	German
FR	French
IT	Italian
ES	Spanish
JA	Japanese

Command line output

Display progress

```
--noprogress
```

Switch off progress information.

Display hits

```
--nohits
```

Switch off output of hits (errors, warnings, information).

Display summary

```
--nosummary
```

Switch off summary information.

Display timestamp

```
--timestamp
```

Show a timestamp for each line of command line output.

Further options

Quick processing

```
-q --quick
```

Processing is stopped after the first detection of an error in the corrected PDF (only applied if no report is generated).

Define the overwrite mode

```
--overwrite
```

New files override existing files with the same name (applies to report files and to created PDF files).

Create output files only for successful conversion

```
--onlypdfa
```

Create an output file only if the file could be converted to PDF/A.

Set the result path

- If neither an output path nor an output folder is defined, any result will be created next to the input file (default: input file name with `suffix _PDFA` or `_NOPDFA`, will be indexed if necessary).
- The use of `--outputfile` together with `--outputfolder` is not supported within one CLI call.

Path to output file

```
-o --outputfile=<path>
```

Defines the absolute path of the destination file. The parent folder must exist.

- Consult section "*Results*" to see if a new file was created.

Parameters

path	absolute path to output file

Path to output folder

```
-f --outputfolder=<path>
```

Defines an absolute path to a folder where the files resulting of an execution are stored.

Parameters

path	absolute path to output folder Output file

Use an additional profile

```
--profile
```

Run additional checks and fixups by defining a full path to a kfp or kfpx file (exported from pdfToolbox/pdfaPilot Desktop or Preflight in Adobe Acrobat). The input file is converted if no warnings or errors occur.

Using response files

To keep the command line call structured and straightforward, pdfaPilot CLI supports the usage of response files. These offer the possibility to define each command line switch line by line and also add some comments.

Example

Response file analyze.rsp:

```
###################
# PDF/A analysis
#
--analyze
#
###################
# EOF
```

Command line call:

```
pdfaPilot @analyze.rsp <PDF file>
```

Structure of command line output

Checks

```
Hit  Type of hit  Name of check  [Name of font]  [Name of glyph]
```

1. Hit

- always visible
- keyword

2. Type of hit

- always visible
- possible values: "PDFA", "Error", "Info", "Warning" (only for additional profile)

3. Name of check

- always visible
- never empty

4. Name of font

- only visible for hits that belong to a font check
- can be empty

5. Name of glyph

- only visible for hits that belong to a font check
- can be empty

Corrections

```
Fix          Name of fixup          [Name of font]
```

1. Fix

- always visible
- keyword

2. Name of fixup

- always visible
- never empty

3. Name of font

- only visible for fixups that belong to a font correction
- can be empty

Failed corrections

```
FixFailure  Name of fixup         [Name of font]
```

1. FixFailure

- always visible
- keyword

2. Name of fixup

- always visible
- never empty

3. Name of font

- only visible for fixups that belong to a font correction
- can be empty

Progress

```
Progress       Value   %
```

Results

Return codes and their usage

All return codes below 100 indicate a successful operation.

When executing kfpx profiles

0	No hit, no fixups executed
1	At least one hit with severity 'info', no fixups executed
2	At least one hit with severity 'warning', no fixups executed
3	At least one hit with severity 'error', no fixups executed
5	No hit, fixups have been executed
6	At least one hit with severity 'info', fixups have been executed
7	At least one hit with severity 'warning', fixups have been executed
8	At least one hit with severity 'error', fixups have been executed; fixup failed

When executing in pdfa mode

0	PDF is valid PDF/A-file additional checks wihtout problems
1	PDF is valid PDF/A-file but additional checks with problems – severity info
2	PDF is valid PDF/A-file but additional checks with problems – severity warning
3	PDF is valid PDF/A-file but additional checks with problems – severity error
4	PDF is not a valid PDF/A-file

When executing an e-mail conversion

0	Successful conversion to PDF/A
2	PDF is valid PDF/A file, but some attachments of the converted e-mail are missing
4	E-mail was converted to PDF, but is not a valid PDF/A-file

When executing the compare action

0	Compared PDFs are equal
1	Compared PDFs have differences

Other commands or actions

0	Successful operation

Errors:

100	Not serialized (no valid serialization found or keycode expired)
101	Command line parameter error
102	Command line syntax error (illegal command)
103	Unknown error (internal error)
104	A file could not be opened
105	An encrypted PDF file could not be opened for writing
106	A file could not be saved

On a Windows computer the return code may be accessed by using the system variable `errorlevel`, on a Linux/Sun Solaris/MacOS computer `$?` can be used.

