



PDF MELD

FyTek, Inc.

Web site: <https://www.fytek.com>

FyTek's PDF Meld

Trademarks

FyTek, FyTek PDF Meld and the FyTek logo are registered trademarks or trademarks of FyTek Incorporated in the United States and/or other countries. Acrobat, Adobe, Adobe PDF and Adobe Reader are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other product names, logos, designs, titles, words or phrases mentioned within this publication may be trademarks, servicemarks, or tradenames of FyTek, Inc. or other entities and may be registered in certain jurisdictions including internationally.

FyTek Disclaimer

FYTEK, INC. MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING THE ENCLOSED COMPUTER SOFTWARE PACKAGE, ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PARTICULAR PURPOSE. THE EXCLUSION OF IMPLIED WARRANTIES IS NOT PERMITTED BY SOME STATES. THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. THIS WARRANTY PROVIDES YOU WITH SPECIFIC LEGAL RIGHTS. THERE MAY BE OTHER RIGHTS THAT YOU MAY HAVE WHICH VARY FROM STATE TO STATE. Copyright © 2000-2024 FyTek, Inc. All rights reserved. This manual may not be copied, photocopied, reproduced, translated, or converted to any electronic or machine-readable form in whole or in part without prior written approval of FyTek, Inc.

This guide may contain links to third-party websites that are not under the control of FyTek, and FyTek is not responsible for the content on any linked site. If you access a third-party website mentioned in this guide, then you do so at your own risk. FyTek provides these links only as a convenience, and the inclusion of the link does not imply that FyTek endorses or accepts any responsibility for the content on those third-party sites.

Acknowledgments

Software Development: Mike Bernardo

Writing: Mike Bernardo

FyTek, Inc.

P.O. Box 71093

Madison Heights, MI 48071


Introduction

The demo version of PDF Meld will place footnote referencing PDF Meld and FyTek, Inc. at the bottom of each page in the output PDF. You must purchase a copy of the software to remove the footnote. Visit <https://www.fytek.com> for information on purchasing a copy.

Please contact us at sales@fytek.com if you plan to use in any business application. This includes any use of the product in a business environment, on a web server or redistribution.

PDF Meld is a program to manipulate pages of existing PDFs. It runs either from the command line (executable version) or you can use the .NET dynamic link library (DLL) version on Windows based systems to integrate with your own software. You may also install as a server (referred to as the PDF Meld server in this document) under Windows or Unix and send commands for processing via TCP/IP. Windows also supports running PDF Meld server as a service. You'll need to use a server key name/code combination or a software subscription to use the program in server mode. Test key names and test subscription codes for use with the demo version are available from the FyTek website. Please contact us at support@fytek.com if you need assistance with these options.

There are a variety of functions the program performs:

- Append two or more PDFs, images, Word/Excel/OpenOffice files*, MP3's, movies or plain text files into a single PDF
- Extract a page or list of pages from a single PDF
- Rearrange the pages of a single PDF
- Resize, rotate or move the position of page contents of a single PDF
- Overlay pages of one PDF onto the pages of a second and set transparency
- Add page numbers, Bates numbering, text or barcodes  to each page
- Extract JPEG, TIFF and U3D images
- Place two pages side-by-side on a single sheet
- Place all pages on a single page or perform 2-up imposition
- Clip several sections of a page to create new pages
- Extract text for mailing labels or other uses
- Add bookmarks and annotations (web links, notes and stamps)
- **Highlight**, underline or **color** specified text
- Make minor touch-up text and image changes
- Add fillable data fields

Introduction

- Create new PDFs containing the modified FDF data
- OCR (optical character recognition) using Google Drive APIs
- Embed files such as Excel spreadsheets or Word documents into PDFs
- Extract embedded files from PDFs
- Encrypt PDFs with 40, 128, or 256-bit encryption
- Password protect PDFs
- Digitally sign PDFs
- Disallow printing or changing the PDF
- Optimize for fast web viewing

* With the free "save as PDF" Microsoft Office plug-in installed along with Microsoft Excel or Word. Or you can have OpenOffice installed (as a server on Unix) on any platform to convert these types to PDF for processing.

The source PDFs in all cases above must be unencrypted or you must know the owner password for the encrypted PDF. The output PDF is always unencrypted unless you're using one of the encryption options. The page text must be either uncompressed or flate (zlib) compressed, common for most PDFs, when overlaying pages. PDFs with form fields and/or PDFs with revisions may not work in all situations.

Bookmarks are retained for each input PDF unless the option to remove them is used. The bookmark structure for each input PDF is placed under a new heading named after the input file itself. There are options to keep the bookmark structure intact without adding the additional level of file name. You may also create your own set of bookmarks to use.

PDF Meld runs as an executable or may be called using the .NET/COM DLL version. The executable is simply run from the DOS or Linux/Unix prompt. You can also create a DOS batch file or Linux/Unix shell script to call the program. The program name is pdfmeld.exe (or pdfmeld64.exe) in DOS and pdfmeld (or pdfmeld64) in Linux/Unix. Be sure to mark the file pdfmeld as executable in Linux/Unix by running a command similar to "chmod 777 pdfmeld".

This version allows you to call methods of PDF Meld that match the command line parameters. You can build all the functionality of PDF Meld into your own application or use via the web in programs such as ColdFusion or ASP. The file pdfmeld_20.dll is a .NET DLL compiled for both 32 and 64-bit that you may also use as a COM DLL by registering it using the Microsoft regasm utility.

PDF Meld is available in different configurations:

PDF Meld

Introduction

Version	Pop-Up Message (message "You are using FyTek's PDF Meld...")	Server/Multi-User (for use on a server or in a web environment)	All Features (SE version does not contain certain functions)
PDF Meld	✓		✓
PDF Meld SE* (Server Edition)		✓	
PDF Meld EE (Enterprise Edition)		✓	✓

* The SE server version does not contain the following:

Executable version: -nochange, -string, -strchange, -textdetailin, -textdetailout, -textdetailoutxy,
-nofillin, -data, -imgout, -embed, -pdfmed

DLL version: setNoChange, setString, setStringChange, setTextDetailIn, setTextDetailOut,
setTextDetailOutXY, setNoFillIn, setData, setImgFileOut, setEmbedFile

Contact FyTek at sales@fytek.com for details on what version is right for you.

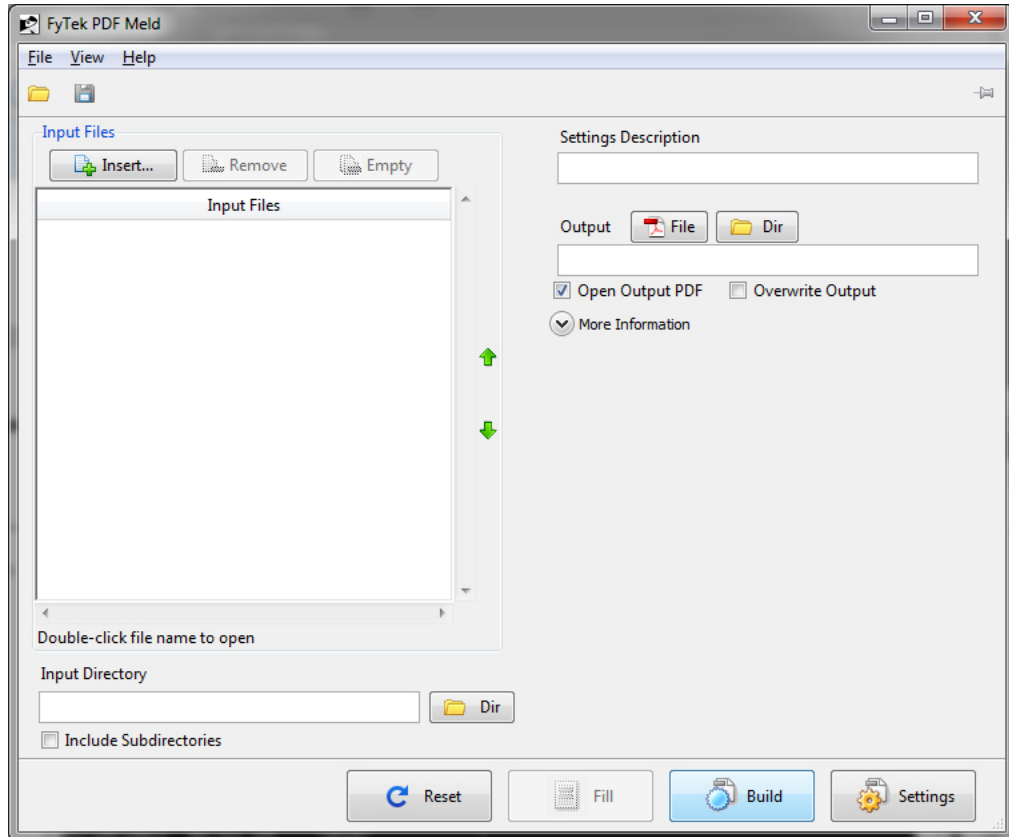
Introduction

PDF Meld contains so many options it may be difficult to know where to start. In its simplest form, PDF Meld takes multiple PDFs and stitches them together to create a new single PDF. When you have the need to perform other functions such as overlay instead of stitching, you'll need to use one or more of the command line or API methods. These are typically named after the function they perform, such as `-overlay` or `setOverlay`.

Other times you might want to add text or form fields to a PDF. In this case, there would be too many options to specify on the command line. Adding text, for example, involves setting the font size, text placement, pages to include on, and so on. Instead of many command line options, an input file is used that allows tag based entry, similar to HTML, to specify all of the options. The various layouts are all documented in the [File Layouts](#) section of this manual. The layouts themselves are loaded via a single command line option. For example, `-strin "filename"` to load a file containing text strings to print on the PDF.

The options for the executable version are detailed under the [Executable](#) section and the DLL methods under the [DLL](#) section. You may find it helpful to spend some time experimenting with various options using the executable version. Or if you're comfortable with what you need to accomplish and want to dive in with the DLL, there are some examples at [GitHub](#) to get you started.

The executable version has a GUI front-end you can use for either prototyping or simply running the program. The GUI front-end is available under Windows, Mac OSX, and graphical versions of Linux. To bring up the GUI window, run the executable (`pdfmeld64.exe` in Windows or `pdfmeld64` in Linux) without passing any parameters. Or pass a single parameter which is a `.mld` (saved settings) file which you can create from within the GUI front-end. The program bypasses the GUI front-end if you send any other options. This is so you can execute in batch or unattended mode when desired. A test license or subscription, which can create or download from FyTek's website, is needed to remove all pop-ups when testing out the demo. The screen below is from Windows though other supported operating systems will appear similar.



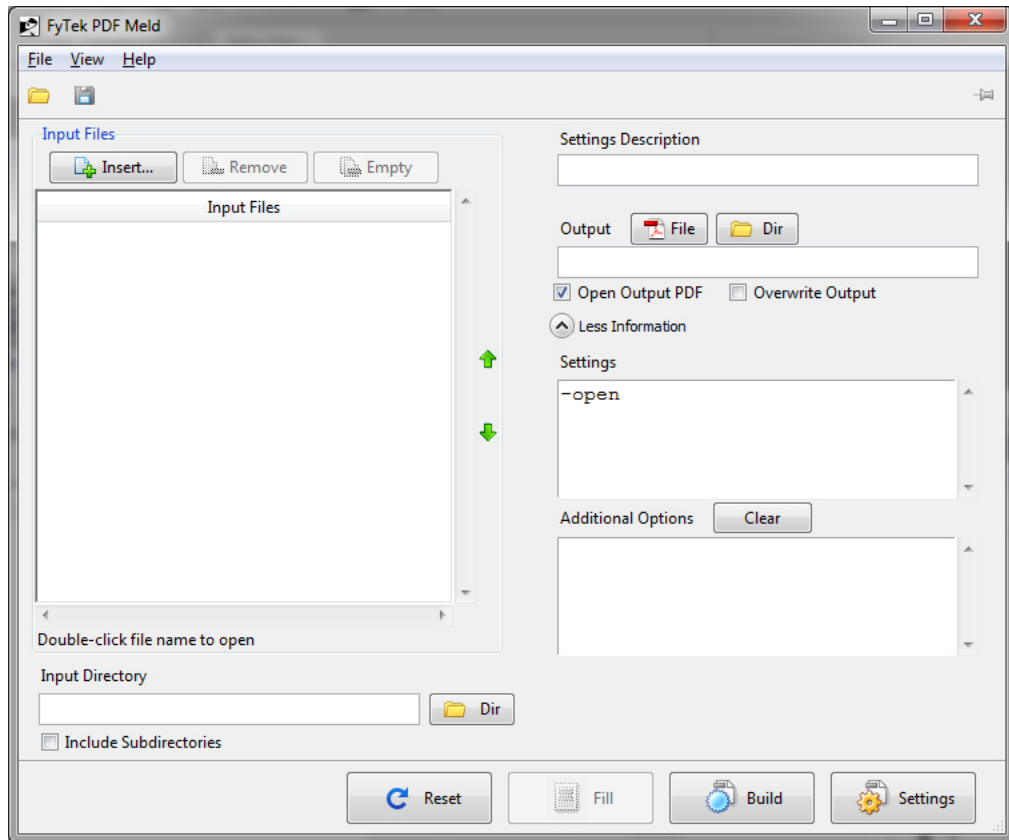
You can insert input files to process on the left. By default, any PDFs or images added will be merged together one after the other in the output PDF. You may see options for adding Microsoft Word and Excel files to the list when running the program under Windows. You must have a version of Microsoft Office installed that includes the free "save as PDF" Office add-in to use these file types. The "save as PDF" add-in is available from Microsoft's website.

Another option to convert Word/Excel on any platform is OpenOffice. Under Windows, PDF Meld will attempt to connect to OpenOffice using the Windows COM sub-system to perform the conversion. This means that the standard OpenOffice installation should be all you need on Windows so PDF Meld can convert Excel/Word documents. The command line utility "unoconv" (part of OpenOffice) can also be used on any operating system if you have OpenOffice installed as a server. Be sure the PATH environment variable for the user running PDF Meld contains the location where unoconv is located. See the OpenOffice documentation for instructions on setting up the OpenOffice server.

You can specify any of the executable parameters in one of two ways.

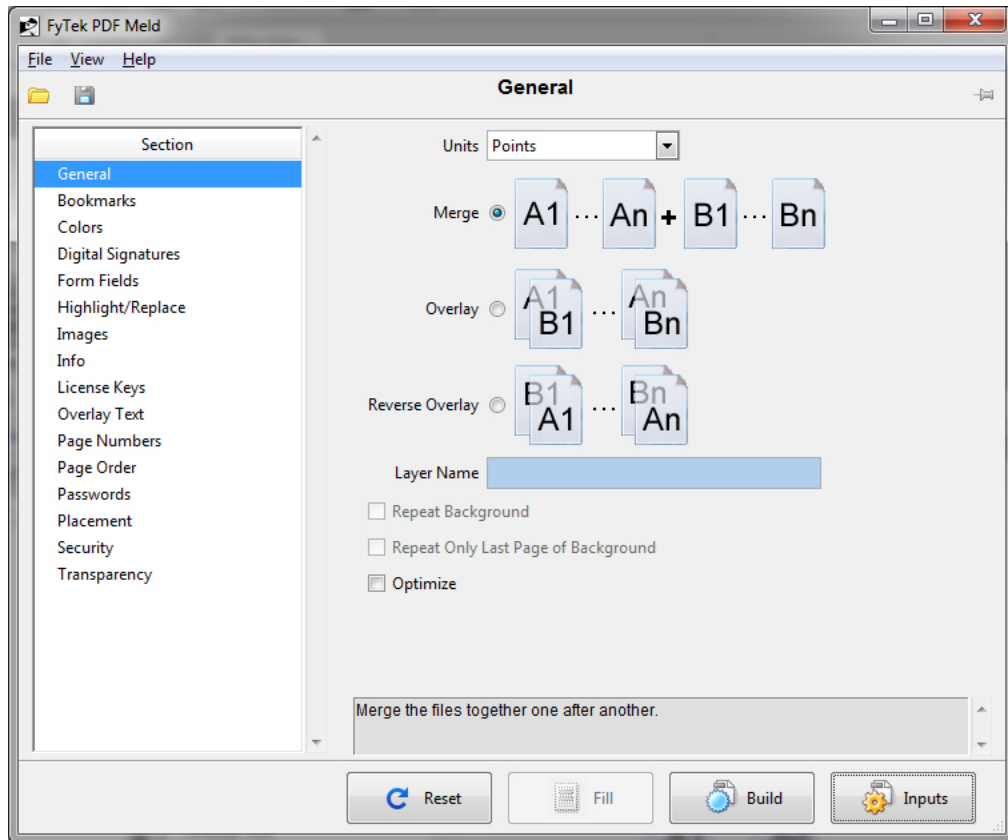
First, you can click on the button on right side labeled "More Information".

Introduction



Two text areas will be shown - the top shows the generated settings (initially empty) that is read-only. The bottom text area is available for you to type in whatever options you wish to pass to PDF Meld. For example, you could enter `-overlay` in the box to perform an overlay of 2 PDFs.

The second way to specify options is to click on the button in the lower left corner labeled "Settings". When you click it, the screen will switch and you can pick and choose what settings to apply in a visual environment.



At any time you may click the button in the lower right corner (now labeled "Inputs") to switch back to your file inputs. Once you have made your settings, click the "Build" button in the lower right corner next to "Inputs" or "Settings". This will start up PDF Meld with the options you specified, run your request and return you back to the application. Not every possible setting is included in the GUI front-end but it contains the more widely used features. If there's something very specific you are trying to accomplish you might need to check the documentation and key in the setting manually.

You can save your settings by clicking the disk icon in the upper left corner or choosing "File Save" from the menu. This will save whatever options you had selected from the settings screen so you can open them at a later time without re-entering.

If you plan on running PDF Meld in batch mode, you can copy the settings the program made from the read-only text area and use that as parameters to pass into PDF Meld. You may also save your settings as a .mld file and use the command line option `-run` or the DLL method `setRun` to execute the saved settings.

Introduction

PDF Meld can also be setup to run on a TCP/IP port in server mode. This can be useful to bypass the startup time needed for the program each time it is ran. The [Client-Server](#) section describes this operation in more detail.

If you have questions or require further assistance, please contact support at support@fytek.com or call us at 248-471-0851 during regular business hours, Monday through Friday 8am to 4pm Eastern Time.

Using the Windows/Linux Executable

The program `pdfmeld.exe` (32-bit) and `pdfmeld64.exe` (64-bit) are the Windows executable programs. For Linux and other flavors of Unix, use `pdfmeld` or `pdfmeld64`. If you are running PDF Meld in server mode, clients can use the program `pdfmeld_tcp` or `pdfmeld_gui_tcp` (or their 64-bit equivalents). You may need to set the program as executable under Linux with the `chmod` command.

The general syntax is:

```
pdfmeld.exe filein1.pdf[,filein2.pdf...] fileout.pdf [options]
```

The above syntax without any options will merge two or more PDFs into a single PDF. `fileout.pdf` is the output PDF in the example above. There are several ways the input files may be specified:

- Use a comma separated list of file names (no space before/after the comma).
For example, "`pdfmeld.exe file1.pdf,file2.pdf,file3.jpg fileout.pdf`"
- Specify a directory rather than a file to process all PDFs in the directory.
For example, "`pdfmeld.exe c:\pdfiles*.pdf fileout.pdf`"
- Specify two or more directories rather than a file to match & process all PDFs.
For example, "`pdfmeld.exe c:\dir1*.pdf;c:\dir2*.pdf c:\dir3 -overlay`"
In this case, the files in `dir1` and `dir2` (and so on if you have more directories you are merging) must be named in such a way that they end with a common number. For example, `c:\dir1\file_17.pdf` will be matched with `c:\dir2\back17.pdf` because they both end with 17 in the file name. The output file for this pair will be `c:\dir3\file_17_back17.pdf`. You must list the input directories followed by the output directory as the first parameters to the program.
- Use an '@' in front of a file name containing a list of files to process.
For example, "`pdfmeld.exe @mylist.dat fileout.pdf`". The file should contain a list of PDF files, one entry per line in the file. Or you can use a tag based file that contains a list of PDFs along with page numbers for each. See the [Input File](#) section for more information on the layout.

The input files may be either PDFs, images, MP3's, flash or movies (`.mov`, `.avi` or `.wmv` for example). For images, only use JPEGs at 72 DPI 256-color grayscale or 24-bit color, GIFs, or PNGs (alpha transparency in PNGs is not supported - the image will show without transparency). Some TIFF images may work as well. You may also use simple text files (`.txt`, `.log` and `.dat`)

Options

though no word wrapping is performed so text may be compressed if the file contains long lines. Microsoft Word, Excel, and PowerPoint documents may work as well under Windows if you have the free Microsoft Office "save as PDF" add-in installed for these products. See Microsoft's website for download information. An installation of OpenOffice can also be used to convert Excel/Word to PDF.

Note you can also use files from the web if you have an active internet connection. For example: `pdfmeld.exe http://www.mysite.com/testfile.pdf::ext=pdf,file.pdf fileout.pdf`. Note the `::ext=pdf` at the end of the URL. This is not passed when retrieving the page but is used to tell PDF Meld what type the file is and that it should retrieve it from the Web. For example, you might create a PDF from a cgi script in which case your URL may look like `http://www.mysite.com/testfile.cgi` along with optional parameters. In this case use: `pdfmeld.exe http://www.mysite.com/testfile.cgi?img=1::ext=pdf`. If you're using an image or other type of file, be sure to set `ext=` to the file type you're expecting back (like `jpg` or `txt`). You can use this on any of the parameters that accept a file name.

Pass the JSON formatted service credentials using `-gdrivejson` or `setGDriveJSON` to download a file from Google Drive. For a Google document/worksheet, use the format `"https://www.googleapis.com/drive/v3/files/(fileid)/export?mimeType=application/pdf::ext=pdf"` as your file name. For a PDF saved in Google Drive, use the format `"https://www.googleapis.com/drive/v3/files/(fileid)::ext=pdf"`. Replace `(fileid)` with the id of the file to download. For example, `"https://www.googleapis.com/drive/v3/files/ABC12345XYZ::ext=pdf"`.

Some of the options take comma separated values or text strings as input. Place quotes around text for options when you need to include spaces. For example, `-title "This is my title"` or `-pages "2, 3, 15, 22"`. Or, leave out any spaces when not using quotes such as `-pages 2,3,15,22`.

The default setting for commands that take a size or set of coordinates (such as `-pagesize`) are points. One point is 1/72 of an inch. To set `pagesize` in points for 8.5 by 11 inches you'd use `-pagesize 612,792`. You may override the default unit setting by using the `-units` command. Set the units command first then any commands that follow will use that setting. For example, to use inches you can specify `-units in -pagesize 8.5,11`. Units may be set to `pt` for points (the default), `in` for inches, `cm` for centimeters or `mm` for millimeters.

Since there are many options and some you might want to use everytime, there is a `-cmds "filename"` option to hold them. Create a file with the options as you would enter them on the command line. Save the file and then use `-cmds` to reference this file. The options will be processed as if you keyed them on the command line.

Options

Use the `-pages` option to pull pages from an existing PDF. Only use one input file in this case (or two if using the `-overlay` option). Separate page numbers with a comma or use a `-` between numbers for a range. For example, `"pdfmeld.exe filein.pdf fileout.pdf -pages 2,7,14-20,35"` will extract pages 2, 7, 14 through 20 and page 35, a total of 10 pages, from `filein.pdf` and place them in `fileout.pdf`. Use negative numbers to refer to the position from the end of the page set. For example, `-1` for the last page or `-3` for the third to the last page. You may also use the words "odd" or "even" to pull all the odd or even pages. Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. The `-pages` option, when used along with the `-overlay` or `-repeat` options, applies only to the second input PDF.

Use the `-pageord` option to rearrange pages from an existing PDF. Only use one input file in this case. Separate page numbers with a comma or use a `-` between numbers for a range. For example, `"pdfmeld.exe filein.pdf fileout.pdf -pageord 1-5,7,15,2"` will extract pages 1 through 5, 7, 15 and then repeat page 2. You may also specify a file in place of the page numbers. The file should contain comma or dash separated page numbers and may contain page numbers on multiple lines.

Use `-pageord "reverse"` to reverse the page ordering from an existing PDF. Only use one input file in this case. In a 10 page PDF, for example, the output PDF will start with page 10 from the input PDF, page 2 will be page 9 and so on.

Use the `-right`, `-down`, `-scale` and `-center` options to change the position of content on each page of a PDF. Only use one input file in this case.

Use the `-angle`, `-rotate`, `-rotateabs` or `-orient` commands to affect rotation of contents or page.

Use the `-overlay` option to overlay pages from one PDF onto a second PDF. Only use two input files in this case. The first PDF specified is the background and the second will be placed on top of the first. The resulting PDF will have the same number of pages as the second PDF. The first PDF should have the same or fewer pages than the second PDF. Both PDFs should have the same page sizes. For example, `"pdfmeld.exe filein1.pdf,filein2.pdf fileout.pdf -overlay"` will take page 1 of `filein1.pdf` and place page 1 of `filein2.pdf` on top and this becomes page 1 of `fileout.pdf`. The same process is applied to the remaining pages of the two input PDFs. You can overlay more PDFs by re-running the overlay process using the output PDF as an input PDF along with a new input PDF if necessary. Use `-reoverlay` to perform a reverse overlay - works similar to `-overlay` except the second PDF is treated as a background. The `-transparency` option can be used to allow the background page to show through on the foreground and create other effects based on the mode used.

Options

The `-repeat` option can be used to overlay PDFs as well. This option will repeat through the first PDF, using it as a background, for as many pages as there are in the second PDF. The resulting PDF will have the same number of pages as the second PDF. The first PDF should have the same or fewer pages than the second PDF. Both PDFs should have the same page sizes. The resulting PDF from using a background PDF with 1 page and a foreground PDF with 5 pages will be 5 pages and contain the single page background PDF on each page with the second PDF ovetop.

Some PDFs, even though they view right side up in the viewer, have their contents rotated as well as the page itself rotated. When you overlay this type of PDF with another that has a different rotation (or no rotation) the result is the page contents from one of the PDFs are rotated sideways or upside down. The `-autorotate` option (used along with `-overlay` or `-reoverlay`) will attempt to compensate for this. Use this option in situations where the result of the overlay is not correct due to a rotation issue. Both PDFs should have the same page size when using this option. If this option still doesn't provide the desired results, you'll need to perform this operation manually in 2 steps by forcing a rotation on one of the PDFs then perform the overlay. That means you'll use the `-angle`, `-right` and/or `-down` options on one of the PDFs and output to a new PDF. Then take this new PDF and use the `-overlay` or `-reoverlay` option with the background (or foreground) PDF.

The `-strin` option can be used to add text to a PDF. You can specify the placement, font size, pages to show on, color and alignment. See the [Text Files](#) section for more information.

The `-textout` and `-textin` options can be used for minor text touch-ups. The `-textout` option will export the text of the PDF to the output file. The text is written in the order it was found and will probably not be ordered the way you see it in the viewer. Some PDFs contain an image only for the page contents so there will be no text to modify in this case. Also, words may be split up across lines as some PDF creation software uses precise positioning for various parts of a word. The output text file may then be modified with text changes. Simply overwrite the text with whatever you want for the new text. Be careful not to add or delete lines from the file as the order the text lines appears in the file will be used to check for changes. The `-json` may be used to create JSON formatted output instead. With JSON you only need to include the sections changed as an array of structures containing the id and text properties. Use the `-textin` option to load in the changes and create a new PDF. Only use one input file in this case.

Here's an example exporting text:

```
pdfmeld.exe filein.pdf fileout.txt -textout
```

Then load in the changes with:

```
pdfmeld.exe filein.pdf fileout.pdf -textin fileout.txt
```

Options

You may add a FONT tag to the input text file to set text [color](#). The format is . For example, use or . Place these tags in front of the text (on the same line - do not add any line breaks) you want to color.

