PDFtk - PDF Toolkit

PDF Toolkit (PDFtk) is a platform for working with PDF files. It is available from PDF Labs in the form of a cross-platform PDFtk command line tool, **PDFtk Server**, and two proprietary Windows-only graphical user interfaces (GUI) for PDFtk Server, PDFtk Free (freeware) and PDFtk Pro (paid).

PDFtk Server can:

- Merge PDF Documents or Interleave PDF Page Scans
- Split a PDF Document into Individual Page PDF Files
- Rotate, Reorder, Extract, Delete or Insert Pages in a PDF Document
- Decrypt Input (Password Required) and Encrypt Output PDF Documents
- Fill PDF Forms with X/FDF Data and/or Flatten Forms
- Generate FDF Data Stencils from PDF Forms
- Apply a Background Watermark or a Foreground Stamp
- Report PDF Metrics, Bookmarks and Metadata
- Add/Update PDF Bookmarks or Metadata
- Attach Files to PDF Pages or the PDF Document; Unpack PDF Attachments
- Uncompress and Re-Compress Page Streams
- Repair Corrupted PDF Files (where possible)

PDFtk Server is a PDF command line interface (CLI) tool based on an old version of the iText PDF library (itext-paulo-155 modified using iText 5.1.3) and compiled from Java to native code. It is open source and free under its GNU GPLv2 license but requires purchase of a redistribution license if distributed as part of a commercial product. The latest release is version 2.02, 24 Jul 2013, an installer for which can be downloaded for Windows XP through 10 from https://www.pdflabs.com/tools/pdftk-server/ .

For users who prefer a GUI, the following is a partial list of available GUIs for the PDFtk command line tool. Some of these GUIs are based on Version 1 (2004-2012) releases of PDFtk and therefore may have compatibility issues with later releases of the PDFtk Server executable (pdftk.exe).

- PDFtk Free and PDFtk Pro (Windows)
- PDFTK Builder (Windows)
- PDFTK Builder Enhanced (Windows) ← Recommended (Windows)
- PDFChain (Linux) ← Recommended (Linux)
- Pdftk4All (Windows)

PDFtk Server Manual

Online: https://www.pdflabs.com/docs/pdftk-man-page/
Command: pdftk –help or -h

This manual documents all options and operations of version 2.02 of PDFtk Server (pdftk.exe). This copy has been reformatted from the online manual as a PDF document and supplemented with copies of the PDFtk Server examples and version history from the PDF Labs website, along with information on some of the available GUIs for PDFtk Server.

Synopsis

pdftk < input PDF files | - | PROMPT >
[input_pw < input PDF owner passwords | PROMPT >]
[< operation > < operation arguments >]
[output < output filename | - | PROMPT >]
[encrypt_40bit | encrypt_128bit]
[allow < permissions >]
[owner_pw < owner password | PROMPT >]
[user_pw < user password | PROMPT >]
[flatten] [need_appearances]
[compress | uncompress]
[keep_first_id | keep_final_id]
[drop_xfa]
[verbose]
[dont_ask | do_ask]

Where:
< operation > can be empty, or:
[cat | shuffle | burst | rotate |
generate_fdf | fill_form |
background | multibackground |
stamp | multistamp |
dump_data | dump_data_utf8 |
dump_data_fields | dump_data_fields_utf8 |
dump_data_annots |
update_info | update_info_utf8 |
attach_files | unpack_files]

Options
A summary of options is included below.

--help, -h
Show this summary of options.

< input PDF files | - | PROMPT >
A list of the input PDF files. If you plan to combine these PDFs (without using handles) then list files in the order you want them combined. Use - to pass a single PDF into pdftk via stdin. Input files can be associated with handles, where a handle is one or more upper-case letters:

< input PDF handle >=< input PDF filename >

Handles are often omitted. They are useful when specifying PDF passwords or page ranges, later.

For example: A=input1.pdf QT=input2.pdf M=input3.pdf

[input_pw < input PDF owner passwords | PROMPT >]
Input PDF owner passwords, if necessary, are associated with files by using their handles:

< input PDF handle >=< input PDF file owner password >

If handles are not given, then passwords are associated with input files by order.

Most pdftk features require that encrypted input PDF are accompanied by the ~owner~ password. If the input PDF has no owner password, then the user password must be given, instead. If the input PDF has no passwords, then no password should be given.

When running in do_ask mode, pdftk will prompt you for a password if the supplied password is incorrect or none was given.

[< operation > < operation arguments >]
Available operations are: cat, shuffle, burst, rotate, generate_fdf, fill_form, background, multibackground, stamp, multistamp, dump_data, dump_data_utf8, dump_data_fields, dump_data_fields_utf8, dump_data_annots, update_info, update_info_utf8, attach_files, unpack_files. Some operations take additional arguments, described below.

If this optional argument is omitted, then pdftk runs in 'filter' mode. Filter mode takes only one PDF input and creates a new PDF after applying all of the output options, like encryption and compression.

cat [< page ranges >]
Assembles ("catenates") pages from input PDFs to create a new PDF. Use cat to merge PDF pages or to split PDF pages from documents. You can also use it to rotate PDF pages. Page order in the new PDF is specified by the order of the given page ranges. Page ranges are described like this:

[< input PDF handle >] [< begin page number > [-< end page number > [< qualifier >]]] [< page rotation >]

For example:

```
pdftk A=in1.pdf B=in2.pdf cat A1 B2-20even output out.pdf
```

The handle identifies one of the input PDF files, and the beginning and ending page numbers are one-based references to pages in the PDF file. The qualifier can be even or odd, and the page rotation can be north, south, east, west, left, right, or down.

If a PDF handle is given but no pages are specified, then the entire PDF is used. If no pages are specified for any of the input PDFs, then the input PDFs' bookmarks are also merged and included in the output.

