



---

# PDF Command Line Suite

## Version 4.12

### User Manual

---

---

Contact: pdfsupport@pdf-tools.com

Owner: **PDF Tools AG**  
Kasernenstrasse 1  
8184 Bachenbülach  
Switzerland  
<http://www.pdf-tools.com>

# Table of Contents

<b>Table of Contents.....</b>	<b>2</b>
<b>1 Overview .....</b>	<b>6</b>
1.1 The Different Tools.....	6
1.2 Installation.....	7
1.3 Using the Tools.....	7
<b>2 License Management .....</b>	<b>8</b>
2.1 Graphical License Manager Tool .....	8
List all installed license keys.....	8
Add and delete license keys .....	8
Display the properties of a license .....	9
Select between different license keys for a single product.....	9
2.2 Command Line License Manager Tool .....	9
List all installed license keys.....	9
Add and delete license keys .....	9
Select between different license keys for a single product.....	9
2.3 License Key Storage.....	10
Windows.....	10
Mac OS X.....	10
Unix / Linux .....	10
<b>3 General Options .....</b>	<b>10</b>
3.1 Usage.....	10
3.2 Encrypted files.....	10
-pw Decrypt the Input Files.....	11
-perm Set Permission Flags .....	11
-ownerSet Owner Password .....	11
-user Set User Password .....	11
-lk Set License Key .....	11
<b>4 pdcat .....</b>	<b>12</b>
4.1 Description .....	12
4.2 Functionality and Options.....	12
Merge documents .....	12
-r Allow replacing the file specified for output .....	12
-e exit with error on processing problem .....	12
-annot Delete Text Annotations .....	12
-bglogo Set a Background Logo.....	12
-box Set the Media Box .....	13
-bs: Border Style of Hyperlinks .....	13
-c Set the Document Information .....	13
-clip Clip Pages.....	13
-crop Set the Crop Box .....	14
-I Create Bookmarks for each Input File.....	14
-n Do not copy bookmarks of next input file.....	14
-l Use a Control File .....	14
Create Link Annotations .....	14
Create Bookmarks with Links .....	16

Create a Free Text Annotation .....	17
-logo Add Logos from a PDF Document.....	19
-m Add Named Destinations for each Page .....	19
-d Do not copy named destinations of next input file(s).....	19
-M Set Metadata .....	19
-oT Set Page Mode .....	19
-p: Copy a Specified Range of Pages .....	19
-pl Set Page Layout .....	19
-R Rotate Pages.....	20
-s Substitute Text Strings in Action Links .....	20
-sp Substitute a Substring in Action Links .....	20
-uc Un-embed File Collection .....	20
-vp Set Viewer Preferences.....	20
-wc Warn about collections .....	21
-uc Unembed collections .....	21
4.3 Common Transformations .....	22
Page Sizes .....	23
Examples for Transformations .....	24
<b>5 pdsplit .....</b>	<b>32</b>
5.1 Description .....	32
5.2 Functionality and Options.....	32
Split Files and name Output Files.....	32
-b Extract all or individual Chapters According to Bookmarks .....	33
-l Restrict the Processing to a Specific Bookmark Level .....	33
-m Exclude low-level Bookmarks .....	33
-p Split a Document into Parts with Specified Number of Pages .....	34
-x Exchange Bookmark Characters in File Names.....	34
<b>6 pdsel.....</b>	<b>35</b>
6.1 Description .....	35
6.2 Functionality and Options.....	35
Select Individual Pages or Ranges of Pages.....	35
-a Remove Annotations .....	35
<b>7 pdw .....</b>	<b>36</b>
7.1 Description .....	36
7.2 Functionality and Options.....	36
-c Break down Text Blocks to Individual Character .....	36
-cr Add Carriage Return before New Lines .....	36
-o List the Annotations in a Separate File.....	36
-r Take Account of the Page Rotation.....	36
-u List the Text in Unicode Encoding .....	36
-w Break down Text Blocks to Blank Separated Words .....	36
<b>8 pdform.....</b>	<b>38</b>
8.1 Description .....	38
8.2 Functionality and Options.....	38
Fill in data .....	38
Add a new form field .....	38
Delete a field .....	38
-l List all Form Fields.....	38

November 27, 2018

<b>9</b>	<b>pdwebl</b> .....	<b>39</b>
9.1	Description .....	39
9.2	Functionality and Options.....	39
	Add URL Links to a PDF Document.....	39
	Add Page links to a PDF Document.....	39
	Add Java Scripts.....	39
-i	Read Input File from Standard Input.....	39
-l:	Define Key/Link Pairs in a File or Standard Input .....	40
-q	Quite Mode .....	40
-s	Set the border style of links .....	40
<b>10</b>	<b>pdtoc</b> .....	<b>42</b>
10.1	Description .....	42
10.2	Functionality and Options.....	42
-b	Create or Omit Bookmarks.....	42
-I	Set New Titles .....	42
-d	Place the Current Date on the Pages .....	42
-w	Set the Page Width .....	42
-t	Place a Header Text on the Pages .....	42
-c	Set the Document Title.....	42
-dest	Create Named Destination Links .....	42
-url	Create URL Links .....	43
-@	Read Input from a Control File .....	43
<b>11</b>	<b>pdbm</b> .....	<b>44</b>
11.1	Description .....	44
11.2	Functionality and Options.....	44
-d	List Named Destinations .....	44
-D	List Named Destinations Tab-separated.....	44
-o	Redirect the Output to a File .....	44
-oa	Page Mode, Initial Page Number, Open Action .....	44
-a	Add Bookmarks from an Input File.....	45
-n	Do not Print Destinations.....	46
-n1	Add a Leading Hyphen .....	46
<b>12</b>	<b>pdpq</b> .....	<b>47</b>
12.1	Description .....	47
12.2	Functionality and Options.....	47
-c	List the CropBox.....	47
-f	List All Fonts on Pages.....	47
-pAll	Get the Total Number of Pages in a PDF Document.....	48
-fAll	List all Fonts in a Document .....	48
-m	List the MediaBox .....	48
-p	Set the Page Range .....	48
-r	List the Page Rotation .....	49
-s	Abbreviate Output .....	49
-u	Disable UserUnit Adjustment.....	49
<b>13</b>	<b>pdxt</b> .....	<b>50</b>
13.1	Description .....	50
13.2	Functionality and Options.....	50

November 27, 2018

Specify a PDF File Containing the Logo .....	50
Select the logo page .....	50
Put the Logo on top of the Page or in the Background .....	50
<b>14 txt2pdf.....</b>	<b>51</b>
14.1 Description .....	51
14.2 Functionality and Options.....	51
-ff Set the font .....	52
-fs Set the font size .....	52
-pb Set the page border size.....	52
-ps Set the page size.....	52
-tw Set the line wrap mode .....	52
-v Set verbose mode.....	52
14.3 Error messages and codes .....	52
<b>15 pdcrop .....</b>	<b>53</b>
15.1 Description.....	53
15.2 Functionality and Options.....	53
-ob Base the Cropping on a Specified Box .....	53
-nb Specify the Box Type to Be Set .....	53
-shrinkShrink the Box .....	53
<b>16 pdmerge .....</b>	<b>55</b>
16.1 Description .....	55
16.2 Functionality and Options.....	55
Simple Merge of PDF Files.....	55
-c Create a Table of Contents.....	55
-t Merge PDF Files and Add Individual Outlines .....	56
-WP Specify Javascript Code .....	56
@ Use a Control File .....	57
-ax Set XMP Metadata .....	57
<b>pdinfo .....</b>	<b>58</b>
<b>17 pdobj .....</b>	<b>59</b>
<b>18 pdls.....</b>	<b>60</b>
<b>19 COM Interface.....</b>	<b>61</b>
19.1 Overview .....	61
19.2 Installation.....	61
19.3 Examples.....	62
Declaration .....	62
Parameter Passing .....	62
Execution .....	62
<b>Appendix A: Security .....</b>	<b>63</b>
<b>Appendix B: Link Definition Files for pdcat .....</b>	<b>63</b>

# 1 Overview

## 1.1 The Different Tools

---

The Command Line Suite consists of a series of tools to manipulate PDF documents in various ways or extract information. The tools are based on the PDF Library SDK.

The following tools are part of the PDF Command Line Suite.

- pdcat**      The pdcat tool concatenates (merges) PDF files. The tool can also merge annotations from another PDF file, generate book marks, add a named destination for each page, set the title text of the output file, rotate pages, add link annotations specified in an ASCII file, substitute file names in action links, clip and crop pages, and more
- pdw**        The pdw tool analyses text and prints it along with position and size information. You can process this output to generate the hyperlink definitions for pdlink. (The pdw sample program does not handle the full set of stream operators and neither all fonts).
- pdxt**        The pdxt tool adds a logo taken from a first PDF file and puts it on all pages of a second input PDF file. The logo can be put behind or on top of the page content.
- pdform**      The pdform tool displays information about text form fields, or fills in data into form fields.
- pdsel**        The pdsel tool allows to select (extract) pages from a PDF file.
- pdwebl**      The pdwebl tools can add web links (URL link annotations) to a PDF file.
- pdtoc**        The pdtoc tool creates a PDF file containing links to existing PDF files. It can also add URL links and links to named destinations.
- pdsplit**      This tool splits a PDF file into single page files, changing links between pages into links between the spited files.
- pdbm**        This tool lists bookmarks contained in a PDF file.
- pdpg**        This tool lists basic information about a PDF file, such as number of pages, fonts, MediaBox, CropBox etc.
- txt2pdf**      The txt2pdf tool creates PDF files from ASCII text
- pdcrop**      This tool can be used to crop PDF files.
- pdinfo\***      The pdinfo tool lists the info object of a PDF file
- pdobj\***        The pdobj tool lists individual objects of the PDF file
- pdls\***        The pdls tool lists all pages objects of a PDF file and optionally their contents streams

(\*) These tools are specialized tools which require profound knowledge about PDF technology.

## 1.2 Installation

---

The PDF Command Line Suite comes as a compressed archive (ZIP for Windows, tar.gz for Unix platforms). Extract the contents of the archive to the file system. There will be a "bin" folder containing a Win32, x86, and/or an x64 subfolder (depending on the platform). These subfolders contain the executables. You may run them directly from this location in the file system, or copy them to a more suitable folder (like /usr/bin on Unix).

The Windows kit also contains the PDApp.DLL file which offers the functionalities of the pdcat and the pdsplit command line tools via a COM interface. To use this functionality, the 32-bit and the 64-bit DLLs need to be registered with COM using the REGSVR32.EXE tool (administrator privileges required).

## 1.3 Using the Tools

---

Each tool prints a short text on how to use it to standard output when executed without parameters. This text consists of the usage and a list of the available parameters.

The command line syntax is Unix like. Command line arguments in brackets ([ ... ]) are optional. Arguments in curly brackets ({ ... }) can be repeated.

The tools have a usage, to provide a brief description of all available features. To retrieve the usage, they the tool name without any arguments.

This documentation does not cover all features, but only common features and meaningful use-cases. For undocumented features, please refer to the usage.