Errors for distributed processing

110	Action is not distributable
111	No Dispatcher was found
112	No Satellite was found or is ready for execution
130	Execution is cancelled after timeout

Reason codes

1000	Unknown reason
1001	A parameter is wrong
1002	A requested file could not be found
1003	A requested folder could not be found
1004	A requested folder is a file
1005	A requested file is a folder
1006	30 days trial period expired
1007	Time limited keycode expired
1008	Invalid activation
1009	PDF does not contain ICC profiles
1010	A file could not be opened
1011	An encrypted PDF file could not be opened for writing
1012	A file could not be saved

Run as a Server / Distributed Processing

Start as a Server

callas pdfaPilot Server/CLI can also be used for processing hotfolders on platforms, where no user interface for the configuration of the require settings is available (like Linux, SunSparc, SunIntel).
Remote access is not available for AIX.
This possiblity to start a Server without a user interface is available on MacOS and Windows also of course.
First, the pdfaPilot CLI has to be started in server mode:

```
--server [--quiet] [--accesskey=accesskey]
[--port=port] [--cachefolder=cachefolder]
```

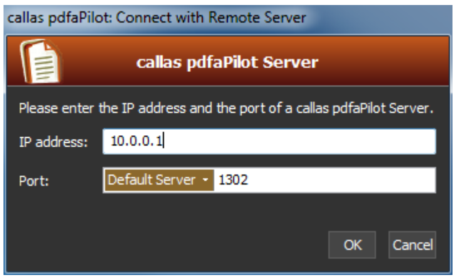
Options

--quiet	optional, suppresses outpput
--accesskey	optional, sets a accesskey for restricting the possibilty to change configuration using the server user interface
--port	optional, defines the port for communication between CLI and server user interface via the network (Default: 1302)
--cachefolder	optional, defines the path to cachefolder

Example:

```
--server --port=1302 --accesskey=123456
```

For setting up a job for hotfolder processing, connect to the remote server using any pdfaPilot Desktop installation in the same network:


callas pdfaPilot: Connect with Remote Server
callas pdfaPilot Server
Please enter the IP address and the port of a callas pdfaPilot Server.
IP address: 10.0.0.1
Port: Default Server 1302

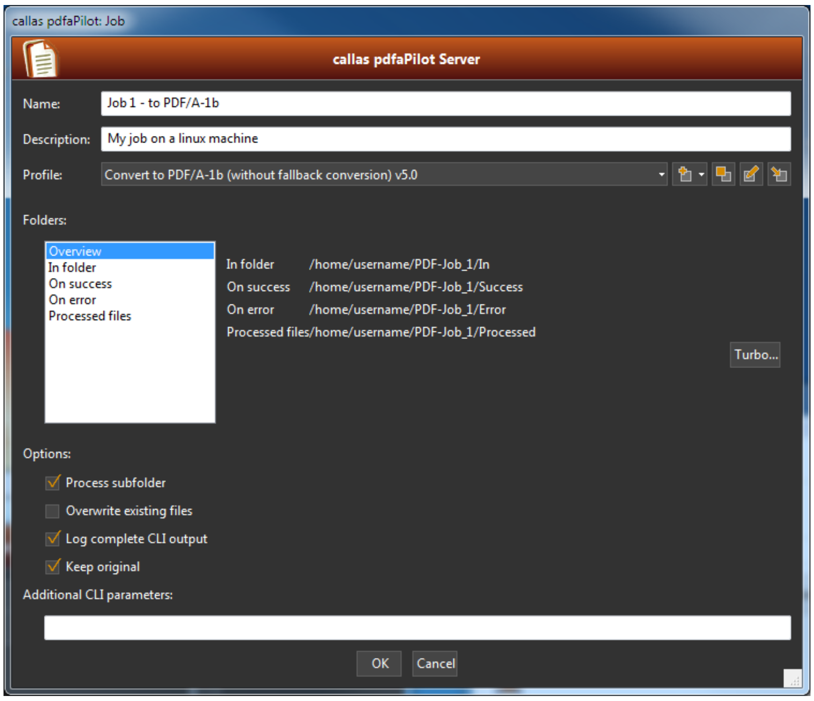
After entering the accesskey, new jobs can be configured, started or stopped. Profiles and Settings will be transfered to the remote server, where the are stored at /usr/share/callas software (path must be writable)

If this location can not be used caused by limitations on the respective environment, the additional option --cachefolder can be used for defining a custom path when starting the server:

Example:

```
--server --port=1302 --cachefolder=<PATH>
```

All paths defined for hotfolder processing need to be entered manually and have to be valid path specifications. (Hotfolder paths of any remote server jobs (IN, OUT, etc.) have to be configured so that they are valid from the service's perspective (the system where the service is running) - and not from the perspective of the controlling standalone application.) The server can also be stopped by remote, but not started.