The -textdetailin and -textdetailout options can be used to locate and extract text from a PDF. For example, you can create a template of where to look for text on a page to pull out name/address information in CSV or your own customized ASCII format. See the [Text Detail Files](#) section for more information.

The -imgout and -imgin options can be used for image modifications or addition of images. This works similar to the -textout and -textin options above. The -imgout option will export a list of image IDs used in the PDF to the output file. The images are written in the order they are found and will probably not be ordered the way you see them in the viewer. The output contains a set of IMG tags with the height and width and image ID as options. For example, . You may also use -allimg to replace all images in a PDF with a single image that you specify.

Modify an image by adding a comma at the end of the line and type in the local path and image file you want to use. Add an image by adding a new IMG tag. Use the -imgin option to load in the changes and create a new PDF. Use the -addback option to place added images behind the current page contents. Only use one input file in this case. Only use JPEGs at 72 DPI 256-color grayscale or 24-bit color, GIFs, or PNGs (alpha transparency in PNGs is not supported - the image will show without transparency). See the [Image Files](#) section for more details.

Use the -imgdir option to export images from the PDF to the output directory specified. Not all images will be able to be extracted. Images are not overwritten (unless -force is used) when re-running the program with the same directory specified. Images are named after the image ID used in the PDF or the PDF file name (based on the -imgname option). The images might be a mix of JPEG, TIFF and U3D depending on how they are stored in the PDF. For example:

```
pdfmeld.exe filein.pdf c:\images\ -imgdir
```

Will place images that could be extracted from filein.pdf in the c:\images directory.

The -rectin option can be used to specify an input file with filled rectangles to draw on the page. A filled rectangle can be used to block out sections of a page when set to the page background color. Any text/images the rectangle covers remains in the document though it is blocked from view. See the [Rectangle File](#) for details on the format of the file.

Options

The `-clip` option can be used to display selected sections of a page. It may be used multiple times on the command line, each time with a different region of the page. Each section specified will be placed on a separate page in the output PDF. If you use `-clip` 3 times, for example, you'll have an output PDF containing 3 times the number of pages as the input PDF. The first 3 pages will be each of the clipped areas from page one of the input PDF. The next 3 pages will be each of the clipped areas from page two of the input PDF and so on.

The `-grid` and related options can be used to display grid markings on page. This can make it easier to determine positioning for commands such as `-clip` or `-fields` that rely on coordinate positions.

The `-twopage` option can be used to place pages from two different (or the same) PDFs next to each other on one sheet. May want to use the `-twoskip` option as well if both input PDFs are the same PDF to alternate pages. Set the page size if desired or use the `-rotate` option in addition.

The `-fields` option is used to add fillable fields to an existing PDF file. See the [Fillable Fields](#) section for detailed information.

The `-fdfout` option is used to create an FDF (Forms Data Format) file. This file can be used to save data for a form. Use this option with a PDF containing fillable fields to create a file you then edit and save with the information you want filled in. The output file should be one with an extension of `.fdf` instead of `.pdf`. Only use one input file in this case. The fields names in the form are placed in a specific format which should not be altered. Only modify the values for the fields. Here's an example FDF file with 3 fields:

```
%FDF-1.2
1 0 obj <<
  /FDF <<
    /Fields
    [
      << /T (First_Name) /V (Bob) /FT /Txt >>
      << /T (Middle_Name) /V ( ) /FT Txt >>
      << /T (Last_Name) /V (Smith) /FT /Txt >>
    ]
  /F (sample.pdf)
  >>
  >>
endobj
trailer
<< /Root 1 0 R >>
%%EOF
```

The file `sample.fdf`, when opened in Reader, will load in `sample.pdf` and display Bob in the `First_Name` field and Smith in the `Last_Name` field. The `/T (...)` is the name of the field as found in the original PDF. Place your data inside the parenthesis after the `/V`. Use a slash in front of any parenthesis

Options

that is part of your data. For example, /V (my \data\)). The /FT is the type of field. The value for it will be /Txt for text, /Btn for button (checkbox or radio), /Ch for choice or /Sig for signature. You may change the file name at the end of this structure, if necessary. It may be a PDF on your local drive or one on the internet (ex. <http://www.someserver.com/file.pdf>). Open the FDF file (not the PDF) with Reader and the data will show up in the PDF.

The -fdfin option can be used to load in the data from an FDF or XFDF file and save it in the output PDF. This is similar to opening the FDF file but the data will be saved in the output PDF in this case. You may also leave the fields open or mark them as read-only in the output PDF with the -fdffixed option. There's also the -flatten and -flattenobox options that will remove the fields from the PDF and retain the field values as text.

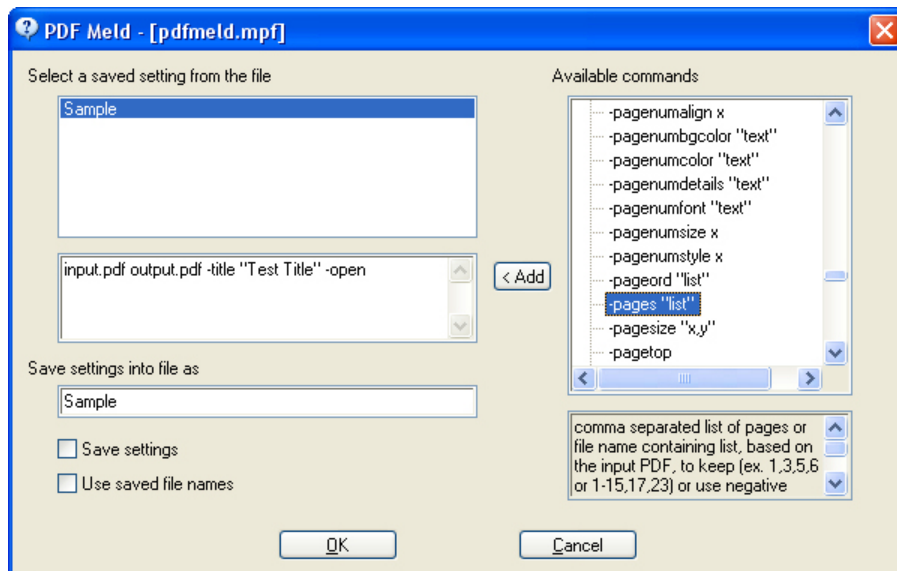
The -fdffield option is another way to pass field data in. In this case, pass one or more name/value pairs (in the form "<< /T (field name) /V (value) >>"). You may repeat this option on the command line with each name/value pair or pass all at once in one string. The text you pass for the option is appended to any prior text passed. See the [Digital Signature](#) section for details on signing PDFs when using -fdfin.

You may use the special keyword "nobuild" for the output file. This does not build an output PDF but lets you perform tasks such as -open, -print or -mail on the input PDF(s). For example: `pdfmeld.exe file1.pdf,file2.pdf nobuild -open` will open the existing PDFs file1.pdf and file2.pdf.

Options

You may store commonly used settings in a file for later use in the Windows executable version. File settings are saved as .mpf files. Running `pdfmeld.exe pdfmeld.mpf` or clicking on a .mpf file in Explorer will open the following dialog box. You may set any of the command line options, recall previously saved commands and make modifications.

The list of saved settings is shown on the left side. Click on an entry and the details of the parameters are shown below. You may modify the parameter details if necessary by typing in the box.



Choose a command from the list on the right side. A description for the command is shown below. Use the `< Add` button to append the command to the parameters on the left. You'll need to modify the sample parameter text once it's added to the parameter string. For example, if you add `-author "text"`, you'll likely want to change the word "text" to something meaningful.

The "Save settings" option will save any changes you make back to the file under the name you specify. Leave the options blank to delete the entry from the file.

The "Use saved file names" option is used to specify whether you want to use the file names from the saved setting or be prompted for a different input/output file. A new file open dialog box will open if you leave this unchecked and click OK. You may then select the input file. Afterwards, you'll be able to select the output file. You won't be prompted for any file names if you check "Use saved file names" (assuming you have file names entered as parameters).

You can also load and save settings using the `-prmload` and `-prmsave` options from the command line.

Options

A DOS batch file (.bat) or Unix shell script may also be used to save commonly run configurations. Place the static commands in the file and pass in any other options you need. For example:

```
c:\pdfmeld.exe -open -author "Test Author" %*
```

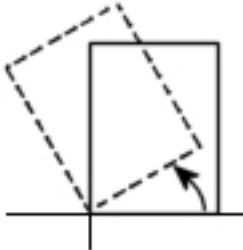
The above settings will auto open the output PDF and set the author to "Test Author". You may then run the script and pass in the file names or other parameters. Suppose the command above is in a file called myparms1.bat. You could run `myparms1.bat filein.pdf fileout.pdf -force -print`. That will auto open and print, set the author and not prompt for overwrite if the output file exists.

Options

The following options are available for the executable version of the program.

<u>Option</u>	<u>Description</u>
-addback	Places any added items such as page numbers, rectangles, grid lines, etc. in the background. Any added text/graphics are placed on top of existing content by default. Using this option does the reverse. Note that you might not be able to see the added items using this option if there is existing text/graphics covering them.
-aes 128 256	Sets AES encryption method. Pass 128 for 128-bit encryption or 256 for 256-bit encryption. Files encrypted with AES 128-bit encryption can only be opened with Acrobat or Adobe Reader 7.0 or above. Files encrypted with AES 256-bit encryption can only be opened with Acrobat or Adobe Reader 9.0 or above.
-allimg <i>file</i>	Pass in a JPEG or GIF image file name. All the images in the source PDF will be removed and replaced with the image specified. The output file size may be significantly reduced if the PDF was composed mostly of image data.
-allowdup	Tells PDF Meld to look for duplicate resources in the merged PDFs. Use this option when you have many PDFs that were generated from some application in a similar fashion to decrease the final output size. For example, suppose you have 100 PDFs to merge and each contains your company logo. By default, PDF Meld will simply append these PDFs together without checking for duplicate images or fonts. This option will instruct the program to search for matching image and fonts and use them only once in the output thus reducing the final PDF size. Note this may not work in all situations so you'll need to test first with any PDFs you intend to use this option with.

Options

<u>Option</u>	<u>Description</u>
<code>-altpages</code>	<p>Alternates the pages in a PDF. You may need to create the input PDF first if you have two PDFs you are starting with. For example, say you have two PDFs (called A and B) with 5 pages each and you want to sort the pages like this:</p> <ul style="list-style-type: none">page 1 = A page 1page 2 = B page 1page 3 = A page 2page 4 = B page 2 <p>and so on...</p> <p>In order to do this, first merge PDF A and PDF B into a new PDF. Next, run PDF Meld on this new PDF along with the <code>-altpages</code> option. Only use one input file with this option.</p>
<code>-angle <i>number</i></code>	<p>The angle to rotate the page contents. Valid range is from 0 to 360. This differs from <code>-rotate</code> in that the page width and length do not change, however the contents rotate about the bottom left corner.</p>
	
	<p>Use <code>-center</code> to rotate about the center of the page. Or, use the <code>-right</code> and/or <code>-down</code> options to put the contents (or portion you want) into view. For example, using <code>-angle 180</code> on a page 8.5 inches by 11 inches will require setting <code>-right</code> to 612 ($8.5 * 72 = 612$) and <code>-down</code> to -792 to view the contents. Only use one input file with this option.</p>
<code>-annotes <i>file</i></code>	<p>File to use for annotations. See the Annotations section for layout details.</p>
<code>-annotesout</code>	<p>Exports note annotations to the output file. In this case, use a file with a <code>.txt</code> extension rather than <code>.pdf</code> for the output.</p>

Options

<u>Option</u>	<u>Description</u>
-artbox " <i>x1,y1,x2,y2</i> "	The rectangle defining the extent of the page's meaningful content as intended by the page's creator. The values are in points (1/72 of an inch) or the unit setting. Default is the crop box setting.
-attribute <i>name,value</i>	Allows you to set your own user defined attributes. May include this option multiple times to assign more than one name/value pair if needed. These attributes are included in the document information object (where document title, subject, keywords, etc. are stored). For example, -attribute "Source,Radio Ad".
-author <i>author</i>	Sets the document author.
-autoclip	Used along with -pagesize. This option will clip the edges of the page for those pages that are smaller than the output page size. For example, if an input page size is 4x6 and the new size is 6x8 and there is no scaling, this option will clip one inch on the top, bottom, left and right. The page size doesn't change but only the 4x6 area of the page shows in the middle of the new 6x8 page. This allows you trim out any printer marks or other extraneous markings.
-autorotate	Used to compensate for page rotation during the overlay process. Use this option when the result of your overlay is one of the PDFs comes out sideways or upside down. This is due to the way the PDF was created and, even though it appears right side up in the viewer, actually has its contents and page rotated. This option will attempt to correct that issue.

Options

<u>Option</u>	<u>Description</u>
-autoscale <i>text</i>	<p>For use with the -pagesize option. Use "D" to scale larger pages down to fit within the new page size. Either the original width or height must be larger than the new page size to scale down. Use "U" to scale smaller pages up to fit within the new page size. Both the original width and height must be smaller than the new page size to scale up. Use "UD" to do both types of scaling.</p> <p>You may also add the letter "L" when using "D". This will be used to scale to the larger of the x or y ratios. The default is to scale larger pages down by the smaller of the ratios.</p> <p>Use an "O" along with "D" and/or "U" to preserve the original orientation but rotate by 90 degrees for those pages oriented differently from the page size being set. Use a "K" along with "D" and/or "U" to preserve the original orientation but do not rotate for those pages oriented differently from the page size being set.</p>
-barthin	Use a thin progress bar in the status dialog box.
-batch <i>text</i>	Used to treat the list of input files from a directory as a set you want to process individually. Normally, the list is operated on all at once - that is, the PDFs are linked together to form a single PDF or two files are specified for overlay. The parameter for this option specifies the details for the input and output file. See the Batch Processing section for details.

Options

<u>Option</u>	<u>Description</u>
-bkglayer <i>text</i>	Used to denote the first (or second when -reoverlay is used) PDF as a separate layer. Supply a name for the background that the user will see in the viewer. Users can selectively turn the background on or off. Requires Acrobat or Reader 6.0 or higher to use. Users of earlier versions of Acrobat or Reader will not be able to turn off the background.
-bleedbox " <i>x1,y1,x2,y2</i> "	The rectangle defining the clipped (cropped) area to display or print in a production environment. The values are in points (1/72 of an inch) or the unit setting. Default is the crop box setting.
-bmadd <i>text</i>	Creates a bookmark for any file being merged that does not have a bookmark structure. Pass in "file" to use the PDF file name for the bookmark description or "title" to use the title from the PDF.
-bmannote	Creates bookmarks based on the text annotations found in the PDF.
-bmauto <i>level</i>	Determines bookmarks based on text in the PDF by examining text font sizes. The level is 1 or higher and is the maximum level of bookmark to export. The higher the number the more entries will potentially be created. The results are written to the output file using the layout for the -bmin option (see the Bookmarks section for layout details). There are many different ways a PDF may have its text structured so this option will not work on every PDF. Even when the option does work it will likely only be close and the output will require some manual editing. Usually there will be missing spaces between words or words broken across several bookmark entries. You can load the bookmarks into the PDF by using the -bmin option on this file when you're done editing.

Options

<u>Option</u>	<u>Description</u>
-bmin <i>file</i>	File to use for bookmark structure. See the Bookmarks section for layout details.
-bmkeep	Retains the bookmark structure from appended PDFs without adding the file name as an upper level. By default, each appended PDF will have a bookmark named as the file name (or title if -bmtitle is used). This option prevents this upper level bookmark from being created and leaves the bookmarks as they appear in the original. You may also set this when you want to keep bookmarks while doing an overlay. Only one PDF should have bookmarks when using an overlay with this option.
-bmkeepnofit	Attempts to remove the automatic zoom scaling performed by existing bookmarks, if they happen to use it. This option also sets -bmkeep.
-bmout	Exports the current bookmark structure to the output file. In this case, use a file with a .txt extension rather than .pdf for the output.
-bmpattern <i>file</i>	Split a PDF into multiple PDFs based on regex pattern matching. The file should contain one or more <PATTERN> blocks with the text of what to match inside. For example: <PATTERN> Form K-1* Statement* </PATTERN> Will match on the pages of a PDF where the first page has a bookmark that starts with "Form K-1". If there next bookmark(s) start with "Statement" then those will be included. You may include multiple PATTERN blocks to match other bookmarks as well, which each page range of matches exported to a new PDF. The output should be a directory to store the PDFs.

Options

<u>Option</u>	<u>Description</u>
<code>-bmsplit level[,level]</code>	Split the original PDF into multiple PDFs based on the bookmark level specified. The topmost level is 1, the next level down is 2 and so on. Pass a 0 to split at all levels. The output PDFs are named after the bookmark text. Optionally include a second level to start the bookmark text from. For example, if the second level is a unique account number and level three is the same static text but you need to split on level three, use 3,2. This will combine the account number from level two with the static text in level three making the output file name unique. The output file specified on the command line must be a directory name and not a file name. The bookmarks should be structured in the input PDF so they all go from lower to higher page number. The output will not be correct if the bookmarks jump around in the document. Only use one input file with this option. Use -burstquick with this option if you have a large PDF and want to speed up the process. The bookmark is not retained in the output PDF in this case.
<code>-bmtitle</code>	Use the title of each document as the bookmark. One bookmark is created for each document in this case. Only works when merging PDFs one after another - not for overlays. The file name is used if the document title is not set.
<code>-bottomtop</code>	Places two pages from the input PDF(s) on one sheet (overlap will occur if the input page size is more than half the size of the output page size). Page 1 goes on bottom and page 2 goes on top when using a single PDF. Page 1 from the first PDF goes on bottom and page 1 from the second PDF goes on the top when using two PDFs. Set the new page size with the <code>-pagesize</code> option. May need to use <code>-scale</code> as well (especially if the input page size is more than half the size of your output page).

Options

<u>Option</u>	<u>Description</u>
-burst <i>text[,group]</i>	Creates one new PDF for each page in the input PDF. Pass in a file name prefix to use or leave blank to use the input file name. The output file specified on the command line must be a directory name and not a file name. The output files will have the suffix <i>_#</i> appended to the name where <i>#</i> is the page number. The page number will be prefixed with 0's if necessary so all PDFs will be the same length for the name. For example, if -burst "" is used and the input file is called mypdf.pdf containing 145 pages then the output directory will contain mypdf_001.pdf, mypdf_002.pdf and so on through mypdf_145.pdf. The group option is used to specify the page grouping. By default, one new PDF is created for each page. The group option takes 3 values - a start, through, and a skip value. For instance, 1-2 means take pages 1 and 2 and create a new PDF, followed by pages 3 and 4, and so on. Optionally use a skip value after the range such as 1-2;4. The skip value is added to the first page when starting the second set. For example, 1-2;4 means pages 1 & 2, then 5 & 6, and so on. Using 3-5;8 means pages 3, 4, and 5 in one PDF then pages 11, 12 and 13 in the next and so on. Only use one input file with this option.
-burstquick	Use along with -burst, -bmsplit, or -textsplit to specify a faster method of bursting. The -comp15 option is not supported when using -burstquick.
-center	Used along with the -scale, -angle or -clip options. Specifies that the page contents should remain centered on the page. When scaling less than 100%, the page contents will generally migrate towards the lower left corner without this option and without -right or -down. This option overrides any -right or -down value. May not work on all PDFs.

Options

<u>Option</u>	<u>Description</u>
-certfile <i>file</i>	<p>The path and file name to store the signing certificate to as an FDF file. Use an extension of .fdf so the file will be opened with Acrobat or Reader on Windows systems. This is used to provide a file to end users they can load into Reader in order to validate signatures from other PDFs you create signed using the certificate. This option is used along with the -signssl and other related options at the time of signing. See the Digital Signature section for details.</p>
-clip " <i>x,y,w,h</i> "	<p>Used to set up a clipping rectangle on the page. Only the contents which fall inside the rectangle are rendered. The values are specified in points (1/72 of an inch) or the unit setting. The x,y position 0,0 is the bottom left corner of the page. Values for w and h set the width (to the right of x) and the height (above y). For example, set to 234, 324, 144, 144 to display only the middle 2 inches of an 8.5 by 11 page. Only use one input file with this option.</p> <p>You may use this command several times to create a new page for each clip region in the original file. For example, using this command twice will create a PDF with twice the pages as the original. The first page of the new PDF will be the first clipped area of page one from the input PDF. The second page of the new PDF will be the second clipped area of page one from the input PDF and so on. You could then take a PDF with two or three columns of text throughout and create a new PDF with each column on its own page (using the -center, -rotate and -scale commands as well).</p>

Options

<u>Option</u>	<u>Description</u>
<code>-cmds <i>path-file</i></code>	The path and file name of a text file containing additional options to apply. For example, you might always use <code>-force</code> and <code>-open</code> . Place the text <code>"-force -open"</code> in a file called <code>"myopts.txt"</code> and reference it by using <code>-cmds "c:\pdf\myopts.txt"</code> . For Windows, you may place <code>pdfmeld.conf</code> and/or <code>fytek.conf</code> in the <code>%userprofile%\appdata\roaming\fytek</code> directory with the commands and they will be processed similar to using <code>-cmds</code> . For Linux, use <code>.pdfmeld.conf</code> and/or <code>.fytek.conf</code> (note <code>"."</code> preceding file name) for the file name. The <code>pdfmeld.conf</code> file is for PDF Meld specific commands and <code>fytek.conf</code> for any FyTek product with this feature.
<code>-colorin <i>file</i></code>	Loads in any changes made to the file created with the <code>-colorout</code> option. Include a leading 0 in front of any decimal number - i.e. use 0.25, not .25. You can use this feature to change colors or change from one format (RGB) to another (CMYK). Note that the format of the colors are in PDF syntax and no checks are made to see if the changes are valid.

Options

	<u>Option</u>	<u>Description</u>
	-colorout	Exports text and line drawing colors used in the PDF. In this case, use a file with a .txt extension rather than .pdf for the output. Each unique color is on a separate line. Each line contains a tag with the old and new color - both set to the same value. Colors are represented by 1 to 4 numbers followed by a code of lower or upper case G, RG, K, SC or SCN. Colors for borders and backgrounds in form fields are represented in brackets with BC or BG in front. You can modify the repeated (second) color in the list then load in the changes with -colorin. Note that the format of the colors are in PDF syntax. The numbers will range from 0 to 1 with a leading 0 in front of any decimal point. G and RG are for RGB colors (1 or 3 numbers), K is for CMYK (4 numbers) and SC and SCN are special color operators. See the Color Fields section for information on the file layout.

Options

<u>Option</u>	<u>Description</u>
-common <i>"text"</i>	Checks for common objects used among a set of similar PDFs and includes those objects only once. For example, if you have 100 PDFs you wish to merge and all contain your company logo then there is no need to include the logo 100 times. This option instructs the program to search for those common objects when building the output and only include them once, thus reducing the output file size. Pass "S" to look for redundant objects in the same PDF. You may also pass the level of checking for matching as L=low, M=medium (default), or H=high. The higher the level, the smaller the output PDF might be (depending on the PDFs involved, there may be no difference with the options). Though at higher settings, more objects may appear to match between the PDFs when they really are quite different. For example, use -common "" or -common "M" for medium level. To also match within the same PDF you might use -common "M,S".
-commondir <i>text</i>	Pass in a common directory to use for various input files. If an input file is not found in the current directory, this directory is used to search for the file. Separate multiple entries with a semi-colon.
-comp15	Uses a compression algorithm compatible with PDF 1.5 (Acrobat 6.0). PDFs with this form of compression can be viewed only with Acrobat or Reader version 6 or higher. The reduction in size is based on the number and type of objects in the PDF but in general is around 10-20%. Not all PDFs will be reduced by the same percentage factor.
-copies <i>number</i>	Number of copies to print when using the -print or -printer commands. Default is 1.

Options

<u>Option</u>	<u>Description</u>
-creationdate <i>YYYYMMDDHHmmSS</i> or <i>today</i>	Sets the document creation date. Provide a date in the format shown or use the word "today" to use the current system date/time. For example, to set to January 15th, 2002 at 4:15:30 PM, enter 20020115161530.
-creator <i>creator</i>	Sets the document creator. Note that Creator is labeled as Application under some versions of Adobe Reader.
-cropbox " <i>x1,y1,x2,y2</i> "	The rectangle defining the clipped (cropped) area to display or print. The values are in points (1/72 of an inch) or the unit setting. Default is the media box setting.
-cwd <i>text</i>	Changes the working directory to the specified directory.
-data	(Not available in SE version) Specifies the input file is a tag based data file containing FDF information. See the Data section for more information on file layout and usage.
-delblank <i>number</i>	Pass in the minimum number of characters a page should have to be considered not blank. Pages with fewer than this number will be removed from the output PDF.
-diropen <i>text</i>	The default directory for the file open dialog box. This dialog box is used when no input files were passed.
-dirsave <i>text</i>	The default directory for the file save dialog box. This dialog box is used when no output file was passed.