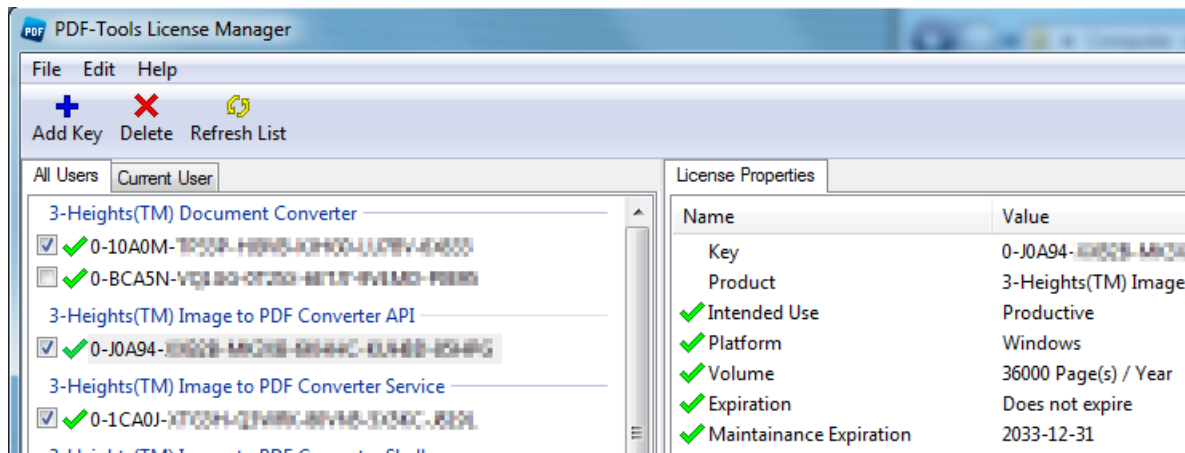
## 2 License Management

There are three possibilities to pass the license key to the application:

1. The license key is installed using the GUI tool (Graphical user interface). This is the easiest way if the licenses are managed manually. It is only available on Windows.
2. The license key is installed using the shell tool. This is the preferred solution for all non-Windows systems and for automated license management.
3. The license key is passed to the application at runtime via the command line switch `-lk` property. This is the preferred solution for OEM scenarios.

### 2.1 Graphical License Manager Tool

The GUI tool *LicenseManager.exe* is located in the *bin* directory of the product kit.



#### List all installed license keys

The license manager always shows a list of all installed license keys on the left pane of the window. This includes licenses of other PDF Tools products.

The user can choose between:

- Licenses available for all users. Administrator rights are needed for modifications.
- Licenses available for the current user only.

#### Add and delete license keys

License keys can be added or deleted with the "Add Key" and "Delete" buttons in the toolbar.

- The "Add key" button installs the license key into the currently selected list.
- The "Delete" button deletes the currently selected license keys.



November 27, 2018

---

## Display the properties of a license

If a license is selected in the license list, its properties are displayed in the right pane of the window.

## Select between different license keys for a single product

More than one license key can be installed for a specific product. The checkbox on the left side in the license list marks the currently active license key.

## 2.2 Command Line License Manager Tool

---

The command line license manager tool *licmgr* is available in the *bin* directory for all platforms except Windows.

A complete description of all commands and options can be obtained by running the program without parameters:

```
licmgr
```

### List all installed license keys

```
licmgr list
```

The currently active license for a specific product is marked with a star '\*' on the left side.

### Add and delete license keys

Install new license key

```
licmgr store X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Delete old license key

```
licmgr delete X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

Both commands have the optional argument **-s** that defines the scope of the action:

- **g**: For all users
- **u**: Current user

### Select between different license keys for a single product

```
licmgr select X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX
```

## 2.3 License Key Storage

---

Depending on the platform the license management system uses different stores for the license keys.

### Windows

The license keys are stored in the registry:

- HKLM\Software\PDF Tools AG (for all users)
- HKCU\Software\PDF Tools AG (for the current user)

### Mac OS X

The license keys are stored in the file system:

- /Library/Application Support/PDF Tools AG (for all users)
- ~/Library/Application Support/PDF Tools AG (for the current user)

### Unix / Linux

The license keys are stored in the file system:

- /etc/opt/pdf-tools (for all users)
- ~/.pdf-tools (for the current user)

Note: The user, group and permissions of those directories are set explicitly by the license manager tool.

It may be necessary to change permissions to make the licenses readable for all users. Example:

```
chmod -R go+rx /etc/opt/pdf-tools
```

## 3 General Options

### 3.1 Usage

---

The usage will be listed by typing a command without parameters.

You may detect useful options for some of the tools that are not described in detail in this manual.

### 3.2 Encrypted files

---

Encrypted input files, that have a user password, can be read with most tools if either the user or owner password is provided.

November 27, 2018

---

A PDF document can have two passwords: The owner password and a user password. The owner password is to protect the document from unwanted changes or actions taken. There are several security flags which can be set for this purpose (see Appendix A: Security). When a PDF document is encrypted with a user password, the content of the document can only be read (opened) when the user password is known.

Most, but not all tools of the PDF Command Line Suite support the following encryption related options:

**-pw Decrypt the Input Files**

Specify either the user or the owner password to read an encrypted PDF input document.

```
pdcat -pw PW1 in1.pdf -pw PW2 in2.pdf out.pdf
```

**-perm Set Permission Flags**

Set the access permission flags. See also Appendix: Security for additional information.

**-ownerSet Owner Password**

Set the owner's password. That's the password required to change the access permissions of the document, such as the passwords and the permission flags.

**-user Set User Password**

Set the user's password. If that password is set, the user is prompted a password to open and read the document.

Specify a user and owner password and access permissions to encrypt the output files.

```
pdcat -user upw -owner opw -permit psca in.pdf out.pdf
```

For the permissions flags see Appendix A: Security

**-lk Set License Key**

Pass a license key to the application at runtime instead of installing it on the system.

## 4 pdcat

### 4.1 Description

---

The main functionality of pdcat is to concatenate pages from different input documents into one new output document. It can also add and replace links, reset the media and crop box, as well as clip and scale pages or add logos.

Please note that if a PDF document is provided multiple times as input file, the resources (such as images or embedded fonts) will no longer be shared. This can lead to an increase of the file size.

### 4.2 Functionality and Options

---

#### Merge documents

Merge the pages of multiple (two or more) PDF input documents into one output document:

```
pdcat in1.pdf in2.pdf in3.pdf out.pdf
```

**User's Tip:** One should try and avoid using the same input document multiple times. The main reason is that resources (such as embedded fonts and images) are embedded from every input file. Hence if the same resource is used on two different pages of the same input document, and these two pages are merged using the same input file twice, the resource is embedded twice. For page re-assembly of the same input document use the tool **pdsel** instead.

#### -r Allow replacing the file specified for output

```
pdcat -r in1.pdf in2.pdf in3.pdf out.pdf
```

This option must be specified as the first option and will allow that the file specified as output can be overwritten.

#### -e exit with error on processing problem

#### -annot Delete Text Annotations

Do not copy any text annotations to the output file.

```
pdcat -annot in.pdf out.pdf
```

#### -bglogo Set a Background Logo

Use *logo.pdf* as background logo, *in.pdf* as input and create a new output document with the logo called *out.pdf*. In the background means behind the content of the page.

November 27, 2018

---

```
pdcat -bglogo:1-1 logo.pdf in.pdf out.pdf
```

**User's Tip:** If the page contains opaque items, such as non-transparent images or vector graphics a logo in the background might remain partially or fully hidden behind these objects. In this situation use the switch **-logo** to put the logo in the foreground.

Add a logo with its size reduced to 50%. The coordinates of the lower left corner of the logo remain the same.

```
pdcat -logo +"0.5 0 0 0.5 0 0" logo.pdf in.pdf out.pdf
```

Add a logo with its size reduced to 25% and move it 200 points to the right and 300 points to the top. Add the logo only to page 2 of *in.pdf*.

```
pdcat -logo:2-2 +"0.25 0 0 0.25 200 300" logo.pdf in.pdf out.pdf
```

**User's Tip:** See also chapter **Common Transformations**.

### **-box Set the Media Box**

The media box marks the size of the page.

Set the media box to a square of 300x300 points at x=100/y=250.

```
pdcat -box "100 250 400 550" in.pdf out.pdf
```

**User's Tip:** The media-box defines the size of the page. If it's changed it has no impact on any objects on the page. This means resetting the media-box does not automatically delete objects outside the media-box. It therefore does not reduce the file size nor should be used to hide any confidential information, since it could simply be reset again.

### **-bs: Border Style of Hyperlinks**

Set the set border style for hyperlinks added via switch **-l**.

See chapter "Border Style" to find out more about border styles.

```
pdcat -bs:33009 in.pdf out.pdf
```

### **-c Set the Document Information**

Set document title attribute to "New Title":

```
pdcat -c "New Title" in.pdf out.pdf
```

Set author attribute to "Peter Pan":

```
pdcat -c "/Author=Peter Pan" in.pdf out.pdf
```

### **-clip Clip Pages**

Set a clipping area of the square 500 x 500 points, with (0,0) at lower left corner to the *out.pdf*.

```
pdcat -clip "0 0 500 500" in.pdf out.pdf
```

November 27, 2018

---

**-crop Set the Crop Box**

The crop box marks the size of the part that is to be displayed.

```
pdcat -crop "100 250 400 550" in.pdf out.pdf
```

If the size of the crop box is increased, the media box should be taken in account too. i.e. needs to be enlarged too, to the same size as the crop box.

Assuming the input file has a size of A4 (0, 0, 595, 842), the following command will increase the crop and media box by 50 points in each direction.

```
pdcat -crop "-50 -50 645 892" -box "-50 -50 645 892" in.pdf out.pdf
```

**-I Create Bookmarks for each Input File**

Create a bookmark called "BM 1" for the first document, create a bookmark called "BM 2" for the second bookmark:

```
pdcat -I "BM 1" in1.pdf -I "BM 2" in2.pdf out.pdf
```

**-n Do not copy bookmarks of next input file****-l Use a Control File**

The switch **-l** can be used for different purposes. It always requires an external control file which contains a series of commands. The control file is a text file containing the commands to create one or several of the items below:

- Link annotation (to another page or an URL)
- Bookmark with a link
- Free Text annotation

Lines in the control file that start with a semicolon are comments.

```
; This is a comment
```

**User's Tip:** The positioning of annotations use the coordinate system of the PDF document to which they are added. Most documents have a media-box, with its origin (0/0) in the lower left corner. However the media-box could also have different coordinate, e.g. lower left corner at (50/-100). If an annotation is added to such a document at position (0/0), its left border is 50 points outside the media-box and its bottom is 100 points above to bottom of the page.

Applies to: Link Annotations and Free Text Annotations.

**Create Link Annotations**

Link annotations supported by pdcat represent either a link to a destination in another PDF document, or URL.

November 27, 2018

---

**Page Number**

A link annotation starts with the required keyword `Page` followed by a number, which defines on what page number the link annotation is to be added. Commands after the keyword `Page` are applied to the selected page number until a new keyword `Page` or `Bookmark` is set.

```
Page 1
```

**Border Style**

Link annotations that point to another page or an URL can optionally have rectangular borders. A border style is specified as

```
.bs DLBGR
```

whereas the number after `.bs` has 5 digits with the following meaning:

D	Distance between dashes in points (0-9)
L	Length of dashes in points (0-9)
B	Blue value (0-9)
G	Green value (0-9)
R	Red value (0-9)

Example: A red dashed line is `.bs 33009`

A grey line is `.bs 00555`

**Destination**

The link annotation pointing to a page in another document or an URL is specified as

```
X Y height width path pagenr X-offset Y-offset
```

or

```
X Y height width URL
```

Where the meaning of the parameters are:

X	x-position of left lower corner of the annotation rectangle
Y	y position of the left lower corner of the annotation rectangle
height	Height of the annotation rectangle
width	Width of the annotation rectangle
Path	Path and filename of the file to which the link points. A relative path is also possible
pagenr	Page number of the linked file to be opened
x-offset	x position on the page of the linked PDF file that has to be places at the upper left corner of the viewing application
y-offset	y position on the page of the linked PDF file that has to be places at the upper left corner of the viewing application
URL	The URL to where the link annotation points

November 27, 2018

---

Note that in PDF the origin of the coordinate system is in the lower left of the page.