If the handle is omitted from the page range, then the pages are taken from the first input PDF.

The even qualifier causes pdftk to use only the even-numbered PDF pages, so 1-6even yields pages 2, 4 and 6 in that order. 6-1even yields pages 6, 4 and 2 in that order.

The odd qualifier works similarly to the even.

The page rotation setting can cause pdftk to rotate pages and documents. Each option sets the page rotation as follows (in degrees): north: 0, east: 90, south: 180, west: 270, left: -90, right: +90, down: +180. left, right, and down make relative adjustments to a page’s rotation.

If no arguments are passed to cat, then pdftk merges all input PDFs in the order they were given to create the output.

NOTES:

< end page number > can be less than < begin page number >.

The keyword end can be used to reference the final page of a document instead of a page number.

Reference a single page by omitting the ending page number in the page range.

The handle can be used alone to represent the entire PDF document, e.g., B1-end is the same as B.

You can reference page numbers in reverse order by prefixing them with the letter r. For example, page r1 is the last page of the document, r2 is the next-to-last page of the document, and rend is the first page of the document. You can use this prefix in ranges, too, for example r3-r1 is the last three pages of a PDF.

Page range examples without handles:

1-endeast - rotate entire document 90 degrees
5 11 20
5-25oddwest - take odd pages in range, rotate 90 degrees
6-1

Page Range Examples Using Handles:

Say A=in1.pdf B=in2.pdf, then:

A1-21 - take range from in1.pdf
Bend-1odd - take all odd pages from in2.pdf in reverse order
A72 - take a single page from in1.pdf
A1-21 Beven A72 - assemble pages from both in1.pdf and in2.pdf
Awest - rotate entire document 90 degrees
B - use all of in2.pdf
A2-30evenleft - take the even pages from the range, remove 90 degrees from each page's rotation
A A - catenate in1.pdf with in1.pdf
Aevenwest Aoddeast - apply rotations to even pages, odd pages from in1.pdf
Awest Bwest Bdown - catenate rotated documents

shuffle [< page ranges >]
Collates pages from input PDFs to create a new PDF. Works like the cat operation except that it takes one page at a time from each page range to assemble the output PDF. If one range runs out of pages, it continues with the remaining ranges. Ranges can use all of the features described above for cat, like reverse page ranges, multiple ranges from a single PDF, and page rotation. This feature was designed to help collate PDF pages after scanning paper documents.

burst
Splits a single input PDF document into individual pages. Also creates a report named doc_data.txt which is the same as the output from dump_data. If the output section is omitted, then PDF pages are named: pg_%04d.pdf, e.g.: pg_0001.pdf, pg_0002.pdf, etc. To name these pages yourself, supply a printf-styled format string in the output section. For example, if you want pages named: page_01.pdf, page_02.pdf, etc., pass output page_%02d.pdf to pdftk.

Encryption can be applied to the output by appending output options such as owner_pw, e.g.:

```
pdftk in.pdf burst owner_pw foopass
```

rotate [< page ranges >]
Takes a single input PDF and rotates just the specified pages. All other pages remain unchanged. The page order remains unchanged. Specify the pages to rotate using the same notation as you would with cat, except you omit the pages that you aren't rotating:

[< begin page number > [-< end page number > [< qualifier >]]] [< page rotation >]

The qualifier can be even or odd, and the page rotation can be north, south, east, west, left, right, or down.

Each option sets the page rotation as follows (in degrees): north: 0, east: 90, south: 180, west: 270, left: -90, right: +90, down: +180. left, right, and down make relative adjustments to a page's rotation.

The given order of the pages doesn't change the page order in the output.

generate_fdf
Reads a single input PDF file and generates an FDF file suitable for fill_form. It saves this FDF file using the output filename. If no output filename is give, it outputs the FDF to stdout. Does not create a new PDF.

fill_form < FDF data filename | XFDF data filename | - | PROMPT >
Fills the single input PDF's form fields with the data from an FDF file, XFDF file or stdin. Enter the data filename after fill_form, or use - to pass the data via stdin, like so:

```
pdftk form.pdf fill_form data.fdf output form.filled.pdf
```

If the input FDF file includes Rich Text formatted data in addition to plain text, then the RichText data is packed into the form fields as well as the plain text. Pdftk also sets a flag that cues Reader/Acrobat to generate new field appearances based on the Rich Text data. So when the user opens the PDF, the viewer will create the Rich Text fields on the spot. If the user's PDF viewer does not support Rich Text, then the user will see the plain text data instead. If you flatten this form before Acrobat has a chance to create (and save) new field appearances, then the plain text field data is what you'll see in the flattened PDF.

Also see the flatten and need_appearances options.

background < background PDF filename | - | PROMPT >
Applies a PDF watermark to the background of a single input PDF. Pass the background PDF's filename after background like so:

```
pdftk in.pdf background back.pdf output out.pdf
```

Pdftk uses only the first page from the background PDF and applies it to every page of the input PDF. This page is scaled and rotated as needed to fit the input page. You can use - to pass a background PDF into pdftk via stdin.