Options

<u>Option</u>	<u>Description</u>
<code>-docaction <i>action,JavaScript</i></code>	<p>Specify a document action and the JavaScript to execute for that action. The JavaScript may be passed directly or pass the name of a file containing the script (include the path) preceded by an ampersand.</p> <p>An action is one of the follow codes: WC - Will Close - executed when the document is about to close. WS - Will Save - executed when the document is about to be saved. DS - Did Save - executed when the document has been saved. WP - Will Print - executed when the document is about to be printed. DP - Did Print - executed after the document has been printed.</p> <p>You may specify this option multiple times to specify two or more commands. For example, if you want to perform an action before and after printing you might use something like this: <code>-docaction WP,"app.alert({ cMsg: 'about to print'});" -docaction DP,"app.alert({ cMsg: 'form has been printed'});"</code></p>
<code>-down <i>number</i></code>	<p>The distance to move the contents of each page down. The number is in units of 1/72 of an inch or the unit setting. May be positive or negative value. May not work on all PDFs. Only use one input file with this option.</p>
<code>-e <i>path-file</i></code>	<p>Sets the error/result file. Use this option to check if any errors were encountered. This file will contain the word OK if the new PDF was created.</p>
<code>-e128</code>	<p>Sets RC4 128-bit encryption method. Files encrypted with 128-bit encryption can only be opened with Acrobat or Adobe Reader 5.0 or above. The default encryption is 40-bit which works with Acrobat and Adobe Reader 4.0 and above.</p>

Options

<u>Option</u>	<u>Description</u>
-embed <i>file</i>	<p>(Not available in SE version)</p> <p>A file containing a list of files to embed in the output PDF. These can be Excel, Word, PDF or any other type of files. End users may extract and save the files from the PDF or open them in the appropriate application. See the Embedded Files section for information on the file structure. You may also pass in the tags rather than a file name. For example, -embed "<EMBED SRC='c:\myexcel.xls' DESCR='Financial Data'>".</p>
-embeddir	<p>Exports embedded files from the PDF. In this case, specify a directory as the output file (the second parameter to the program).</p>
-exclude	<p>Specifies the values entered for -pages are the pages to exclude rather than include.</p>
-expire YYYYMMDD	<p>Pass in the year, month and day the document should expire. Use 2 digits for both the month and day (i.e. 06 for 6). When the date is reached (based on the current date of the computer where the PDF is opened) the message set using -expiremsg is displayed and the PDF is closed. Note the user can still modify their system clock or uncheck the execute JavaScript preference in Reader to get around the message and thus access the PDF. This is intended as a way to restrict access for most casual users of Adobe Reader only.</p>

Options

<u>Option</u>	<u>Description</u>
<code>-expiremsg "msg title icon"</code>	<p>Used to display a message in a dialog box when the output PDF is expired. Enter the message and, optionally, the dialog box title and an icon. For example, <code>-msg "Test message"</code> displays the words "Test Message" in a dialog box with a default title. The values for icon are:</p> <ul style="list-style-type: none">0 - Error1 - Warning2 - Question3 - Status <p>To set the title and icon, use <code>-msg "Test message Test title 1"</code>.</p>
<code>-fdffield <i>text</i></code>	<p>Set to the name/value pair using the layout for the FDF file. The format is "<code><< /T (field name) /V (value) >></code>". You may repeat this option on the command line with each name/value pair or pass all at once in one string. The text you pass for the option is appended to any prior text passed. The output PDF will contain the modified field values.</p>
<code>-fdffixed</code>	<p>Mark all the form fields in the output PDF as read-only. This option may only be used along with the <code>-fdfin</code> option.</p>
<code>-fdfin <i>file</i></code>	<p>Set to an FDF or XFDF file you want to use with a PDF containing the corresponding form fields. It's not necessary to have every field in the FDF file. Those not in the file will retain their current setting. The output PDF will contain the modified field values. See the Multiple FDF section for information on merging a group of FDF files. Use the <code>-update</code> option if you need to maintain document rights on a rights-enabled PDF.</p>
<code>-fdfout</code>	<p>Creates an FDF file used to store data for populating a PDF containing fillable fields. All fields, even blank, are exported. In this case, use a file with a <code>.fdf</code> extension rather than <code>.pdf</code> for the output. Edit the FDF file with the data you want to show up in the PDF. Open the FDF file in Reader.</p>

Options

<u>Option</u>	<u>Description</u>
<code>-fdfoutflds text</code>	Used to specify a list of existing field names to export with <code>-fdfout</code> . Separate the list with a comma (be careful not to leave a space before or after the comma). Use an asterisk (the <code>*</code> character) in the string to match any number of characters. For example, <code>c.*</code> to exclude any fields that start with "c.". The match is case-sensitive when using the wildcard. Place an '@' in front of the value if it is instead a file containing a list of fields (one field per line).
<code>-fdfoutfldsrev</code>	Reverses the meaning of <code>-fdfoutflds</code> (and <code>-fdfouttypes</code>). When specified, the list of fields or field types listed in <code>-fdfoutflds</code> and <code>-fdfouttypes</code> are the ones to keep rather than discard.
<code>-fdfoutjson</code>	Similar to <code>-fdfout</code> , creates a file of the fields and values in the PDF in JSON format.
<code>-fdfouttypes text</code>	Used to specify a list of existing field types (such as text or button) to export with <code>-fdfout</code> . Separate the list with a comma (be careful not to leave a space before or after the comma). Place an '@' in front of the value if it is instead a file containing a list of fields (one field per line). The values to use are: Tx - for text fields Ch - for choice (drop-down list) fields Btn - for button fields
<code>-fdfoutxml</code>	Similar to <code>-fdfout</code> , creates a file of the fields and values in the PDF in XML format.
<code>-fdfskipflds text</code>	Used to specify a list of existing field names to skip or leave out with <code>-fdfout</code> . Separate the list with a comma (be careful not to leave a space before or after the comma). Use an asterisk (the <code>*</code> character) in the string to match any number of characters. For example, <code>c.*</code> to exclude any fields that start with "c.". The match is case-sensitive when using the wildcard. Place an '@' in front of the value if it is instead a file containing a list of fields (one field per line).

Options

<u>Option</u>	<u>Description</u>
<code>-fieldbits <i>text</i></code>	<p>Pass a set of lists (pipe separated) where each list contains a field name followed by a comma then the list of bit flags to set (or clear) which are also comma separated. Any existing bits currently set are retained if they are not present in the set of flags to clear. Use an asterisk (the * character) for a field name to match any number of characters. For example, <code>c.*</code> to match any fields that start with "c.". The match is case-sensitive when using the wildcard. Example: <code>"c.*,3,26 d.*,-3"</code> means set bit flag 3 (no export) and bit flag 26 (richtext) for any field that starts with "c.". For any field that starts with "d.", clear bit flag 3.</p> <p>Place an '@' in front of the field if it is instead a file containing a list of fields and bits (one field per line). The file must contain a single field name on a line optionally followed by a comma with a list of bit flags to set or clear. If the field appears alone, the bit values will be taken from the command line. Also, values entered on the command line will override the file bit flag values. Example: <code>"@f1.dat @f2.dat,3"</code> means read the list of fields from <code>f1.dat</code> as well as the bit flags since none are specified on the command line. Then, read the list of fields from <code>f2.dat</code>, ignore the flag settings in the file and set bit flag 3 for those fields. See the PDF Reference Manual from Adobe for details on the various flags.</p>
<code>-fields <i>file</i></code>	<p>Set to a file containing fields to add to the PDF. See the Fillable Fields section for details on the format of the file.</p>
<code>-filesep <i>text</i></code>	<p>The file separator to use for the list of input files. The default is a comma if this option is not used. You may use more than one character. This option should be the first one passed as some other options may use this value. For example, <code>pdfmeld -filesep "-z-" "filein1.pdf-z-filein2.pdf" fileout.pdf</code>.</p>

Options

<u>Option</u>	<u>Description</u>
-flatten	Flattens the fields in the PDF. This removes the fields and replaces the field with its text value. The effect is the PDF now contains a standard text string where the field was originally. See the -flattenflds option to specify a list of fields to flatten.
-flattenfile <i>file1,file2,...</i>	Flattens the fields in the listed PDFs (names must match a subset of the input files). This removes the fields and replaces the field with its text value. The effect is the PDF now contains a standard text string where the field was originally. Use -flatten to flatten all fields in all of the PDFs.
-flattenflds <i>field1,field2,...</i>	Flattens the listed fields in the PDF. This removes the fields and replaces the field with its text value. The effect is the PDF now contains a standard text string where the field was originally. Place an '@' in front of the value if it is instead a file containing a list of fields (one field per line). Use -flatten to flatten all fields in the PDF.
-flattenfldsrev	Reverses the meaning of -flattenflds . The fields listed in -flattenflds are the the ones to keep when this option is used.
-flattenobox <i>text</i>	Same as -flatten except any drawing around the field, such as a rectangle, is removed. Pass "B" to remove borders, "F" to remove any background fill color, or blank to remove both borders and fill. Note this applies only to borders and background fills drawn as part of the widget. This option has no effect if the border or fill is instead part of the page contents.
-flattensig	Same as -flatten except any signature fields are not removed. This leaves the signature boxes alone for signing later.

Options

<u>Option</u>	<u>Description</u>
-fonts <i>file</i>	A tag based file with TrueType, OpenType or Type 1 fonts to embed. This is for use with the -pagenumfont option or to replace existing fonts. See the Font Files section for information on the file layout.
-force	Turns off the prompt to overwrite the output file if it already exists. Also, this option will allow overwrite of exported images (-imgdir option).
-gdrivedescr <i>text</i>	Optional. A short description of your PDF when saving to Google Drive.
-gdriveencfile <i>path-file</i>	Pass your Google generated JSON file with -gdrivejson and use this option to create an encrypted file of your JSON data. Only PDF Meld can read the contents of this file so you may use it in place of the plain text file used with -gdrivejson. You only need to use this option once with your plain text JSON file to generate your encrypted output.
-gdriveimp <i>text</i>	The user (email address) to impersonate as when using the Google Service. Note your Google Business Administrator must have already set up your Client ID as a trusted app in order to use this option.
-gdriveinsfolder (text)	Inserts (creates) a new folder in Google Drive to store the PDF into. The -gdriveparents in this case are used as the upper level folder for this one. Upon success the PDF is then stored in this folder.
-gdriveinsid <i>text</i>	The file ID for the upload. Only use this if you have pre-generated a file ID from your Google account.
-gdrivejson <i>path-file</i>	The path and name of the JSON formatted file containing your Google Drive service credentials. Pass this if you wish to save or retrieve a document in your Google Drive. Note you may leave this off if you simply want to OCR your document and just use -gdriveocrsave. See the Google Drive section for more details.

Options

<u>Option</u>	<u>Description</u>
-gdrive <code>log path-file</code>	The file to log the results returned from Google. This file is overwritten each time it is used. When used with -gdrive <code>json</code> it will contain the results from the Google Drive post in JSON format that include the file id and other useful information.
-gdrive <code>ocr</code>	Perform OCR (optical character recognition) when saving to your Google Drive account by using the -gdrive <code>json</code> option. Useful for PDFs that contain scanned images so you can retrieve the plain text.
-gdrive <code>ocrpdf text</code>	Pass "A" for annotation, or "T" for text, or "P" for PDF. Runs the OCR processing against the PDF to generate the text for each page. The text is then saved back to the PDF as hidden text or an annotation. The text won't line up to the text in the image on the page but can be used to search within the PDF. Using the "P" (PDF) option returns just the text as a PDF (without the original formatting and images).
-gdrive <code>ocrsave</code>	Saves various formats such as rtf, HTML and docx from your PDF output named after the base output file. For example, if your output is "myfile.pdf" then, if successful, you will also have files such as "myfile.pdf.html" and "myfile.pdf.docx". Note you may use this option by itself without any of the other -gdrive* options if you simply want to OCR your output PDF. Your PDF is still transmitted to Google for decoding so an internet connection is required.
-gdrive <code>parents text</code>	Optional. A comma separated list of parents (folders) to add the PDF into.
-gdrive <code>proxy text</code>	The proxy server, if needed, such as http://proxy.mycomp.com:9000 . If not sure, you can generally check using Internet Explorer under Tools Internet Options Content Lan Setting.

Options

<u>Option</u>	<u>Description</u>
-gdrivesave	Specify this option along with -gdrivejson to save the output PDF to your Google Drive. See the Google Drive section for more details.
-gdriveshareaccess <i>text</i>	The type of access to give - either "reader" or "writer". If not specified this defaults to "writer".
-gdriveshareemail <i>text</i>	The email address of the user or group to share the PDF with.
-gdrivesharetype <i>text</i>	The type is the type of user the email parameter refers to - "user", "group", "domain", "anyone". If not specified this defaults to "user".
-gdrivetitle <i>text</i>	The title of your PDF when saving to Google Drive.
-gdriveupdid <i>text</i>	The file ID for the upload. This overwrites an existing file saved in your Google Drive.
-gdrivewsllog <i>path-file[,text]</i>	The id of a Google worksheet to append a row to when PDF Meld creates a new PDF. Optionally include the message to use. If not specified a generic message like "PDF Meld Created File <filename>" is used.
-grid "x,y,z"	Used to draw grid markings on each page. The x and y values are in points (1/72 of an inch) or unit settings. If inches have been set with the -units command, for example, then setting -grid 1,1 will draw grid lines one inch apart. The third parameter is optional and is used to sub-divide the main grid blocks. Setting z to 3 will draw 2 lines (for 3 equal spaces) between each set of grid lines.
-gridcolor "c"	Sets the color for the grid lines. Valid values are r=red (default), g=green, b=blue, y=yellow, l=black and w=white.
-gridmargins "l,b,r,t"	Used to set margins for the grid lines. The values, in order, are for the left, bottom, right and top margins. Values are specified in points (1/72 of an inch) or unit settings. The grid is drawn using the entire page when margins are not set.

Options

<u>Option</u>	<u>Description</u>
-gridnum	Used to display grid numbers next to the grid lines.
-gridyrev	Reverses the Y grid direction from the default of bottom to top to be from top to bottom.
-groupend	Used to end a group session. See the Group Processing section for details.
-groupstart	Used to start a group session for PDF Meld. See the Group Processing section for details.
-guioff	Suppresses the dialog window that shows the current build progress. Note this is the default on Linux. Use -guion in Linux to turn on the dialog window. Not all non-Windows operating systems may support the graphical interface.
-guion	Turns on the graphical front end and message windows in Unix/Linux operating systems. Not all non-Windows operating systems may support the graphical interface. You may see a message that simply says "aborted" when attempting to use a graphical front end which means it will not run on that system.
-hidemenubar	Hides the menu bar in the PDF viewer when the output PDF is opened.
-hidetoolbar	Hides the tool bar in the PDF viewer when the output PDF is opened.
-hidewindowui	Hides the window interface elements in the PDF viewer when the output PDF is opened.
-imgbright <i>number</i>	Brightens or darkens images. Pass a number from -100 to 100. Numbers less than 0 darken while greater than 0 brighten. Will not work on all images in all types of PDFs.

Options

<u>Option</u>	<u>Description</u>
-imgcolor <i>text</i>	<p>Modifies images by reversing color or changing to grayscale or black and white. The parameters for this option are:</p> <ul style="list-style-type: none">GRAY - GrayscaleBW - Black & whiteREV - Reverses colors (X-Ray effect)REVGGRAY - Reversed grayscaleREVBW - Reversed black & white <p>Will not work on all images in all types of PDFs.</p>
-imgdecor "src,wl,wr,wt,wb[,tp[,tpm]]"	<p>Shorthand method of setting an image border (decoration) without using an image input file. Pass in the image file name and the border widths in points (1/72 of an inch) or the unit setting. The widths are in the order of left, right, top and bottom. You may optionally include a transparency percent value and mode after the widths. For example, "myimg.jpg,36,36,72,72" will use an image named myimg.jpg as a border for all pages. The border will be half an inch of the left and right and one inch on the top and bottom. Use "myimg.jpg,36,36,72,72,50" for a 50% transparency on the image border. See the Image Files section for more details on image input files.</p>
-imgdecorscale <i>number</i>	<p>The amount as a percent to scale the border image. For example, use 50 for 50%. Use with the -imgdecortile option.</p>
-imgdecortile	<p>Tiles the border image rather than stretching the image to fit. Use with the -imgdecor option.</p>
-imgdir	<p>Exports images from the PDF. In this case, specify a directory as the output file (the second parameter to the program). This option may not be able to extract all images depending on how they are stored in the PDF. The exported images will be JPEG, TIFF or U3D format depending on how they are stored in the PDF.</p>

Options

<u>Option</u>	<u>Description</u>
<code>-imgerr "pg,pct[,n]"</code>	<p>A PDF with a damaged image may display the error "insufficient data for image" in Reader when viewing the page with the damaged image. This option may help to fix the problem by reducing the height of the image and thus require less data. This will not work in all cases. Currently this affects all images on the given page so if you have good images on the same page they will be shortened as well. Pass in the page number (the "pg" variable) and the percentage to reduce the image height by. Try values between 95-99 at first as the lower the value the shorter the image. Continue to work your way down from there if 95 still is not working. A value of 0 for "pct" will remove all images from the page. Optionally, pass in the image number. This is simply the order of the images as Meld determines them on a page. If you have 3 images on a page and only one is damaged you'll need to experiment by passing different values for n (from 1 to 3 in this case) to determine which is the problem image.</p>
<code>-imgfaxbcolor "text"</code>	<p>Specifies a background color in RGB (Red-Green-Blue) format for scanned/fax type images. Has no effect on images that are not recognized as a black and white fax image. The valid range is from #000000 (black) to #FFFFFF (white). The value must include all 3 color components (3 hex values or 6 characters in length).</p>
<code>-imgfaxcolor "text"</code>	<p>Specifies a foreground color in RGB (Red-Green-Blue) format for scanned/fax type images. Has no effect on images that are not recognized as a black and white fax image. The valid range is from #000000 (black) to #FFFFFF (white). The value must include all 3 color components (3 hex values or 6 characters in length).</p>

Options

<u>Option</u>	<u>Description</u>
-imgin <i>file</i>	Loads in any changes made to the file created with the -imgout option. May not work on all PDFs. You may also pass in the tags rather than a file name. For example, -imgin "". See the Image Files section for more details.
-imgmargin " <i>x,y</i> "	Sets the page margin for input images in points (1/72 of an inch) or the unit setting. The x value is for the left/right margin and y is for the top/bottom margin. For example, use -imgmargin 36,36 for a half-inch margin all around (this is the default when this option is not used). This setting is used for images that are passed as input.
-imgname <i>text</i>	Use "file" to name exported images based on the PDF file name or "name" (default) to name images based on their image ID used in the PDF.
-imgonly " <i>text</i> "	Pulls the images from the PDF(s) and places one image per page in the output PDF. The "text" value can be used to place a text string below each image. Use "%f" as a variable in the string that will be replaced with the image ID. The -imgpagesize and -imgpagepos options can be used to determine the page size and placement.

Options

<u>Option</u>	<u>Description</u>
<code>-imgout ["pos"]</code>	<p>(Not available in SE version)</p> <p>Exports a list of images found in the PDF to the output file. In this case, use a file with a .txt extension rather than .pdf for the output. The images are written in the order they were found and not necessarily the order they appear on screen when viewing the PDF. You may then add an image replacement to any line in the output file. Use <code>-imgout "pos"</code> (this is fixed text, not a variable) to export image positioning as well. This involves extra processing so is not enabled by default. Use the <code>-imgin</code> option to load in your changes. May not work on all PDFs. Only use JPEGs at 72 DPI 256-color grayscale or 24-bit color, GIFs, or PNGs (alpha transparency in PNGs is not supported - the image will show without transparency). See the Image Files section for more details.</p>
<code>-imgpagepos "x,y"</code>	<p>Sets the page position for input images in points (1/72 of an inch) or the unit setting. For example, use <code>-imgpagepos 72,72</code> to place the lower left corner of the image 1 inch from the left page edge and 1 inch from the page bottom. Images are centered on the page by default. This setting is used for images that are passed as input or with the <code>-imgonly</code> option.</p>
<code>-imgpagesize "x,y"</code>	<p>Sets the page size for input images in points (1/72 of an inch) or the unit setting. For example, use <code>-imgpagesize 612,792</code> for a size of 8.5 by 11 inches when <code>-units</code> have not been set. This setting is used for images that are passed as input or with the <code>-imgonly</code> option.</p>
<code>-imgres number</code>	<p>Pass in the resolution in DPI for any exported TIFF images. The default is 300.</p>

Options

<u>Option</u>	<u>Description</u>
-imgscale "x"	Sets the scaling factor for images that are passed as input. The default is 100 for 100%. Pass a number less than 100 to shrink the image or greater than 100 to increase.
-impose "2-up"	Performs 2-up imposition. Currently only 2-up is supported so any other value than "2-up" will still result in 2-up. This may change in the future so use "2-up" for compatibility with later versions. Two pages from the input PDF are placed on a single page in the output PDF. In addition, the pages are ordered in such a way that they can be printed front and back then bound together in a booklet. For example, a PDF with 4 pages will be output with pages 1 and 4 on the first sheet and pages 2 and 3 together on the second. The output page size by default will have a width equal to the height of the input PDF and a height of double the width of the input PDF. A PDF that has a page size of 8.5 by 11 inches will output a PDF with a page size of 11 by 17 inches. Use only one input PDF with this option.
-in	Optional command to use in front of the input file(s). Running "pdfmeld a.pdf b.pdf" is the same as "pdfmeld -in a.pdf b.pdf". Also see -out .

Options

<u>Option</u>	<u>Description</u>
-infoout	<p>Exports file information for the PDF. This is a tag based file with information for each input file. The data is written to the output file (the second parameter to the program). For example:</p> <pre data-bbox="857 493 1235 806"><PDFFILE SRC="myfile.pdf" PAGES="3" BYTESIZE="78275" CREATED="2010-09-15-120530" Title="Document title" Author="Document author" Creator="Document creator" Producer="Document producer" Keywords="Document keywords"></pre> <p>The dates/times listed for CREATED, MODIFIED and ACCESSED are in year, month, day, hour, minute, second order. CreationDate and ModifiedDate parameters are in PDF syntax. Note that Creator is labeled as Application under some versions of Adobe Reader.</p>
-insertat <i>number</i>	<p>The starting page number to insert the second of two PDFs into the first PDF. For example:</p> <pre data-bbox="857 1178 1187 1205">f1.pdf,f2.pdf out.pdf -insertat 3</pre> <p>Will create an output PDF with the first 2 pages of f1.pdf followed by all of the pages of f2.pdf then the rest of f1.pdf starting from page 3. You may also use a negative number to mean the number of pages before the last page of first PDF. So -1 means all but the last page of the first PDF then all of the second PDF and finally the last page of the first PDF. Use only two input PDFs with this option. You may use the -pages1 and -pages2 options to limit the number of pages from each PDF.</p>

Options

<u>Option</u>	<u>Description</u>
<code>-insertbmfile <i>file</i></code> <code>-insertbmfileq <i>file</i></code>	<p>Use along with the -bmpattern option. Pass a PDF or image file to insert into the source PDF either before or after the bookmark specified with the <code>-bmpattern</code> file. Use <code>::before</code> after the file name to insert this file before the matching bookmark or <code>::after</code> (the default) to insert after. For example, assume you have a 5 page PDF with a bookmark "Section 2" and you want to place a 3 page PDF inside this PDF just before this bookmark. In the pattern file for <code>-bmpattern</code>, use something like this: For example: <PATTERN> Section 2* </PATTERN></p> <p>If your insert file is called <code>myinsert.pdf</code>, use <code>-insertbmfile "myinsert.pdf::before"</code> to place this PDF just before the bookmark. The output will contain 8 pages with the 3 page PDF inserted just before the page that bookmark "Section 2" references. Include -bmkeep to retain the bookmarks from the input PDF.</p> <p>The <code>-insertbmfileq</code> option (q for quick) works the same but uses a quicker method for building the PDF. It's recommended you use the <code>-insertbmfileq</code> option and if you have any issues then use the <code>-insertbmfile</code> option.</p>
<code>-json ["<i>details</i>"]</code>	<p>For use with <code>-textout</code> and <code>-textdetailout</code> to export in JSON format. Optionally include the string "<code>details</code>" to export x/y positioning and string fragments. This is necessary if you plan on modifying any text.</p>
<code>-keywords <i>keywords</i></code>	<p>Sets the document keywords.</p>
<code>-layersin <i>file</i></code>	<p>Loads in any changes made to the file created with the <code>-layersout</code> option.</p>

Options

<u>Option</u>	<u>Description</u>
-layersout	Exports layer information for the PDF. In this case, use a file with a .txt extension rather than .pdf for the output. Each layer is on a separate line. Each line contains a tag with the name and description. You can add a REMOVE option to the tag then load in the changes with -layersin. See the Layer Fields section for information on the file layout.
-log <i>file</i> or <i>file,a</i>	Use to create a log file containing the list of PDFs merged and the page number they start on. The following fields are exported, tab separated, on each line: <ol style="list-style-type: none">1. File name2. Document title3. Page #4. Current system date/time The date/time is formatted as YYYY-MM-DD-HH:MI:SS. The file name is used if the document title is missing or cannot be determined. The page number is the starting page number for that PDF in the output. Only the first PDF is listed in the log file if an overlay option was used. Use a ",a" after the file name to append to the log file rather than overwrite it. For example, -log "myfile.log,a".
-mail	Opens the user's e-mail program to a composition window with the newly created PDF attached. May not work with all e-mail programs. None of the other mail options (such as -mailsmtp) are necessary with this option. The default settings for to, cc and bcc addresses as well as the subject and body are taken from the options described below.
-mailauth <i>text</i>	The authentication protocol to use. The default is LOGIN but you may use NTLM. Only needed if the mail server requires authentication. (SMTP only)

Options

<u>Option</u>	<u>Description</u>
-mailauthid <i>text</i>	The user name used to log into the server. Only needed if the mail server requires authentication. (SMTP only)
-mailauthpwd <i>text</i>	The password for the user name used to log into the server. Only needed if the mail server requires authentication. (SMTP only)
-mailbcc <i>text</i>	The address(es) to BCC (blind carbon-copy) the email to. Must be an address in the form of name@somecompany.com. Separate multiple addresses with a comma. May also specify a file containing a list of addresses. Each address must be on a separate line by itself. Pass in the name of the file preceded by an ampersand. For example, -mailbcc @addr.txt.
-mailbody <i>text</i>	The body text of the email. Enclose in quotes. This may also be a file name. If so, the contents of the file will be used as the body. Use a \n for a new line when the body is entered using this option. You may also send HTML formatted body text. Put the <HTML> tag as the first line of the body text and it will be sent as HTML rather than plain text. Avoid using references to other local files in the HTML body, such as images, as they will not be sent with the message. You may use images with a web location as the source however.
-mailcc <i>text</i>	The address(es) to CC (carbon-copy) the email to. Must be an address in the form of name@somecompany.com. Separate multiple addresses with a comma. May also specify a file containing a list of addresses. Each address must be on a separate line by itself. Pass in the name of the file preceded by an ampersand. For example, -mailcc @addr.txt.
-maildebug <i>text</i>	The path and name of a file to use for debugging purposes. The data (email message) that is sent to the SMTP server will be logged here. (SMTP only)

Options

<u>Option</u>	<u>Description</u>
-mailfakecc <i>text</i>	The CC address to show for the email. The default is the CC address(es). (SMTP only)
-mailfakefrom <i>text</i>	The from address to show for the email. The default is the FROM address. (SMTP only)
-mailfaketo <i>text</i>	The to address to show for the email. The default is the TO address(es). (SMTP only)
-mailfiles <i>text</i>	A comma separated list of file names to include with the mailing. The path must be fully qualified for each file.
-mailfrom <i>text</i>	The from address for the email. Must be an address in the form of somename@mycompany.com. (SMTP only)
-maillog <i>text</i>	The name of a log file to use for date/time emails were sent as well as any errors. This is optional. (SMTP only)
-mailnodialog	Sends the email via MAPI without a composition window. The user may still receive a dialog box asking if it's OK to send the message on their behalf. The message is send via MAPI if they decide they want to send it. Use the -mailsmtp option instead to send the email via SMTP without user intervention.
-mailpri <i>text</i>	The message priority. Set to either HIGH or LOW. Leave this option off for normal priority. (SMTP only)
-mailreply <i>text</i>	The reply to address for the email. Must be an address in the form of somename@mycompany.com. The default is the FROM address. (SMTP only)
-mailretry <i>number</i>	The number of times to retry sending the email if an error is encountered. The default is 0, meaning try sending the email only once.

Options

<u>Option</u>	<u>Description</u>
-mailscr <i>file</i>	For Unix/Linux systems where -mail is not available. Specify a script that will receive as a parameter the output PDF file name. Create a script for your operating system that will be used to bring up an email window with an attached PDF.
-mailsmtp <i>text</i>	The SMTP server to use for sending the mail. Used to send the PDF via SMTP rather than interactively. For example, mail.yourdomain.com. Must also supply the -mailfrom and -mailto options. The -mailfrom must be a valid email account on the SMTP server. You may also pass the port if necessary after the server name or IP address by adding a colon followed by the port number. For example, 192.168.0.30:25.
-mailsub <i>text</i>	The subject of the email. Enclose in quotes.
-mailto <i>text</i>	The address(es) to send the email to. Must be an address in the form of name@somecompany.com. Separate multiple addresses with a comma. May also specify a file containing a list of addresses. Each address must be on a separate line by itself. Pass in the name of the file preceded by an ampersand. For example, -mailto @addr.txt.
-mediabox " <i>x1,y1,x2,y2</i> "	The rectangle defining the boundaries of the physical medium the page is intended to be displayed or printed. The values are in points (1/72 of an inch) or the unit setting. Overrides the -pagesize setting. See the Page Layout Files section for information on setting these values individually for pages in the PDF.
-mediaimg <i>file</i>	An image to use as the display before the media is activated.