### **Example 1: Create a Link Annotation to another Page**

#### Step 1: Create Control File

Create a control text-file, name it *link.tmp*, add the following four lines:

```
Page 1
.bs 33009
; replace the target.pdf by an existing PDF document
100 100 200 200 "target.pdf" 2 100 150
```

The above control files does the following:

- Add a link annotation on page 1
- The annotation has a dashed, red border
- Its position is 100 from the left and 100 points from the bottom (100 points = 1.39 inch = 35.3 mm)
- It is 200 points wide and high
- It points to the second page of the document *target.pdf* and on that page to offset 100 points from the left and 150 points from the bottom

#### Step 2: Add Link Annotation

Add the link annotation defined in the control file using the following shell command:

```
pdcat -l link.tmp in.pdf out.pdf
```

### **Example 2: Create a Link annotation to an URL**

A link annotation pointing to an URL is constructed in a very similar way as a link annotation to another file (see Example 1). The first 4 parameters for the link rectangle remain the same. As 5<sup>th</sup> and last parameter the URL is provided as shown in the sample below:

#### Step 1: Create Control File

Create a text file with the following content and name it *link.tmp*:

```
Page 1
.bs 33009
100 100 200 200 http://www.pdf-tools.com
```

#### Step 2: Add Link Annotation

Add the link using the following command:

```
pdcat -l link.tmp in.pdf out.pdf
```

### **Create Bookmarks with Links**

Bookmarks are created by using the keyword `Bookmark` without parameters in the control file. Subsequent lines after the keyword define one bookmark each. Each line consists of three parameters:



November 27, 2018

level	description	destination
-------	-------------	-------------

Where the meaning of the parameters are:

level	The depth of the bookmark. A level 0 bookmark is a top-level (or root) bookmark. PDF supports a maximum bookmark depth of 28 levels.
description	The displayed text of the bookmark. If the text contains blanks, this parameter must be in "quotes".
destination	The destination of the bookmark. If the text contains blanks, this parameter must be in "quotes".

## Example

### Step 1: Create Control File

Create a text file with the following content and name it *link.tmp*:

Bookmarks
0 PDF-Tools http://www.pdf-tools.com
1 "CLS" http://www.pdf-tools.com/asp/products.asp?name=CLS

### Step 2: Add Bookmarks with Links

Add the link using the following command:

```
pdcat -b -l link.tmp in.pdf out.pdf
```

Note, that on Unix platforms, the end of a link is defined by a CR, whereas on Windows it is CR LF.

## Create a Free Text Annotation

A Free Text annotation is an rectangular annotation that contains text and is always visible, i.e. does not have an open or close status.

### Page Number

The page number of a Free Text annotation is set using the keyword `Page` followed by the actual page number.

Page 1
--------

### Style

The font style is defined using the keyword `font`, followed by a series of parameters.

font	FontName	FontSize	Alignment	TextColor	BgColor
------	----------	----------	-----------	-----------	---------

Where the meaning of the parameters are:

FontName	The font name. Must be one of the 14 PDF standard fonts. These are:			
	Times-Roman	Helvetica	Courier	Symbol
	Times-Bold	Helvetica-Bold	Courier-Bold	ZapfDingbats
	Times-Italic	Helvetica-Oblique	Courier-Oblique	

November 27, 2018

	Times-BoldItalic Helvetica-BoldOblique Courier-BoldOblique
FontSize	The font size in points.
Alignment	The alignment, supported values are L, R and C for left, right, center.
TextColor	The color of the text. The color must be set as PDF command, e.g. "0.5 g"
BgColor	The color of the rectangle in the background. The background color must be set as PDF command, e.g. "0.8 0.8 1 rg"

**Text**

The actual text an position is defined using the keyword text, followed by the two parameter for position and the actual text.

```
text X Y Text
```

Where the meaning of the parameters are:

X	The horizontal position, 0 being at the left hand side.
Y	The vertical position, 0 being at the bottom.
Text	The actual text. If the text contains blanks, this parameter must be in "quotes". If the text shall contain quotes as part of the text they must be prefixed with a backslash character \".

**Example****1. Create Control File**

Create a text file with the following content and name it *control.txt*:

```
Page 1
; comment lines start with a semicolon
; use the keyword "font" to set the font name, font size, alignment, text
color, and background color
; only the 14 PDF standard fonts are supported; if less parameters are
specified, previous or default values will be in effect
font Courier 24 L "0.5 g" "0.8 0.8 1 rg"
; use the keyword "text" to set the x/y coordinates and the actual text
to be printed; optionally, a width parameter can be passed to set the
width of the rectangle (default 0 = adjust width to text width)
text 10 700 "this is a piece of text with a \" inside"
font Courier 12 C(enter)
text 300 600 "centered" 80
```

**2. Add the Free Text Annotation**

Add the textbox using the following command:

```
pdcat -r +I1 -l control.txt in.pdf out.pdf
```

**-logo Add Logos from a PDF Document**

Add a logo to the foreground of every page of *in.pdf*.

```
pdcat -logo logo.pdf in.pdf out.pdf
```

**User's Tip:** See also chapter **Common Transformations** and switch **-bglogo**.

**-m Add Named Destinations for each Page**

A named destination for each of a document can be set using **-mPre**, where "Pre" is the name of the prefix. The following example sets the named destinations: "FirstBook1" on page 1, "FirstBook2" on page 2, etc.

```
pdcat -mFirstBook in.pdf out.pdf
```

**-d Do not copy named destinations of next input file(s)****-M Set Metadata**

Set document Metadata from a XML file. If the parameter 'none' is passed, no meta data is added.

```
pdcat -M meta.xml in.pdf out.pdf
```

**-oT Set Page Mode**

-oT: show thumb nails

-oO: show outlines (bookmarks)

-oP: page only

-oF: full screen

-oC: show layers

**-p: Copy a Specified Range of Pages**

Copy pages range 1 through 3:

```
pdcat -p:1-3 in.pdf out.pdf
```

Copy page 1 from first file and pages 3 to 4 from second file:

```
pdcat -p:1-1 in1.pdf -p:3-4 in2.pdf out.pdf
```

**-pl Set Page Layout**

The **-pl** option takes one of the following values: SinglePage, OneColumn, TwoColumnLeft, TwoColumnRight, TwoPageLeft, TwoPageRight

```
pdcat -pl TwoPageRight in.pdf out.pdf
```

November 27, 2018

---

**-R Rotate Pages**

Rotate the pages by 90 degree clockwise:

```
pdcat -R 90 in.pdf out.pdf
```

Rotate the pages of the first file by 90 degree clockwise, don't rotate the pages of the second file:

```
pdcat -R 90 in1.pdf -R 0 in2.pdf out.pdf
```

**-s Substitute Text Strings in Action Links**

If *in.pdf* contains for a web link like "http://www.google.com", the link can be replaced like this:

```
pdcat -s http://www.google.com http://www.yahoo.com in.pdf out.pdf
```

Please note this option replaces only the web link. If the link is related to a textual string in the content of the PDF document, this string will remain as is.

**-sp Substitute a Substring in Action Links**

The parameter **-sp** is similar to **-s**, but can be used to replace substrings. The following example replaces all (sub-) strings "tools" by "fools".

```
pdcat -sp tools fools in.pdf out.pdf
```

**-uc Un-embed File Collection**

This switch is used for un-embedding of attached files in a collection.

**-vp Set Viewer Preferences**

This option permits the setting of the following viewer preferences:

- HideToolbar (Tool)
- HideMenubar (Menu)
- HideWindowUI (UI)
- FitWindow (Fit)
- CenterWindow (Center)
- DisplayDocTitle (Title)

Any of these preferences can be activated by listing the corresponding key word (or the significant part thereof as indicated above in parenthesis) as part of the option's value.

```
pdcat -vp Fit,Center in.pdf out.pdf
```

**-wc Warn about collections**

Issue a warning if one of the input files contains collections. pdcats will terminate with exit code 2 if an input file containing a PDF collection is encountered.

**-uc Unembed collections**

If the input file is a PDF collection, append the document(s) from the collection.

### 4.3 Common Transformations

---

The following information applies to all types of transformation, in particular to the logo functions `-logo` and `-bglogo`.

In order to scale, rotate, skew and translate the logo, one must apply a transformation matrix. A transformation matrix describes a linear transformation from one coordinate system to another. For our purpose the transformation matrix contains 6 elements and has the form  $[a\ b\ c\ d\ e\ f]$ . The elements "a b c d" are for scaling, rotating and skewing, the elements "e f" are vertical and horizontal translation.

The identity matrix is  $[1\ 0\ 0\ 1\ 0\ 0]$  which means, no scaling, rotating, skewing nor translating is applied.

**Scale**      Scaling is achieved by multiplying with the matrix  $[s_x\ 0\ 0\ s_y\ 0\ 0]$ .

For uniform scaling up or down,  $s_x$  and  $s_y$  must be equal.

Or put in simpler words: The first 4 numbers are multiplied, all with the same value. A value larger than 1 means scaling up, a value smaller than 1, but larger than 0 means scaling down. (A value of 0 means the size becomes 0, a value below 0 means it becomes inverted.)

*Examples:*

Increase the size to 200%:       $[2\ 0\ 0\ 2\ 0\ 0]$

Decrease the size to 50%:       $[0.5\ 0\ 0\ 0.5\ 0\ 0]$

**Rotate**      Rotating is achieved by multiplying with  $[\cos(\theta)\ \sin(\theta)\ -\sin(\theta)\ \cos(\theta)\ 0\ 0]$ . Where  $\theta$  is the angle of the counter-clockwise rotation.

*Examples:*

Rotate by 30 degrees:       $[0.866\ 0.5\ -0.5\ 0.866\ 0\ 0]$

Rotate by 45 degrees:       $[0.707\ 0.707\ -0.707\ 0.707\ 0\ 0]$

Rotate by 90 degrees:       $[0\ 1\ -1\ 0\ 0\ 0]$

Rotate by 180 degrees:       $[-1\ 0\ 0\ -1\ 0\ 0]$

Rotate by 270 degrees:       $[0\ -1\ 1\ 0\ 0\ 0]$

**Skew**      Specifying an matrix of the form  $[1\ \tan(\alpha)\ \tan(\beta)\ 1\ 0\ 0]$  results in skewing the x-axis by angle  $\alpha$  and y-axis by angle  $\beta$ . Due to its rare use, this is not described further here.

**Translate**      Translation is specified by the form  $[1\ 0\ 0\ 1\ t_x\ t_y]$  where  $t_x$  represents the translation in points along the x-axis, and  $t_y$  represents the translation in points along the y-axis.

1 PDF point = 1/72 inch

1 inch = 25.4 mm

November 27, 2018

**Advanced Tip:** The following information is slightly mathematical and assumes the reader is familiar with matrix multiplication. It is not required to understand how to apply transformation, it just adds some mathematical background.