Name: Job 1 - to PDF/A-1b

Description: My job on a linux machine

Profile: Convert to PDF/A-1b (without fallback conversion) v5.0

Folders:

In folder	/home/username/PDF-Job_1/In
On success	/home/username/PDF-Job_1/Success
On error	/home/username/PDF-Job_1/Error
Processed files	/home/username/PDF-Job_1/Processed

Options:

- ✓ Process subfolder
- ☐ Overwrite existing files
- ✓ Log complete CLI output
- ✓ Keep original

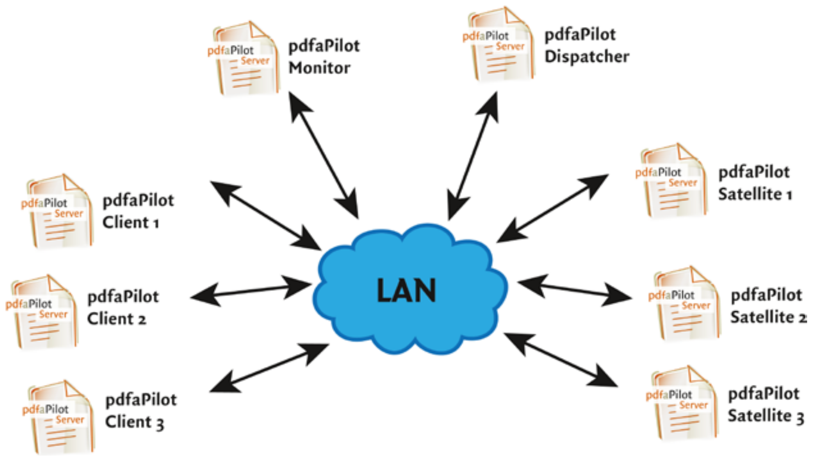
Additional CLI parameters:

Distributed Processing mode

callas pdfaPilot Server/CLI can be used in distributed processing mode in which all jobs are distributed over the network to as many "satellites" as being present and results are send back to the originator. Therefore pdfaPilot Server/CLI may be started in different modes:

- "Dispatcher" must to be present exactly one time in the network. This node controls which jobs are to be processed by which machines: the "satellites".
- "Satellite" receives jobs from the clients or directly from the dispatcher (if the dispatcher is run with hotfolders), processes them and sends them back to the clients.
- "Client" asks the dispatcher for satellites and after receiving the next satellite it sends jobs to the satellites and receives the results.
- "Monitor" monitors the dispatcher and displays the current situation.

All of these modules can run on the same or on different machines. There needs to be exactly one dispatcher and at least one satellite. In order to submit jobs at least one client is required.
Distributed processing is supported for Windows, MacOS, Linux, SunSolaris and SunIntel. It is not available on AIX.



Starting a Dispatcher

```
--dispatcher [--port=<port number>] [--noserver]
```

Example:

```
--dispatcher [--port=1300] --noserver
```

Port is the port number on which the Dispatcher can be called over the network. This port is set to 1300 as default.

By the setting the `--noserver` option, the Dispatcher will not observe existing hotfolders, but only distribute jobs to Satellites sent in by Clients. This option is only available using the CLI.

Starting a Dispatcher using the ServerUI

There is also the possibility to start a server as a dispatcher on Windows and MacOS using the user interface. Also hotfolder-processing can be set up here. In this mode, the Dispatcher will also distribute jobs which are send by other Clients.

Starting a Satellite

```
--satellite --endpoint=<dispatcher ip
number>[:<dispatcher port>] [--port=<port number>]
[--connections=<number of concurrent connections]
```

Example:

```
--satellite --endpoint=10.0.0.100:1300 --port=1301
```

In order to process jobs at least one Satellite is required. Endpoint is the IP number and the port of the Dispatcher. Port is the port that the Satellite is using in order to communicate with the clients. The port of the Satellite is 1301 as default and can be defined optionally to another one.