Options

<u>Option</u>	<u>Description</u>
-mediaplay <i>text</i>	<p>Specifies when the media (such as an MP3 or movie file, when used as input instead of a PDF) is to be played. The options are:</p> <ul style="list-style-type: none">O When the page containing the media is openedV When the page containing the media is viewedM When the cursor moves into the media area <p>You may specify one or more options comma separated.</p>
-mediarepeat	<p>Use this option to repeat or loop the embedded media. For use when the input is an MP3 or movie file rather than a PDF.</p>
-moddate <i>YYYYMMDDHHmmSS</i> or <i>today</i>	<p>Sets the document modification date. Provide a date in the format shown or use the word "today" to use the current system date/time.</p>
-msg " <i>msg title icon</i> "	<p>Used to display a message in a dialog box when the output PDF is opened. Enter the message and, optionally, the dialog box title and an icon. For example, -msg "Test message" displays the words "Test Message" in a dialog box with a default title. The values for icon are:</p> <ul style="list-style-type: none">0 - Error1 - Warning2 - Question3 - Status <p>To set the title and icon, use -msg "Test message Test title 1".</p>
-ndkeep	<p>Keeps named destinations. Normally these are flattened, that is the name is removed and the destination from a link or bookmark is simply set to the page it should go to. Only use this when you don't have a naming conflict among the PDFs you're merging and it's important that the name is kept. Unless you're debugging links in the output PDF it's best to not use this option.</p>

Options

<u>Option</u>	<u>Description</u>
-noannotate	Disables add/change of form fields or annotations.
-noannotes	Specifies that annotations should be removed from the PDF(s).
-noapp	Suppresses the PDF entry for "NeedAppearances" that may be added from stamps which causes Acrobat to prompt to save changes when closing the PDF.
-noassemble	<i>(128-bit only)</i> Disables assembly (insert, rotate, delete pages or create bookmarks) when -nochange is used.
-nobm	Turns off pulling of bookmarks from source PDFs. This is set by default if -pages or -overlay is used.
-nochange	<i>(Not available in SE version)</i> Disables changes to the document.
-nocopy	<i>40-bit :</i> Disables copying of text and/or graphics from the document. <i>128-bit :</i> Disables copying of text and/or graphics from the document other than in support of accessibility to disabled users or for other purposes.
-nodigital	<i>(128-bit only)</i> Disables printing at digital quality - can only print low resolution. The -noprnt option overrides this option so you'll want to use -noprnt or -nodigital but not both.
-noerrmsg	Hides any pop-up error messages that would normally occur such as "cannot open output file". You should consider using the -e option in this case to write errors out to a log file.
-noextract	<i>(128-bit only)</i> Disables extraction of information in support of accessibility to disabled users or for other purposes.
-nofillin	<i>(128-bit only)</i> <i>(Not available in SE version)</i> Disables fill in interactive fields when -noannotate is used.

Options

<u>Option</u>	<u>Description</u>
-noglyphcomp n	Use along with -noglyphpos as this tag has no effect by itself. This entry can be used to specify a text width compression amount. If a font was replaced with something other than the original, the text in the PDF may be too narrow or too wide as to flow off the page. A value of 100 means 100% - no compression. Use lower values to squeeze the text so it is not as wide or a value greater than 100 to expand the width. This affects all text on the page and not just a particular section.
-noglyphpos	Removes glyph positioning from the text. This might be necessary if a PDF was damaged or modified in such a way that a font was replaced and the characters in words no longer line up.
-noimg	Specifies that images should be removed from the PDF(s). Line drawn images are not removed - only images that came from an external source such as a JPEG, TIFF, BMP, etc.
-nometaenc	When used with -e128 this option keeps the document metadata unencrypted in the output PDF. This allows other applications to read the metadata while the rest of the PDF is encrypted.
-noprint	Disables printing of the document (even low resolution). To create a PDF with both printing and copying disabled for the user you would run something similar to: <pre>pdfmeld.exe filein.pdf fileout.pdf -o abc123 -u xyz -noprint -nocopy</pre> The file could only be opened by someone who knows one of the two passwords (abc123 or xyz). Using a password of abc123 gives full access while using the password of xyz does not allow printing or copying of text.

Options

<u>Option</u>	<u>Description</u>
-noprintprefs	Do not pull printer preferences from the first input PDF. Some PDFs include default settings for printer options. These are settings that include the number of copies to print, duplex printing, page scaling, paper tray and page range. This option keeps the output PDF from inheriting those settings.
-norights	Turns off all rights (default is all are granted). Setting of options such as -noprint or -nocopy turns those rights on rather than off. Use this if you typically are turning off most or all of the rights. Note that setting -norights and -noprint will allow high resolution printing. Setting -norights and -nodigital will allow low resolution printing. Setting only -norights will disallow printing.
-notblank	Used with -fdfout, -fdfoutjson, -fdfoutxml to specify only fields with a non-blank value are to be exported.
-noxml	Removes the metadata XML (if any) from the output PDF.
-o <i>password</i>	Sets the owner password for the PDF. If not specified but the user password is, this is set to the user password. Also, when not specified, the owner has only the rights granted when the document was created. So for example, if -noprint was specified, then it is impossible for the owner to print the document.
-objin <i>file</i>	Loads the modified objects into the PDF.

Options

<u>Option</u>	<u>Description</u>
-objout [<i>text</i>]	<p>Un-compresses the streams in the PDF and writes all objects to the output file. You may then modify any object(s) and load in the changes with the -objin option. You should remove any unmodified objects as some may contain binary data that could become corrupt after saving from a text editor. Objects start with a pair of numbers followed by "obj" (without the quotes), the object itself, then ending with "endobj". This option is meant for advanced users with at least a basic understanding of the PDF syntax.</p> <p>Optionally pass in a text string. The program will use this as a regular expression match and only export objects matching the string. For example, you may want to only export annotations. In this case, you could pass in "/Annot" and only those objects containing the string would be exported.</p> <p>For some more complex examples, to match all entries that have /Annot or /JS, use "(?:/Annot /JS)". To match all entries that have /Annot but not /JS, use "^(!.*(?:/JS)).*/Annot(?:!.*(?:/JS))". To match all entries that have both /Annot and /JS, use "((?:/Annot.* /JS)(?:/JS.* Annot))". Certain characters such as the . and * are special for Perl regular expressions so you will need to place a backslash in front of them to search for that character. For example, "my\\.string" to match on my.string.</p>

Options

<u>Option</u>	<u>Description</u>
-objsizeout	<p>Exports object size information for the PDF. Use this to gauge what objects are taking up the most space in your PDF. This option includes everything from -infoout plus extra information about object sizes. The tab OBJSIZE is used to convey size information. For example: <OBJSIZE TYPE="XObject" SIZE=592978></p> <p>The TYPE attribute is based on the type of object as labeled in the PDF. Not every object may be labeled so there will be some sized listed under "Other". Some common entries are: Font - The font names and styles. FontDescriptor - Additional description information for fonts. Page - The page objects (not the content, just the page identifiers). Pages - The pointers to all the page objects. Stream Data - The (usually compressed) data that makes up the page contents. XObject - Image data.</p> <p>The SIZE attribute is in bytes.</p>
-onepage "text"	<p>Takes all pages from the input PDF and places them on a single page in the output PDF. Pass in "noscale" to expand the output page size big enough that the individual pages won't need to be scaled down. Otherwise, use "scale". Use the -pagesize option if you want to set the page size (pass in "scale" in this case). Use only one input PDF with this option. May not work on all PDFs.</p>
-onepageborder	<p>Draws a border around the pages when using the -onepage option.</p>

Options

<u>Option</u>	<u>Description</u>
-onepage <i>"text"</i>	Optional layout to use with the -onepage option. Pass in a comma separated list of page numbers. For example, "1,5,6,18,19,10" will pull only those pages and in the order specified. You may also specify an angle of rotation for each page. Place the angle after the page number with a colon in front. For example, "1,5:90,18,19:180,10:270" will pull the pages and rotate page 5, 19 and 10. The angle must be 0, 90, 180 or 270 only.
-ooserver <i>server:port</i>	The server and port that OpenOffice server is running on. For when you have an OpenOffice server running to convert Excel/Word/etc. to PDF for use by PDF Meld. Pass the server and port number. For example, -ooserver "myooserver:2050".
-open	Automatically opens Acrobat and loads the newly created PDF.
-openactfile <i>text number</i>	The file number (starting from 1) or name from your list of input files that you wish to take the open action from for the open action on the output PDF.
-openscr <i>file</i>	For Unix/Linux systems where -open is not available. Specify a script that will receive as a parameter the output PDF file name. Create a script for your operating system that will be used to open PDFs.

Options

<u>Option</u>	<u>Description</u>
-opt	<p>Optimize the output PDF for fast web viewing. Note this typically increases the size of the output by a few hundred bytes or so. The PDF is optimized for viewing on the web as opposed to shrinking the physical size. Additionally, you must create the PDF to a file rather than stream the output to the browser. The setting "fast web view" will be set to yes for optimized PDFs when you open in Reader and check the properties. This means the first page of the PDF is sent to the user and made viewable while the rest of the pages continue to download in the background.</p>
-opt15	<p>Optimize the output PDF for fast web viewing and use higher compression (same as setting -opt and -comp15). The PDF is optimized for viewing on the web and further compressed for a smaller file size. Additionally, you must create the PDF to a file rather than stream the output to the browser. The setting "fast web view" will be set to yes for optimized PDFs when you open in Reader and check the properties. This means the first page of the PDF is sent to the user and made viewable while the rest of the pages continue to download in the background.</p>
-orient <i>text</i>	<p>Set to P or L for portrait or landscape. This will apply a rotation of 90 degrees to any pages that are opposite of what you set -orient to. For example, if you use -orient P then any pages that are landscape will be rotated (if not already) to portrait. The contents will remain landscape or portrait but the page itself will be rotated in the viewer. You may use 270 in front of the P or L such as -orient 270P. This will set the rotation to 270 degrees rather than 90 degrees for those pages that require rotation.</p>

Options

<u>Option</u>	<u>Description</u>
-out	Optional command to use in front of the output file. Running "pdfmeld a.pdf b.pdf" is the same as "pdfmeld a.pdf -out b.pdf". Also see -in .
-overlay	Used to specify overlay mode rather than append mode. This option places the contents of each page from one PDF on top of the corresponding page from a second PDF. The first input PDF is the background and the second input PDF is placed over top.
-overlay2	An alternate method for overlaying pages. May work with PDFs that have problems overlaying using the -overlay option. Also, depending on the PDFs, may execute faster than -overlay.
-overlayfile <i>file</i>	Used along with -overlay. Specify a file that contains tags with text to look for on the page. The page of the background PDF can then be set based on whether or not the text was found. See the Overlay File section for details.
-overlaypages <i>list</i> or <i>file</i>	A range of pages to perform the overlay or stamp. Use a comma separated list with no spaces before or after the comma. A dash may be used to separate a range of numbers. For example, -overlaypages 2,7,14-20,35. Use negative numbers to refer to the position from the end of the page set. For example, -1 for the last page or -3 for the third to the last page. You may also use the words "odd" or "even" to overlay on all the odd or even pages. The odd and even options may be combined with page numbers as well, as in "1,even,-1,-2". Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. To overlay only on pages 15 to the last page of a document you can use "15--1". May also specify a file name containing the list in place of the list.

Options

<u>Option</u>	<u>Description</u>
<code>-pagebmargin <i>number</i></code>	The amount of space from the bottom edge of the page for the page number. Each unit is 1/72 of an inch (or the unit setting) so a number of 144 means 2 inches from the bottom edge of the page. This value is used to mean from the top of the page if <code>-pagetop</code> is specified.
<code>-pageduplex ["x,y[,z]"]</code>	Adds a blank page at the end of any PDF with an odd number of pages (except the last PDF when multiple files are passed as input) when appending PDFs. Optionally, pass in the added page size in points (1/72 of an inch) or the unit setting. For example, use <code>-pageduplex 612,792</code> for a size of 8.5 by 11 inches when <code>-units</code> have not been set. The default is to use the same size as the last page in the PDF so you can leave the size off. The last ("z") parameter is used to specify you want a blank page even on the last PDF in the list if it has an odd number of pages. Pass a 1 for the z parameter in this case. This will happen automatically if you only have one PDF as input. Set x and y to 0 if you only want to set the z value to 1.

Options

<u>Option</u>	<u>Description</u>
-pagefmt <i>text</i>	<p>Format string for the page number. Use %1 for the current page and %2 for the total number of pages. You can use %B or %nB for Bates numbering (legal document numbering with a six digit page number, padded on the left with 0's). For example, you could use "Page %1 of %2" to have "Page 1 of 3" print on the first page, "Page 2 of 3" on the second and so on. The default string is "%1/%2" (or just "%1" when the number format is roman). The n in the %nB format is the optional starting page number. For example, using "AP%312B" will number the first page as AP000312, the next as AP000313, etc. Note you don't have to include the page number if all you want to do is print a text string on each page. Use two %'s in a row if you're running the program from within a DOS batch file.</p> <p>See the Page Format Variables section for information on other variables you may use.</p>
-pageinfoin <i>file</i>	<p>Loads in any changes for the pages specified. You may use the -pageinfoout option to generate a file containing the current settings. See the Page Layout Files section for information on the file layout.</p>
-pageinfoout	<p>Exports page layout information for each page in the PDF. This is a tag based file with the values for MediaBox, CropBox, BleedBox, TrimBox, ArtBox and Rotate. For example:</p> <pre><PAGEINFO PAGE=1 MEDIABOX="0,0,612,792" CROPBOX="0,0,612,792" ROTATE=90></pre> <p>See the Page Layout Files section for information on the file layout.</p>

Options

<u>Option</u>	<u>Description</u>
-pagelmargin <i>number</i>	The amount of space from the left edge of the page for the page number. Each unit is 1/72 of an inch (or the unit setting) so a number of 144 means 2 inches from the left edge of the page. Note that when -pagenumalign is used, this value is used as the padding amount on the left or right instead.
-pagemode <i>text</i>	Optional string for how to display the document when opened. Set to one of the following: <u>UseNone</u> Neither document outline nor thumbnail images visible <u>UseOutlines</u> Document outline visible <u>UseThumbs</u> Thumbnail images visible <u>FullScreen</u> Full-screen mode, with no menu bar, window controls, or any other window visible <u>UseOC</u> Optional content group panel visible <u>UseAttachments</u> Attachments panel visible
-pagenum	Adds page numbers to each page. Default position is bottom left hand corner.
-pagenumalign <i>text</i>	L for left, C for center or R for right. The setting made with -pagelmargin is used as padding on the left or right rather than an absolute position when this method is used.
-pagenumbgcolor <i>color</i>	The color for the background of the page number string. Default is no background.
-pagenumcolor <i>color</i>	The color for the page number string. Default is black text.

Options

<u>Option</u>	<u>Description</u>
-pagenumdetails <i>text</i>	<p>Pass in the details on what pages to print page numbers on, the starting page number for each range and the style. The format is a pipe separated main list with a comma separated inner list. Specify the following comma separated values for each inner list:</p> <ul style="list-style-type: none">• Starting page number• Ending page number• Number to begin range with• Through page offset (may be negative)• Style to use <p>For example, "1,5,,r 7,-5,1" means for pages 1 to 5 inclusive, start numbering at 1 (since the current page number is used if this value is omitted) and use lowercase roman numerals. For page 6, no page number will print. Pages 7 up to 5 pages before the end of the PDF (since -5 was used - note that -1 means the last page of the PDF) will use Arabic numbering with page 7 numbered as page 1, page 8 as page 2, etc. If "1,5,,r 7,,1" was used instead, pages 7 and on would all be numbered, no matter how many.</p> <p>Use "20,,1,-19" To start numbering at page 20 and show 1 as the page number and the through pages be the remaining pages in the PDF (-19 meaning subtract 19 from the actual total pages in the PDF). So for example, suppose there are 50 pages in the PDF. Page 20 will read "1/30", page 21 will read "2/30" and so on.</p> <p>Use -pagefmt to format how the page number and through page number should be formatted. You can also use -pagefmt to exclude the through page number.</p>
-pagenumeven	<p>Adds page numbers to each even numbered page. Default position is bottom left hand corner.</p>

Options

<u>Option</u>	<u>Description</u>
-pagenumfont <i>text</i>	<p>The font to use for the page number string. Set to one of the following:</p> <p>Courier Helvetica (Default) Times-Roman Courier-Bold Helvetica-Bold Times-Bold Courier-Oblique Helvetica-Oblique Times-Italic Courier-BoldOblique Helvetica-BoldOblique Times-BoldItalic</p> <p>Or use the -fonts option to embed your own font.</p>
-pagenumnorotate	<p>Page numbers, by default, are rotated to match the current page's rotation angle. This setting turns off that processing meaning page numbers may display on any edge depending on the current page's rotation.</p>
-pagenumodd	<p>Adds page numbers to each odd numbered page. Default position is bottom left hand corner.</p>
-pagenumoutside	<p>Places added items such as page numbers and rectangles outside of any page scale or sizing specified. By default, these added items will scale or adjust with page size or scaling operations.</p>
-pagenumreset	<p>Resets the current page number to 1 on each PDF. In addition, the total number of pages changes for each PDF appended to be the number of pages in that PDF rather than the total pages in the output.</p>
-pagenumsize <i>number</i>	<p>Point size of the font for the page number. Default is 10.</p>

Options

<u>Option</u>	<u>Description</u>
-pagenumstyle <i>text</i>	The style for the page number. Currently, the valid values are R for uppercase roman and r for lowercase roman. Roman numerals of 4000 and higher are not converted since the necessary symbols are not in the ASCII character set. The default is Arabic.
-pageord <i>list</i> or <i>file</i>	Specifies a new page ordering for the resulting PDF. Only use one input PDF with this option. Use a comma separated list with no spaces before or after the comma. A dash may be used to separate a range of numbers. For example, -pageord 1-5,7,15,2. You may also use -pageord "reverse" to reverse the page order from the input PDF. May also specify a file name containing the list in place of the list.
-pages <i>list</i> or <i>file</i>	A range of pages to pull from the input PDF. Use a comma separated list with no spaces before or after the comma. A dash may be used to separate a range of numbers. For example, -pages 2,7,14-20,35. Use negative numbers to refer to the position from the end of the page set. For example, -1 for the last page or -3 for the third to the last page. You may also use the words "odd" or "even" to pull all the odd or even pages. The odd and even options may be combined with page numbers as well, as in "1,even,-1,-2". Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. To pull pages 15 to the last page of a document you can use "15--1". May also specify a file name containing the list in place of the list.
-pages1 <i>list</i> or <i>file</i>	A range of pages to pull from the first of two input PDFs. Similar to the pages option option, this option applies to the pages of the first PDF when merging two PDFs.

Options

<u>Option</u>	<u>Description</u>
-pages2 <i>list</i> or <i>file</i>	A range of pages to pull from the second of two input PDFs. Similar to the pages option option, this option applies to the pages of the second PDF when merging two PDFs.
-pagesize "x,y"	Sets a new page size in points (1/72 of an inch) or the unit setting. For example, use -pagesize 612,792 for a size of 8.5 by 11 inches when -units have not been set. All information about the current page size and cropping is removed so setting to the same page size currently used will not necessarily display the same. You can make adjustments, if necessary, with the -right or -down commands.
-pagestopdf	Used to remap pages from one PDF to page 1 of other PDFs. For example, assume you have a PDF with three pages. Page 1 has a link to page 2 and a link to page 3. You can use this option to pull only page 1 of this PDF and append two other PDFs to it. Normally, the links on page 1 would no longer function since page 2 and page 3 have been removed. With this option set, the link to page 2 in the first PDF will now point to page 1 of the first appended PDF (which is still page 2 in this case). The link to page 3 will now point to page 1 of the second appended PDF. This will be 1 plus the number of pages in the second PDF. This allows you to setup a table of contents in one PDF that can be used to link to contents that are created when you run PDF Meld.
-pagetop	Place the page numbers at the top of the page rather than the bottom. Note that -pagebmargin then specifies space from the top rather than bottom edge of the page.

Options

<u>Option</u>	<u>Description</u>
-parse1	Use this option only if the program isn't able to use the input PDF(s) given. This option will parse through the input files using a different method. Will not work on all PDFs. Usually this is only needed for input PDFs which are damaged (the cross reference table isn't valid, for example).
-pdfmeld	<i>(Not available in SE version)</i> Used in-between sets of commands to simulate calling the program several times. See the Multiple Call section for details.
-print	Automatically prints the newly created PDF to the default printer. Must have Acrobat or Reader installed.
-printcfgin <i>file</i>	Used to specify the print configuration file. This information is stored in the PDF and later can be used by PDF Meld for custom printing parts of the document to certain printers or trays. Use the -printcomp option to later print the PDF based on these settings. See the Printer Configuration Files section for information on the file structure.
-printcfgout	Exports the current print configuration settings used in the PDF. In this case, use a file with a .txt extension rather than .pdf for the output. See the Printer Configuration Files section for information on the file structure.
-printcomp <i>file</i>	Used to print a PDF that contains printer configuration information. Pass in the name of a configuration file or leave blank to use the file contained in the PDF (the PDF must have been built using the -printcfgin option in this case). This is only valid on Windows operating systems and requires Adobe Reader to print. The printer configuration information can be added to a PDF with the -printcfgin option. There is no output specified when using this option, nor should any other options be specified other than the input PDF.

Options

<u>Option</u>	<u>Description</u>
-printcopies n	Works with Acrobat or Reader 8.0 or higher. The number of copies to be printed when the print dialog is opened for this file. Supported values are the integers 2 through 5. Values outside this range are ignored.
-printdlg	Brings up the Acrobat print dialog box and allows printer selection. This only works when the user has Acrobat or Reader associated with PDFs on their machine. Otherwise the user's viewer is opened with the document and they will need to print from there.
-printduplex "text"	Works with Acrobat or Reader 8.0 or higher. The paper handling option to use when printing the file from the print dialog. The following values are valid: <u>Simplex</u> Print single-sided <u>DuplexFlipShortEdge</u> Duplex and flip on the short edge of the sheet <u>DuplexFlipLongEdge</u> Duplex and flip on the long edge of the sheet
-printer <i>printer [device port]</i>	Used to print the PDF to the specified printer. There is no print dialog box in this case. This option takes three parameters: printer, device and port. You may pass in just the printer and leave device and port blank to use the default settings for the printer. For example: -printer "Accounting Printer" "HP LaserJet 5" "lpt1:" or -printer "Shipping Printer" You may also use the printer port as the first parameter and leave the last two off if you are using a network printer or don't know the printer name. For example: -printer "\\server\printer"

Options

<u>Option</u>	<u>Description</u>
-printerlist <i>file</i>	Used to generate a list of printers available on the system. This can be used to verify what printers the program finds and what they are called. The list generated is tab separated and includes the printer name, device name and port. Use any of the printer names in the file with the -printer option. This option is only available under Windows systems. Use this option by itself as the program will exit after generating the list.
-printpagerange "from,to,..."	Works with Acrobat or Reader 8.0 or higher. The page numbers used to initialize the print dialog box when the file is printed. The first page of the PDF file is denoted by 1. Each pair consists of the first and last pages in the sub-range. An odd number of integers causes this entry to be ignored. Negative numbers cause the entire array to be ignored.
-printpicktray	Works with Acrobat or Reader 8.0 or higher. Specifies the PDF page size is used to select the input paper tray. This setting influences only the preset values used to populate the print dialog presented by a PDF viewer application. If used, the check box in the print dialog associated with input paper tray is checked.
-printscales "text"	Works with Acrobat or Reader 7.0 or higher. Valid values are None, which indicates that the print dialog should reflect no page scaling, and AppDefault, which indicates that applications should use the current print scaling.
-printscr <i>file</i>	For Unix/Linux systems where -print is not available. Specify a script that will receive as a parameter the output PDF file name. Create a script for your operating system that will be used to print PDFs.

Options

<u>Option</u>	<u>Description</u>
<code>-prmencc <i>password</i></code>	Encrypts the parameter file with the password. All keys and values are encrypted. To unencrypt, run <code>pdfmeld -prmunenc "file password"</code> .
<code>-prmload <i>file key[,key2,...]</i> or <i>key[,key2,...]</i></code>	Loads the command line parameters. The file used to save the settings are in <code>pdfmeld.mpf</code> by default. Pass in a file name followed by the pipe character <code> </code> to use that file instead. The "key" is the string value you used along with the <code>-prmsave</code> option. You may run <code>pdfmeld -prmload "key"</code> to re-run a previous save using the same input/output file names. Also, you may change the input/output files (or specify additional parameters) by including those on the line. For example, <code>pdfmeld filein.pdf fileout.pdf -prmload "key" -open</code> . Specify a comma separated list of key names if you wish to load options from more than one group.
<code>-prmsave <i>file key</i> or <i>key</i></code>	Saves the command line parameters for recall later using the <code>-prmload</code> option. The file used to save the settings are in <code>pdfmeld.mpf</code> by default. Pass in a file name followed by the pipe character <code> </code> to use that file instead. The "key" is any text string you want to refer to the settings as. This allows you to save multiple settings in a single file. The input/output file names are saved with the parameters but you may override them during the load.
<code>-prmunenc <i>file password</i> or <i>password</i></code>	Un-encrypts the parameter file when the correct password is used. All keys and values become un-encrypted.
<code>-producer <i>producer</i></code>	Sets the document producer.

Options

<u>Option</u>	<u>Description</u>
-pwdlist <i>text</i>	Used to specify a list of owner passwords (no particular order) for any encrypted input PDFs. Separate the list with a comma (be careful not to leave a space before or after the comma). Place an '@' in front of the value if it is instead a file containing a list of passwords (one password per line). Use a \ in front of a comma if the comma itself is part of the password when passing the list. You do not need to do this when using a file containing a list of passwords. Only PDFs encrypted with RC4 40-bit to 128-bit encryption or AES 128/256-bit will work. The output PDF will be unencrypted unless you're re-encrypting with the -o or -u options.
-quick	Make fewer passes though the input files to speed up processing. This option is ignored when performing any type of overlay.
-quickscan	Skip processing that scans each page for used objects. This is only noticeable on large PDFs where standard processing is taking a long time. Otherwise it's recommended not to use this option. The output PDF may be slightly larger when using this option.
-rectin <i>file</i>	Specifies the file to load rectangle information from. Use a pipe character, the symbol, to separate files if you wish to pass more than one file name. See the Rectangle Files section for information on the file structure.
-reduce	Checks page contents and removes any images or fonts in the PDF that are not used. Normally this process isn't necessary but if the PDF has been modified or updated there is the potential for left over images or fonts to be retained in the PDF.

Options

<u>Option</u>	<u>Description</u>
<code>-reduceimg <i>number</i></code>	Reduces image size by removing pixels. Does not work on all image types so this feature won't work on every PDF. Only works on images stored as a matrix of pixels. Not on JPEG images, for example. Pass a value for the reduction amount. The bigger the number the more reduction (and smaller PDF) but the more "blocky" or "jagged" the images will appear. The number is any integer 1 or greater but for practical purposes it should be no more than 3.
<code>-reducemax <i>number</i></code>	Sets the maximum size an image must be (in 1,000 pixels) before applying the <code>-reduceimg</code> function. The default is to not use this option and convert everything over the specified minimum. There may be cases where some larger images are not to be converted. Use this option in that case.
<code>-reducemin <i>number</i></code>	Sets the minimum size an image must be (in 1,000 pixels) before applying the <code>-reduceimg</code> function. Typically you won't want to reduce small images since there is little benefit. The default is 62.5 (for 62,500 pixels) which works out to an image size of 250 x 250. Any images smaller than this will not be converted. Note it's the total number of pixels and not a certain height or width. An image with a size of 5,000 x 10 won't be converted using the default.
<code>-removeattr <i>name</i></code>	Removes an attribute. May include this option multiple times to remove more than one name/value pair if needed. These attributes are the ones included in the document information object (where document title, subject, keywords, etc. are stored). For example, <code>-removeattr "Source"</code> to remove the attribute called Source.

Options

<u>Option</u>	<u>Description</u>
-removebkg <i>text</i>	Used with the -overlay or -reoverlay option. Will attempt to remove any filled rectangles from the PDFs. Pass 1 to remove from the first PDF, 2 to remove from the second or * for both. Use this when there is a block of color (including white) behind the text keeping the background from showing through. Only works when there is actual text on the page and not a scanned image containing text. Otherwise see the -transparency option.
-removeflds	Removes the fields in the PDF as opposed to flattening. There is no text replacement with the value as in -flatten. Use with other options such as -flattenflds.
-removestruct	Removes the structured tree (if one exists) from the PDF. This can speed up processing and reduce the output file size if this additional information is not needed in the finalized PDF.
-removevr <i>text</i>	Removes vector or raster components. Pass in either V to remove vector components or R to remove raster components. Raster components are images, like jpeg for example. Vector components are text and any line drawing or fill commands. May not work on all PDFs.
-repeat	Used to specify overlay mode rather than append mode and repeat the first PDF. This option places the contents of each page from one PDF on top of the corresponding page from a second PDF. The first input PDF is the background and the second input PDF is placed over top. The background is repeated, if necessary, for as many pages as there are in the second PDF.