A transformation matrix to transform from one coordinate system to another is represented by a 3 by 3 matrix as shown below:

$$TM = \begin{vmatrix} a & b & 0 \\ c & d & 0 \\ e & f & 1 \end{vmatrix}$$

The 3<sup>rd</sup> column does not contain any relevant elements, which is why it is left away in the simplified form [a b c d e f]. However it is required to multiply matrices and receive a result which is a 3 by 3 matrix again.

In order to combine multiple operations, e.g. rotating and scaling, the corresponding matrices are multiplied.

*Example:*

Matrix R to rotate by 90 degree, matrix S to scale by factor 2.

$$R = \begin{vmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{vmatrix} \quad S = \begin{vmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

The transformation matrix that does both, rotate and scale is:

TM = R x S, which in the simplified version is [0 2 -2 0 0 0].

To gain a deeper knowledge about coordinate transformation please refer to the PDF Reference chapter 4.2.

## Page Sizes

When dealing with transformation matrices, specially with translation, it is a requirement to know how large the pages are. The coordinate system of a PDF page is in points. 1 point corresponds to 1/72 inch, 1 inch is equal to 25.4mm. Below is a table with the most frequently used paper sizes, and their size in PDF points.

Name	Size	PDF Points
A3	297 x 420 mm	842 x 1190
A4	210 x 297 mm	595 x 842
Letter	8.5 x 11 inch	612 x 792
Legal	8.5 x 14 inch	612 x 1008

## Examples for Transformations

The following commands show how to use the `-logo` and `-bglogo` switches, and how the corresponding results look. Let's assume you have a PDF document with a page of size A4 and a PDF logo document with the same dimensions.

(A4: width  $w = 595$  points, height  $h = 842$  points).

These two documents are displayed below image (1) and (2):

(1)



filename = page.pdf

Width = 595, Height = 842

viewing rotation = 0

(2)



filename = logo.pdf

Width = 595, Height = 842

viewing rotation = 0

## No Transformation

Stamping the logo without transformation on top is done using `-logo` without a transformation matrix, stamping in the background using `-bglogo`. The commands to achieve this are:

```
pdcat -logo logo.pdf page.pdf foreground.pdf
```

```
pdcat -bglogo logo.pdf page.pdf background.pdf
```

If no transformation matrix is provided, the default transformation matrix  $[1\ 0\ 0\ 1\ 0\ 0]$  is applied. The above commands are therefore equal to the following commands:

```
pdcat -logo +"1 0 0 1 0 0" logo.pdf page.pdf foreground.pdf
```

```
pdcat -bglogo +"1 0 0 1 0 0" logo.pdf page.pdf background.pdf
```

(3)



foreground.pdf

(4)



background.pdf



November 27, 2018

### Scale

The origin of any transformation is the lower left corner. This also applies to scaling. In order to scale down the logo to 50% use the matrix [0.5 0 0 0.5 0 0].

```
pdcat -logo +"0.5 0 0 0.5 0 0" logo.pdf page.pdf output5.pdf
```

With this command the lower left corner of the page and the logo will be at the same location, however due to the scaling the upper left corner of the logo is in the middle of the page, see image (5).

### Scale and Translate

To translate the logo, use the 5<sup>th</sup> value of the matrix to move it horizontally (positive value = right), and the 6<sup>th</sup> value to move it vertically (positive value = upwards).

The command below scales the image by 0.5 and moves it by 297.5 points to the right. (297.5 points = w/2)

```
pdcat -logo +"0.5 0 0 0.5 297.5 0" logo.pdf page.pdf output6.pdf
```

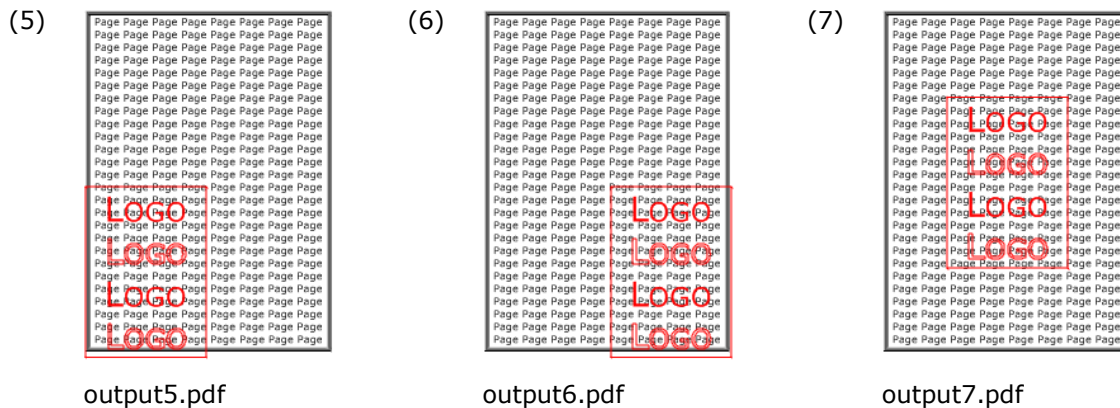
The result is shown in (image 6).

In order to scale down a logo by 50% and center it, it needs to be translated by a quarter of its original width to the right and by a quarter of its original height upwards.

The command for that is:

```
pdcat -logo +"0.5 0 0 0.5 148.5 210.5" logo.pdf page.pdf output7.pdf
```

If the pages of the document and the log are different, or the scaling is different, the values need to be adjusted accordingly.



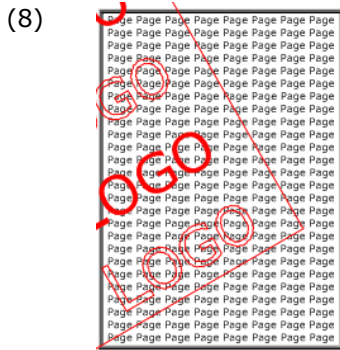
### Rotate

The origin of the rotation is the lower left corner. The commands below rotate by 30 and by -30 degrees.

```
pdcat -logo +"0.866 0.5 -0.5 0.866 0 0" logo.pdf page.pdf output8.pdf
```

```
pdcat -logo +"0.866 -0.5 0.5 0.866 0 0" logo.pdf page.pdf output9.pdf
```

November 27, 2018



output8.pdf



output9.pdf

As one can see, part of the logo is outside the page. In fact if the rotation angle is larger than 90 or smaller then -90, the entire logo is outside the page.

A common issue is "logos disappearing". This often happens due to the initial viewing rotation of the page, which may differ from 0. See also chapter "Rotated Pages".

### Rotate, Scale and Translate

Rotating a document as in the sample above image (8) and additionally scaling with 0.5 is achieved by multiplying the two matrices, result see image (10).

```
pdcat -logo +"0.433 -0.25 0.25 0.433 0 0" logo.pdf page.pdf output10.pdf
```

In order to translate the rotated logo upwards and to the right, so it is placed in the middle of the page, one needs to apply some trigonometry.

Page width w = 595, height h = 842

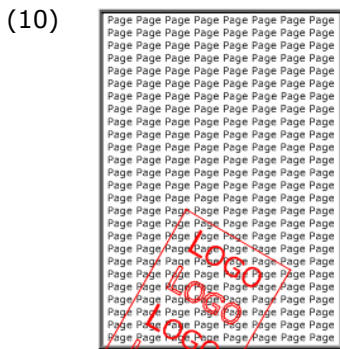
The center of the logo needs to be moved x to the right and y upwards, where after some calculation one ends up with:

$$x = (4-\sqrt{3})/8 * w - 1/8 * h = 63.4$$

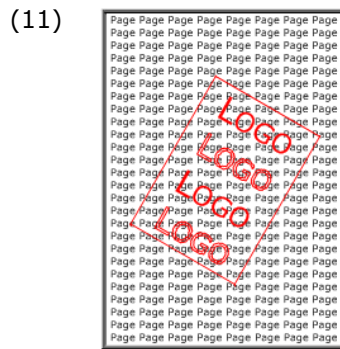
$$y = (4-\sqrt{3})/8 * h + 1/8 * w = 313.1$$

```
pdcat -logo +"0.433 -0.25 0.25 0.433 63.4 313.1" logo.pdf page.pdf output11.pdf
```

Again, this assumes, logo and page have the identical dimensions. The result can be seen in image (11). If this is not the case, or the rotation is different from 30 degrees, other equations apply.



output10.pdf



output11.pdf

November 27, 2018

### Portrait vs. Landscape vs. Rotated Portrait

Every page in a PDF document can have a so called viewing rotation. The viewing rotation can take the following values: 0, 90, 180, 270. It defines how the page is to be displayed when viewed in a PDF viewing application.

In this chapter we treat three types of pages. These are:

- Portrait: The width of the page is smaller than its height. The viewing rotation is 0.
- Landscape: The width of the page is larger than its height. The viewing rotation is 0.
- Rotated Portrait: The width of the page is smaller than its height. The viewing rotation is 90. As a result the page appears as landscape.

(The most common example of a rotated portrait is using the PDFMaker 7.0 plug-in for MS Word and convert a landscape MS Word document to a PDF document.)

There exist many more types, such as rotated landscape, or other viewing rotations (e.g. 270). Those cases occur infrequently and are not discussed here.

The rotation and page sizes of documents can be retrieved using the command line tool **pdpg**.

When one deals with rotated pages, one needs to be aware how rotated pages are handled. The steps always are as follows:

- The viewing rotation of the PDF page and the logo are set to 0.
- The logo is added to the page.
- The viewing rotation is set back to the initial rotation of the page.

There are nine different combinations of a logo added onto a page. Three of them are trivial, i.e. when the page and the logo are of the same type. The other six are explained in the following samples.

		Page		
		Portrait	Landscape	Rotated Portrait
Logo	Portrait	trivial	<b>Example 3</b>	<b>Example 5</b>
	Landscape	<b>Example 1</b>	trivial	<b>Example 6</b>
	Rotated Portrait	<b>Example 2</b>	<b>Example 4</b>	trivial

The six samples handle the non-trivial combinations of the three different pages and three different logos. These six input documents are displayed in images (1), (2), (12), (13), (14) and (15).