Starting a Satellite using the ServerUI

There is also the possibility to start a Server as a Satellite on Windows and MacOS using the user interface. In this mode, the Satellite will not process any hotfolder jobs on the computer.

A Satellite will always use the number of CPUs on the respective machine as the number of concurrent connections/processes. To limit this number, the Satellite has to be started by CLI with the `--connections` parameter. The number of connections should not exceed the number of CPUs, as this might reduce the performance per process and could result in some system stability problems.

Distribute a process using a Client

The client is called using any regular pdfaPilot command line command. In order to distribute the call over the network the command line parameters --dist and --endpoint are added. The client will then first ask the Dispatcher to receive a Satellite connection and then send the command to the Satellite and wait until the result is sent back from the Satellite.

```
pdfaPilot --dist --endpoint=<dispatcher ip
number>[:<dispatcher port>] <any regular pdfaPilot
call>
```

Examples:

```
pdfaPilot --dist --endpoint=10.0.0.100:1300 <myPDF.pdf>
```

```
pdfaPilot --dist --endpoint=10.0.0.100:1300 --level=2b
--analyze <myPDF.pdf>
```

Set type of satellite

As some kinds of jobs shall only be processed on a defined type of Satellite, it is possible to start a Satellite with one or more types set.
Every CLI call can also be amended with one or more typification of allowed types of Satellites the job shall be processed by.

Set typification for Satellite:

```
pdfaPilot --satellite --endpoint=<dispatcher IP number>
--satellite_type=<type> [--satellite_type=<type>]
```

for example:

```
pdfaPilot --satellite --endpoint=10.0.0.100
--satellite_type=A
```

```
pdfaPilot --satellite --endpoint=10.0.0.100
--satellite_type=A --satellite_type=B
```

Set typification for Client:

```
pdfaPilot --dist --endpoint=<dispatcher IP number>
--satellite_type=<type> [--satellite_type=<type>] <any
regular pdfaPilot call>
```

for example:

```
pdfaPilot --dist --endpoint=10.0.0.100 --satellite_
type=A <any regular pdfaPilot call>
```

Implementation details:

- If a Satellite has been started with a typification, only Client calls with the same type set will be send to this satellite.
- If a Client call contains a number of typifications, all typifications must match with those set for a satellite.
- If a Client call has no typfication set, it can be processe on all satellites, even they have been started with a typfication.
- The `<type>`-string has to be alpha-numeric and is case sensitive.

Avoid local processing

As a fallback, processing can be performed locally if either the action can not be distributed, a Satellite can not be assigned within a timeframe or if no Dispatcher is available.
This type of local processing might be not desired for several reasons.
To avoid such local processing, the Client call can be amended as well as the start of a Dispatcher (if run as a server with hotfolders) with the option `--nolocal`.

Example for Client:

```
pdfaPilot --dist --endpoint=<dispatcher IP number>
--nolocal <any regular pdfaPilot call>
```

Example for Dispatcher:

```
pdfaPilot --dispatcher --nolocal
```

Fallback for Dispatcher

In some workflow systems, a fallback for a Dispatcher might be required to ensure production stability.
To cover this, a number of Dispatcher can be set up, which will run individually. One or multiple Dispatcher can be assigned to a Satellite.

Define multiple Dispatcher to a Satellite

Connects a satellite to two (or more) Dispatcher.

```
pdfaPilot --satellite --endpoint=<dispatcher 1 IP>
[--endpoint=<dispatcher 2 IP> [--endpoint=<dispatcher
IP>]
```

Set multiple Dispatcher in a Client call

Distributes a Client call via two (or more) Dispatcher. First reachable Dispatcher with free satellite will process the job.

```
pdfaPilot --dist --endpoint=<dispatcher 1 IP>
--endpoint=<dispatcher 2 IP> [--endpoint=<dispatcher
IP>] <any regular pdfaPilot call>
```

Define a timeout for processing

In some workflow systems, long running processes might not be allowed and shall be cancelled if a give timeframe is reached.
Due to the flexibility of distributed processing, a variety of timeouts for the individual parts can be set:

- for the Cient call
- for the Satellite
- for the Dispatcher

Timeout for processing on a Satellite

When defining a timeout for the Client call, the execution will be cancelled after the given period.
When defining a timeout when starting a Satellite, all jobs processed by this Satellite will be cancelled after the given period.
If both are defined, the shorter timeframe will be used.