Options

	<u>Option</u>	<u>Description</u>
	-repeatlast	Same as -repeat except the last page of the background PDF is repeated, not cycling again from page 1. For example, you might have a 5 page PDF that is to be overlaid with a logo contained in a second PDF. Suppose there are 2 pages in the logo PDF - page 1 has the logo for page 1 of the output and page 2 has the logo for all other pages. You would use -repeatlast in this case to continue repeating page 2 of logo PDF in the output. Using -repeat instead would result in page 1 of the logo PDF being used on pages 1, 3 and 5 of the output.
	-reoverlay	Used to specify reverse overlay mode. Works like -overlay except the second PDF is treated as a background. The resulting PDF still has the same number of pages as the second PDF. You'll need to use a reverse overlay when you have a form field on a single page PDF you wish to overlay on multiple pages of another PDF and still retain the form field. In this case, the single page PDF should be the second PDF. Any changes to the form field on one page will be reflected in the copy of the field on all other pages. You may instead use -flatten along with either -overlay or -reoverlay to flatten and repeat the field which will be converted to plain text.
	-reoverlay2	Same as -overlay2 except the second PDF is treated as the background for overlay.
	-richtext <i>text</i>	Used to specify a list of existing field names to convert to rich text. Separate the list with a comma (be careful not to leave a space before or after the comma). Place an '@' in front of the value if it is instead a file containing a list of fields (one field per line). Pass an empty list to mean all fields should be converted to rich text.

Options

<u>Option</u>	<u>Description</u>
-right <i>number</i>	The distance to move the contents of each page to the right. The number is in units of 1/72 of an inch or the unit setting. May be positive or negative value. May not work on all PDFs. Only use one input file with this option.
-ro	Sets any interactive fields in the PDF to read-only.
-rofields <i>field1,field2,...</i>	Sets the listed interactive fields in the PDF to read-only.
-rofieldsrev	Reverses the meaning of -rofields. The fields listed in -rofields are the the ones to keep fillable when this option is used.
-rotate <i>number</i>	The angle to rotate the pages when opening in the viewer. Must use either 90, 180 or 270. This value is added to any existing rotation.
-rotateabs <i>number</i>	The absolute angle to rotate the pages when opening in the viewer. Must use either 0, 90, 180 or 270. The current rotation is ignored and replaced with this value.
-run "text"	Uses the saved settings file supplied as input parameters. You may use this in addition to any other parameters you wish to pass to the program. For example, output.pdf -run mysettings.mld -open. Setting files are created by saving a setting file (.mld) from the GUI front-end which can accessed by running the executable version without passing it other parameters.
-s	Include subdirectories when the input is a directory rather than individual files.
-scale <i>number</i>	The scaling factor to apply to each page. The default is 100 or no scaling. A value of 50 will scale the contents to 50% of their original size. May not work on all PDFs. Only use one input file with this option.

Options

<u>Option</u>	<u>Description</u>
-scalex <i>number</i>	The scaling factor to apply to each page along the x-axis only. The default is 100 or no scaling. A value of 50 will scale the contents to 50% of their original size. May not work on all PDFs. Only use one input file with this option.
-scaley <i>number</i>	The scaling factor to apply to each page along the y-axis only. The default is 100 or no scaling. A value of 50 will scale the contents to 50% of their original size. May not work on all PDFs. Only use one input file with this option.
-server	Starts up PDF Meld in server mode. PDF Meld will run as a separate process and take commands from the TCP/IP port it is connected to. The Client-Server section describes this operation in more detail. _000082__000082_
-showannotes	Draws a box around annotations and displays the field name in the box.
-shrink	Attempts to remove unused fonts or graphics - under most circumstances this option is not necessary. Mainly of use when you are splitting out a page or smaller section from a PDF and unnecessary objects are being included. Fonts and graphics are normally kept if the page definition has a reference to them. This option scans the page contents to see if they are actually used so there is more processing involved though you may get a smaller PDF. You won't see a size reduction if the input PDF is already structured to not include unused objects in the page definition.
-signbgcolor <i>text</i>	Optional. The background color for the signature field. See the Digital Signature section for details.
-signcolor <i>text</i>	Optional. The text color for the signature field. See the Digital Signature section for details.

Options

<u>Option</u>	<u>Description</u>
-signdate <i>text</i>	<p>Use this option to force a particular date and time for the signing. Pass a comma separated string in the form "yyyy,mm,dd,hr,mi,ss,+ or -,offset hr,offset mi".</p> <p>yyyy = year mm = month ss = seconds hr = hour (optional) mi = minutes (optional) ss = seconds (optional) + or - = the time zone offset from UTC (optional) offset hr = the number of hours offset from UTC (optional) offset mi = the number of minutes offset from UTC (optional)</p> <p>Only the year, month and day are required. Any values not provided will be taken from the local computer settings. Be careful not to set a date before the start date of the signing certificate or in the future as the signature will not show as valid in Acrobat or Reader.</p>
-signimg <i>text</i>	<p>Optional. The path and name of an image to use for the signature. Set this option to "none" to not place any image in the signature field. See the Digital Signature section for details.</p>
-signkeepratio	<p>Optional. Keep the image x/y scaling ratio when using an image with a signature field. See the Digital Signature section for details.</p>

Options

<u>Option</u>	<u>Description</u>
<code>-signname text</code>	<p>Optional. The name of the signature field in the PDF to sign. For example, "sig1". Use the <code>-fdfout</code> option to export field names if you are not sure. If not used then PDF Meld will look for the first unsigned signature field. Or you may pass in <code>:"new"</code> or <code>:"hide"</code> to create a new signature field in the PDF and then sign it. Only use <code>:"new"</code> or <code>:"hide"</code> when there are no existing signed signatures in the PDF or you will invalidate them.</p> <p>The <code>:"new"</code> option places a visible signature field in the lower left corner of the first page. You may optionally specify the position along with <code>:"new"</code>. In this case, place the coordinates after <code>:"new"</code> like this - <code>:"new@x1,y1,x2,y2"</code>. X1 is the left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature field. The position 0,0 is the lower left corner of the page. Y1 is the top Y coordinate, X2 is the right X coordinate, and Y2 is the lower Y coordinate. For example, <code>:"new@200,300,250,400"</code>.</p> <p>The <code>:"hide"</code> option does not place a visible signature on the page; it only shows in the signature pane. See the Digital Signature section for details.</p>
<code>-signpemfile text</code>	<p>The path and name of the signing certificate. For example, <code>"c:\keys\mykey.pem"</code>. See the Digital Signature section for details.</p>
<code>-signpkfile text</code>	<p>The path and name of the private key file. For example, <code>"c:\keys\mykey_pk.pem"</code>. See the Digital Signature section for details.</p>
<code>-signrsn text</code>	<p>Optional. The reason for signing the document. Default is "Attestation to the accuracy and integrity of this document". See the Digital Signature section for details.</p>

Options

<u>Option</u>	<u>Description</u>
-signscript	Use a script font that looks like this - <i>Jane Doe</i> - and place just the name from the certificate into the signature field. See the Digital Signature section for details.
-signsize <i>text</i>	Optional. The fontsize for the text of the signature (0 for no text). Default is 12. See the Digital Signature section for details.
-signsrc <i>text</i>	Optional. A TrueType font file to use for the signature text. See the Digital Signature section for details.
-signssl <i>text</i>	The path and file name of the OpenSSL program. For example, "c:\openssl\bin\openssl.exe". See the Digital Signature section for details.
-signtimestamp <i>url</i>	The path and name of your timestamp server if you want to include an embedded timestamp on the signature. Use "1" to use the default of https://freetsa.org/tsr. Pass your user name and password if your timestamp server requires it like this: "myuserid:mypassword@https://mytimestamp.org". Use the -signs option with -signtimestamp.
-signs <i>file</i>	The path and name of the ts.exe (or just ts for Linux) executable that is used to perform the timestamping function. This is a separate download available here . Use this program for both Windows and Linux.
-skipfieldrename	Prevents interactive field names from changing when merging fillable PDFs. Normally, when multiple fillable PDFs are merged, the names of the fields in each are changed so as not to conflict with each other. This option turns off that feature.
-skipmissing	Skips over any missing input files rather than give an error and stop processing. The list of skipped files will be written to the error log file (set via the -e option).

Options

<u>Option</u>	<u>Description</u>
-sort <i>sortby</i>	Used to specify the sort order. The default sort order is the order given in the list of input files when a comma separated list of names is used. Pass in one of the following for sortby: f = file name n = file name numeric b = file size in bytes d = last modify date You may also use an r along with any of the above to sort in decending order. For example, rf or fr sorts by name in decending order.

Options

<u>Option</u>	<u>Description</u>
-splittext "params"	<p>Split a PDF into multiple PDFs based on a string value at a given location on each page. The output file specified on the command line must be a directory name and not a file name. To find the string location in your PDF, first run PDF Meld on a sample input and use the options -textdetailout -json "details". You may want to extract just a few pages first so the output is not too big. Check the output for the string you are interested in splitting the text on and note the X/Y and X2/Y2 bounding box location. Pick any point that falls within the bounding rectangle as the point to pass to this option. The "params" option is a space separated list of options similar to a tag.</p> <p>Options:</p> <ul style="list-style-type: none">X - The X location on the pageY - The Y location on the pageDETAILS - (Optional) uses the "details" section when searching for text.FROMPG - (Optional) the page to start looking for the string.THRUPG - (Optional) the page to stop looking for the string.STARTPOS - (Optional) the starting position (0 based) in the string to use.STARTPOS - (Optional) the starting position (0 based) in the string to use.STRLEN - (Optional) length of the string to use. <p>For example, -splittext "details X=500 Y=300 frompg=5 thrupg=100". May not work on all PDFs depending on how the text is arranged. The output PDF will be named after the string found at the location specified.</p>
-sql <i>text</i>	<p>Specifies the file containing the QUERY command to issue. See the SQL section for details on queries.</p>
-sqlcmds	<p>Load the needed SQL libraries on Unix. Because of differences among Unix systems, the libraries may not be compatible so they are not loaded unless this option is used.</p>

Options

<u>Option</u>	<u>Description</u>
-sqlcomb <i>text</i>	An output file to combine the multiple PDFs generated from a SQL into a single PDF. Pass in the path-name of the output PDF. Use -flatten as well to flatten the fields into plain text.
-sqldb <i>text</i>	The database schema or driver information. See the Database Connection section for details.
-sqldriver <i>text</i>	The data source. This is a case-sensitive string. Entries with a * are only available for Windows operating systems. Valid values are: Oracle mysql* mysqlPP ODBC* mysql may give slightly better performance over mysqlPP on Windows systems. The Oracle client software will need to be installed when using Oracle.
-sqlparams <i>text</i>	Default values to pass in for dynamic processing. This overrides any settings from the environment variable RWQUERY_STRING. Pass name/value pairs with an ampersand in-between. Start all variables with a \$. For example, "\$reg='A','B'&\$drink=ice%20water" will set the variable \$reg to "'A','B'" and \$drink to "ice water". You can then use \$reg and \$drink as part of dynamic SQL queries. See the SQL section for details on queries.
-sqlpwd <i>text</i>	The password for the user when making a database connection. This may also be specified on the QUERY tag.
-sqluser <i>text</i>	User name to use when making a database connection. This may also be specified on the QUERY tag.

Options

<u>Option</u>	<u>Description</u>
-stamp "text"	Stamps a PDF with the specified stamp. Use one of the following: Approved* Not Approved Draft Final Completed Confidential For Public Release Not For Public Release For Comment Void Preliminary Results Information Only Revised* Reviewed* Received* Witness Initial Sign Here Check Cross

You may use the -right, -down, -angle, -scale and -center options with the stamp, if desired. Add the text ",rev" (without the quotes) to the end of the select stamp (for example, "draft,rev") to reverse overlay the stamp. That is, the stamp will be placed on the page before any existing text or graphics. Only use one input PDF with this option. See the [Stamps](#) section for examples of the various stamps.

*See the -stamptext option for these.

Options

<u>Option</u>	<u>Description</u>
-stamptext <i>"text"</i>	<p>Text for dynamic stamps. The following stamp types can accept dynamic text to show in the stamp:</p> <ul style="list-style-type: none">ApprovedRevisedReviewedReceived <p>Note that there are two "approved" stamps - if you don't supply text for it then it is shown without the extra room for additional text. Use a small text string for the text. For example, on reviewed, you might want to set the text to "By <name> on September 9th, 20xx". You may use -angle of 0, 90, 180, or 270 only with this option.</p>
-strchange <i>text</i>	<p>(Not available in SE version)</p> <p>Replacement text for text specified with -string. Note the alignment of text will likely change on lines where the text is replaced. You may not get the correct characters if the PDF doesn't contain them due to font subsetting. In some PDFs, only a subset of the font is embedded meaning the PDF doesn't have enough information to properly show the added character(s). The characters will show only if a base font is used or the font was embedded in its entirety.</p>
-strcolor <i>color</i>	<p>The color for the -string text. The text color is set to black after the text is colored so only use in cases where the text is black to start with (this applies only when not using -strcolormode). The tag may be used for more control over color settings when using -textout/-textin.</p>
-strcolor2 <i>color</i>	<p>A second color for the -string text. This is used to create a gradient effect going from -strcolor to -strcolor2. Use -strcolormode HI in this case.</p>

Options

<u>Option</u>	<u>Description</u>
<code>-strcolorary x0,y0,x1,y1</code>	An array of 4 numbers used to express the starting and ending coordinates for the gradient shade. Use <code>-strcolor</code> along with <code>-strcolor2</code> and <code>-strcolormode HI</code> for a gradient shade. The default value is "0,0,1,0". This creates a shading pattern going from left to right. Use "0,0,0,1" for a shading pattern that goes from top to bottom, which works best with <code>-strcolormode HI-R</code> . Other values, like ".2,.1,.8,.7" will give other effects.
<code>-strcolormode text[,number,number]</code>	<p>The text itself is colored by default using <code>-string</code> and <code>-strcolor</code>. This option is used to highlight or underline the text instead. Set to one of the following:</p> <ul style="list-style-type: none">HI for highlightHI-R for highlight with rounded endsOL for outlineUL for underlineCO for cross-out <p>You may optionally pass in a number pair as well. These numbers (in units of points or 1/72 of an inch) are used to adjust the width and height of the highlight. A positive number will increase the width or height of the highlight and a negative number will decrease. The first number is the X axis amount and the second number is the Y axis amount. You may optionally use the transparency or background layer functions with this also.</p> <p>Text in a PDF is not necessarily contiguous so this option may not work on your PDF or on all occurrences.</p>

Options

<u>Option</u>	<u>Description</u>
<code>-string <i>text</i>[<i>scale,xoff,yoff</i>]</code>	Places the specified image to the left of the text matched from <code>-string</code> . Pass in the full path and image name. Optionally pass in a scale amount for the image (for example, use 50 for 50%) and an offset in points (1/72 of an inch) for the image in the X and Y direction (positive or negative numbers are allowed). For example, use <code>-string "sign.jpg" and -string "Sign Here"</code> to place an image to the left of the text "Sign Here" in the document. Place optional scaling and offset inside of the quotes like this: <code>-string "sign.jpg,50,-6,8"</code> . You may use transparency or a background layer with this option. Text in a PDF is not necessarily contiguous so this option may not work on your PDF or on all occurrences.
<code>-strin <i>file</i></code>	Used to load in text from the specified tag based file. See the Text Files section for the layout. You may also pass in the tags rather than a file name. For example, <code>-strin "<TEXT X=72 Y=72 SIZE=15>Here's some text</TEXT>"</code> .

Options

<u>Option</u>	<u>Description</u>
-string <i>text</i>	<p>(Not available in SE version)</p> <p>A text string to color or highlight in the PDF. Use along with -strcolor to set the color for the text or highlighting. Or, use with -strchange to change the text. Text in a PDF is not necessarily contiguous so this option may not work on your PDF or on all occurrences. Text that is an image or part of an image will not be affected.</p> <p>Use an asterisk (the * character) in the string to match any number of characters. Only use one asterisk. For example, to highlight a passage starting with "Run the setup program" and ending with "then reboot." use "Run the setup program*then reboot." as the string. All of the text in the section will be highlighted. Note that you may not span pages however. If a passage does span a page or more, break up the command and re-run PDF Meld several times, once for each page using an appropriate string value. To match on an asterisk and not use as a wildcard, use a backslash, the \ character, in front of the asterisk.</p>
-stringcase	<p>(Not available in SE version)</p> <p>Used to specify the string search from -string or -stringlist is case-sensitive. By default the search is case-insensitive.</p>
-stringlist <i>text</i>	<p>(Not available in SE version)</p> <p>A list of text strings to color or highlight in the PDF. Separate each entry with a pipe (the vertical bar or symbol). Be sure to place quotes around the entire string like this: -stringlist "cat dog mouse". Use along with -strcolor to set the color for the text or highlighting. Text in a PDF is not necessarily contiguous so this method may not work on your PDF or on all occurrences. Text that is an image or part of an image will not be affected.</p>

Options

<u>Option</u>	<u>Description</u>
-stringword	<p>(Not available in SE version)</p> <p>Used to specify the string search from -string or -stringlist should match on the whole word only. For example, if -string "light" is used it will match "light" by itself or in words like "highlight". This option attempts to prevent that and only matches when the word "light" is not preceded or followed by another letter. This may not always work depending on how the input PDF is structured.</p>
-stringwordre <i>text</i>	<p>(Not available in SE version)</p> <p>Use along with -stringword to specify other characters that should be included as part of a word. This is a regular expression so you could use -stringwordre "\xE4" to include the character ï¿½. You may pass multiple characters like -stringwordre "[\xE4\xFC]" - any string that is a valid regular expression may be used.</p>
-strpages <i>list</i> or <i>file</i>	<p>A range of pages to apply any -string modifications on. Use a comma separated list with no spaces before or after the comma. A dash may be used to separate a range of numbers. For example, -strpages 2,7,14-20,35. May also specify a file name containing the list in place of the list.</p>
-strrend <i>number</i>	<p>The rendering mode for -string. The default is 0 (fill the character). Other options are:</p> <ul style="list-style-type: none">1 - stroke (or outline) the character only2 - stroke and fill the character3 - no stroke or fill (invisible)
-strscolor <i>color</i>	<p>The -strscolor option can be used to set the color for the stroking operation.</p> <p>The stroking color used to set the text color for -string when using -strrend set to 1 or 2.</p>

Options

<u>Option</u>	<u>Description</u>
-strurl <i>text</i>	Converts the text matched from -string to a URL link. Pass in the URL to use. Use a * in the URL to have it replaced by the value matched from -string. For example, use -string "www.*.com" to match all text in the document starting with "www." and ending with ".com". You can set -strurl to "http://*" and the link will use whatever text the -string option matched. Text in a PDF is not necessarily contiguous so this option may not work on your PDF or on all occurrences.
-subject <i>subject</i>	Sets the document subject.
-textdetailin " <i>file</i> "[, " <i>hex</i> "]	(Not available in SE version) Uses the file layout passed in to extract text from the PDF. Optionally pass the string "hex" as a second parameter to export the text in hex format. See the Text Detail Files section for more details.
-textdetailout	(Not available in SE version) Exports to a tag based file the set of text fragments found in the PDF. Optionally include the option -json to export in JSON format. You can use this information for -textdetailin to export CSV or other ASCII files from text in the PDF. For example, to export a CSV file of mailing addresses. See the Text Detail Files section for more details.
-textdetailoutxy <i>x1,y1,x2,y2</i>	(Not available in SE version) Exports to a tag based file the set of text fragments found in the PDF. Pass in the X/Y coordinates specified in points. Only the text that falls within the specified rectangle is exported. You may repeat this option if there are multiple rectangles you are interested in. You might need to use -textdetailout for a few pages of your PDF first to find out what the coordinates are for the text you are interested in.

Options

<u>Option</u>	<u>Description</u>
-textdetailread <i>break,blank</i>	<p>(Not available in SE version)</p> <p>Attempts to create a readable text file as output. Pass a non-zero value for "break" to include the page break, zero otherwise. Pass a non-zero value for "blank" to include blank lines, zero otherwise. The default for "break" and "blank" are 1 so page breaks and blank lines will be included. This works best when the text in the PDF is fixed-width so columnar line up better. Use the -textscalex or -textscaley options to adjust the spacing. Use the -textheaderx or -textheadery options to remove headers or footers.</p>
-textheaderx	<p>(Not available in SE version)</p> <p>The starting line number when exporting text with -textdetailread. Export the report without this option first to see where the detail of the pages start. Set this value to the first line number of where you want to capture text. For example, if lines 1-3 contain heading information, set -textheaderx to 4.</p>
-textheadery	<p>(Not available in SE version)</p> <p>The ending line number when exporting text with -textdetailread. Export the report without this option first to see where the detail of the pages end. Set this value to the last line number of where you want to capture text. For example, if lines 50 is the last line containing detail text to capture, set -textheaderx to 50.</p>
-textin <i>file</i>	<p>Loads in any changes made to the file created with the -textout option. Do not add or remove any lines from the file as the ordering is used to determine what line of the PDF to change. May not work on all PDFs.</p>

Options

<u>Option</u>	<u>Description</u>
-textout ["DOS"]	Exports any text in the PDF to the output file for minor touch-up. Optionally pass "DOS" to export with a carriage return followed by a line feed. The default is just a line feed at the end of each line. In this case, use a file with a .txt extension rather than .pdf for the output. The text is simply written in the order it was found and not necessarily the order it appears on screen when viewing the PDF. Optionally include the option -json to export in JSON format. You may then make changes to the file in order to modify the text of the original PDF. Use the -textin option to load in your changes. May not work on all PDFs. Some PDFs use an image for the page contents rather than text so nothing will export in this case.
-textscalex	<i>(Not available in SE version)</i> The adjustment to make along the x-axis when exporting text with -textdetailread. Larger numbers will space text further apart and smaller numbers will push it closer together. The default is 7.5.
-textscaley	<i>(Not available in SE version)</i> The adjustment to make along the y-axis when exporting text with -textdetailread. Larger numbers will space text further apart and smaller numbers will push it closer together. The default is 11.

Options

<u>Option</u>	<u>Description</u>
<code>-textsplt <i>text</i></code>	Split the original PDF into multiple PDFs based on the text string specified. For example, you might have a PDF with the text "Page 1 of " on each new section to break on. Pass the text "Page 1 of " in this case. The output files are named after the base PDF with the page number range appended. May not work on all PDFs depending on how the text is internally structured. The output file specified on the command line must be a directory name and not a file name. Only use one input file with this option. May want to use -reduce with this option. Use -burstquick with this option if you have a large PDF and want to speed up the process. The bookmark is not retained in the output PDF in this case.

Options

<u>Option</u>	<u>Description</u>
<code>-textspltxy</code> <code>x1,y1,x2,y2[,s1,s2]</code>	Split the original PDF into multiple PDFs based on any change of the text in the specified rectangle. Pass in the X/Y coordinates specified in points. Optionally include <code>s1</code> and <code>s2</code> for the substring position within the text. The <code>s1</code> value is the starting position (beginning with 0 for the first character) and the <code>s2</code> value is the length. Set <code>s2</code> to 0 to mean to the end of the string or a negative number to mean up to that many characters from the end of the string. Also, you may set <code>s1</code> to a negative value to mean that number of characters from the end of the string. The output files are named after the located text with the page number range appended. You may repeat this option if there are multiple rectangles you are interested in. May not work on all PDFs depending on how the text is internally structured. The output file specified on the command line must be a directory name and not a file name. Only use one input file with this option. May want to use -reduce with this option. It may be helpful to use -textdetailout first to determine the X/Y locations of the text you are interested in. Use -burstquick with this option if you have a large PDF and want to speed up the process. The bookmark is not retained in the output PDF in this case.
<code>-textspltxyib</code>	Use with <code>-textspltxy</code> to ignore blanks. This is useful in situations where the text to break on is only displayed on the first page to break on.
<code>-timeout secs</code>	Number of seconds to process before timing out. You may also specify the number of minutes by using an "m" at the end, such as "5m" meaning 5 minutes. Use this option to limit how long PDF Meld will run before exiting. This is mainly for batch mode processing where a rogue PDF could potentially hold up processing. The default is 0 meaning there is no timeout setting. A timeout returns code -11.

PDF Meld

Options

	<u>Option</u>	<u>Description</u>
	-title <i>title</i>	Sets the document title.
	-topbottom	Places two pages from the input PDF(s) on one sheet (overlap will occur if the input page size is more than half the size of the output page size). Page 1 goes on top and page 2 goes on bottom when using a single PDF. Page 1 from the first PDF goes on top and page 1 from the second PDF goes on the bottom when using two PDFs. Set the new page size with the -pagesize option. May need to use -scale as well (especially if the input page size is more than half the size of your output page).

Options

<u>Option</u>	<u>Description</u>
-transparency <i>x[,text]</i>	Used to specify the transparency value and mode. Requires Acrobat or Reader 5.0 or higher to view the transparency. You likely don't need to use this option if you are only trying to overlay text on a background PDF. The default mode, Normal, uses the transparency value to determine how transparent the top level will be. Note this applies not only to overlays, but you can use this when adding page numbers or text with the -pagenum option. The opacity value is from 1 to 100. For example, in Normal mode, the higher the number, the more of the background shows through. Optionally, specify the mode of transparency. The valid values are: Normal (Default) Multiply Screen Overlay Darken Lighten ColorDodge ColorBurn HardLight SoftLight Difference Exclusion Hue Saturation Color Luminosity
-trimbox " <i>x1,y1,x2,y2</i> "	Example: -transparency 50 or -transparency 75,Screen The rectangle defining the finished page after trimming. The values are in points (1/72 of an inch) or the unit setting. Default is the crop box setting.

Options

	<u>Option</u>	<u>Description</u>
	-twopage	Places two pages from the input PDFs next to each other onto one sheet in the output PDF. The original pages are scaled down to fit next to one another. Page one from each of the inputs are placed side-by-side on page 1 of the output PDF. Page two from each of the inputs are placed side-by-side on page 2 of the output PDF and so on. Use the -twoskip option to skip over alternating pages from the inputs. This can be used when the two input PDFs are the same PDF and you want to put page 1 and 2 from the input PDF on page 1 of the output PDF. May want to use the -rotate 90 option as well with this option. Use two input PDFs with this option.
	-twoskip	Used optionally along with the -twopage option. This option takes alternating pages from the input PDFs. You would typically want to use this option if both input PDFs are the same PDF. This allows you to place pages 1 and 2 from the input PDF next to each other on page 1 of the output PDF. Following would be pages 3 and 4 together on page 2 of the output PDF and so on.
	-u <i>password</i>	Sets the user password for the PDF. No password is prompted for when opening the PDF if only an owner password was specified. This will allow you to restrict users from printing, for example, without requiring a password to open the document.