November 27, 2018



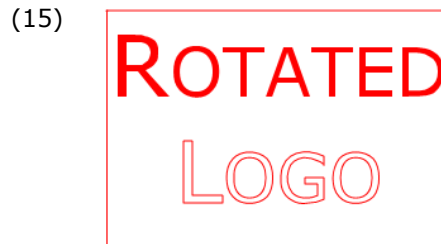
filename = page2.pdf  
 Width = 842, Height = 595  
 viewing rotation = 0



filename = logo2.pdf  
 Width = 842, Height = 595  
 viewing rotation = 0



filename = page3.pdf  
 Width = 595, Height = 842  
 viewing rotation = 90



filename = logo3.pdf  
 Width = 595, Height = 842  
 viewing rotation = 90

**Example 1: (Page = Portrait, Logo = Landscape)**

If the logo is applied without a transformation matrix, the result looks as displayed in image (16).

```
pdcat -logo logo2.pdf page.pdf output16.pdf
```

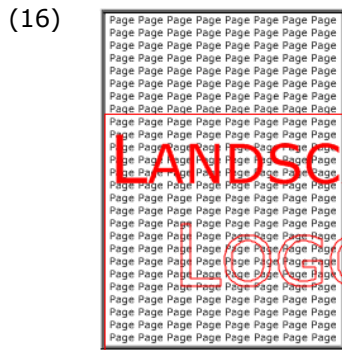
If the logo should fit on the page, there are the following options:

- Shrink the logo and translate it to the middle of the page
- Rotate the logo clockwise and move it upwards
- Rotate the logo counter-clockwise and move it to the right

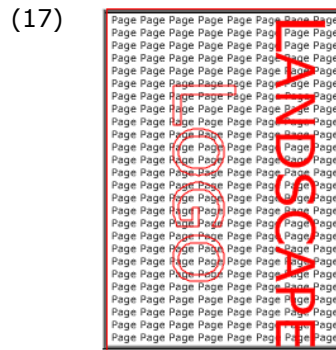
The second option is explained in this paragraph. If the landscape-logo is rotated by 90 degrees clockwise around the origin in the lower left corner, it needs to be translated upwards by the height of the page, otherwise the logo ends up outside (below) the page. The corresponding command for the rotation  $[0 \ -1 \ 1 \ 0 \ 0 \ 0]$  and translation  $[1 \ 0 \ 0 \ 1 \ 0 \ 842]$  is given below. The result is displayed in image (17).

```
pdcat -logo +"0 -1 1 0 0 842" logo2.pdf page.pdf output17.pdf
```

November 27, 2018



output 16.pdf

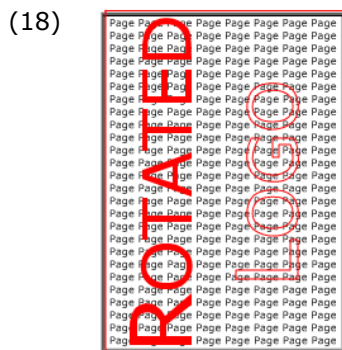


output 17.pdf

**Example 2: (Page = Portrait, Logo = Rotated Portrait)**

The viewing rotation of the logo is resolved. The result is displayed in image (18).

```
pdcat -logo logo3.pdf page.pdf output18.pdf
```



output 18.pdf

**Example 3: (Page = Landscape, Logo = Portrait)**

If the logo is added without transformation matrix, the result looks as displayed in image (19).

```
pdcat -logo logo.pdf page2.pdf output19.pdf
```

If the logo should fit on the page, there are the following options:

- Shrink the logo and translate it to the middle of the page
- Rotate the logo clockwise and move it upwards
- Rotate the logo counter-clockwise and move it to the right

The third option is explained in this paragraph. If the portrait-logo is rotated by 90 degrees counter-clockwise around the origin in the lower left corner, it needs to be translated to the right by the width of the page, otherwise the logo ends up outside (to the left) the page. The corresponding command for the rotation  $[0 \ 1 \ -1 \ 0 \ 0 \ 0]$  and translation  $[1 \ 0 \ 0 \ 1 \ 842 \ 0]$  is given below. The result is displayed in image (20).

November 27, 2018

```
pdcat -logo +"0 1 -1 0 842 0" logo.pdf page2.pdf output20.pdf
```



output 19.pdf



output 20.pdf

**Example 4: (Page = Landscape, Logo = Rotated Portrait)**

If the logo is added without transformation matrix, the result looks as displayed in image (21).

```
pdcat -logo logo3.pdf page2.pdf output21.pdf
```

If the logo should fit on the page, it makes most sense to rotate the logo clockwise and move it upwards.

The corresponding command for the rotation [0 -1 1 0 0 0] and translation [1 0 0 1 0 595] is given below. The result is displayed in image (22).

```
pdcat -logo +"0 -1 1 0 0 595" logo3.pdf page2.pdf output22.pdf
```



output 21.pdf



output 22.pdf

**Example 5: (Page = Rotated Portrait, Logo = Portrait)**

The logo inherits the viewing rotation of the page as shown in image (23).

```
pdcat -logo logo.pdf page3.pdf output23.pdf
```



output 23.pdf

November 27, 2018

**Example 6: (Page = Rotated Portrait, Logo = Landscape)**

If the logo is added without transformation matrix, the result looks as displayed in image (24).

```
pdcat -logo logo.pdf page3.pdf output24.pdf
```

First off, one needs to be aware where the origin is. Due to the rotation of the page, the origin in the image appears in the upper left. The logo must be rotated counter-clockwise and moved to the right. (Not downwards, remember the page is rotated!)

The corresponding command for the rotation  $[0 \ 1 \ -1 \ 0 \ 0 \ 0]$  and translation  $[1 \ 0 \ 0 \ 1 \ 595 \ 0]$  is given below. The result is displayed in image (25).

```
pdcat -logo +"0 1 -1 0 595 0" logo.pdf page3.pdf output25.pdf
```



output 24.pdf



output 25.pdf

## 5 pdsplit

### 5.1 Description

---

The pdsplit program splits large PDF files into smaller pieces – either one page files, or according to the bookmarks stored in the file. This is controlled by the option `-b` (meaning do split according to bookmarks).

The program can also selectively extract a particular chapter, if the corresponding bookmark title is specified (e.g. `-b:'Chapter one'=one.pdf`).

For other options supported by pdsplit, see the usage message which can be retrieved when calling the program without any parameters.

### 5.2 Functionality and Options

---

#### Split Files and name Output Files

By default the created files have the following file name: `<xx><page number>.pdf`

`<xx>` is a prefix that can be modified and `<page number>` is at least two digits long.

The following command splits the file `big.pdf` into individual pages and names the resulting files `xx01.pdf`, `xx02.pdf`, etc.

```
pdsplit big.pdf
```

When you specify a second argument, the links and file names will be generated according to given the format, in this case `small001.pdf`, `small002.pdf`, etc.

```
pdsplit big.pdf small%3.3d.pdf
```

Note that you must specify a "printf" conforming format. The argument supplied to the format is always an integer number (the page number). Therefore, you can use `'d'` (for decimal), `'x'` (for hexadecimal), `'u'` (unsigned decimal), or `'o'` (octal). If you specify `'s'` (char string), pdsplit will crash, because the page number will not be a valid string address.

By using pdsplit as shown in the command below, you can change the links between the resulting PDF files to web links that invoke a servlet method on the web server, which can do some on the fly processing while serving the desired PDF page.

```
pdsplit big.pdf http://myweb/servlet/big?page=%d big-%d.pdf
```

It is also possible to change links to relative web links:

```
pdsplit big.pdf http:big?page=%d big-%d.pdf
```

or

```
pdsplit big.pdf http:/servlet/big?page=%d big-%d.pdf
```



November 27, 2018

---

**-b Extract all or individual Chapters According to Bookmarks**

Extract the chapter "Chapter One" - if it exists in the file *in.pdf* - according to bookmarks.

```
pdsplit -v -b:"Chapter One" in.pdf
```

```
Creating xx-Chapter One.pdf
```

In order to extract by bookmarks and not create a prefix, define an output file and name it *%s.pdf*.

```
pdsplit -v -b:"Chapter One" in.pdf %s.pdf
```

```
Creating Chapter One.pdf
```

If the file *big.pdf* contains a bookmark tree pointing to the individual chapters of the file, *pdsplit* will extract all sections into separate files. The name of the files being created will carry the prefix "chapter-" followed by the bookmark text, and have the extension ".pdf" in the example below.

```
pdsplit -b big.pdf chapter-%s.pdf
```

**-l Restrict the Processing to a Specific Bookmark Level**

The option **-l** can only be used in combination with the option **-b**.

Assume you have a bookmark tree that looks as this:

Chapter A

- Page 1

- Page 2

Chapter B

- Page 3

- Page 4

The following command splits the input file according to the first level bookmarks:

```
pdsplit -v -b -l 1 in.pdf
```

```
Creating xx-Chapter A.pdf
```

```
Creating xx-Chapter B.pdf
```

**-m Exclude low-level Bookmarks**

Use the option **-m n** to exclude bookmarks which have a lower level than *n*.

The option **-m** can only be used in combination with the option **-b**. The following command splits the input file according to the first and second level bookmarks:

```
pdsplit -v -b -m 2 in.pdf
```

```
Creating xx-Chapter A.pdf
```

```
Creating xx-Page 2.pdf
```

```
Creating xx-Chapter B.pdf
```

```
Creating xx-Page 4.pdf
```

The following command splits the input file according to the second level bookmarks:

November 27, 2018

---

```
pdsplit -b -l 2 -m 2 in.pdf
```

```
Creating xx-Page 1.pdf
```

```
Creating xx-Page 2.pdf
```

```
Creating xx-Page 3.pdf
```

```
Creating xx-Page 4.pdf
```

## **-p Split a Document into Parts with Specified Number of Pages**

Split the input document and create output documents, which are two pages long each.

```
pdsplit -p 2 in.pdf
```

## **-x Exchange Bookmark Characters in File Names**

This option allows to replace characters in bookmark. This is useful for bookmarks which contain characters that are not allowed in file names, such as "/" or ":".

For example to replace all "\" by a "\_" and all "/" by a "-" use a command like this:

```
pdsplit -x \/=_- in.pdf
```

## 6 pdsel

### 6.1 Description

---

Split a PDF document into several documents or extract pages from a document.

### 6.2 Functionality and Options

---

#### Select Individual Pages or Ranges of Pages

Select page 1.

```
pdsel 1 in.pdf out.pdf
```

Select pages 1 and 3-5.

```
pdsel 1,3-5 in.pdf out.pdf
```

Selecting a specific page more than once which repeats the page without using more space

```
pdsel 1-3,2 in.pdf out.pdf
```

#### -a Remove Annotations

Remove all annotations on pages 1 to 3.

```
pdsel -a 1-3 in.pdf out.pdf
```

## 7 pdw

### 7.1 Description

---

pdw is a tool to list all text tokens in a PDF document. The output is x/y/font-size/text-width/rotation.

Note there is a newer and more enhanced text extraction tool available by PDF Tools AG, the 3-Heights™ PDF Extract Shell:

<http://www.pdf-tools.com/asp/products.asp?name=EXPS>

### 7.2 Functionality and Options

---

#### **-c Break down Text Blocks to Individual Character**

List all characters separately.

```
pdw -c in.pdf
```

#### **-cr Add Carriage Return before New Lines**

Depending on the platform, this option should be used in combination with the option **-u**.

```
pdw -u -cr in.pdf
```

#### **-o List the Annotations in a Separate File**

```
pdw -o out.txt in.pdf
```

#### **-r Take Account of the Page Rotation**

The option **-r** takes account of page rotation. This is useful for documents with rotated page and rotated text on the rotated pages, which visually appears to be not rotated. However, this option can always be turned on.

```
pdw -r in.pdf
```

#### **-u List the Text in Unicode Encoding**

```
pdw -u in.pdf
```

#### **-w Break down Text Blocks to Blank Separated Words**

This option starts a new text token when one of the following criteria is fulfilled:

November 27, 2018

---

- There is one or more blank characters
- The distance between two characters is 20% or more of the font size.  
(only applies if there is kerning in the TJ operator)

If the option `-w` is not set, a new text token starts when two characters are at least 50% of the font size apart.

```
pdw -w in.pdf
```

## 8 pdform

### 8.1 Description

---

The pdform tool displays information about text form fields, or fills in data into form fields.

Note, there is a professional form filling tool available by PDF Tools AG, which also supports flattening (merging the form fields with the page content, i.e. burn the fields on the page, so they are no longer editable).