Example for Client:

```
pdfaPilot --dist --endpoint=<dispatcher IP> --timeout_
satellite=<seconds> <any regular pdfaPilot call>
```

Example for Satellite:

```
pdfaPilot --satellite --endpoint=<dispatcher IP>
--timeout=<seconds>
```

Timeout for local processing of Dispatcher or Client

A processing timeout (if no satellite is available or if the type of job can not be distributed) for the fallback to local processing on the Client or the Dispacher (when used as a server for hotfolders) can also be defined.
If both are defined, the shorter timeframe will be used.

Example for Client:

```
pdfaPilot --dist --endpoint=<dispatcher IP>
--timeout=<seconds> <any regular pdfaPilot call>
```

Example for Dispatcher:

```
pdfaPilot --dispatcher --timeout=<seconds>
```

Timeout for Dispatcher to search for Satellites

Additionally, also a timeout for the Dispatcher can be set, which will define the timeframe in which is searched for Satellites.
This can also be set individually for every Client call or when starting the Dispatcher (will have effect on all distributed files then).
If both are defined, the shorter timeframe will be used.

Example for Client:

```
pdfaPilot --dist --endpoint=<dispatcher IP> --timeout_
dispatcher=<seconds> <any regular pdfaPilot call>
```

Example for Dispatcher:

```
pdfaPilot --dispatcher --timeout_dispatcher=<seconds>
```

If a timeout for satellites or dispatcher is set and the --nolocal option has been defined, it will not be tried to process the job locally. Processing will end up in an error.

Setting --timeout... or --nolocal parameters in the "Additional CLI parameter" area of the Server UI is not supported at the moment.

Using the CLI-Monitor

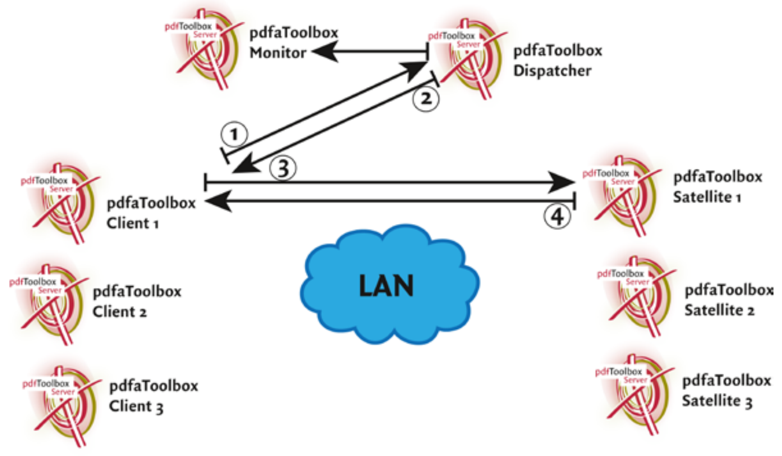
```
pdfaPilot --monitor --endpoint=<dispatcher ip
number>:<dispatcher port> [--endpoint=<dispatcher
IP>:<dispatcher port>]
```

Example:

```
--monitor --endpoint=10.0.0.100:1300
```

Monitor is optional and mirrors the command line output of the dispatcher to another computer. Endpoint is the IP number and the port of the dispatcher.
When using more than one Dispatcher, also multiple Dispatcher IPs can be entered and observed.