If the input PDF does not have a transparent background (such as a PDF created from page scans) then the resulting background won't be visible — use the stamp operation instead.

multibackground < multibackground PDF filename | - | PROMPT >
Same as the background operation, but applies each page of the background PDF to the corresponding page of the input PDF. If the input PDF has more pages than the stamp PDF, then the final stamp page is repeated across these remaining pages in the input PDF.

stamp < stamp PDF filename | - | PROMPT >
This behaves just like the background operation except it overlays the stamp PDF page on top of the input PDF document's pages. This works best if the stamp PDF page has a transparent background.

multistamp < multistamp PDF filename | - | PROMPT >
Same as the stamp operation, but applies each page of the stamp PDF to the corresponding page of the input PDF. If the input PDF has more pages than the stamp PDF, then the final stamp page is repeated across these remaining pages in the input PDF.

dump_data
Reads a single input PDF file and reports its metadata, bookmarks (a/k/a outlines), page metrics (media, rotation and labels) and other data to the given output filename or (if no output is given) to stdout. Non-ASCII characters are encoded as XML numerical entities. Does not create a new PDF.

dump_data_utf8
Same as dump_data except that the output is encoded as UTF-8.

dump_data_fields
Reads a single input PDF file and reports form field statistics to the given output filename or (if no output is given) to stdout. Non-ASCII characters are encoded as XML numerical entities. Does not create a new PDF.

dump_data_fields_utf8
Same as dump_data_fields except that the output is encoded as UTF-8.

dump_data_annots
This operation currently reports only link annotations. Reads a single input PDF file and reports annotation information to the given output filename or (if no output is given) to stdout. Non-ASCII characters are encoded as XML numerical entities. Does not create a new PDF.

update_info < info data filename | - | PROMPT >
Changes the bookmarks and metadata in a single PDF's Info dictionary to match the input data file. The input data file uses the same syntax as the output from dump_data. Non-ASCII characters should be encoded as XML numerical entities. This does not change the metadata stored in the PDF's XMP stream, if it has one. For example:

```
pdftk in.pdf update_info in.info output out.pdf
```

update_info_utf8 <info data filename | - | PROMPT>
Same as update_info except that the input is encoded as UTF-8.

attach_files < attachment filenames | PROMPT > [to_page < page number | PROMPT >]
Packs arbitrary files into a PDF using PDF's file attachment features. More than one attachment can be listed after attach_files. Attachments are added at the document level unless the optional to_page option is given, in which case the files are attached to the given page number (the first page is 1, the final page is end). For example:

```
pdftk in.pdf attach_files table1.html table2.html to_page 6 output out.pdf
```

unpack_files
Copies all of the attachments from the input PDF into the current folder or to an output directory given after output. For example:

```
pdftk report.pdf unpack_files output ~/atts/
```

or, interactively:

```
pdftk report.pdf unpack_files output PROMPT
```

[output < output filename | - | PROMPT >]
The output PDF filename can't be set to the name of an input filename. Use - to output to stdout. When using the dump_data operation, use output to set the name of the output data file. When using the unpack_files operation, use output to set the name of an output directory. When using the burst operation, you can use output to control the resulting PDF page filenames (described above).

[encrypt_40bit | encrypt_128bit]
If an output PDF user or owner password is given, output PDF encryption strength defaults to 128 bits. This can be overridden by specifying encrypt_40bit.

[allow < permissions >]
Permissions are applied to the output PDF only if an encryption strength is specified or an owner or user password is given. If permissions are not specified, they default to 'none,' which means all of the following features are disabled.

The permissions section can include one or more of the following features:

Printing – Top Quality Printing
DegradedPrinting – Lower Quality Printing
ModifyContents – Also allows Assembly
Assembly
CopyContents – Also allows ScreenReaders
ScreenReaders
ModifyAnnotations – Also allows FillIn
FillIn
AllFeatures – Allows the user to perform all of the above, and top quality printing.

[owner_pw < owner password | PROMPT >]

[user_pw < user password | PROMPT >]

If an encryption strength is given but no passwords are supplied, then the owner and user passwords remain empty. This means that the resulting PDF can be opened and its security parameters altered by anybody.

[flatten]
Use this option to merge an input PDF's interactive form fields (and their data) with the PDF's pages. Only one input PDF can be given. Sometimes used with the fill_form operation.

[need_appearances]
Sets a flag that cues Reader/Acrobat to generate new field appearances based on the form field values. Use this when filling a form with non-ASCII text to ensure the best presentation in Adobe Reader or Acrobat. It won't work when combined with the flatten option.

[compress | uncompress]
These are only useful when you want to edit PDF page code in a text editor like vim or emacs. Remove PDF page stream compression by applying the uncompress filter. Use the compress filter to restore page stream compression.

[keep_first_id | keep_final_id]
When combining pages from multiple PDFs, use one of these options to copy the document ID from either the first or final input document into the new output PDF. Otherwise pdftk creates a new document ID for the output PDF. When no operation is given, pdftk always uses the ID from the (single) input PDF.

[drop_xfa]
If your input PDF is a form created using Acrobat 7 or Adobe Designer, then it probably has XFA data. Filling such a form using pdftk yields a PDF with data that fails to display in Acrobat 7. The workaround solution is to remove the form's XFA data, either before you fill the form using pdftk or at the time you fill the form. Using this option causes pdftk to omit the XFA data from the output PDF form.

This option is only needed when running pdftk on a single input PDF. When assembling a PDF from multiple inputs using pdftk, any XFA data in the input is automatically omitted.

[verbose]
By default, pdftk runs quietly. Append verbose to the end and it will speak up.

[dont_ask | do_ask]
Depending on the compile-time settings (see ASK_ABOUT_WARNINGS), pdftk might prompt you for further input when it encounters a problem, such as a bad password. Override this default behavior by adding dont_ask (so pdftk won't ask you what to do) or do_ask (so pdftk will ask you what to do).

When running in dont_ask mode, pdftk will over-write files with its output without notice.

--

PDFtk Server Examples

Online: **https://www.pdflabs.com/docs/pdftk-cli-examples/**

These examples show you how to perform common PDF tasks from the command-line using pdftk.