### 8.2 Functionality and Options

---

#### Fill in data

Assuming "in.pdf" contains a form field called "field01", its content can be set like this:

```
pdform in.pdf out.pdf field01="Hello World!"
```

#### Add a new form field

Add a new form at page 1, coordinates (50/50), width = 200, height = 25, font = 4, font size = 10 and write "Hello World!" into the form.

```
pdform in.pdf out.pdf +field02@1,50,50,250,75,4,10="Hello World!"
```

#### Delete a field

Assuming "in.pdf" contains a form field called "field01"

```
pdform in.pdf out.pdf -field01
```

#### -l List all Form Fields

pdform lists the name, the box coordinates, and current data for each field in the file. If there are multiple instances of the same field, the box of an arbitrary instance is shown.

```
pdform -l formtemplate.pdf
```

```
F1 [177,399/374,51] : Data of first field
```

```
F2 [104,399/150,51] : Data of second field
```

## 9 pdwebl

### 9.1 Description

---

The first parameter specifies the input file, the second one the output file. Subsequent parameters define the key/link pair: pdwebl will scan the text on all pages for a text fragment containing the key text, and will put the corresponding link over the text.

If a key contains spaces, you have to put it into quotes (so the shell treats the whole key/link expression as one argument). pdwebl currently cannot match keys that wrap over the end of a line.

The default link type for pdwebl is a web link, but it supports also other link types. These are specified with a prefix in the link specification. For a link to another PDF file in the file system (GoToR link), substitute the prefix `http://` by `file:.` To launch any other file or program in the file system, use `"launch:"`.

pdwebl joins text fragments before performing match; text can have varying font size - the size of the last fragment determines the height of the link box.

### 9.2 Functionality and Options

---

#### Add URL Links to a PDF Document

Add the web link `http://www.pdf-tools.com` to all keywords "PDF-Tools" in the document *in.pdf*.

```
pdwebl in.pdf out.pdf PDF-Tools=http://www.pdf-tools.com
```

#### Add Page links to a PDF Document

To all keywords "file2" in *in1.pdf*, add a link to the document *in2.pdf*

```
pdwebl in1.pdf out.pdf file2=in2.pdf
```

To all keywords "command" in *file1*, add a link, which launches the command shell

```
pdwebl in1.pdf ouin.pdf command=launch:C:\winnt\system32\command.com
```

#### Add Java Scripts

Add Java Scripts to a PDF document stored on a text file and add a link to it:

```
pdwebl infile.odf out.pdf action=js.file.js
```

#### -i Read Input File from Standard Input

See switch `-l`:

November 27, 2018

---

**-l: Define Key/Link Pairs in a File or Standard Input**

Instead of providing the key/link pairs as parameter on the command, they can be passed as file parameter using switch **-l:**.

**Example using a File**

Create a text file with one key/link pair per line (case sensitive), e.g. like this:

```
PDF-Tools=http://www.pdf-tools.com
google=http://www.google.com
```

Save that text file and name it for example *"pairs.txt"*. Then execute the command below:

```
pdweb1 in1.pdf out.pdf -l:pairs.txt
```

This has the same effect as the following command with text file:

```
pdweb1 in1.pdf out.pdf PDF-Tools=http://www.pdf-tools.com google=
http://www.google.com
```

Note: use a tab in the control file instead of the equal sign to delimit a match text that contains an equal sign. Example:

```
E=mc2      http://en.wikipedia.org/wiki/E%3Dmc2
```

**Example using Standard Input**

If no file name parameter is provided with the switch **-l:**, the input is read from the standard input. One line corresponds to one key/link pair. Completing the input is achieved by pressing Ctrl+Z twice.

```
pdweb1 in1.pdf out.pdf -l:
PDF-Tools=http://www.pdf-tools.com
google=http://www.google.com
^Z
^Z
```

Alternatively the key/link pairs can be piped. This is achieved using the following command:

```
(echo PDF-Tools=http://www.pdf-tools.com && echo
google=http://www.google.com) | pdweb1 in1.pdf out.pdf -l:
```

**-q Quite Mode**

Omit the log to standard out.

```
pdweb1 -q in1.pdf out.pdf PDF-Tools=http://www.pdf-tools.com
```

**-s Set the border style of links**

0: black solid

1: red dashed



November 27, 2018

---

2: solid red

3: green dashed

4: solid green

5: blue dashed

6: solid blue

Set the border to solid blue

```
pdweb1 -s6 in.pdf out.pdf PDF-Tools=http://www.pdf-tools.com
```

## 10 pdtoc

### 10.1 Description

---

pdtdoc create a table of content for PDF files.

### 10.2 Functionality and Options

---

#### **-b Create or Omit Bookmarks**

To not generate any bookmarks use the option **-b**:

```
pdtdoc -b in1.pdf in2.pdf out.pdf
```

#### **-I Set New Titles**

Set or replace titles.

```
pdtdoc -I "my first PDF" in1.pdf -I "my second PDF" in2.pdf out.pdf
```

#### **-d Place the Current Date on the Pages**

```
pdtdoc -d in1.pdf in2.pdf out.pdf
```

#### **-w Set the Page Width**

Set the page width to 300 points.

```
pdtdoc -w300 in1.pdf in2.pdf out.pdf
```

#### **-t Place a Header Text on the Pages**

```
pdtdoc -t "Table of content" in1.pdf in2.pdf out.pdf
```

#### **-c Set the Document Title**

Set the document title attribute to "My Title".

```
pdtdoc -c "My Title" in1.pdf in2.pdf out.pdf
```

#### **-dest Create Named Destination Links**

If *in.pdf* has a named destination "mydest", a link to it can be created like this:

```
pdtdoc -dest mydest in.pdf out.pdf
```

November 27, 2018

---

To replace the title to the named destination link (default is document title), use the option **-I**

```
pdtoc -dest mydest -I "My Link" in.pdf out.pdf
```

Note that Acrobat 6.0 can only link directly to the correct page number when the linked file is already opened. Acrobat 5.0 links correctly in any case.

## **-url Create URL Links**

```
pdtoc -url http://www.pdf-tools.com "PDF-Tools Homepage" out.pdf
```

## **-@ Read Input from a Control File**

The option `-@filename` allows read data from a control text file.

Lets assume you have a text file named `link.txt` with the following content:

```
-I "Link to File 1"  
in1.pdf  
-I "Link to File 2"  
in2.pdf
```

The following command will then create a table of content PDF with the two files specified in the control file.

```
pdtoc -r -@link.txt out.pdf
```

## 11 pdbm

### 11.1 Description

---

List bookmarks in a PDF document. Add bookmarks from a text file to a PDF document.

### 11.2 Functionality and Options

---

#### -d List Named Destinations

```
pdbm -d in.pdf
```

#### -D List Named Destinations Tab-separated

This is the default option. Providing no option has the same effect as setting the option -D.

```
pdbm in.pdf
```

Is equal to:

```
pdbm -D in.pdf
```

#### -o Redirect the Output to a File

This has the same effect as the pipe command >.

```
pdbm -o bookmarks.txt in.pdf
```

#### -oa Page Mode, Initial Page Number, Open Action

This switch sets the page mode, the initially displayed page and the open action for the initial view. It must be set before the switch -a.

-oa takes three parameters which have the following syntax:

```
-oa [OpenMode],[PageNumber],[OpenAction]
```

The individual parameters are optional and comma-separated

OpenMode: o=Outlines, t=Thumbs, f=Fullscreen, n=None

PageNumber: The initially displayed page number: 1, 2, 3...

OpenAction: Window=fit window, Width=fit width, Visible=fit visible

Example that sets the initial view to bookmarks, page 1 and fit window:

```
pdbm -oa o,1,Window -a bookmarks.txt in.pdf out.pdf
```

Example that sets the initial view to full screen, page 2:

November 27, 2018

---

```
pdbm -oa f,2, -a bookmarks.txt in.pdf out.pdf
```

## **-a Add Bookmarks from an Input File**

The input file has the same syntax as the output created with the option **-D**. (Remove the first line.)

```
[Expand State][Level]{Tab}[Bookmark Title]{Tab}[Link]{CRLF}
```

[Expand State] is the expand state of the bookmark node. There are three possible options:

- The '+' indicates that node should be expanded (showing child nodes).
- The '-' indicates that the node should be collapsed (sub node should not be shown).
- '' means the node has no children.

[Level] is the indent of branch level of the bookmark. The value is numeric and zero based.

{Tab} is a tab character (hex 09).

[Bookmark Title] is the title of the bookmark.

[Link] The link type of the bookmark.

GoTo [Page Number]: A reference to a page number within the document.

GoToR [Page Number]: A reference to a page number which is in another document.

{CRLF} is a carriage return line feed control set (hex 0d0a).

The destination such as /Fit, /FitH, /XYZ, etc, can only be extracted, it cannot be set with pdbm -a.

Create a text file with the following content. There must be a Tab before and after the bookmark title.

+ 0	Part 1	GoTo 0 /FitH 844
1	First Page	GoTo 0 /FitH 844
+ 1	Second Page	GoTo 1 /FitH 839
+ 2	Bookmark1	GoTo 1 /FitH 700
3	Bookmark2	GoTo 1 /FitH 505
3	Bookmark3	GoTo 1 /FitH 341
+ 0	Part 2	GoTo 2 /FitH 843
1	Bookmark4	GoTo 2 /FitH 676

```
pdbm -a bookmarks.txt in.pdf out.pdf
```

November 27, 2018

---

**-n Do not Print Destinations**

This option only lists the titles, and not the destinations.

```
pdbm -n in.pdf
```

**-n1 Add a Leading Hyphen**

This option is equal to the option `-n`, but adds a leading hyphen for compatibility reasons with older versions.

```
pdbm -n1 in.pdf
```

## 12 pdpg

### 12.1 Description

---

pdpg can be used to list:

- The total number of pages in a PDF document
- The size of the media and crop-box
- The viewing rotation of the pages
- The fonts per page or per document

The output of pdpg has the following format:

pagenumber, type, values1, value2, value3...

pagenumber: The page number in the document. At document level this value is 0.

type: The type can be *page* or *font*.

valueX: A row of type *font* has always one value: the name of the font. A row of type *page* can have several values: e.g. Media Box, Crop Box, Rotation.