Communication



1) Clients sends a request for Satellite to Dispatcher

2) Dispatcher assigns a Satellite and send the address to the Client

3) Client send the job to the Satellite

4) Satellite send the result back to the Client

Licensing

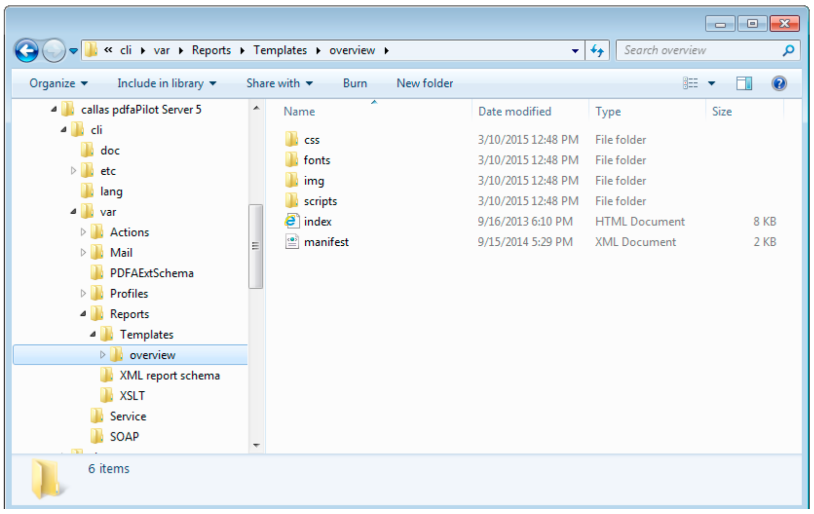
- Server: Regular pdfaPilot Server/CLI license required
- Dispatcher: Dispatcher pdfaPilot Server/CLI license required
- Satellite: Regular pdfaPilot Server/CLI license required
- Monitor: No license required
- Client: No license required

HTML-based custom reports

To adjust PDF-reports easily, HTML-based custom reports can be used. The visual appearance is controlled by a HTML-Template and Custom Style Sheets (CSS), while the reported details are directly requested from pdfaPilot or (optionally) parsed from an internally created XML-report.

Structure of related files

A predefined HTML-template is contained in all installer packages for Desktop and Server/CLI.


« cli › var › Reports › Templates › overview ›

Search overview

Organize ▾ Include in library ▾ Share with ▾ Burn New folder

- callas pdfaPilot Server 5
  - cli
    - doc
    - etc
    - lang
    - var
      - Actions
      - Mail
      - PDFAExtSchema
      - Profiles
      - Reports
        - Templates
          - overview
          - XML report schema
          - XSLT
      - Service
      - SOAP

Name	Date modified	Type	Size
css	3/10/2015 12:48 PM	File folder	
fonts	3/10/2015 12:48 PM	File folder	
img	3/10/2015 12:48 PM	File folder	
scripts	3/10/2015 12:48 PM	File folder	
index	9/16/2013 6:10 PM	HTML Document	8 KB
manifest	9/15/2014 5:29 PM	XML Document	2 KB

6 items

This predefined template can be found in:

- Server/CLI:
  ../cli/var/Reports/Templates
- Desktop/PlugIn:
  <User Preferences>/callas software/pdfaPilot <version>/Reports/Templates
  (using Desktop/PlugIn, a HTML-based report must have been generated at least once, in order to have these these files created)

The predefined template contains several folders and files

• index.html	the template in HTML format
• manifest.xml	a XML file which defines information needed as content for the report, to be delievered by the pdfaPilot engine
• /css	contains a style sheet
• /fonts	contains used fonts
• /img	contains used images
• /scripts	contains used JavaScripts

It is highly recommended to create a copy of the original template in a separate folder when starting to adjust a HTML-template based report.

The manifest.xml

The manifest.xml defines the set of information to be provided by the pdfaPilot engine. This information will be used to to fill up the details in the report based on the HTML-Template.
Basic document information as well as all results of the processed profiles are provided by default. Other parts like a preview image, comparision images or an XML report can be also requested here. Even a XML report can be ordered to enable picking up additional information about the PDF or executed fixups or checks using JavaScript.
The display name in the user interface is defined here as well.
For developing purposes, the internally generated, filled HTML representation of the report can be maintained to review changes in the template files also in a browser.

The HTML converter is using WebKit, so it is recommended to use Safari (or Chrome, which is based on a spin-off of WebKit) as a browser.

Request basic informations about PDF

```
<x:dict>
        <x:overview/>
</x:dict>
```

Purpose

If contained, document information and results of the performed profile will be available for using them in the HTML template.

Preview images of pages

```
<x:results>
        <x:preview resolution="150" page="1"/>
</x:results>
```

Purpose

Rendering of images of one or more pages for visual represenation of the PDF in the report.

Visual comparision of original and processed file

```
<x:compare>
        <x:document_a resolution="20"/>
        <x:document_b resolution="20"/>
        <x:diffresult resolution="20"/>
</x:compare>
```

Purpose:

Include compare tree if comparison resources are used inside index.html.

Parameters

resolution	resoultion used in ppi for rendering the comparision

Keep the temporarily generated files

```
<x:settings>
        <x:keeptemp>true</x:keeptemp>
</x:settings>
```

Purpose

Temporarily generated files like the filled index.html, CSS-files, images, XML-reports and used JavaScripts will not become deleted after finnishing the PDF report.