Collate even and odd scanned PDF pages

```
pdftk A=even.pdf B=odd.pdf shuffle A B output collated.pdf
```

or if odd.pdf is in reverse order:

```
pdftk A=even.pdf B=odd.pdf shuffle A Bend-1 output collated.pdf
```

Decrypt a PDF

```
pdftk secured.pdf input_pw foopass output unsecured.pdf
```

Encrypt a PDF using 128-bit strength (the default), withhold all permissions (the default)

```
pdftk 1.pdf output 1.128.pdf owner_pw foopass
```

Same as above, except password baz must also be used to open output PDF

```
pdftk 1.pdf output 1.128.pdf owner_pw foo user_pw baz
```

Same as above, except printing is allowed (once the PDF is open)

```
pdftk 1.pdf output 1.128.pdf owner_pw foo user_pw baz allow printing
```

Join in1.pdf and in2.pdf into a new PDF, out1.pdf

```
pdftk in1.pdf in2.pdf cat output out1.pdf
```

or (using handles):

```
pdftk A=in1.pdf B=in2.pdf cat A B output out1.pdf
```

or (using wildcards):

```
pdftk *.pdf cat output combined.pdf
```

Remove page 13 from in1.pdf to create out1.pdf

```
pdftk in.pdf cat 1-12 14-end output out1.pdf
```

When using the Windows command-prompt, it helps to use drag-and-drop from the file manager: drag the input PDF file from the file manager onto the command-prompt, and its full pathname will appear at the prompt.

Apply 40-bit encryption to output, revoking all permissions (the default). Set the owner PW to foopass.

```
pdftk 1.pdf 2.pdf cat output 3.pdf encrypt_40bit owner_pw foopass
```

Join two files, one of which requires the password foopass. The output is not encrypted.

```
pdftk A=secured.pdf 2.pdf input_pw A=foopass cat output 3.pdf
```

Uncompress PDF page streams for editing the PDF in a text editor (e.g., vim, emacs)

```
pdftk doc.pdf output doc.unc.pdf uncompress
```

Repair a PDF's corrupted XREF table and stream lengths, if possible

```
pdftk broken.pdf output fixed.pdf
```

Burst a single PDF document into pages and dump its data to doc_data.txt

```
pdftk in.pdf burst
```

Burst a single PDF document into encrypted pages. Allow low-quality printing

```
pdftk in.pdf burst owner_pw foopass allow DegradedPrinting
```

Write a report on PDF document metadata and bookmarks to report.txt

```
pdftk in.pdf dump_data output report.txt
```

Rotate the first PDF page to 90 degrees clockwise

```
pdftk in.pdf cat 1east 2-end output out.pdf
```

Rotate an entire PDF document to 180 degrees

```
pdftk in.pdf cat 1-endsouth output out.pdf
```

Article Author: Sid Steward

PDFtk Version History

Online: **https://www.pdflabs.com/docs/pdftk-version-history/**

Note: The yellow highlighting below denotes significant enhancements incorporated in PDFtk releases beyond version 1.41 (Nov 2006) which is the release of PDFtk included in the distributions of PDFtk Builder by Angus Johnson.

2.02 - July 24, 2013

- Added drop_xmp output option for removing the document XMP metadata stream from a PDF.
- Added dump_data output of custom page data embedded by STAMPtk tool. See the embed option in STAMPtk for more information.
- Improved PDF bookmark merging logic so it can handle more input cases.
- Fixed a password bug where some 'upper-ASCII' characters weren't being mapped to the correct code points.
- Fixed a 40-bit decryption bug introduced in version 2.00.
- Fixed a bug in the bookmark merging logic that caused bookmarks to be omitted from the merged PDF.
- Added a test to ensure that encryption passwords use permitted characters only. (Decryption attempts still allow a larger set of input characters.)
- Rewrote the wide-to-utf8 code for Windows to make it more rigorous.
- Organized our calls of JvInitClass() in main().
- Added descriptions to some exception reports.
- Reviewed some code from pdftk.cc, PdfReader.java, PdfWriter.java and friends.

2.01 - June 5, 2013

- Fixed an uncompress bug introduced in 2.00 that corrupted some image streams.
- Updated the Windows pdftk.exe compiler settings to remedy an elusive NullPointerException reported in the field. This problem first appeared in version 2.00.

2.00 - May 22, 2013

- Added AES decryption of input PDFs. The 'owner' password is still required when decrypting any PDF.
- Added merging of bookmarks/outlines when merging full PDFs.
- Added new rotate operation, which is a convenient way of rotating select pages of a single PDF.
- Added new dump_data_annots operation. Currently it reports only link annotation information.
- Added new need_appearances output option. Use this when filling a form with non-ASCII text to ensure the best presentation in Adobe Reader/Acrobat. It won't work when combined with the flatten option.
- Improved the compress option so that output PDFs are more compact and efficient.
- Added page media information to dump_data output: page rotation, page media bounds and page crop bounds.
- Improved the performance of dump_data so it works better with very large PDFs.
- Improved the memory management in the Windows binary. This fixes the rare "Too many heap sections" error.
- Fixed a bug where form fields with multiple values were not being properly reported by dump_data_fields.

- Fixed a burst bug that was corrupting the output PDF pages.
- Fixed an input bug to allow interactive prompting of both the user and owner passwords.
- Fixed a burst bug so that doc_data.txt is now output to the same directory as the PDF's pages when an output directory is given.
- Fixed a bug where indirect references to the PDF ID in the trailer would cause a crash.
- Added a test to fill_form so it checks that an input PDF is a form before trying to fill it with data.
- Added a return value of 3 for warnings 'PDF information not added' or 'PDF form not filled.'
- Improved the error message for cat page range errors.
- Fixed the error report when an input page number is out of range.
- Fixed a burst bug where document metadata wasn't being copied properly to the output PDFs.
- Updated the Bouncy Castle library to 1.48.
- When using the cat operation, the output PDF version number is now set to the maximum PDF version of all of the input PDFs. If any of the input PDFs have PDF extension levels, then the greatest extension level is also copied to the output PDF.