### 12.2 Functionality and Options

---

#### -c List the CropBox

This option lists the crop-boxes of the pages comma separated. If there are no crop-boxes set, it returns the media-boxes.

```
pdpg -m in.pdf
```

```
1, page, 0.00, 0.00, 595.00, 842.00
```

```
2, page, 0.00, 0.00, 595.00, 842.00
```

```
3, page, 0.00, 0.00, 595.00, 842.00
```

```
4, page, 0.00, 0.00, 595.00, 842.00
```

#### -f List All Fonts on Pages

List all fonts on all pages:

```
pdpg -f in.pdf
```

```
1, font, "TimesNewRomanPSMT"
```

```
2, font, "Verdana"
```

```
2, font, "Verdana-Bold"
```

```
2, font, "Arial-BoldMT"
```

November 27, 2018

---

```
3, font, "Arial-BoldMT"
```

### **-pAll Get the Total Number of Pages in a PDF Document**

The option `-s` generates a shortened output message.

```
pdpg -pAll in.pdf
```

```
0, page, 4
```

```
pdpg -pAll -s in.pdf
```

```
4
```

### **-fAll List all Fonts in a Document**

The option `-s` generates a shortened output message.

```
pdpg -fAll in.pdf
```

```
0, font, "Verdana"
```

```
0, font, "ArialMT"
```

```
0, font, "Verdana-Bold"
```

```
0, font, "TimesNewRomanPSMT"
```

```
pdpg -fAll -s in.pdf
```

```
"Verdana"
```

```
"ArialMT"
```

```
"Verdana-Bold"
```

```
"TimesNewRomanPSMT"
```

### **-m List the MediaBox**

This option lists the media-boxes of the pages comma separated.

```
pdpg -m in.pdf
```

```
1, page, 0.00, 0.00, 595.00, 842.00
```

```
2, page, 0.00, 0.00, 595.00, 842.00
```

```
3, page, 0.00, 0.00, 595.00, 842.00
```

```
4, page, 0.00, 0.00, 595.00, 842.00
```

### **-p Set the Page Range**

List all fonts on page 2-3:

```
pdpg -f -p 2 2 in.pdf
```



November 27, 2018

---

```
2, font, "Verdana"
```

```
2, font, "Verdana-Bold"
```

```
2, font, "Arial-BoldMT"
```

```
3, font, "Arial-BoldMT"
```

### **-r List the Page Rotation**

Option `-m` lists the media box, Option `-c` the crop box and option `-r` the rotation.

```
pdpg -m -r in.pdf
```

```
1, page, 0
```

```
2, page, 0
```

```
3, page, 0
```

```
4, page, 90
```

### **-s Abbreviate Output**

Do not output the result with additional information comma separated. See options `-pAll` and `-fAll` for samples.

### **-u Disable UserUnit Adjustment**

Disable the adjustment of box sizes by UserUnit.

## 13 pdxt

### 13.1 Description

---

The pdxt tool adds a logo taken from a first PDF file and puts it on all pages of a second input PDF file. The logo can be put behind or on top of the page content.

This functionality is also available in pdcat. pdxt can only be applied once per document. It is generally suggested to use pdcat instead.

### 13.2 Functionality and Options

---

#### Specify a PDF File Containing the Logo

```
pdxt logo.pdf in.pdf out.pdf
```

#### Select the logo page

Take page 3 of "login.pdf" as logo

```
pdxt -3 login.pdf in.pdf out.pdf
```

#### Put the Logo on top of the Page or in the Background

Put the logo in the foreground

```
pdxt login.pdf in.pdf -top out.pdf
```

## 14 txt2pdf

### 14.1 Description

---

As of version 4.2, the txt2pdf tool is replaced by a new implementation that supports not only ASCII input, but also UTF-8 and Unicode, as well as the use of arbitrary fonts.

The old version of txt2pdf is deprecated and will be shipped as "oldtxt2pdf" for a limited period of time.

### 14.2 Functionality and Options

---

The basic usage of txt2pdf is to call it with the input and output file names:

```
txt2pdf input.txt output.pdf
```

The input text file may contain certain control characters with formatting effects:

- Tab            advance the start of the next character to the next tab position (which is a multiple of 8 space characters per column).
- Form feed    start a new page
- Byte order marker: Unicode and UTF-8 text files should contain the appropriate byt order marker at the beginning of the file. If this byte order marker is missing, txt2pdf tries to guess the encoding, prior to attempt to process it as ASCII.

The txt2pdf tool supports a number of options as listed by the usage text when called without any parameters:

#### txt2pdf

```
txt2pdf.exe [options] in.txt out.pdf
```

options:

-ff font	Font name (default: CourierNew)
-fs size	Font size (default: 11)
-pb border	Page border in points (default: 20)
-ps width height	Page width and height in points (default: 595 842)
-tw wrap	Set line wrap mode (0: off, 1: on, default: 1)
-v	Verbose mode

Units:

1 point	1/72 inch
1 inch	25.4 mm

Error codes:

0	Successful completion
1	Couldn't open input file
2	Couldn't create output file
3	Option error
13	Unexpected error

**-ff Set the font**

Use this option to specify the name of the font to be used. The default is "CourierNew".

On Windows platforms, the selected font must be installed in the system's Fonts directory (usually C:\WINDOWS\Fonts).

On Mac OSX platforms, the font must be installed in /System/Library/Fonts or /Library/Fonts. On other UNIX platforms, fonts are searched in /usr/share/fonts, /usr/local/share/fonts, including subdirectories. Use the environment variable **PDFFONTDIR** to specify an additional location (if not set, /usr/lib/X11/fonts/Type1 is used).

Note that the spelling of fonts as shown in the Windows explorer may not always be identical to the way it must be passed to txt2pdf. While the Windows font explorer lists msmncho.ttf (msmncho.ttc) as "MS Mincho", it must be passed as "MS-Mincho" to txt2pdf.

**-fs Set the font size**

Use this option to set the font size. Default is 11 (points).

**-pb Set the page border size**

Use this option to set the size of the page border (default 20).

**-ps Set the page size**

Use this option with two values to set the width and height of the pages in points (default: 595 842 for A4)

**-tw Set the line wrap mode**

Use a value of 0 to disable automatic line wrapping of long lines (default: 1)

**-v Set verbose mode**

Print verbose log messages

**14.3 Error messages and codes**

---

The txt2pdf tool returns with an error code of 0 on successful completion.

Other error codes are:

1	Couldn't open input file	Verify that the input file exists and can be opened for read
---	--------------------------	--

November 27, 2018

2	Couldn't create output file	Verify that the specified folder can be written to. Also, make sure that a font with the specified name (or "CourierNew") exists in the font folder (see description for option -ff).
3	Option error	Verify the spelling and permitted values of the options on the command line
13	Unexpected error	An internal error of the tool was encountered. You may want to contact PDF Tools support, if you cannot resolve the problem.

## 15 pdcrop

### 15.1 Description

---

As the name says, pdcrop crops pages in PDF documents. It can set various boxes, such as the media box, the crop box, the bleed box, the art box and the trim box. New vales can be based on existing values of any existing box.

### 15.2 Functionality and Options

---

#### **-ob Base the Cropping on a Specified Box**

This option defines based on which box the cropping is done. Any type of box can be used, the most common and supported are:

MediaBox, CropBox, TrimBox, BleedBox, Artbox

#### **-nb Specify the Box Type to Be Set**

This option defines to which box the cropping is applied. Any type of box can be used, the most common and supported are:

MediaBox, CropBox, TrimBox, BleedBox, Artbox

#### **-shrinkShrink the Box**

The shrink value defines the value by which the box is cropped.

```
pdcrop -ob MediaBox -nb CropBox -shrink <left>:<bottom>:<right>:<top>
in.pdf out.pdf
```

The following example uses the MediaBox as base, and sets a CropBox which is 30 points smaller than the MediaBox in all four directions.

```
pdcrop -ob MediaBox -nb CropBox -shrink 30 in.pdf out.pdf
```

November 27, 2018

---

Alternatively the values can be provided as absolute values. The following command will shrink the CropBox by 10 points from the left and right border, and 20 points from the bottom and top border. This is assuming the input page is A4. (A4 = 595 x 842 points)

```
pdcrop -nb CropBox -shrink 10:20:10:20 in.pdf out.pdf
```

The shrink option can also be used to increase a box, this is simply done by providing negative numbers. The following command sets the MediaBox 20 points larger in each direction as A4.

```
pdcrop -nb MediaBox -shrink -20 in.pdf out.pdf
```

## 16 pdmerge

### 16.1 Description

---

pdmerge is a tool that can merge several PDF documents into one. It also provides the functionality to insert Java scripts for "Will Save" and "Will Print", to generate a table of contents and to add outlines (bookmarks) for each input document.

### 16.2 Functionality and Options

---

#### Simple Merge of PDF Files

The following command merges the documents *in1.pdf*, *in2.pdf* and *in3.pdf* to *out.pdf*.

```
pdmerge in1.pdf in2.pdf in3.pdf out.pdf
```

#### -c Create a Table of Contents

A table of contents is automatically created when a template file is defined using the option -c.

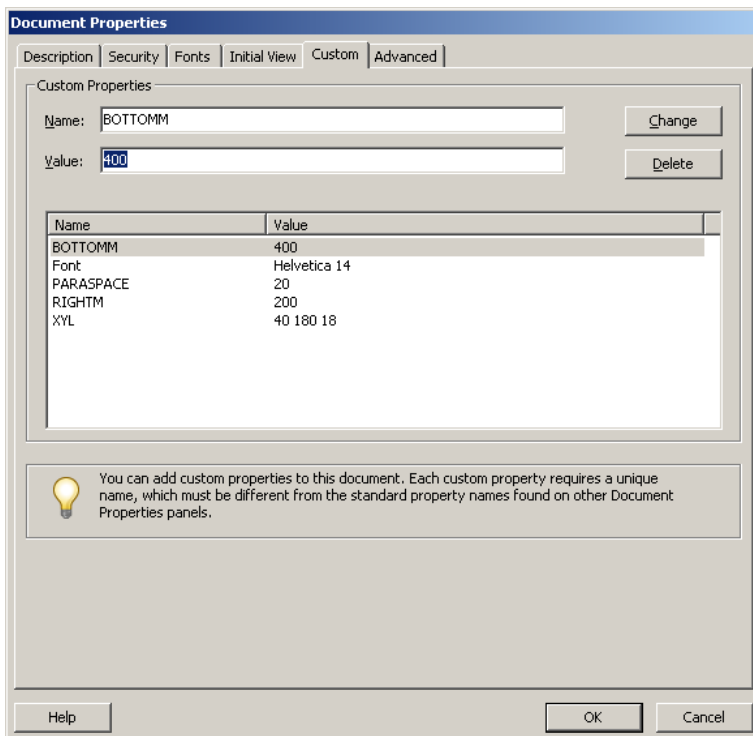
The template document has two functionalities:

- The template should contain one page. This page acts as template of the created table of contents.
- The template defines the layout of the table of contents. The defining values are set in the custom properties of the PDF document. These settings are optional.

BOTTOMM	Margin from bottom
Font	Two vales that define the font and font size in points
PARASPACE	The distance between two lines in points
RIGHTM	Margin from left border
XYL	Three values that define the position of the first entry (X position, Y position, line feed)

Custom properties can be created in different ways. A simple way is using the Document Properties window of Adobe Acrobat, see screenshot:

November 27, 2018



Example that creates a table of contents as first page, it contains the same entries as the outlines, "one" and "two":

```
pdmerge -c template.pdf -t one input1.pdf -t two input2.pdf out.pdf
```

The table of contents can be inserted at any position. Each document that is added a title using the switch `-t` is listed in the table of contents.

## **-t Merge PDF Files and Add Individual Outlines**

The following command merges the two input files *in1.pdf* and *in2.pdf*. It adds the outlines (= bookmarks) "Chapter 1" and "Chapter 2", which point to the beginning of each document.

```
pdmerge -t "Chapter 1" input1.pdf -t "Chapter 2" input2.pdf out.pdf
```

## **-WP Specify Javascript Code**

**-WS** These options specify a Javascript for "Will Save" (WS) and "Will Print" (WP) document actions. The Javascript is read from a file. The following command specifies the Javascript from the file *willsave.script* as the "Will Save" document action of the output file.

```
pdmerge -WS willsave.script input1.pdf input2.pdf out.pdf
```

Where "willsave.script" is a text file containing for example the following script:

```
app.alert('saving the document clears field data');
this.getField("NAME").value = "";
```



November 27, 2018

---

**@ Use a Control File**

A control file containing a list of commands can be inserted to the command using the prefix @. A command can contain multiple control files.