Parameters

false	files become deleted (default)
true	files will not become deleted

Creating a XML report for additional content

```
<x:results>
        <x:xmlreport path="xml/report.xml"
inkcovres="72" inkcovbox="TrimBox"/>
</x:results>
```

Purpose

Requests a XML report of the performed profile to extract additional information using JavaScript which can be used in the report.
Determining the ink coverage will only take place if one of the respective parameters exists.

Parameters

inkcovres	resolution in ppi, used for determining the effective ink coverage of each page in the PDF (optional, default: 300)
inkcovbox	page geometry box of which the effective ink coverage will be determined (optional, default: CropBox)

The HTML template

The HTML template can easily be modified using an appropriate code-editor or enhanced text-editor.

The `index.html` of the default "overview" template references a stylesheet, two JavaScripts as well as a number of images.

The provided HTML-template already contains some "dummy" data, which is automatically replaced by actual content when a new report is generated. So, when doing adjustments to the template with custom profiles and PDF files, it is recommended to keep the temporarily generated files for debugging, as a basis for modifications and their review in a browser.

It is possible to use image formats (like JPEG or PNG) as well as PDFs for positioning visual content like logos. If you want to debug your HTML in an HTML Browser you may want to display an image instead of the PDF reference. You can do this by putting identically named files for images and PDF next to each other. The PDF file must be referenced in the <img> tag in the HTML-template. The usage of PDF files allows for higher quality of logos in the resulting PDF report.

How the HTML-template works

The provided HTML-template contains already all document and processing information which can be supplied directly from the application.

You will find `<cals_params.js>` and `<cals_overview.js>` which are used by the engine to create and render the HTML. Please do not modify these files.
For adding more functionality it is recommended to do this in new, own JavaScript files, which have to be linked in the `<index.html>` of course.

Examples for template modification

callas software is providing a number of sample templates, which can be downloaded using the following link:

http://www.callassoftware.com/manuals/callas_Tutorial_HTML-reports.zip

This package contains samples as complete template folders including comments in the HTML-, CSS- or JavaScript-files and the result as a PDF.

Run as a Service

Start as a server, dispatcher or satellite

The callas pdfaPilot Services application is only available for Windows at the moment.

Installation

1.
Ensure there is an installation of callas pdfaPilot Server/CLI on the system and the application has been activated successfully.

2.
A special executable, which is needed to run pdfaPilot as a service, is located in /cli/var/Service.
Copy the executable into the subfolder "/cli" of the application folder of the server installation.

3.
To install, the following command has to be executed on the command line:

```
pdfaPilotService.exe --install
```

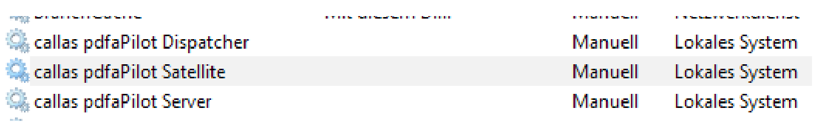
Please confirm the security question of Windows if shown.

4.
Open the "Services dialog" of Windows. This dialog can easily opened by typing the following string into the search field of the Windows start menu or use the following command on the command line:

```
Services.msc
```

5.
There should show up 3 new services:

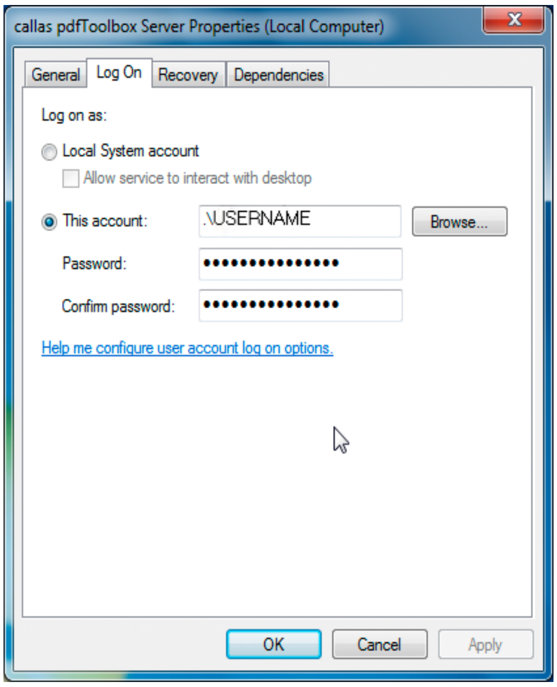
- callas pdfaPilot Server
- callas pdfaPilot Satellite
- callas pdfaPilot Dispatcher



callas pdfaPilot Dispatcher	Manuell	Lokales System
callas pdfaPilot Satellite	Manuell	Lokales System
callas pdfaPilot Server	Manuell	Lokales System

6.