1.45 - December 6, 2012

- You can now add or change a PDF's bookmarks using update_info.
- Added record delimiters to dump_data output to help make parsing more reliable.
- The changes to dump_data output (described above) are also now required for the input to update_info.
- You can now use multi-character input handles. Prior versions were limited to a single character, imposing an arbitrary limitation on the number of input PDFs when using handles. Handles still must be all upper-case ASCII.
- Added means of referring to PDF pages in reverse order. By prefixing a page number with an r, it counts from the end of the document. For example, r1 is the last page, r2 is the next-to-last page, etc.
- Changed the syntax for page rotation. Instead of N, S, E, W, L, R and D, now use: north, south, east, west, left, right and down.
- Fixed a problem reading input PDF filenames that have non-ASCII characters on Windows 7.
- Fixed a stream parsing issue with troublesome PDFs that don't strictly follow the PDF specification.

1.44 - October 28, 2010

- Added new feature for collating PDF page scans: shuffle. Please see the man page for usage details.
- Introduced update_info_utf8, dump_data_utf8 and dump_data_fields_utf8 to provide UTF-8 companions to update_info, dump_data and dump_data_fields. These latter operations use XML numerical entities to encode non-ASCII characters. In version 1.43, we changed the encoding for update_info to UTF-8, but that made it incompatible with dump_data and also broke some downstream applications. By introducing these UTF-8 operations, we can revert update_info to its original behavior.
- Burst feature now copies the metadata (including XMP) from the input file to the output pages.
- Updated Bouncy Castle library to 1.45.
- Removed or replaced third-party code that wasn't compatible with pdftk's GPL license.
- Updated third-party license information.

1.43 - September 30, 2010

- Improved input handle detection to reduce false hits.
- Improved keyword detection logic to eliminate false hits when input filenames happen to include pdftk keywords even, odd and end.
- Added option of prompting the user for the output when bursting a PDF. Also reviewed other filename prompting code.
- Changed the PDF parser to accept name tokens longer than 127 characters — the PDF Specification says that 127 is the limit. This isn't related to file names. The issue arose with PDFs created by Acrobat Web Capture 9.0.
- Fixed a problem with filling form choice fields in some PDFs where the old form value was 'sticking.'
- Changed pdftk behavior when handling subset fonts so it doesn't alter font name "tags." This was causing printing problems with Acrobat 3.01 on Windows.
- Fixed a stream parsing bug that was causing page content to disappear after merge of PDFs generated by Microsoft Reporting Services PDF Rendering Extension 10.0.0.0.
- Added multistamp and multibackground features provided by a Debian patch — thanks!
- Clear the signal mask as workaround to environments that turn off signals before calling pdftk. This problem is known to cause pdftk to hang in some Python web setups as well as in PHP.
- Set locale to C as workaround to an unusual exception. This is a Debian patch. Please let me know if it causes any troubles.
- Improved reporting of output errors via Debian patch — thanks!
- Added support for UTF-8 data in update_info via Debian patch — thanks!
- Added support for UTF-8 filenames via Debian patch — thanks!
- Updated build procedure to work better with newer versions of GCC. Maintained compatibility with older versions of GCC.
- Added license information to the source tree for the third-party libraries that pdftk uses.

1.41 - November 28, 2006 *(Version provided with distributions of PDFTK Builder by Angus Johnson.)*

- Fixed a bug that corrupted output PDF xref tables. This corruption was mild but universal. Most PDF tools can cope with the corrupted PDFs, but I recommend upgrading from 1.40 to 1.41 as soon as possible. This bug was introduced in version 1.40 — version 1.12 does not have this bug.
- Fixed a bug that prevented XFDF form data from being passed to pdftk via stdin.
- Commented out some unused code from pdftk.cc.

1.40 - September 19, 2006

- Added the stamp operation, a natural complement to the existing background operation.
- Added the page rotating patch provided by David Fabel &mdash thanks! Tweaked the patch so it handles a greater variety of input syntax (e.g., 1-20evenE).
- Added the generate_fdf patch provided Bernhard R. Link &mdash thanks! I actually rewrote the patch so it uses FDF features built into the iText library. Please let me know my changes break anything downstream.
- The fill_form operation can now take XFDF data as well as FDF data. This feature was sponsored by Vesaria — thanks!
- Added the drop_xfa option so pdftk could fill forms created with newer versions of Acrobat or Adobe Designer. Read more about this above.
- Added the keep_first_id and keep_final_id options for more PDF fun.

- Upgraded the iText library we use to itext-paulo rev. 155. This makes pdftk harder to compile on older versions of gcc.
- Added the -O2 optimizing switch to Makefile GCJFLAGS. This should make pdftk leaner and meaner, but could be dropped if your build acts funny (like segfaulting).
- Fixed a bug that caused pdftk to create bloated PDFs when input PDF pages had links on their pages.
- Added License-Adobe.txt to the fonts folder, as required for distribution of Adobe's AFM files.

1.12 - November 9, 2004

- Fixed a bug where the presense of page annotations in some PDFs would cause pdftk to crash. This bug first emerged when processing a PDF created by FPDF (version 1.52) that contained web links. Turns out that pdftk erroneously expected all page annotations to be indirect objects. This assumption has been removed from the code.

1.11 - November 3, 2004

- Fixed a couple bugs in the dump_data_fields form field reporting. Also improved this feature so it now reports all possible settings for check box, radio button, list box and combo box form fields.