Options

<u>Option</u>	<u>Description</u>
-units <i>text</i>	<p>Used to set the unit of measure you want to specify certain options in. Set this option first so latter options use the proper unit setting. Valid settings are:</p> <ul style="list-style-type: none">pt - points (1/72 of an inch)in - inchescm - centimetersmm - millimeters <p>Most of the options that use the units setting can be specified with the units setting on the option itself. This allows you to enter the units with one setting. For example, rather than:</p> <pre>-units "in" -pagesize "8.5,11"</pre> you can simply enter: <pre>-pagesize "8.5,11 in"</pre> <p>Note using this option sets the "units" value. This means if another option came after -pagesize (in this example) and that option didn't specify the units, then inches would be used.</p>
-untaint <i>number</i>	<p>Untaints file names. Use this on Unix systems if you get errors about "insecure dependency while running setgid" or want to restrict what file names may be used. The parameter takes a number from 1 to 3. A value of 1 allows the least characters and 3 the most. For example, if you don't want to allow files from other directories, use 1. Use 2 if you do allow file names that contain slashes (so directories can be used) or 3 for any character in a file name (such as ! or \$).</p> <ul style="list-style-type: none">1 - Only letters and numbers as well as -, _ and @ will be allowed2 - Same as 1 except also allow \, / and :3 - Allow all characters <p>Note this applies to all files - both input and output.</p>

Options

<u>Option</u>	<u>Description</u>
-update	Updates fillable field values by appending them to the end of the input PDF as an incremental update rather than recreate a new PDF. This is necessary when you have a rights-enabled PDF (such as allowing users of the free version of Reader to save the PDF) and you want to keep those rights after updating field values. Note the demo version of the program will not work for this as it modifies existing pages. You may only fill in field values when using this option and have only one input PDF for the rights to remain valid in the output PDF.
-updateapp	Updates fillable field values by appending them to the end of the input PDF as an incremental update rather than recreate a new PDF. This option forces form fields to be evaluated and refreshed by Reader before display.
-urlname <i>name</i>	The user name to authenticate with on websites when using a web page as input and authorization is required.
-urlpwd <i>name</i>	The password to authenticate with on websites when using a web page as input and authorization is required.
-urls <i>file</i>	File to use for modifying URLs. See the URL section for layout details. You may also pass in the tags rather than a file name. For example, -urls "".

Options

<u>Option</u>	<u>Description</u>
-viewlo <i>text</i>	<p>The page layout for the PDF viewer. Set to one of the following:</p> <p><u>SinglePage</u> Display one page at a time</p> <p><u>OneColumn</u> Display the pages in one column</p> <p><u>TwoColumnLeft</u> Display the pages in two columns, with odd-numbered pages on the left</p> <p><u>TwoColumnRight</u> Display the pages in two columns, with odd-numbered pages on the right</p> <p><u>TwoPageLeft</u> Display the pages two at a time, with odd-numbered pages on the left</p> <p><u>TwoPageRight</u> Display the pages two at a time, with odd-numbered pages on the right</p>
-xmlin <i>file</i>	<p>Loads the XML data from the file specified to the PDF. The file layout is defined as part of a framework called the Extensible Metadata Platform (XMP) and described in the Adobe[®] document XMP: Extensible Metadata Platform.</p>
-xmlout	<p>Exports the metadata XML (if any) from the PDF to the output file. In this case, use a file with a .xml extension rather than .pdf for the output. Use the -xmlin option to load in your changes. The file layout is defined as part of a framework called the Extensible Metadata Platform (XMP) and described in the Adobe[®] document XMP: Extensible Metadata Platform.</p>
-xmloutq	<p>Same as -xmlout but should work faster on larger PDFs. Does not require the owner password to be passed in on encrypted PDFs (unless there is also a open or permission password). May not work on all PDFs depending on how they are structured.</p>

Options

	<u>Option</u>	<u>Description</u>
	-zoom	The zoom factor to open the document at. Enter 100 for 100 percent.
		FITPAGE = open the document sized so the entire page fits in the window.
		FITWIDTH = open the document sized so the width of the page fits in the window.

Options by Category

Attachments

[embed](#) [embeddir](#)

Annotations

[annotes](#) [annotesout](#) [noannotes](#)
[showannotes](#)

Bookmarks

[bmadd](#) [bmannote](#) [bmauto](#)
[bmin](#) [bmkeep](#) [bmkeepnofit](#)
[bmout](#) [bmsplit](#) [bmtitle](#)
[nobm](#)

Color (Text/Lines)

[colorin](#) [colorout](#)

Data/Fields

[data](#) [fdffield](#) [fdffixed](#)
[fdfin](#) [fdfout](#) [fdfoutflds](#)
[fdfoutfldsrev](#) [fdfoutjson](#) [fdfouttypes](#)
[fdfoutxml](#) [fdfoutflds](#) [fieldbits](#)
[fields](#) [flatten](#) [flattenfile](#)
[flattenflds](#) [flattenfldsrev](#) [flattennobox](#)
[flattensig](#) [keepap](#) [keepmk](#)
[notblank](#) [removeflds](#) [richtext](#)
[ro](#) [rofields](#) [rofields](#)
[skipfieldrename](#) [update](#) [xmlin](#)
[xmlout](#) [xmloutq](#)

Digital Signature

[certfile](#) [signbgcolor](#) [signcolor](#)
[signdate](#) [signing](#) [signkeepratio](#)

Options by Category

signature	signpemfile	signpkfile
signrsn	signscript	signsize
signsrc	signssl	signtimestamp
signts		

Document

attribute	author	creationdate
creator	expire	expiremsg
fonts	hidemenubar	hidetoolbar
hidewindowui	keywords	moddate
msg	pagemode	producer
removeattr	subject	title
viewlo	zoom	

Encryption

aes	e128	noannotate
noassemble	nochange	nocopy
nodigital	noextract	nofillin
nometaenc	noprint	norights
o	pwdlist	u

Error Log

e	log
-------------------	---------------------

Google Drive/OCR

gdrivedescr	gdriveencfile	gdriveimp
gdriveinsfolder	gdriveinsid	gdrivejson
gdrivelog	gdriveocr	gdriveocrpdf
gdriveocrsave	gdriveparents	gdriveparents
gdriveshareemail	gdriveshareaccess	gdrivesharetype
gdrivetitle	gdriveupdid	gdrivewslog

Grid Drawing

grid	gridmargins	gridnum
gridcolor	gridyrev	

Options by Category

Images

allimg	imgbright	imgcolor
imgdecor	imgdecorscale	imgdecortile
imgdir	imgfaxbcolor	imgfaxfcolor
imgin	imgmargin	imgname
imgonly	imgout	imgpagepos
imgpagesize	imgres	noimg
reduce	reduceimg	reducemax
reducemin		

Layers

layersin	layersout
--------------------------	---------------------------

Mail

mail	mailauth	mailauthid
mailauthpwd	mailbcc	mailbody
mailcc	maildebug	mailfakecc
mailfakefrom	mailfaketo	mailfiles
mailfrom	maillog	mailnodialog
mailpri	mailreply	mailscr
mailsmtp	mailsub	mailto

Media

mediaimg	mediaplay	mediarepeat
--------------------------	---------------------------	-----------------------------

Open/Print

copies	noprintprefs	open
openscr	print	printcfgin
printcfgout	printcomp	printcopies
printdlg	printduplex	printer
printerlist	printpagerange	printpicktray
printscale	printscr	

Optimization

Options by Category

[opt](#)

[opt15](#)

Overlay

[autorotate](#)

[bkglayer](#)

[overlay](#)

[overlay2](#)

[overlayfile](#)

[overlaypages](#)

[pagestopdf](#)

[removebkg](#)

[repeat](#)

[repeatlast](#)

[reveroverlay](#)

[reveroverlay2](#)

[transparency](#)

Page Layout

[bottomtop](#)

[impose](#)

[objin](#)

[objout](#)

[onepage](#)

[onepageborder](#)

[onepagelo](#)

[topbottom](#)

[twopage](#)

[twoskip](#)

Page Numbering

[pagebmargin](#)

[pagefmt](#)

[pagelmargin](#)

[pagenum](#)

[pagenumalign](#)

[pagenumbgcolor](#)

[pagenumcolor](#)

[pagenumdetails](#)

[pagenumeven](#)

[pagenumfont](#)

[pagenumnrotate](#)

[pagenumodd](#)

[pagenumoutside](#)

[pagenumreset](#)

[pagenumsize](#)

[pagenumstyle](#)

[pagetop](#)

Page Size

[artbox](#)

[autoclip](#)

[bleedbox](#)

[clip](#)

[cropbox](#)

[mediabox](#)

[pagesize](#)

[autoscale](#)

[trimbox](#)

Pages

[altpages](#)

[burst](#)

[burstquick](#)

[exclude](#)

[insertat](#)

[pageduplex](#)

[pageord](#)

[pages](#)

[pages1](#)

[pages2](#)

[removestruct](#)

[removevr](#)

[textsplit](#)

[textsplitxy](#)

Placement

Options by Category

angle	autorotate	center
down	orient	right
rotate	rotateabs	scale
scalex	scaley	

Processing & Misc.

addback	allowdup	common
comp15	delblank	force
imgerr	infoout	ndkeep
noerrmsg	noxml	objsizeout
ooserver	parse1	rectin
s	shrink	skipmissing
pageinfoin	pageinfoout	units
untaint	urlname	urlpwd
urls		

Stamps

stamp	stamptext
-----------------------	---------------------------

SQL

sql	sqlcomb	sqldb
sqldriver	sqlparams	sqlpwd
sqluser		

Text Strings

noglyphpos	noglyphcomp	strchange
strcolor	strcolor2	strcolormode
strcolorary	string	strin
string	stringcase	stringlist
stringword	stringwordre	strpages
strrend	strscolor	strurl
textdetailin	textdetailout	textdetailoutxy
textdetailread	textin	textout
textscalex	textscaley	

Using the Windows DLL (Dynamic Link Library)

The DLL is compiled as a 32 and 64-bit .NET DLL. You may register it with regasm to make it available from a COM client such as VBScript. The file pdfmeld_20.dll is the compiled .NET DLL. The source code is available on [GitHub](#) if you want to make changes.

The .NET DLL is a wrapper for the executable pdfrw.exe or pdfmeld64.exe (which is the default) or as a means to send and receive data when running PDF Meld in server mode. It allows you to easily add the functionality of PDF Meld to any existing code that can access a .NET or COM DLL.

The DLL also has methods to start and stop a PDF Meld server though you may also do so from the command line (see the -server option). This provides the added benefit of keeping the program in memory for quick access without the cost of program startup and shutdown each time you build a PDF. In addition you can control the number of simultaneous builds that happen at one time to minimize memory or CPU usage. If your license allows, you may run multiple servers on the same or different boxes and the DLL will cycle requests between the running servers. Each server instance for PDF Meld is a separate license when purchasing individually.

Use the startServer method to bring up a PDF Meld server. PDF Meld then waits in memory listening for commands on a port that you specify when starting the server. The DLL sends commands to PDF Meld on that port in order to build the PDF. You may have PDF Meld save the PDF to disk or send it back to the DLL as byte array.

The setFileDataB or setFileDataH methods can be used to pass files (like images or PDFs) in from memory rather than using a file on disk. This is useful when you're generating such information elsewhere and want to pass it into the software but not have to create a file on disk. You pass in an id value, any user defined string that uniquely identifies the data, along with the data itself. Any method or tag options that use a file can then pull this data rather than read from a file. For example, setFileDataB "file1.pdf", pdfdata where "file1.pdf" is what the file is called and pdfdata is a variable or byte array containing the PDF.

The default setting for methods that take a size or set of coordinates (such as setPageSize) are points. One point is 1/72 of an inch. To set page size in points for 8.5 by 11 inches you'd use setPageSize 612,792. You may override the default unit setting by using the setUnits method. Call the setUnits method first then any methods using the setting. For example, to use inches you can first call setUnits "in" then call setPageSize 8.5,11. Units may be set

to pt for points (the default), in for inches, cm for centimeters or mm for millimeters.

Use the `setPages` method to pull pages from an existing PDF. Only use one input file in this case (or two if using the `setOverlay` method). Separate page numbers with a comma or use a - between numbers for a range. For example, `setPages "2,7,14-20,35"` will extract pages 2, 7, 14 through 20 and page 35, a total of 10 pages, from the input PDF. Use negative numbers to refer to the position from the end of the page set. For example, -1 for the last page or -3 for the third to the last page. You may also use the words "odd" or "even" to pull all the odd or even pages. Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. The `setPages` method, when used along with the `setOverlay` or `setRepeat` methods, applies only to the second input PDF.

Use the `setPageOrder` method to rearrange pages from an existing PDF. Only use one input file in this case. Separate page numbers with a comma or use a - between numbers for a range. For example, `setPageOrder "1-5,7,15,2"` will extract pages 1 through 5, 7, 15 and then repeat page 2. You may also specify a file in place of the page numbers. The file should contain comma or dash separated page numbers and may contain page numbers on multiple lines.

Use `setPageOrder "reverse"` to reverse the page ordering from an existing PDF. Only use one input file in this case. In a 10 page PDF, for example, the output PDF will start with page 10 from the input PDF, page 2 will be page 9 and so on.

Use the `setPageRight`, `setPageDown`, `setPageScale` and `setPageCenter` methods to change the position of content on each page of a PDF. Only use one input file in this case.

Use the `setAngle`, `setRotate`, `setRotateAbs` or `setOrient` methods to affect rotation of contents or page.

Use the `setOverlay` method to overlay pages from one PDF onto a second PDF. Only use two input files in this case. The first PDF specified is the background and the second will be placed on top of the first. The resulting PDF will have the same number of pages as the second PDF. The first PDF should have the same or fewer pages than the second PDF. Both PDFs should have the same page sizes. You can overlay more PDFs by re-running the overlay process using the output PDF as an input PDF along with a new input PDF if necessary. Use `setRevOverlay` to perform a reverse overlay - works similar to `setOverlay` except the second PDF is treated as a background. The `setTransparency` method can be used to allow the background page to show through on the foreground and create other effects based on the mode used.

Some PDFs, even though they view right side up in the viewer, have their contents rotated as well as the page itself rotated. When you overlay this type of PDF with another that has a different rotation (or no rotation) the result is the page contents from one of the PDFs are rotated sideways or upside down. The `setAutoRotate` method (used along with `setOverlay` or `setRevOverlay`) will attempt to compensate for this. Use this method in situations where the result of the overlay is not correct due to a rotation issue. Both PDFs should have the same page size when using this method. If this method still doesn't provide the desired results, you'll need to perform this operation manually in 2 steps by forcing a rotation on one of the PDFs then perform the overlay. That means you'll use the `setAngle`, `setPageRight` and/or `setPageDown` methods on one of the PDFs and output to a new PDF. Then take this new PDF and use the `setOverlay` or `setRevOverlay` method with the background (or foreground) PDF.

The `setRepeat` method can be used to overlay PDFs as well. This method will repeat through the first PDF, using it as a background, for as many pages as there are in the second PDF. The resulting PDF will have the same number of pages as the second PDF. The first PDF should have the same or fewer pages than the second PDF. Both PDFs should have the same page sizes. The resulting PDF from using a background PDF with 1 page and a foreground PDF with 5 pages will be 5 pages and contain the single page background PDF on each page with the second PDF overtop.

The `setStrFileIn` can be used to add text to a PDF. You can specify the placement, font size, pages to show on, color and alignment. See the [Text Files](#) section for more information.

<u>Method</u>	<u>Description</u>
<code>object buildPDF bool waitForExit = true ,String saveFile = ""</code>	<p>Call this method to build the PDF after setting all of your other options. The server will be used if <code>setServer</code> or <code>setServerFile</code> was used. Set the first parameter to true when you want to wait for Report Writer to finish building the PDF before returning from this method when using the executable (non-server) version. The <code>waitForExit</code> is used to mean wait and return the byte array of the PDF when using the server version. Optionally pass a <code>saveFile</code> which will be used to save the file locally when running as a server. You may also set <code>saveFile</code> when not using a sever to bypass the <code>setOutFile</code> method call. Returns an object (when <code>waitForExit</code> is true) with these properties: <code>byte[] Bytes</code> - the PDF output <code>String msg</code> - error message or "OK" <code>String result</code> - code (see below) <code>int pages</code> - number of pages in output</p> <p>The return values are: Success: 0 = Output PDF was built successfully 10 = Alternate parsing method used Generally, a code of 10 still worked but it's possible an input PDF contained some incorrect pointers.</p> <p>Error: -1 = Can't open input file -2 = Can't open output file -3 = Can't decrypt input PDF -4 = Can't parse input PDF -5 = No pages for output PDF -6 = Can't find OpenSSL executable -7 = Can't find private key file -8 = Can't find certificate file -9 = Can't find signature field to sign -11 = Timeout</p> <p>This method is also part of the output object when using <code>setGDriveOCR</code>: <code>byte[] getOcrFile(String mimeType)</code></p> <p>Pass one of the following for <code>mimeType</code>: "html", "docx", "txt", or "odt".</p>

<u>Method</u>	<u>Description</u>
resetOpts(bool resetServer = false)	If you using the DLL object to build more than one PDF in the same instance you should call this method to clear your current settings. That is, any DLL methods you have called so you may set them again without retaining any settings from the previous run. Optionally pass true to reset any server information you sent with setServer.
sendFileTCP String fileName ,String filePath = ""	The name and location of a file to send when using PDF Meld as a service. Use this when the server is on a different box or you don't have knowledge of what directory the service is running under. For example, you might use setInFile("somefile.pdf") where you are using "somefile.pdf" as a placeholder. You then call sendFileTCP "somefile.pdf", "c:\temp\myrealfile.pdf". This sends c:\temp\myrealfile.pdf to the server and instructs it to treat it as a file named somefile.pdf.
setAuthor(String author)	Sets the document author.
setAutoClip	Used along with setPageSize. This option will clip the edges of the page for those pages that are smaller than the output page size. For example, if an input page size is 4x6 and the new size is 6x8 and there is no scaling, this option will clip one inch on the top, bottom, left and right. The page size doesn't change but only the 4x6 area of the page shows in the middle of the new 6x8 page. This allows you trim out any printer marks or other extraneous markings.
setAutoRotate	Used to compensenate for page rotation during the overlay process. Use this method when the result of your overlay is one of the PDFs comes out sideways or upside down. This is due to the way the PDF was created and, even though it appears right side up in the viewer, actually has its contents and page rotated. This method will attempt to correct that issue.

<u>Method</u>	<u>Description</u>
setAutoScale(String text)	<p>For use with the setPageSize option. Use "D" to scale larger pages down to fit within the new page size. Either the original width or height must be larger than the new page size to scale down. Use "U" to scale smaller pages up to fit within the new page size. Both the original width and height must be smaller than the new page size to scale up. Use "UD" to do both types of scaling.</p> <p>You may also add the letter "L" when using "D". This will be used to scale to the larger of the x or y ratios. The default is to scale larger pages down by the smaller of the ratios.</p> <p>Use an "O" along with "D" and/or "U" to preserve the original orientation but rotate by 90 degrees for those pages oriented differently from the page size being set. Use a "K" along with "D" and/or "U" to preserve the original orientation but do not rotate for those pages oriented differently from the page size being set.</p>
setAutoSendFiles	<p>This is optionally used when running a PDF Meld server on a different box from where the request is being made. Used to send all the input files without using sendFileTCP for each one. You only need to set this option once. When the server program looks for a file and this option was used it will send a request back to the client requesting the file. The assumption on the server is none of the files being processed are local files.</p>
setBookmarkFile(String fileName)	<p>Path and file to use for bookmark structure. See the Bookmarks section for layout details.</p>
setBookmarkTitle	<p>Use the title of each document as the bookmark. One bookmark is created for each document in this case. Only works when merging PDFs one after another - not for overlays. The file name is used if the document title is not set.</p>
setCmdlineOpts(String text)	<p>Used to pass executable command line options into the program. You may use this option instead of or in addition to the normal methods for setting options. For example, you may pass "-pagenum -pages '2,5,8-15'" to this method instead of calling setPageNum and setPages. Any of the options listed in the executable command line option set may be used. These will override any options you set with the corresponding method.</p>

<u>Method</u>	<u>Description</u>
setComp15	Uses a compression algorithm compatible with PDF 1.5 (Acrobat 6.0). PDFs with this form of compression can be viewed only with Acrobat or Reader version 6 or higher. The reduction in size is based on the number and type of objects in the PDF but in general is around 10-20%. Not all PDFs will be reduced by the same percentage factor.
setCreationDate(String text)	Sets the document creation date. Send a string formatted as YYYYMMDDHHmmSS or pass "today" for current date/time
setCreator(String creator)	Sets the document creator. Note that Creator is labeled as Application under some versions of Adobe Reader.
setData	(<i>Not available in SE version.)</i> Specifies the input file is a tag based data file containing FDF information. The setDataPDF and setDataField methods can be used as an alternative for this method to pass data directly. See the Data section for more information on file layout and usage.
setDataField String name String value	Use to pass the NAME and VALUE options for the FDFFIELD tag. setDataPDF and setDataField methods can be used as an alternative to the setData method. These methods allow you pass the data directly without creating a tag based input file. Do not use setDataPDF and/or setDataField along with setData. See the Data section for more information on file layout and usage.
setDataPDF String src [, String options])	Use to pass in the SRC option for the PDF tag. Optionally pass in a string of options that are available with the PDF tag (from the Data section) to include. For example, setDataPDF "myfile.pdf", "KEEPNAMES". setDataPDF and setDataField methods can be used as an alternative to the setData method. These methods allow you pass the data directly without creating a tag based input file. Do not use setDataPDF and/or setDataField along with setData. See the Data section for more information on file layout and usage.

<u>Method</u>	<u>Description</u>
setEncrypt128(bool noMetaEnc = false)	<p>Sets RC4 128-bit encryption method. Files encrypted with 128-bit encryption can only be opened with Acrobat or Adobe Reader 5.0 or above. The default encryption is 40-bit which works with Acrobat and Adobe Reader 4.0 and above.</p> <p>You may optionally pass true to this method. When set, the document metadata remains unencrypted in the output PDF. This allows other applications to read the metadata while the rest of the PDF is encrypted.</p>
setEncryptAES(String 128 256)	<p>Sets AES encryption method. Pass 128 for 128-bit encryption or 256 for 256-bit encryption. Files encrypted with AES 128-bit encryption can only be opened with Acrobat or Adobe Reader 7.0 or above. Files encrypted with AES 256-bit encryption can only be opened with Acrobat or Adobe Reader 9.0 or above.</p>
setFieldsFlatten ([String "sig"])	<p>Flattens the fields in the PDF. This removes the fields and replaces the field with its text value. The effect is the PDF now contains a standard text string where the field was originally. Optionally pass the text "sig" to not flatten signature fields. This has the effect of removing all the basic input fields but leaves the signature boxes alone for signing later. See the setFieldFlatten method if you only want to flatten certain fields.</p>
setFieldsRemove	<p>Removes the fields in the PDF as opposed to flattening. There is no text replacement with the value as in setFieldsFlatten. Use with other options such as setFieldFlatten.</p>
setFileDataB String id ,byte[] byteArray	<p>Used to pass data (images, PDFs, etc.) in from a byte array in memory. Pass the byte array containing the data. The "id" value is what you will be using to refer to the file which can be the actual file name or some other unique ID. The "byte()" value is the raw file data contained in the byte array. You can set the id value to an image name (like myimg.jpg or file1.pdf) or some unique value (like myimage1). You then use the id anyplace an input file name is accepted such as with the setInFile method or the SRC parameter of an IMG tag.</p>

<u>Method</u>	<u>Description</u>
setFileDataH String id ,String hexString	Used to pass data (images, PDFs, etc.) in from a hex string in memory. Pass an ASCII hex string containing the data. The "id" value is what you will be using to refer to the file which can be the actual file name or some other unique ID. The "hexstring" value is the raw file data in hex format (0-9 and a-f/A-F characters only - any other characters will be removed before processing). You can set the id value to an image name (like myimg.jpg or file1.pdf) or some unique value (like myimage1). You then use the id anyplace an input file name is accepted such as with the setInFile method or the SRC parameter of an IMG tag.
setFileDataS String id ,String fileContents	Used to pass an ASCII string in from memory. Similar to setFileDataH except the data is in a plain ASCII string and not for binary type data. The "id" value is what you will be using to refer to the file which can be the actual file name or some other unique ID. The "string" value is the text of the file. You can use this method to pass in items such as a bookmark structure or other similar plain text file.
setFileSep(String text)	The file separator to use for the list of input files. The default is a comma if this option is not used. You may use more than one character. This method should be the first one called as some other methods such as setInFile use this value. For example: PDFMeld.setFileSep("-z-") PDFMeld.setInFile ("c:\f1.pdf-z-c:\f2.pdf")
setFontSize(Double number)	Point size of the font for the page number. Default is 10.
setGDriveDescr(String text)	Optional. A short description of your PDF when saving to Google Drive.
setGDriveInsFolder(String text)	Inserts (creates) a new folder in Google Drive to store the PDF into. The setGDriveParents in this case are used as the upper level folder for this one. Upon success the PDF is then stored in this folder.
setGDriveInsID(String fileid)	The file ID for the upload. Only use this if you have pre-generated a file ID from your Google account.

<u>Method</u>	<u>Description</u>
setGDriveJSON String path-file [,String impersonate-email]	The path and name of the JSON formatted file containing your Google Drive service credentials. Pass this if you wish to save or retrieve a document in your Google Drive. Use the command line program and the -gdriveencfile option if you want to create an encrypted version of your JSON file for use with PDF Meld. Note you may leave this off if you simply want to OCR your document and just use setGDriveOCR. Also optionally pass the user (email address) to impersonate as when using the Google Service. Note your Google Business Administrator must have already set up your Client ID as a trusted app in order to use this option. See the Google Drive section for more details.
setGDriveLog(String path-file)	The file to log the results returned from Google. This file is overwritten each time it is used. When used with setGDriveJSON it will contain the results from the Google Drive post in JSON format that include the file id and other useful information.
setGDriveOCR	Perform OCR (optical character recognition) when saving to your Google Drive account by using the setGDriveJSON option. Useful for PDFs that contain scanned images so you can retrieve the plain text. Set this if you plan on using getOcrFile once you have built your output PDF.
setGDriveOCRPDF(char letter)	Pass "A" for annotation, or "T" for text, or "P" for PDF. Runs the OCR processing against the PDF to generate the text for each page. The text is then saved back to the PDF as hidden text or an annotation. The text won't line up to the text in the image on the page but can be used to search within the PDF. Using the "P" (PDF) option returns just the text as a PDF (without the original formatting and images).
setGDriveOCRSave	Saves various formats such as rtf, HTML and docx from your PDF output named after the base output file. For example, if your output is "myfile.pdf" then, if successful, you will also have files such as "myfile.pdf.html" and "myfile.pdf.docx". The getDriveOCR method will contain the various OCR documents as well. Note you may use this option by itself without any of the other setGDrive... methods if you simply want to OCR your output PDF. Your PDF is still transmitted to Google for decoding so an internet connection is required.

<u>Method</u>	<u>Description</u>
setGDriveParents(String text)	Optional. A comma separated list of parents (folders) to add the PDF into.
setGDriveProxy(String text)	The proxy server, if needed, such as http://proxy.mycomp.com:9000. If not sure, you can generally check using Internet Explorer under Tools Internet Options Content Lan Setting.
setGDriveSave	Specify this method along with setGDriveJSON to save the output PDF to your Google Drive. See the Google Drive section for more details.
setGDriveShare String email ,String type ,String access	Use to share the PDF uploaded to Google Drive with a user or group. The email parameter is the email address of the user or group to share the PDF with. The type is the type of user the email parameter refers to - "user", "group", "domain", "anyone". If not specified this defaults to "user". The access parameter is the type of access to give - either "reader" or "writer". If not specified this defaults to "writer".
setGDriveTitle(String text)	The title of your PDF when saving to Google Drive.
setGDriveUpdID (String fileid)	The file ID for the upload. This overwrites and existing file saved in your Google Drive.
setGDriveWSLog String path-file [,String text]	The id of a Google worksheet to append a row to when PDF Meld creates a new PDF. Optionally include the message to use. If not specified a generic message like "PDF Meld Created File <filename>" is used.
setGUIOff	Suppresses the dialog window that shows the current build progress.