Select "pdfaPilot Server" and use the right-click menu item "Action" [or "Properties" depending on the Windows-version] in order to use this service and set in "General" the "Startup Type" to "Automatic" or "Manual".
When "Automatic" is choosen, every started job will continue processing, even when no user is logged on the system. It will even start processing, when the operating system is started.
When using "Automatic", also user details have to be entered into the "Log On" tab.
It must be ensured, that all folders used in the jobsettings can be accessed by the defined user (especially when network paths shall be used by the job).



callas pdfToolbox Server Properties (Local Computer)

Log on as:

Local System account

Allow service to interact with desktop

[✓ This account:] .\USERNAME

Password: •••••••••••••••

Confirm password: •••••••••••••••

Help me configure user account log on options.

Configration of a job

Now a job can be configured using the ServerUI, which can be accessed using the Standalone version (Menu: Tools - Server).
When a job is started, pdfaPilot Standalone can be closed. The services application ensures, that the processing will continue even when the user is logging off.

Access by remote

It is possible to connect to a Server running as a service by remote via the local network.
Start pdfaPilot standalone, select menu: "Tools" - "Server" and choose "Connect with remote server".
Enter the IP of the remote server where the service is running.
After connecting all jobs on the remote server are shown and can be started, stopped or even modified. (Hotfolder paths of any server jobs (IN, OUT, etc.) have to be configured so that they are valid from the service's perspective (the system where the service is running) - and not from the perspective of the controlling standalone application.)

Troubleshooting

If network paths are used for processing jobs, the user should have sufficient rights to access them.

There may be special requirements for converting Office files to PDF when using pdfaPilot as a service. Check http://www.callassoftware.com/goto/apl_ENU_faqs for the latest details.

Additional settings for Office conversion with restricted user permissions

If Office conversion is needed out of services using Windows it is recommended to grant the respective service user administrator privileges.
If this level of rights can not set due to internal regulations, some additional settings within the operating system are recommended.

The following folders should allow the user the respective access right:

For 64-bit machines	
C:\Windows\Temp	`Modify`
C:\Windows\syswow64\config	`Read`
C:\Windows\syswow64\config\systemprofile	`Read`
C:\Windows\syswow64\config\systemprofile\AppData	`Modify`
C:\Windows\syswow64\config\systemprofile\Desktop	`Modify (Create it if it does not exist)`

For 32-bit machines	
C:\Windows\Temp	`Modify`
C:\Windows\system32\config	`Read`
C:\Windows\system32\config\systemprofile	`Read`
C:\Windows\system32\config\systemprofile\AppData	`Modify`
C:\Windows\system32\config\systemprofile\Desktop	`Modify (Create it if it not exist)`

Additional settings:

- Set the 32-bit folder preferences (detailed above) in addition to the 64-bit preferences on 64-bit systems running 64-bit versions of Microsoft Office
- Set the default printer to XPS Document Writer

DCOM settings:

- Launch DCOMCNFG by using:

```
C:\WINDOWS\SysWOW64> mmc comexp.msc /32
```

- Go to Computers > MyComputer > DCOM Config.
- Right-click the application that you want to automate.
  The application names are listed in the table below:

Application	DCOM Name
MS Access 97	`Microsoft Access Database`
MS Access 2000/2002/2003	`Microsoft Access Application`
MS Office Access 2007	`Microsoft Office Access Application`
MS Excel 97/2000/2002/2003	`Microsoft Excel Application`

Application	DCOM Name
MS Office Excel 2007	`Microsoft Excel Application`
MS Office Excel 2010	`Microsoft Excel Application`
MS Word 97	`Microsoft Word Basic`
MS Word 2000/2002/2003	`Microsoft Word Document`
MS Office Word 2007	`Microsoft Office Word 97 - 2003 Document`
MS Office Word 2010	`Microsoft Word 97 - 2003 Document`

- On some systems Microsoft Word is not displayed and you will have to use {00020906-0000-0000-C000-000000000046} instead.
- Click Properties to open the property dialog box for this application.
- Verify Identity and Security tabs