1.10 - October 27, 2004

- Fixed the background feature so it handles rotated pages (input PDF or background PDF) better. Pdftk will transform the background PDF page so that its orientation is preserved on every page of the output PDF, even on input PDF pages that are rotated. I chose this logic so as to give the user greater control over the results; rotate pages before processing to achieve the desired output. Let me know if this logic is too inconvenient for you.
- Fixed form field handling when combining PDF pages. Pdftk used to permit duplicate form field names, which is illegal PDF. Now, pdftk detects duplicates and adds name prefixes as needed. If no duplicates occur, then no changes are made. If an input PDF has a field represented by multiple annotations, then that is respected and preserved in the output.
- An especially nice upshot to this new handling is that you can now assemble duplicate PDF forms and not end up with all of their fields echoing each other (as you get with Acrobat). Run pdftk A=form.pdf cat A A A output formX3.pdf and you'll get a form that behaves as you would expect.
- Added stdin support for input PDF, FDF, or Info files (thanks to Bart Orbons for this patch).
- Added a means for users to control the output PDF filenames when using the burst feature: pass in a printf-styled format string via output (documented above).
- Changed background command-line syntax, so it is an operation instead of an output option. The old syntax also works, for backward compatibility.
- Now shuffling subset font name prefixes for input PDFs, to prevent duplicates.
- Updated Makefile.Mandrake according to feedback from Larry Gilliland.
- Reduced the Windows EXE filesize using UPX, as suggested by Ralf Koenig.

1.00 - August 14, 2004

- Upgraded the iText library we use to itext-paulo rev. 132, which resolved a bug involving bookmark page references in dump_data output.
- Fixed the problem of form fields getting corrupted by splitting or merging PDF form pages.

- Building the Windows binary using libgcj 3.4 seems to have fixed the problem of using accented characters in filenames and paths.
- Added these new operations: fill_form, update_info, attach_file, and unpack_file.
- Added the background and flatten output options.
- Added the do_ask interactive mode (the default on Windows) that asks before overwriting files and asks for passwords to input PDFs, if necessary. Also added the dont_ask mode (the default on Linux), for hands-free operation.
- Many input fields can be substituted with PROMPT, which cues pdftk to ask the user for a filename or password upon execution.
- Added output to stdout via output -.
- Using the uncompress option now also adds page numbers to page dictionaries, for easy lookup. Find page N (1-based) by searching for /pdftk_PageNum N. Using the compress option removes these markers.
- Added Mac OS X Makefile, and removed the optimization flag from the GCJ flags (which would cause trouble on older versions of gcc, such as 3.2.2) in all Makefiles.
- Now catching PDF output open exceptions.
- Builds now pack iText font afm files into pdftk, which are required for the new form filling feature.

0.941 - March 28, 2004

- Fixed the 'Input_UnicodeBig not found' error encountered by Windows users when using the dump_data or the burst operations on some PDFs.
- Added an optimization flag to the gcj arguments. This can be adjusted or omitted by editing your platform-specific Makefile.
- Renamed the CC_OPTS Makefile macro to CXXFLAGS, for uniformity.

0.94 - March 24, 2004

- Fixed a string copy bug in pdftk.cc.
- Fixed unicode string output so it drops initial, signature character.
- Fixed nagging gnu.java.locale.Calendar static linking problem (Windows).
- Made more improvements towards gcc/gcj 3.2 compatibility (e.g., RedHat 8, 9).
- Added macros to Makefiles, for easier porting.
- Added simple return codes: 0 --> Success, 1 --> Error, 2 --> Exception. Some exceptions will return an "Error" return code.
- Removed warning issued when an input PDF has no ID string.
- Empty Info fields no longer reported on dump_data.
- Added newline to end of --help output.

0.93 - March 7, 2004

- Removed restriction on the number of input documents. For example, you can now run:
- pdftk *.pdf cat output combined.pdf
- to assemble any number of PDFs into a single document.
- Made pdftk run silently by default, and added the verbose output option for when you want detailed feedback.
- Changed the encryption strength default from 40-bit to 128-bit.
- Improved file open error handling and reporting. If pdftk can't open a PDF, it tells you why.

- Added RedHat9 and Mandrake makefiles (Thanks to Andre Gompel and Pablo Rodríguez). Support for these platforms is still experimental.
- Copied the MD5 code from libgcj into our tree, to improve support for older compilers/libraries. This should improve RedHat9 and Mandrake support.
- Removed pointless warning sometimes issued by the libgcj security class.
- Added debian directory and Aurélien's man page, updated man page.
- Reorganized Makefiles (thanks Andre).

0.92

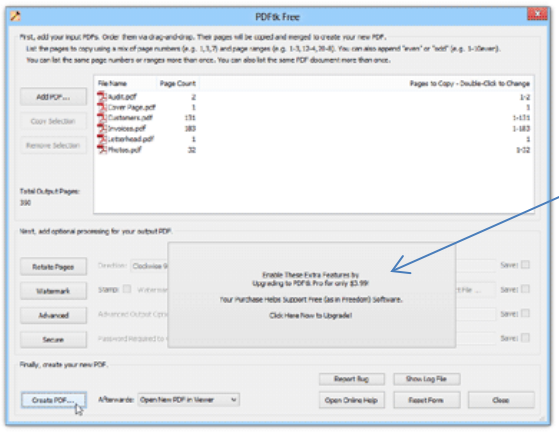
- Added logical page numbering (a/k/a page labels) to dump_data operation.
- Appended .omit extension to a few iText files we don't use, to speed compiling.

0.91

- Removed restriction on adding the same page from the same PDF more than once.
- Added Solaris Makefile.
- AllFeatures permission now implies 'top quality' printing, not 'degraded' printing.
- Fixed handling of 'empty owner password' case during output encryption.
- Added test to make sure the user password is not the same as the owner pw.
- CopyContents also allows ScreenReaders.
- ModifyAnnotation also sets the FillIn bit.
- ModifyContents also sets the Assembly bit.
- Updated docs.
- Added --version switch.