Example with 1 control file containing all commands:

```
pdmerge @control.txt
```

Where the text file *control.txt* could look like this:

```
-t "Chapter 1"  
input1.pdf  
-t "Chapter 2"  
input2.pdf  
out.pdf
```

Example with 2 control files, each containing a list of input files.

```
pdmerge -owner mypassword @list1.txt @list2.txt out.pdf
```

In order to merge all PDF documents in a directory, the files should first be listed and written to a file and can then be merged using a control file. Here is a sample for the CMD Shell of Windows that merges all files in the folder *C:\pdf*.

```
C:\> dir C:\pdf\*.pdf /B > list.txt
```

```
C:\> pdmerge @list.txt out.pdf
```

**-ax Set XMP Metadata**

An XMP Metadata file can be provided for embedding as document metadata. The file shall be encoded in UTF-8.

November 27, 2018

---

## pdinfo

---

pdinfo print Catalog and Info object.

The following sample prints the Catalog and Info object of the file `acrobat.pdf`.

```
pdinfo c:/acrobat3/reader/acrobat.pdf
```

```
PDF 1.7
<<
/Type /Catalog
/Pages 274 0 R
/OpenAction 279 0 R
/AcroForm 280 0 R
/Outlines 243 0 R
>>
<<
/Type /Pages
/Kids [ 273 0 R 275 0 R ]
/Count 11
>>
<<
/CreationDate (D:19960913110306)
/Producer (Acrobat Distiller 3.0 for Macintosh)
/Author (Adobe Systems Incorporated)
/Creator (PageMaker 6.0)
/Title (Why To Buy)
/ModDate (D:19961023133930)
/Subject (Why to Buy Adobe Acrobat 3.0)
/Keywords (Adobe Acrobat 3.0)
>>
```

With the option `'-o output.txt'` the output can be redirected to a specified file.

```
pdinfo -o output.txt c:/acrobat3/reader/acrobat.pdf
```

## 17 pobj

---

Prints the attributes of a PDF object. The objects can be printed by providing the object number as parameter.

For example use pobj to print the /OpenAction object of the file acrobat.pdf

```
pobj c:/acrobat3/reader/acrobat.pdf 279
```

```
<<
```

```
/S /GoTo
```

```
/D [ 281 0 R /Fit ]
```

```
>>
```

With the option '-o output.txt' the output can be redirected to a specified file.

pobj allows you to walk through all objects of a PDF document and see their content and properties. The use of this tool requires profound knowledge about PDF. See PDF Reference manual:

<http://www.pdf-tools.com/public/downloads/pdf-reference/pdfreference16.pdf>

## 18 pdls

---

pdls prints the pages tree and content streams of PDF documents.

For example:

```
pdls -s c:/acrobat3/reader/acrobat.pdf
Info object = 276
/CreationDate = (D:19960913110306)
/Producer = (Acrobat Distiller 3.0 for Macintosh)
/Author = (Adobe Systems Incorporated)
/Creator = (PageMaker 6.0)
/Title = (Why To Buy)
/ModDate = (D:19961023133930)
/Subject = (Why to Buy Adobe Acrobat 3.0)
/Keywords = (Adobe Acrobat 3.0)
page 1 is object 281
Font /F6 /NKAAAF+NuevaMM-It_540_200_
Font /F9 /NKBDDBI+NuevaMM_200_200_
Font /F11 /MyriadMM_400_600_
Font /F12 /MyriadMM_700_600_
% 0 0 612 792 RC
/GS1 gs
q
0 -530.05 402.5 0 103.4999 661.3 cm
/Im1 Do
Q
BT
/F6 1 Tf
... (rest omitted)
```

With the option '-o output.txt' the output can be redirected to a specified file.

## 19 COM Interface

### 19.1 Overview

---

There is a COM interface available which acts as a wrapper to the two command line tools *pdcat.exe* and *pdsplit.exe*.

The COM interface is named *pdapp.dll*, it is a Windows COM DLL. It can be used with any programming language that supports COM, such as Visual Basic or Delphi.

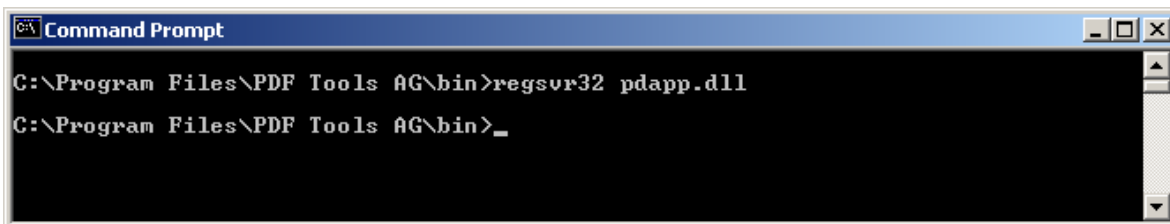
The COM interface does not support any kind of encryption opposed to the executables *pdcat.exe* and *pdsplit.exe*.

The COM interface used to be sold as a stand-alone product called PDCAT COM DLL. This product is no longer available, instead it is bundled to the PDF Command Line Suite.

### 19.2 Installation

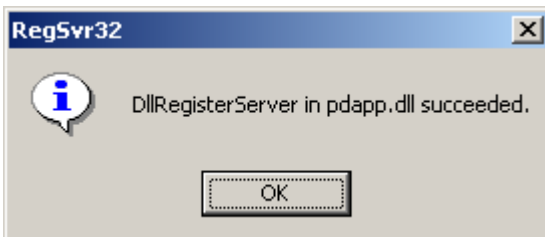
---

Before the COM interface can be used, it must be registered using the Windows tool *regsvr32.exe*. Copy the *pdapp.dll* into the installation directory (e.g. *C:\Program Files\PDF Tools AG\bin*). In the command prompt, type *regsvr32.exe* and specify the name of the DLL as a parameter. (If you are on Windows Vista, the command prompt must run under administrator).



```
Command Prompt
C:\Program Files\PDF Tools AG\bin>regsvr32 pdapp.dll
C:\Program Files\PDF Tools AG\bin>_
```

Upon this command you should receive a message box confirming the successful registering of the DLL.



## 19.3 Examples

---

The COM interface works very much like the command line interface, except that the parameter list must be constructed via the API. It is not possible to pass it as one single string. In Visual Basic 6, the COM interface is used as shown below.

### Declaration

```
Dim obj As New PDAPPLib.PDCat
```

Note that you must add the PDAPPLib type library to the references of the Visual Basic project to make this work. Alternatively, you can use the CreateObject method:

```
Dim obj As Object
```

```
Set obj = CreateObject("PDApp.PDCat")
```

### Parameter Passing

The AddParameter method is used to compose the arguments just as they would be specified on the command line for the PDCAT executable (but without the restriction of the maximum length imposed by the command shell).

```
obj.AddParameter "-r"
```

```
obj.AddParameter "input.pdf"
```

```
...
```

```
obj.AddParameter "output.pdf"
```

### Execution

The command is executed by calling the method Execute.

```
Dim res As PDErrorType, osErr As Long
```

```
res = obj.Execute()
```

```
If res = pdOSErr Then osErr = obj.GetErrorCode
```

The Execute method returns a result code as defined in the IDL file (and the *PDApp.h* file).

If an error is returned from the operating system, the corresponding error number can be retrieved using the GetErrorCode method. This typically happens when there is a problem with file input/output.

The class pdsplit works in the same way.

## Appendix A: Security

---

Encrypting a PDF file is useful in combination with permission flags that define what actions the user may or may not perform. A document must have an owner password in order to set permission flags. A user password is not related to permission flags, but only required to open the document. Here is a list of the permission flags:

"p": Do not print the document from Acrobat.

"c": Changing the document is denied in Acrobat.

"s": Selection and copying of text and graphics is denied.

"a": Adding or changing annotations or form fields is denied.

The following flags are defined for 128 bit encryption (PDF 1.4, Acrobat 5.0):

"i": Disable editing of form fields.

"e": Disable extraction of text and graphics.

"d": Disable document assembly.

"q": Disable high quality printing.

The flag "5" can be used in combination with one of the "old" flags to force 128 bit encryption without setting any of the i, e, d, or q flags. Note that using any of these Acrobat 5 related flags will produce a file that cannot be opened with older versions of Acrobat.

The flag "6" can be used to have the output AES-256 encrypted.

## Appendix B: Link Definition Files for pdcat

---

The structure of a link definition file is composed as follows (EBNF syntax):

```
LINK_FILE           ::= { LINK_SECTION | BOOKMARK_SECTION }
LINK_SECTION       ::= PAGE_HEADER NL { (BOX_SETTING | LINK) NL }
NL                 ::= new line in text file
PAGE_HEADER        ::= "Page" PAGE_NUMBER /* omit quotes in file */
PAGE_NUMBER        ::= INTEGER_NUMBER
INTEGER_NUMBER     ::= a number (decimal, e. g. 1, 2, 3, ... 99, ...)
LINK               ::= RECTANGLE DESTINATION
RECTANGLE          ::= X_COORD Y_COORD HEIGHT WIDTH
X_COORD            ::= NUMBER
Y_COORD            ::= NUMBER
```

November 27, 2018

---

```

HEIGHT           ::=  NUMBER
WIDTH            ::=  NUMBER
DESTINATION      ::=  PAGE_IN_SAME_DOC | EXTERNAL_DEST
PAGE_IN_SAME_DOC ::=  INTEGER_NUMBER /* the page number */
EXTERNAL_DEST    ::=  FILE_NAME [ (PAGE_NUMBER | DST_NAME)
                             [ VIEW_X [ VIEW_Y ] ] ]
FILE_NAME        ::=  a quoted string, e. g. "../in.pdf"
VIEW_X           ::=  INTEGER_NUMBER
VIEW_Y           ::=  INTEGER_NUMBER
BOOKMARK_SECTION ::=  "Bookmark" NL { BOOKMARK_ENTRY NL }
BOOKMARK_ENTRY  ::=  LEVEL TITLE BM_DESTINATION
LEVEL            ::=  NUMBER /* starting at 0, max. 4, max. one more than
                        previous */
TITLE            ::=  a quoted string, e. g. "Overview and Introduction"
BM_DESTINATION  ::=  PAGE_IN_SAME_DOC | FILE_NAME [ PAGE_NUMBER ]
BOX_SETTING      ::=  ".bs" NUMBER /* sets the /Border value of link
                        annotations; 0 = no border box, 1 = red dashed, 2 =
                        solid red, 3 = green dashed, 4 = solid green, 5 = blue
                        dashed, 6 = solid blue, other numbers of the form YXBGR
                        will result in box with red = R/10, green = G/10, blue =
                        B/10, dash length = X, dash space = Y; values of 9 are
                        mapped to 1 rather than 0.9 */

```

The separator between symbols is one or several blanks (ASCII 32). The comment character is the number sign (#).

Example:

```

# Sample link definition file for pdcat
Bookmark
0 "Overview" 1
0 "Chapter 1" 2
1 "Section 1.1" 3

Page 1
.bs 1 # 22211
56.40 682.80 12.0 37.0 2 # link to pdinfo
56.40 607.20 12.0 32.4 "pdftools.pdf" 3 92 278 # pdcat
.bs 12900 # blue dashed
227.76 773.76 30.24 127.68 "http://www.pdf-tools.com" # web link
Page 2

```



November 27, 2018

---

```
227.76 773.76 30.24 127.68 "http://www.pdf-tools.com" # web link
```