<u>Method</u>	<u>Description</u>
setInFile(String fileName)	<p>Used to specify the input files which may be either PDFs or images. For images, only use JPEGs at 72 DPI 256-color grayscale or 24-bit color, GIFs, or PNGs (alpha transparency in PNGs is not supported - the image will show without transparency). Some TIFF images may work as well. You may also use simple text files (.txt, .log and .dat) though no word wrapping is performed so text may be compressed if the file contains long lines. Microsoft Word, Excel, and PowerPoint documents may work as well under Windows if you have the free Microsoft Office "save as PDF" add-in installed for these products. See Microsoft's website for download information. An installation of OpenOffice can also be used to convert Excel/Word to PDF. There are several ways the input files may be specified:</p> <ul style="list-style-type: none">• Use a comma separated list of file names (no space before/after the comma). For example, setInFile ("file1.pdf,file2.pdf,file3.jpg")• Call the method multiple times. Each file will be appended to the list. For example, setInFile ("file1.pdf") setInFile ("file2.pdf")• Specify a directory rather than a file to process all PDFs in the directory. For example, setInFile ("c:\pdffiles*.pdf")• Use an '@' in front of a file name containing a list of files to process. For example, setInFile ("@mylist.dat").• Use files from the web by placing ::ext= <i>type</i> at the end. For example, setInFile ("http://www.site.com/pdfs/myfile.pdf::ext=pdf") setInFile ("file2.pdf") <p>The file should contain a list of PDF files, one entry per line in the file. Or you can use a tag based file that contains a list of PDFs along with page numbers for each. See the Input File section for more information on the layout.</p>

<u>Method</u>	<u>Description</u>
setKeywords(String keywords)	Sets the document keywords.
setLogFile(String fileName)	Use to create a log file containing the list of PDFs merged and the page number they start on. The following fields are exported, tab separated, on each line: <ol style="list-style-type: none">1. File name2. Document title3. Page #4. Current system date/time The date/time is formatted as YYYY-MM-DD-HH:MI:SS. The file name is used if the document title is missing or cannot be determined. The page number is the starting page number for that PDF in the output. Only the first PDF is listed in the log file if an overlay option was used. Use a ",a" after the file name to append to the log file rather than overwrite it. For example, setLogFile ("myfile.log,a").
setModDate(String text)	Sets the document modification date. Send a string formatted as YYYYMMDDHHmmSS or pass "today" for current date/time
setNoAnnote	Disables add/change of form fields or annotations.
setNoAssemble	(128-bit only) Disables assembly (insert, rotate, delete pages or create bookmarks) when setNoChange is used.
setNoBookmarks	Turns off pulling of bookmarks from source PDFs. This is set by default if setPages or setOverlay is used.
setNoChange	(Not available in SE version) Disables changes to the document.
setNoCopy	40-bit : Disables copying of text and/or graphics from the document. 128-bit : Disables copying of text and/or graphics from the document other than in support of accessibility to disabled users or for other purposes.
setNoDigital	(128-bit only) Disables printing at digital quality - can only print low resolution. The setNoPrint method overrides this option so you'll want to use setNoPrint or setNoDigital but not both.

<u>Method</u>	<u>Description</u>
setNoExtract	<i>(128-bit only)</i> Disables extraction of information in support of accessibility to disabled users or for other purposes.
setNoFillIn	<i>(128-bit only)</i> <i>(Not available in SE version)</i> Disables fill in interactive fields when setNoAnnote is used.
setNoPrint	Disables printing of the document (even low resolution).
setOpen	Automatically opens Acrobat and loads the newly created PDF.
setOptimize(bool compress = true)	Optimize the output PDF for fast web viewing. Note this typically increases the size of the output by a few hundred bytes or so. Pass true to further compress the contents for a slightly smaller PDF. The PDF is optimized for viewing on the web as opposed to shrinking the physical size. Additionally, you must create the PDF to a file rather than stream the output to the browser. The setting "fast web view" will be set to yes for optimized PDFs when you open in Reader and check the properties. This means the first page of the PDF is sent to the user and made viewable while the rest of the pages continue to download in the background.
setOrient(String text)	Set to P or L for portrait or landscape. This will apply a rotation of 90 degrees to any pages that are opposite of what you pass for setOrient. For example, if you use setOrient("P") then any pages that are landscape will be rotated (if not already) to portrait. The contents will remain landscape or portrait but the page itself will be rotated in the viewer. You may use 270 in front of the P or L such as setOrient("270P"). This will set the rotation to 270 degrees rather than 90 degrees for those pages that require rotation.
setOutFile(String fileName)	Full path and name of the output PDF file. Other options such as "nobuild" or "membuild" may be used as well. See the Output Options for details.
setOverlay	Used to specify overlay mode rather than append mode. This method places the contents of each page from one PDF on top of the corresponding page from a second PDF. The first input PDF is the background and the second input PDF is placed over top.

<u>Method</u>	<u>Description</u>
setOverlay2	An alternate method for overlaying pages. May work with PDFs that have problems overlaying using the setOverlay method. Also, depending on the PDFs, may execute faster than setOverlay.
setOwner(String password)	Sets the owner password for the PDF. If not specified but the user password is, this is set to the user password. Also, when not specified, the owner has only the rights granted when the document was created. So for example, if setNoPrint was specified, then it is impossible for the owner to print the document.
setPageBMargin(Double number)	The amount of space from the bottom edge of the page for the page number. Each unit is 1/72 of an inch (or the unit setting) so a number of 144 means 2 inches from the bottom edge of the page. This value is used to mean from the top of the page if setPageTop is specified.
setPageCenter	Used along with the setPageScale, setAngle or setClip methods. Specifies that the page contents should remain centered on the page. When scaling less than 100%, the page contents will generally migrate towards the lower left corner without this method and without setPageRight or setPageDown. This method overrides any setPageRight or setPageDown value. May not work on all PDFs.
setPageDown(Double number)	The distance to move the contents of each page down. The number is in units of 1/72 of an inch or the unit setting. May be positive or negative value. May not work on all PDFs. Only use one input file with this method.

<u>Method</u>	<u>Description</u>
setPageFmt(String text)	<p>Format string for the page number. Use %1 for the current page and %2 for the total number of pages. You can use %B or %nB for Bates numbering (legal document numbering with a six digit page number, padded on the left with 0's). For example, you could use "Page %1 of %2" to have "Page 1 of 3" print on the first page, "Page 2 of 3" on the second and so on. The default string is "%1/%2" (or just "%1" when the number format is roman). The n in the %nB format is the optional starting page number. For example, using "AP%312B" will number the first page as AP000312, the next as AP000313, etc. Note you don't have to include the page number if all you want to do is print a text string on each page.</p> <p>See the Page Format Variables section for information on other variables you may use.</p>
setPageLMargin(Double number)	<p>The amount of space from the left edge of the page for the page number. Each unit is 1/72 of an inch (or the unit setting) so a number of 144 means 2 inches from the left edge of the page. Note that when setPageNumAlign is used, this value is used as the padding amount on the left or right instead.</p>
setPageNum	<p>Adds page numbers to each page. Default position is bottom left hand corner.</p>
setPageNumAlign(char letter)	<p>L for left, C for center or R for right. The setting made with setPageLMargin is used as padding on the left or right rather than an absolute position when this method is used.</p>
setPageNumBGColor(String color)	<p>The color for the background of the page number string. Default is no background.</p>
setPageNumColor(String color)	<p>The color for the page number string. Default is black text.</p>

<u>Method</u>	<u>Description</u>
setPageNumDetails(String text)	<p>Pass in the details on what pages to print page numbers on, the starting page number for each range and the style. The format is a pipe separated main list with a comma separated inner list. Specify the following comma separated values for each inner list:</p> <ul style="list-style-type: none">• Starting page number• Ending page number• Number to begin range with• Through page offset (may be negative)• Style to use <p>For example, "1,5,,,r 7,-5,1" means for pages 1 to 5 inclusive, start numbering at 1 (since the current page number is used if this value is omitted) and use lowercase roman numerals. For page 6, no page number will print. Pages 7 up to 5 pages before the end of the PDF (since -5 was used - note that -1 means the last page of the PDF) will use Arabic numbering with page 7 numbered as page 1, page 8 as page 2, etc. If "1,5,,,r 7,,1" was used instead, pages 7 and on would all be numbered, no matter how many.</p> <p>Use "20,,1,-19" To start numbering at page 20 and show 1 as the page number and the through pages be the remaining pages in the PDF (-19 meaning subtract 19 from the actual total pages in the PDF). So for example, suppose there are 50 pages in the PDF. Page 20 will read "1/30", page 21 will read "2/30" and so on.</p> <p>Use setPageFmt to format how the page number and through page number should be formatted. You can also use setPageFmt to exclude the through page number.</p>

<u>Method</u>	<u>Description</u>
setPageNumFont(String text)	<p>The font to use for the page number string. Set to one of the following:</p> <p>Courier Helvetica (Default) Times-Roman Courier-Bold Helvetica-Bold Times-Bold Courier-Oblique Helvetica-Oblique Times-Italic Courier-BoldOblique Helvetica-BoldOblique Times-BoldItalic</p> <p>Or use the setFontsFile method to embed your own font.</p>
setPageNumReset	<p>Resets the current page number to 1 on each PDF. In addition, the total number of pages changes for each PDF appended to be the number of pages in that PDF rather than the total pages in the output.</p>
setPageOrder String pagelist ,bool exclude = false	<p>Specifies a new page ordering for the resulting PDF. Only use one input PDF with this method. Use a comma separated list with no spaces before or after the comma. A dash may be used to separate a range of numbers. For example, setPageOrder "1-5,7,15,2". You may also use setPageOrder "reverse" to reverse the page order from the input PDF. May also specify a file name containing the list in place of the list. Pass true for "exclude" to specify these are the pages to exclude rather than include.</p>
setPageRight(Double number)	<p>The distance to move the contents of each page to the right. The number is in units of 1/72 of an inch or the unit setting. May be positive or negative value. May not work on all PDFs. Only use one input file with this method.</p>

<u>Method</u>	<u>Description</u>
setPages(String pagelist)	A range of pages to pull from the input PDF. Use a comma separated list with no spaces before or after the comma. A dash may be used to separate a range of numbers. For example, setPages ("2,7,14-20,35"). Use negative numbers to refer to the position from the end of the page set. For example, -1 for the last page or -3 for the third to the last page. You may also use the words "odd" or "even" to pull all the odd or even pages. The odd and even options may be combined with page numbers as well, as in "1,even,-1,-2". Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. To pull pages 15 to the last page of a document you can use "15--1". May also specify a file name containing the list in place of the list.
setPageScale Double number [, Double number])	The scaling factor to apply to each page. The default is 100 or no scaling. A value of 50 will scale the contents to 50% of their original size. May not work on all PDFs. Only use one input file with this method. Pass one number to scale along both the x and y-axis. Otherwise, pass in the value to scale along the x-axis followed by the value to scale along the y-axis.
setPageSize Double x , Double y	Sets a new page size in points (1/72 of an inch) or the unit setting. For example, use setPageSize 612, 792 for a size of 8.5 by 11 inches when setUnits have not been specified. All information about the current page size and cropping is removed so setting to the same page size currently used will not necessarily display the same. You can make adjustments, if necessary, with the setPageRight or setPageDown methods.
setPageTop	Place the page numbers at the top of the page rather than the bottom. Note that setPageLMargin then specifies space from the top rather than bottom edge of the page.
setPrint	Automatically prints the newly created PDF to the default printer. Must have Acrobat or Reader installed.
setProducer(String producer)	Sets the document producer.

<u>Method</u>	<u>Description</u>
setPwdList(String text)	Used to specify a list of owner passwords (no particular order) for any encrypted input PDFs. Separate the list with a comma (be careful not to leave a space before or after the comma). Place an '@' in front of the value if it is instead a file containing a list of passwords (one password per line). Use a \ in front of a comma if the comma itself is part of the password when passing the list. You do not need to do this when using a file containing a list of passwords. Only PDFs encrypted with RC4 40-bit to 128-bit encryption or AES 128/256-bit will work. The output PDF will be unencrypted unless you're re-encrypting with the setOwner or setUser methods.
setRepeat	Used to specify overlay mode rather than append mode and repeat the first PDF. This method places the contents of each page from one PDF on top of the corresponding page from a second PDF. The first input PDF is the background and the second input PDF is placed over top. The background is repeated, if necessary, for as many pages as there are in the second PDF.
setRepeatLast	Same as setRepeat except the last page of the background PDF is repeated, not cycling again from page 1. For example, you might have a 5 page PDF that is to be overlaid with a logo contained in a second PDF. Suppose there are 2 pages in the logo PDF - page 1 has the logo for page 1 of the output and page 2 has the logo for all other pages. You would use setRepeatLast in this case to continue repeating page 2 of logo PDF in the output. Using setRepeat instead would result in page 1 of the logo PDF being used on pages 1, 3 and 5 of the output.

<u>Method</u>	<u>Description</u>
setRevOverlay	Used to specify reverse overlay mode. Works like setOverlay except the second PDF is treated as a background. The resulting PDF still has the same number of pages as the second PDF. You'll need to use a reverse overlay when you have a form field on a single page PDF you wish to overlay on multiple pages of another PDF and still retain the form field. In this case, the single page PDF should be the second PDF. Any changes to the form field on one page will be reflected in the copy of the field on all other pages. You may instead use setFieldsFlatten along with either setOverlay or setRevOverlay to flatten and repeat the field which will be converted to plain text.
setRevOverlay2	Same as setOverlay2 except the second PDF is treated as the background for overlay.
setSkipFieldRename	Prevents interactive field names from changing when merging fillable PDFs. Normally, when multiple fillable PDFs are merged, the names of the fields in each are changed so as not to conflict with each other. Calling this method turns off that feature.
setStrFileIn(String path-file)	Used to load in text from the specified tag based file. See the Text Files section for the layout. You may also pass in the tags rather than a file name. For example, setStrFileIn("<TEXT X=72 Y=72 SIZE=15>Here's some text</TEXT>").
setSubject(String subject)	Sets the document subject.
setSubset(String text)	Subsets fully embedded fonts making the output PDF smaller. This only works on PDFs where an entire font program (the TrueType file) has been embedded. Pass "quick" for a faster reduction or "scan" to scan the input file and reduce the size further. The "quick" option assumes that ASCII characters in the range of 32-127 are used and removes the rest. The "scan" option checks each page of the PDF to determine what characters are used and removes the rest. Using "scan" will generally make for a smaller output PDF though it will take longer to generate. Does not alter Unicode fonts.
setTitle(String title)	Sets the document title.

DLL Methods

<u>Method</u>	<u>Description</u>
setUnits(String text)	Used to set the unit of measure you want to specify certain options in. Call this method first so latter method calls use the proper unit setting. Valid settings are: pt - points (1/72 of an inch) in - inches cm - centimeters mm - millimeters
setUser(String password)	Sets the user password for the PDF. No password is prompted for when opening the PDF if only an owner password was specified. This will allow you to restrict users from printing, for example, without requiring a password to open the document.
setZoom(String text)	The zoom factor to open the document at. Enter 100 for 100 percent. FITPAGE = open the document sized so the entire page fits in the window. FITWIDTH = open the document sized so the width of the page fits in the window.

Output Options

The `setOutFile(text)` method is used to pass in the output file to create or to instruct the program to pass back the output as a string or perform further processing. Be sure to specify the full path name when creating an output file however.

Some methods (such as `setBookmarkOut` or `setFDFFileOut`) also use the file set here as their output when not building a PDF.

You may use the special keywords `"?"`, `"nobuild"`, `"membuild"`, `"membuildhex"` or `"save:filename"` with `setOutFile` instead of a file name.

Using `"?"` instructs PDF Meld to create a file name in the temporary directory. The return value from `setOutFile` in this case will be the file name created by PDF Meld.

Using `"nobuild"` does not build an output PDF but lets you perform tasks such as `setOpen`, `setPrint` or `setMail` on the input PDF(s).

Using `"membuild"` returns the output PDF or other file (depending on what methods you're using) to the `buildPDF` method.

Using `"membuildhex"` returns the output PDF or other file to the `buildPDF` method in hexadecimal format. Each character is represented by 2 characters in this case so you'll need to convert the hex back to a decimal then take the character value for that decimal. For example, the letter A (or ASCII value 65) will be coded as 41 (the hex value of 65). The characters will range from hex values 00 to FF. This allows you to build the PDF or other file in memory and perform other tasks with it such as stream to a browser or store in a database.

DLL Examples

Here is an example of calling the DLL using Visual Basic.

```
Set PDF = CreateObject("pdf.Meld")
' Note there two input files
PDF.setInFile "c:\temp\filein1.pdf"
PDF.setInFile "d:\mypdfs\filein2.pdf"
PDF.setOutFile "c:\temp\fileout.pdf"
PDF.setPageNum
PDF.setPageTop
PDF.setPageLMargin (144)
PDF.setPageBMargin (72)
PDF.setPageFmt ("Page %1 of %2")
PDF.setCreationDate 2002, 2, 15
PDF.setModDate 0
rslt = PDF.buildPDF
If rslt < 0 Then
    MsgBox ("Error " & rslt)
End If
Set PDF = Nothing
```

Here is an example of calling the DLL using PowerBuilder.

```
OLEObject PDF
PDF = CREATE OLEObject
li_rc = PDF.ConnectToNewObject("pdf.Meld")
PDF.setInFile "c:\temp\filein1.pdf,d:\mypdfs\filein2.pdf"
PDF.setOutFile "c:\temp\fileout.pdf"
PDF.setPageNum
PDF.setPageTop
PDF.setPageLMargin 144
PDF.setPageBMargin 72
PDF.setPageFmt "Page %1 of %2"
PDF.setOverlay
PDF.buildPDF
```

Input File Types

PDF Meld can accept a variety of file types. Some natively and others with a converter. You can mix and match these types for your input to create a single PDF from multiple sources or to use a PDF as a background to an Excel file or whatever else you need. Below is a summary of the different file types and how they are handled.

PDF

PDF Meld will handle most any PDF without an issue and no other software (such as Adobe Acrobat) is necessary to process them. You will need to pass in the owner password if the PDF is encrypted or password protected.

Images

Most JPEG, GIF, PNG, and some TIFF images will import as-is. Other image types will need to be converted manually to one of these formats (preferably JPEG).

Text

Plain text files will convert as-is. No word wrapping is performed so text may be compressed if the text contains long lines.

Excel/Word/PowerPoint/OpenOffice

PDF Meld can use these types of files if there is a way to convert them to PDF installed on the system. There are several ways this can be done.

First, if you have Excel/Word 2007 or higher with the free Office "Save as PDF" add-on installed, the Windows version of PDF Meld can connect to Excel or Word to perform the conversion.

Second, OpenOffice can be used to convert these document types to PDF. Under Windows, PDF Meld will attempt to connect to OpenOffice using the Windows COM sub-system to perform the conversion. This means that the standard OpenOffice installation should be all you need on Windows so PDF Meld can convert Excel/Word documents. The command line utility "unoconv" (part of OpenOffice) can also be used on any operating system if you have OpenOffice installed as a server. Be sure the PATH environment variable for the user running PDF Meld contains the location where unoconv is located. This is useful on Linux systems to convert Excel/Word to PDF. See the OpenOffice documentation for instructions on setting up the OpenOffice server.

While you can manually use one of these options to convert to PDF first then pass the PDF to PDF Meld, it may be more convenient to let PDF Meld run this process for you by simply passing your Office document to PDF

Input File Types

Meld. Users do not need anything special on their local machine to read or convert Excel/Word to PDF as the process will be handled by the PDF Meld server with one of these options setup on a server running PDF Meld in Client-Server TCP/IP mode.

Input File

You may use an input file that contains a list of PDFs and images along with the set of pages to use from the PDFs. This gives you more flexibility in pulling page ranges from a set of PDFs.

Each tag, or command, starts with an angle bracket < and closes with >. The tag name (always FILE in this case) comes directly after the opening <. Options are then listed, space separated, with an = sign between it and its value.

PDFs are merged in the order found in the input file. Here's a sample input file with 3 PDFs and an image - all pages from myfile3.pdf will be used in this example:

```
<FILE SRC="c:\myfile1.pdf" PAGES="2">
<FILE SRC="c:\myfile2.pdf" PAGES="3,5,9-12">
<FILE SRC="c:\myimage.jpg">
<FILE SRC="c:\myfile3.pdf">
```

Here's how you'd use the file (assume it's saved as myinput.dat):
pdfmeld.exe @myinput.dat output.pdf *[options]*

Here are the options for the FILE tag:

<u>Option</u>	<u>Description</u>
SRC="text"	The input PDF or image file name.
PAGES="text"	The page numbers to pull from the PDF. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include all pages.

Keep the following in mind when creating your tagged file:

- No spaces between the option and = sign
- No spaces between = and the value

Batch Processing

The `-batch` option or `setBatch` method allows you to process input files from a directory individually rather than as a group. The program will run once with the options given for each matching file in the input directory. The parameter for `-batch` or `setBatch` is a string with some wildcards for the input and output file.

The variable `%1` is used to refer to the input file name (without the path) and `%2` is the input file name without the extension. For example, `"%1 %2.pdf"` can be used to name the output the same as the input PDF. The output PDF will be placed in the output directory.

If you're also processing subdirectories, you can use `%3` to refer to the directory below the main one. In that case you might have `"%1 %3%2.pdf"`. If your input directory is `c:\in` and you have a PDF `c:\in\sub\test.pdf`, then `%1` is `test.pdf`, `%2` is `test` and `%3` is `sub\`. If your output directory is `c:\out`, then a directory called `"sub"` will be created if it doesn't exist and the output file will be named `c:\out\sub\test.pdf`.

For the executable, here's a sample to put page numbers on all PDFs in the `c:\in` directory:

```
pdfmeld.exe c:\in c:\out -batch "%1 %2.pdf" -pagenum
```

If the `c:\in` directory contains `file1.pdf` and `file2.pdf`, the `c:\out` directory will contain files named `file1.pdf` and `file2.pdf` with page numbers.

Here's an example appending a PDF to the end of each PDF in the `c:\in` directory and appending `"_final"` to the output PDF file name:

```
pdfmeld.exe c:\in c:\out -batch "%1,c:\legal.pdf %2_final.pdf"
```

Using the same `file1.pdf` and `file2.pdf`, the `c:\out` directory will contain `file1_final.pdf` and `file2_final.pdf`.

Here's an example overlaying a PDF with each PDF in the `c:\in` directory:

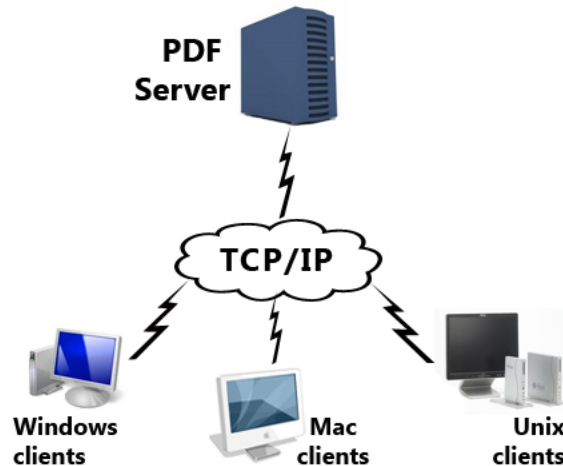
```
pdfmeld.exe c:\in c:\out -batch "%1,c:\bkg.pdf %2.pdf" -overlay -repeat
```

Note that the string for `-batch` or `setBatch` should only contain the input and the output parameters. Do not include other parameters in with this one as they won't be used. Use two `%`'s in a row (i.e. `%%1`) if you're running the program from within a DOS batch file.

Client-Server TCP/IP

Running PDF Meld as a server is a way to startup the program and have it remain idle until it receives a request via TCP/IP to perform a merge or other operation. Once it completes its request it will process any other waiting requests (unless a pool is specified to allow multiple users at once) until there

are no more. The program will then go back into wait mode until another request comes in. The advantage to running PDF Meld this way is you bypass the startup time for each run of the program. This may not be an issue if you perform a few builds each day but if you are running hundreds it could add up. In addition, the processing happens on the server so client machines are not using CPU time building reports.



There are several programs (note do not include the .exe under Unix) used for running in server mode. They are:

Server Programs

pdfmeld.exe (or pdfmeld64.exe) - used to start a server from the command line (by passing -server as the first option)

pdfmeld_srv.exe (or pdfmeld_srv64.exe) - used to install a server as a Windows service

Client Programs

pdfmeld_tcp.exe (or pdfmeld_tcp64.exe) - used to submit a client request to the server

pdfmeld_gui_tcp.exe (or pdfmeld_gui_tcp64.exe) - same as pdmeld_tcp.exe but with a progress dialog box

The -server option is used to startup PDF Meld in server mode like this:

```
C:\>pdfmeld -server -v -pool 5
-log "c:\logs\meldlog.txt" -host "localhost"
-port 7070 -licname "fytek-inc" -licpwd "abc12345"
```

-licweb

This starts the program in a DOS or Unix command session where it will remain until cancelled or a -quit command is sent. The preferred way to run under Windows is to use the [PDF Meld Service](#). The program is installed as a Windows service that any user with network access and permission may use. This section contains all the options that apply to both this method and the service.

You can run in the background like this (note the & at the end of the command) on Unix platforms:

```
$ pdfmeld -server -v -pool 5
  -log "/logs/meldlog.txt" -host "localhost"
  -port 7070 -licname "fytek-inc" -licpwd "abc12345"
  -licweb &
```

The program will startup and wait for commands on the specified port (7070 is the default if not set). The -server option must be the first option passed to the program. In addition, you must pass in your subscription (-licname and -licpwd options) or server key (-kn and -kc options). You do not need to include the subscription or server key information on client requests.

PDF Meld should then start and wait for commands. You issue commands by sending them to the TCP/IP port. Requests will be handled in sequence as they arrive unless the -pool option is used. This may allow for much faster processing as the program is already running in the background waiting for a request rather than starting up a separate process, performing its task, then shutting back down each time.

You may use any program to send the commands to the TCP/IP port. Pass BUILDPDF followed by a line feed (ASCII 10) to the port to indicate all information has been sent and PDF Meld should start processing. Or you may use the included pdfmeld_tcp.exe (pdmeld_tcp on Unix) to perform the call to the server. This program will take care of opening the port, sending the parameters you give it and including the BUILDPDF command. The program pdfmeld_tcp.exe does not build the output - it simply sends the commands to the port for processing by the server. To use pdfmeld_tcp, first start the server as described above. Pass any valid PDF Meld commands to pdfmeld_tcp and optionally include the -host and -port commands. Here is an example:

```
pdmeld_tcp -host localhost -port 7070
  file1.pdf,file2.pdf fileout.pdf
```

If the PDF Meld server is running, it will process the command otherwise an error will be returned. Note that the PDF Meld server is processing the request so you may need to provide the full path of your input and output files otherwise file names will be relative to the directory where the PDF

Meld server is running. You can also use the `-cwd` or `-currdir` options to change the working directory. Also, the files must be available from the server rather than the client. That is, if you are sending commands from a local Windows client to process on a Linux server, the PDF files must be available on the Linux server (vs. the Windows client) since PDF Meld is running on the Linux server in this case. The file pathing in this example should be based on the Linux directory structure and not Windows.

You may wish to send PDFs to the server if the PDF Meld server is running on a different computer from the client. To send files to the server for processing you will need to pass them to the TCP/IP port with a special syntax if you are writing your own program (`pdfmeld_tcp` handles this behind the scenes for you). Issue the command `-send --binaryname--<filename>--binarybegin--<binary data here>--binaryend--`. Note that base64 encoding may be used as well - substitute the text "base64" for "binary" in `binaryname`, `binarybegin`, and `binaryend`. The `<filename>` must match the name of a file being processed as input. The binary data for that file can come from any file on the client you wish to use to represent that file. For example, here's how you would pass 2 files (using Perl syntax):

```
use IO::Socket;
my $host = 'localhost'; # host server is running on
my $port = '12345'; # port server is running on
my $sock = new IO::Socket::INET (
    PeerAddr => $host,
    PeerPort => $port,
    Reuse => 1,
    Type => SOCK_STREAM,
    Proto => 'tcp',
);
print $sock "a.pdf,b.pdf t.pdf -send
--binaryname--a.pdf--binarybegin--(a.pdf contents)--binaryend--";
print $sock " -send
--binaryname--b.pdf--binarybegin--(b.pdf contents)--binaryend--";
print $sock "\nBUILDPDF\n";
```

You may use the option `-return` to receive the file back via TCP/IP from the PDF Meld server. Specify the path and file name you wish to store the output under on the client. The output will not be stored on the server in this case. This allows you to receive the output PDF on the client side that you can then save or process accordingly. Of course, the larger your files the longer it will take to process as your connection speed will play a role in the time it takes to send and receive large PDFs.