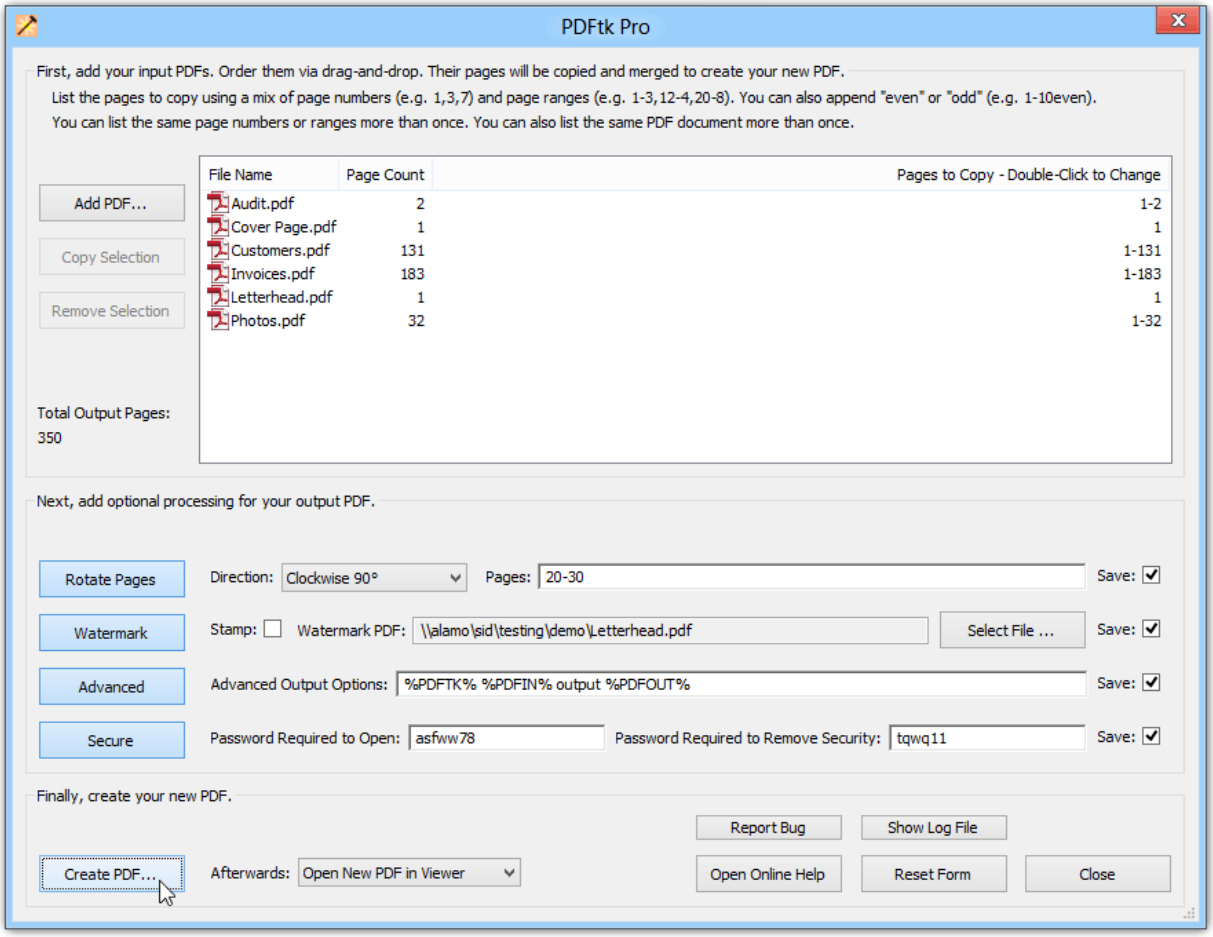
PDFtk GUI Front Ends

PDFtk Free (Windows): https://www.pdflabs.com/tools/pdftk-the-pdf-toolkit/

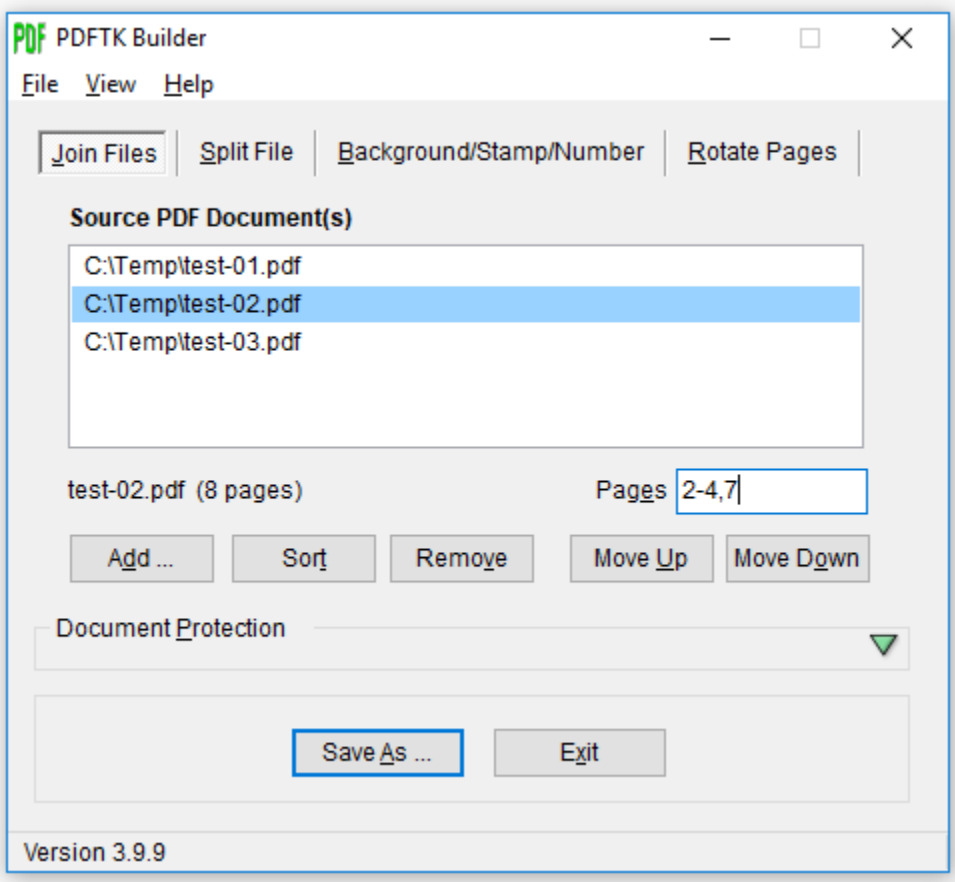


The free version of the PDFTK GUI is the same as the Pro version (below) except that the optional processing section (Rotate, Watermark, Advanced, Secure) is disabled.

PDFtk Pro ($3.99; Windows): https://www.pdflabs.com/tools/pdftk-pro/



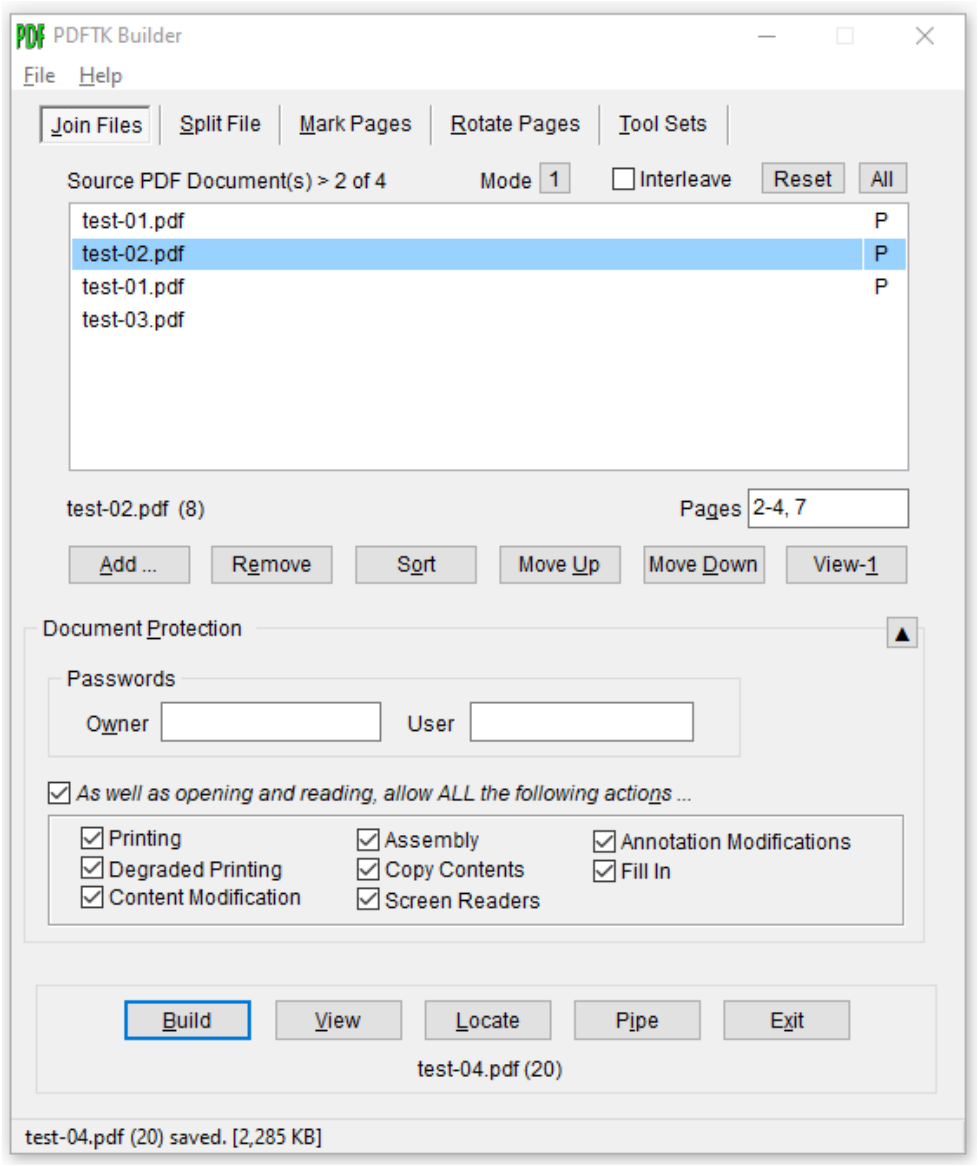
PDFTK Builder (Windows): http://www.angusj.com/pdftkb/#pdftkbuilder



PDFTK Builder is still based on and distributed with binaries for version 1.41 (Nov 2006) of the PDFtk command line tool. Its 'Rotate Pages' function will not work with PDFtk v1.45 or higher because of changes in the syntax of PDFtk's 'cat' (catenate) operation introduced in v1.45.

PDFTK Builder Enhanced (Windows):

PDFTK Builder Enhanced updates PDFTK Builder Ver. 3 to be compatible with version 2 of the PDFtk command line tool, PDFtk Server, and adds a new tab sheet (Tool Sets) for access to additional PDFtk operations. Additional enhancements include buttons for viewing input and output PDF files, extension of file drag and drop to all tab sheets, improvements to the 'Join Files' list, new options for splitting files and numbering (marking) pages, and extensive documentation.



PDF Chain (Linux): http://pdfchain.sourceforge.net/

PDF Chain is a GUI for the PDFtk command line tool for Linux systems. Its interface is similar to PDFtk Builder but adds Attachment and Tools tabs for access to additional PDFtk operations. It is written in C++, using the GTKmm library and supports the latest version of PDFtk.