The `pdfmeld_tcp` program makes it easier to accomplish the above when transferring files. You may use `-send a.pdf=filename.pdf` where `filename.pdf` is the name of the PDF to send. The program will take care of sending the contents of the file in this case. For example:

PDF Meld

Client-Server TCP/IP

```
pdfmeld_tcp a.pdf,b.pdf t.pdf -send a.pdf=c:\myfile.pdf  
-send b.pdf=c:\another.pdf -return c:\out.pdf
```

In this case, a.pdf is the client file c:\myfile.pdf and b.pdf is c:\another.pdf. The output as referenced by t.pdf on the server will be sent back to the client and saved as c:\out.pdf. The file t.pdf will not be stored on the server in this case. The data will come back over the same socket connection as binary data if you are writing your own program to communicate with the server. The content length will be passed back first formatted as "Content-Length: n" where n is the number of bytes followed by a blank line and then the data stream. Once the port is closed that is the end of the file.

You may use -sendcache filename.pdf to send the file only the first time you call the server program. The filename.pdf should be the same path and file name of one of your input files. The server will cache the file the next time you need it on future calls to the server. Include the -sendcache option each time you run the program with the file name even though the file will only need to be transmitted once. This can be useful when you have the same background PDF, for example, you wish to reuse many times.

Do not include interactive options such as -open as part of the commands sent to process unless PDF Meld server is running locally. Otherwise, the PDF will open on the remote server which is probably not what you intend.

Here are some entries from the log file. In this case, there are 5 simultaneous processes allowed at any one time. The number in parenthesis such as the (1) and (2) below are the pool ids. For example, pool id 1 is used to start a build. While this build is happening, another request comes in to build a PDF. The "result: 0" line is the result status with 0 being a successful build. If there were any issues with the input PDFs then the result would be some number other than zero.

```
[2010-07-24 16:00:44] Creating pool of 5 entries  
[2010-07-24 16:00:44] Accepting commands on port 7070  
[2010-07-24 16:01:59] (1) (127.0.0.1) a.pdf a2.pdf  
[2010-07-24 16:02:00] (2) (127.0.0.1) b.pdf b2.pdf  
[2010-07-24 16:02:03] (2) (127.0.0.1) result: 0  
[2010-07-24 16:02:03] (1) (127.0.0.1) result: 0  
[2010-07-24 16:02:11] (1) (127.0.0.1) -quit
```

The return result values are:

Success:

0 = Output PDF was built successfully

10 = Alternate parsing method used

Generally, a code of 10 still worked but it's possible an input PDF contained some incorrect pointers.

Error:

-1 = Can't open input file

- 2 = Can't open output file
- 3 = Can't decrypt input PDF
- 4 = Can't parse input PDF
- 5 = No pages for output PDF
- 6 = Can't find OpenSSL executable
- 7 = Can't find private key file
- 8 = Can't find certificate file
- 9 = Can't find signature field to sign
- 11 = Timeout

The following are the options to use when setting up PDF Meld to run as a server. Remember to also include your key name/code combination using `-kn` and `-kc` or your software subscription information with `-licname`, `-licpwd`, and `-licweb`.

<u>Option</u>	<u>Description</u>
<code>-server</code>	Used to specify server processing mode. Must be the first option passed.
<code>-host <i>hostname</i></code>	The host name of the computer. The default is localhost.
<code>-port <i>number</i></code>	The port number to use. The default is 7070. You may want to setup a descriptive name in <code>etc/services</code> to use instead. For example: <code>pdfmeld 7070/tcp</code> Then, use <code>-port pdfmeld</code> . By adding this entry in the services files on your clients, you can connect in the same manner by using <code>-port pdfmeld</code> . The server will not start if the port is already in use.
<code>-pool <i>number</i></code>	Optional. Pass the number of simultaneous builds to allow at a time. You should start with 5 and increase if you find users are waiting on connections. The log file will show the pool id number used for each build. If you see the maximum number of pool entries being used most of the time then you may want to increase. Keep in mind more processor time will be needed to handle more simultaneous requests so you'll need to balance the two.
<code>-log <i>path-file</i></code>	Optional. The path and name of a file to log requests to.
<code>-logmax <i>number</i></code>	Optional. The maximum size in bytes for a logfile. Once the file reaches the specified size it is renamed with the current date/time appended to the end and a new log file is started.
<code>-stacksize <i>number</i></code>	Optional. Number of bytes to allocate to each pool entry. The operating system will default (typically between 8 and 16 megs) if this is not specified.
<code>-v</code>	Optional. Echoes requests to the screen. Not used when running PDF Meld Service.

PDF Meld

Client-Server TCP/IP

The client programs `pdfmeld_tcp` and `pdfmeld_gui_tcp` have the same options as PDF Meld. There are a few additional options you may use shown in the following table. There are DLL and .NET DLL versions available on the FyTek website as well as versions for all compiled operating systems at <https://www.fytek.com/#/detail/clienttcp>. The DLL and .NET DLL versions also contain all of the standard [DLL methods](#) available with PDF Meld. The actual location of the PDF Meld server doesn't matter when using the client DLL in Windows. That is, the PDF Meld server itself may reside on a Linux box but you can use the client DLLs under Windows to call the server. The DLL object to create in your code is `PDFMeld.TCP` or `PDFMeld.GUITCP` depending on whether you are using the command line or GUI version. The method to call once all others have been set is "runPDF" to start the build.

<u>Option</u>	<u>Description</u>
-host <i>host</i>	The host PDF Meld server is running on.
-port <i>number</i>	<p>The port number PDF Meld server is listening on. The default is 7070. You may want to setup a descriptive name in <code>etc/services</code> to use instead. For example:</p> <pre>pdfmeld 7070/tcp</pre> <p>Then, use <code>-port pdfmeld</code>. By adding this entry in the services files on your clients, you can connect in the same manner by using <code>-port pdfmeld</code>.</p> <p>You may also setup a file of ports if you have multiple server instances listening on various ports. In this case, type each port on a single line in a file then reference the file name with a '@' in front. For example:</p> <pre>pdfmeld_tcp <options...> -port @/tmp/myports.dat</pre> <p>The file will need to be writable by the user running <code>pdfmeld_tcp</code> as it will take the top entry for the port, move it to the bottom of the list and re-create the file. In this way, each time <code>pdfmeld_tcp</code> is called it will cycle through the list.</p>
-clopen	Client open. Opens the output PDF file in reader on the local machine. You may use <code>-open</code> if you are running off of the same box PDF Meld server is running on.
-clprint	Client print. Prints the PDF to the default printer on the local machine. Using <code>-print</code> will print the PDF to the default printer from the machine PDF Meld server is running on.

PDF Meld

Client-Server TCP/IP

<u>Option</u>	<u>Description</u>
-currrdir	Sets the working directory for the server or service to be the current directory. That way your file pathing can be relative to the directory you are currently in and not from where the server or service is running from. This will likely only work if you are running off the same machine the server is running on.
-send <i>name=path-file</i>	Used to send files to the machine PDF Meld server is running on. Set the name to an input file name and path-file to the path and file name you wish to use for that file. For example, "pdfmeld_tcp a.pdf,b.pdf t.pdf -return t.pdf -send a.pdf=c:\temp\afile.pdf -send b.pdf=c:\temp\bfile.pdf". The file a.pdf will be taken as c:\temp\afile.pdf and b.pdf will be taken as c:\temp\bfile.pdf. These will be sent to the server for processing.
-sendcache <i>path-file</i>	Used to send a file once to the machine PDF Meld server is running on. Once cached, the server will read from a copy it has kept for itself rather than ask for the file to be transmitted again. Include this option with the same file on each build that you wish to use it. The server cache is cleared once the server program is restarted.
-autosend	Used to send all the input files without using -send or setSend for each one. You only need to set this option once. When the server program looks for a file and this option was used it will send a request back to the client requesting the file. The assumption on the server is none of the files being processed are local files.
-return <i>filename</i>	Used to return to your local machine the output PDF from PDF Meld. The filename specified should be a local file to save the PDF under.

<u>Option</u>	<u>Description</u>
-serverstat	<p>Returns a report of the server status. The report contains the following information:</p> <pre>Current date time : 2010-08-01 15:00:00 Server started : 2010-08-01 12:00:00 Requests received : 75 Bytes received : 4090 Bytes sent : 0 Pool size : 5 Available pool threads : 5 Highest pool thread use: 2 Requests that waited : 0</pre> <p>"Server started" = the date and time the server was started. "Requests received" = the total number of requests the server has received to process. "Bytes received" = the total number of bytes sent into the server for requests. "Bytes sent" = the total number of bytes in returned PDFs sent back to clients. "Pool size" = the total number of pool entries the server was started with. "Available pool threads" = the current number of available threads. "Highest pool thread use" = the most threads that were in use at any one time. "Requests that waited" = the total number of requests that have had to wait for an entry in the pool to become available in order to run.</p>
-wait	<p>Set this option to cause the non-GUI version of pdfmeld_tcp to wait until PDF Meld has finished processing before returning. Normally, when you are not receiving back the resulting PDF, pdfmeld_tcp will simply send the request to the TCP/IP port and not wait for PDF Meld to perform its processing. This option causes the program to wait until finished so you know the PDF has been built and you can take some further action with it.</p>

PDF Meld Service

The PDF Meld service is another option when running PDF Meld as a server under Windows. See the [Client-Server](#) section as the details on the various parameters are covered there. The difference with running as a service is the server program is available to anyone with network access to the server. Plus you don't need to manually start up PDF Meld in server mode each time you log in. The service can be set to start whenever the machine is booted so it can be made available without logging in first.

The program `pdfmeld_srv.exe` (or `pdfmeld_srv64.exe` for 64-bit) is the program for the service. You pass in `-install` as the first option (rather than `-server` like when running `pdfmeld.exe`) followed by the normal options (such as `-pool` or `-host`) that you would use to start in server mode. You'll likely need administrative privileges in order to initially setup the service. Select the "Run as Administrator" option for the DOS box when you go to install.

You'll need to allow TCP traffic on the port if you want to make the service available to other computers. Go into your Windows firewall program and create an entry to allow traffic on that port. You can restrict access by computer and/or user if you like.

Note you still need to pass in a key name/code combination using `-kn` and `-kc` or your software subscription information with `-licname`, `-licpwd`, and `-licweb`.

For example:

```
C:\>pdfmeld_srv.exe -install auto -pool 5
-log "c:\logs\meldlog.txt" -host "mymachine"
-port 7070 -licname "fytek-inc" -licpwd "abc12345"
-licweb
```

Replace "mymachine" in `-host` with the actual name of your computer or leave out `-host` to use the default of localhost. This should start up the service and you will then be ready to start servicing requests. Other options you can use are:

```
c:\>net start PDFMeldSrv
```

This will start the service if `-install` is used without the "auto" option. For example, you can run `pdfmeld_srv.exe -install` to simply install the service without starting it.

To stop the service run:

PDF Meld Service

```
c:\>net stop PDFMeldSrv
```

This will stop the service.

To remove the service run:

```
C:\>pdfmeld_srv.exe -remove -service PDFMeldSrv
```

This will remove or un-install the service.

Be sure to fully qualify your file names as the service is not running out of the directory you are running the program pdfmeld_tcp from. You can also use the `-cwd` or `-currdir` options to change the working directory. The [Client-Server](#) section also discusses how you can send files from a remote machine to the server running the service. You can use the `-send` and/or `-return` options with pdfmeld_tcp in order to send and receive your files to and from the server.

The options for startup are the same as those found in the [Client-Server](#) section. The following are additional options for the service.

<u>Option</u>	<u>Description</u>
<code>-username <i>username</i></code>	Optional. The username to run the service as.
<code>-password <i>password</i></code>	Optional. The password for the username. You may leave this option off and, if <code>-username</code> is passed, the program will prompt for the password.

Multiple Call

The `-pdfmeld` option (or `resetOpts` DLL method) is used to break up sets of commands for a related PDF build. For example, you may have a situation where you typically overlay 3 files. Since the `-overlay` option only accepts 2 files at a time, you would need to run `pdfmeld.exe` twice. This option can be used to still run twice, but all in one call so it runs faster than if you run the program twice from the command line. Note this option is not available when running PDF Meld in server mode (via TCP/IP).

For example, to take these 3 calls:

```
pdfmeld.exe f1.pdf,f2.pdf f3.pdf -overlay  
pdfmeld.exe f3.pdf,f4.pdf f5.pdf -overlay  
pdfmeld.exe f5.pdf,f6.pdf f7.pdf
```

And convert to one call, you'd run this instead:

```
pdfmeld.exe f1.pdf,f2.pdf f3.pdf -overlay -pdfmeld  
f3.pdf,f4.pdf f5.pdf -overlay -pdfmeld  
f5.pdf,f6.pdf f7.pdf
```

The `-pdfmeld` option is the separator between subsequent calls to the program. The above command would be run without the line breaks shown, it is a single command to execute. The steps are run in sequence, not parallel. You can have several `-pdfmeld` options but each set of commands is processed before starting the next set. There is no limit in PDF Meld on the number of command sets you may pass but your operating system or environment settings may limit you on how much can be passed on the command line. You only need to pass the `-kn` and `-kc` server key-code options (if used) once on the first call (that is, before the first `-pdfmeld` option) and they will be retained.

You may also want to place the set of calls in a script and pass the file names in. For example, you could create a batch (`.bat`) file in DOS with the commands:

```
pdfmeld.exe %1,%2 %3 -overlay -pdfmeld %3,%4 %5  
-overlay -pdfmeld %5,%6 %7
```

Then run the batch file and pass in the actual file names. You'd pass in 7 PDF file names to the batch script in the example above.

You'll notice the most speed improvement when you have lots of command sets linked and are running a high volume of PDFs. Note that this option is meant for one logical group of PDFs to be processed - not several groups of unrelated PDFs. The output PDF from one set is used in the next step in the

Multiple Call

example above (though you don't have to use it that way). See the [Group Processing](#) section if you have groups of unrelated PDFs. You can combine `-pdfmeld` and `-groupstart` for volume processing.

Group Processing

The `-groupstart` option (or `resetOpts` DLL method) allows you to call the program and keep it active while passing different PDFs to the program. This is used to avoid the startup processing when the program is launched if you are converting many PDFs. You use the `-groupend` option to signal that you are finished and the program should shut down. For example, suppose you have a set of calls you want to make such as:

```
pdfmeld f1.pdf,f2.pdf f3.pdf
pdfmeld g1.pdf,g2.pdf g3.pdf
pdfmeld k1.pdf,k2.pdf k3.pdf
```

You can instead set up a file with the commands to be sent, like this:

```
f1.pdf,f2.pdf f3.pdf
g1.pdf,g2.pdf g3.pdf
k1.pdf,k2.pdf k3.pdf
-groupend
```

Each set of parameters to pass is on its own line. The line breaks are used to break up the calls to the program. Each line is taken as the set of commands to pass for that particular run.

From the DOS command line, pipe the file contents into `pdfmeld`:

```
type merge.dat | pdfmeld.exe -groupstart
```

Or, from Linux/Unix:

```
cat merge.dat | pdfmeld -groupstart
```

Where `merge.dat` is the file containing the commands above. Note the `-groupstart` option must be the first parameter passed to `pdfmeld` and must be included on the call to `pdfmeld` rather than in the `.dat` file. Also see the [Multiple Call](#) section for information on the `-pdfmeld` option. You can combine `-pdfmeld` and `-groupstart` for volume processing.

Multiple FDF Processing

The `-fdfin` option (or `setFDFFileIn` method) allows you to pass in a file containing field names and values to merge with a fillable PDF containing those fields. In this case there is a single input file containing the fields/values to set.

PDF Meld also allows you to merge many field/value files with a single PDF and create a new filled in PDF for each field/value file. For example, you might have 3 files of information and you want to merge each with the same base fillable PDF. In a real world situation you could simply run the program 3 times to create the output. However, this would be tedious if you had hundreds or thousands of field/value files you wanted to merge.

There are two ways you can specify a set of FDF files to process automatically. The first is to create a flat file containing the path and name of each field/value (or FDF) file you want to merge. In our example, assume this file called `mymerge.ini` contains the following:

```
c:\fields\file1.fdf
c:\fields\file2.fdf
c:\saveddata\mydata.fdf
```

You would then run PDF Meld like this:

```
pdfmeld mypdf.pdf c:\out -fdfin @mymerge.ini
```

Use an '@' in front of the file name containing the list of field/value files to merge (the `@mymerge.ini` file above). The "c:\out" parameter is where you would normally give an output file name. In this case, it is a directory. This directory will contain 3 files when the program finishes - `file1.pdf`, `file2.pdf` and `mydata.pdf`. The `file1.pdf` will be the result of merging `mypdf.pdf` with `file1.fdf`, `file2.pdf` contains data from `file2.fdf` and `mydata.pdf` contains `mydata.fdf` data.

The second way to process a set of files is to point `-fdfin` to a directory containing files to merge. In this example, assume `c:\data` contains the following files:

```
file1.fdf
file2.fdf
mydata.fdf
```

You would then run PDF Meld like this:

```
pdfmeld mypdf.pdf c:\out -fdfin c:\data
```

Multiple FDF Processing

Again, 3 PDF files will be created in the c:\out directory that match the name of the input files. Only file names that end with .fdf will be considered when using a directory.

Page Format Variables

The `-pagefmt` option or `setPageFmt` method allow for the use of variables in the format string. These variables are replaced with their values at runtime. A percent sign (%) is used to denote a variable.

The variable `%1` represents the current page number and `%2` is the total number of pages. You may use `%B` or `%nB` for Bates numbering (legal document numbering with a six digit page number, padded on the left with 0's).

For example, you could use "Page %1 of %2" to have "Page 1 of 3" print on the first page, "Page 2 of 3" on the second and so on. The default string is "%1/%2" (or just "%1" when the number format is roman). The `n` in the `%nB` format is the optional starting page number. For example, using "AP%312B" will number the first page as AP000312, the next as AP000313, etc. Note you don't have to include the page number if all you want to do is print a text string on each page.

Use two %'s in a row if you're running the program from within a DOS batch file.

<u>Variable</u>	<u>Description</u>
<code>%1</code>	The page number from the current PDF. This is reset with each PDF in the input list.
<code>%2</code>	The total number of pages in the output PDF.
<code>%B</code> or <code>%nB</code>	Bates numbering.
<code>%file</code>	The file name of the input PDF.
<code>%pathfile</code>	The full path and file name of the input PDF.
<code>%relfile</code>	The requested or relative path and file name of the input PDF.
<code>%yy</code>	The year (2 digits).
<code>%yyyy</code>	The year (4 digits).
<code>%m</code>	The month.
<code>%mm</code>	The month (2 digits).
<code>%mon</code>	The month name (3 letter abbreviation).
<code>%month</code>	The month name.
<code>%d</code>	The day.
<code>%dd</code>	The day (2 digits).

Page Format Variables

<u>Variable</u>	<u>Description</u>
%wd	The day of the week (3 letter abbreviation).
%weekday	The day of the week.
%h	The hour.
%hh	The hour (2 digits).
%hh24	The hour (24 hour clock).
%mi	The minutes (always 2 digits).
%ss	The seconds (always 2 digits).
%am	AM or PM.
%varname	Replace "varname" with an environmental variable defined for the user running the software (for example, %username).

Specifying Colors

Various options and methods are used to supply a color. For example, `-strcolor` or `setStringColor`. In addition, certain tags such as `<INPUT>` that are used in files have options that take a color as the value. Colors in all these cases may be entered in any of the following ways:

- You may specify the red, green and blue components as values from 0 to 255, separated by a comma. In this case 0,0,0 is black and 255,255,255 is white.
- You may specify the red, green and blue components as a hex string preceded by a # sign. In this case #000000 is black and #FFFFFF is white.
- You may specify one of the colors from the table below.

Color	Name	Color	Name
	Black		Green
	Silver		Lime
	Gray		Olive
	White		Yellow
	Maroon		Navy
	Red		Blue
	Purple		Teal
	Fuchsia		Aqua

You may use CMYK (Cyan, Magenta, Yellow and Black) for string colors as well. In this case, you use four numbers for the color setting instead of three. Colors in these cases may be entered as follows:

- You may specify the cyan, magenta, yellow and black components as values from 0 to 255, separated by a comma. In this case 0,0,0,255 is black and 0,0,0,0 is white. Other examples are 0,0,255,0 for yellow and 0,255,255,0 for red.
- You may specify the cyan, magenta, yellow and black components as a hex string preceded by a # sign. In this case #000000FF is black and #00000000 is white. Other examples are #0000FF00 for yellow and #00FFFF00 for red.
- You may use the Pantone® color chart on the following page. In this case, pass the code for color shown such as "486" or "black 5". (Pantone is a registered trademark of Pantone, Inc.)

Colors

Use the number or name shown as the color value such as "100" or "PROCESS CYAN". Pantone® colors shown are converted to CMYK values.

Color chart table with 20 columns and 72 rows of color swatches and labels.

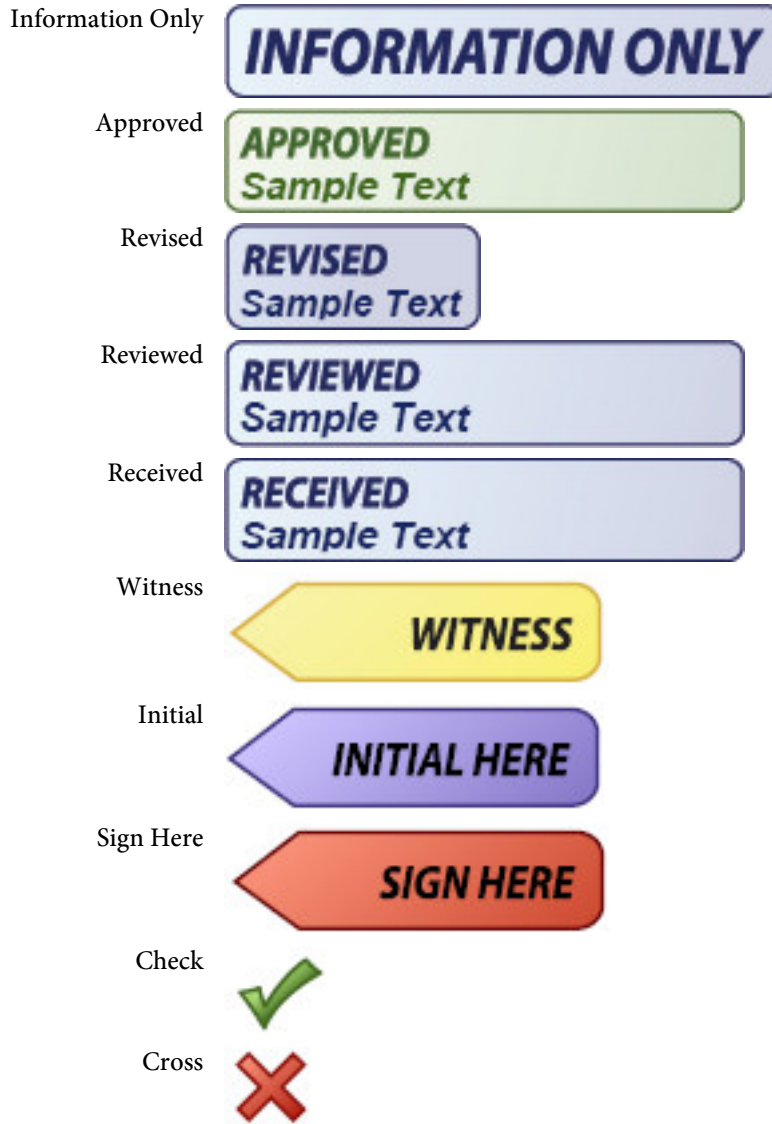
Stamps

You may need to include the `-noapp` option or `setNoApp` method when adding stamps to a PDF to prevent Acrobat from prompting to save changes when closing. Do not include it when you have other input fields in the PDF.

Below are examples of the various stamps that can be use using `-stamp` or `setStamp`.

Approved	
Not Approved	
Draft	
Final	
Completed	
Confidential	
For Public Release	
Not For Public Release	
For Comment	
Void	
Preliminary Results	

Stamps



File Layouts

This section describes the various file layouts used by PDF Meld. These are tag based files similar to HTML. Many of the options and methods have similar named counterparts for creating and importing from these files. For example, the `-bmout` option for creating a bookmark file from an exiting PDF and `-bmin "file"` to import a bookmark file.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (such as `BOOKMARK`) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<BOOKMARK LEVEL=2 CLOSED PAGE=5 DESCR="A Bookmark">
```

Note that files can be local or from the web. You may pass a web URL along with the file type (text in the case of these file type). For example, you can use `-strin "c:\file.txt"` or `-strin "http://www.mysite.com/file.txt::ext=txt"`. The `::ext=txt` tells PDF Meld that this is file to pull from the Web and that it is a text file type.

Note you must place a slash in front of any `<` or `>` character when you want to use it as part of the option text. For example:

```
<OUTLINE LEVEL=1 DESCR="Click here \>" PAGE=5>
```

Or you may use `<` and `>` instead such as:

```
<OUTLINE LEVEL=1 DESCR="Click here &gt;" PAGE=5>
```

Lastly, you may supply your own string begin and terminator sequence. Use the `SEP` tag to enter the string itself. For example:

```
<SEP LEFT="qx" RIGHT="qz">
```

The value for `RIGHT` defaults to `LEFT` when not supplied. This can be even more useful when you have a string of JavaScript code to enter and do not want to escape your `<` and `>` characters. Using the above tag, you could then enter something like this:

```
<OUTLINE LEVEL=1 DESCR=qxClick here >qy PAGE=5>
```

Keep the following in mind when creating or updating your tagged file:

- No spaces between the option and `=` sign
- No spaces between `=` and the value

Page Layout Files

The `-pageinfoin/-pageinfoout` or `setPageInfoFileIn/setPageInfoFileOut` methods allow you to modify page sizes and rotation in a PDF. The base file is created by using the `-pageinfoout` option or `setPageInfoFileOut` method. A set of tags will be created in the output file based on the page settings found in the PDF. You may then change the settings to modify the page sizes or rotation on an individual basis.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `PAGEINFO` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<PAGEINFO PAGE=1 MEDIABOX="0,0,612,792">
```

is a tag to set the page size to 8.5 by 11 inches.

Note there are two different tags, `PAGEINFO` and `PAGELABEL`. `PAGEINFO` is to set the page size and `PAGELABEL` is to set the style of page numbers used by the viewer application. Only the `PAGEINFO` tags are exported. You may use either tag or both in your input file when loading. In addition you may use the `UNITS` and/or `SHADING` tag.

Here's a sample file:

```
<UNITS VALUE="in">
<PAGELABEL PAGE=1 STYLE="R">
<PAGELABEL PAGE=3 STYLE="a" START=1>
<PAGEINFO PAGE=1 MEDIABOX="0,0,8.5,11"
  CROPBOX=".5,.5,8,10.5">
<PAGEINFO PAGE=2 MEDIABOX="0,0,8.5,11">
<PAGEINFO PAGE=3 MEDIABOX="0,0,8.5,11" ROTATE=90>
```

Here's how to create a page layout file:

```
pdfmeld.exe input.pdf layout.txt -pageinfoout
```

And here's how to load in any changes made to it:

```
pdfmeld.exe input.pdf output.pdf -pageinfoin layout.txt
```

Keep the following in mind when updating your tagged file:

- No spaces between the option and `=` sign
- No spaces between `=` and the value

```
<PAGEINFO  
  PAGE=number  
  MEDIABOX="x1,y1,x2,y2"  
  CROPBOX="x1,y1,x2,y2"  
  BLEEDBOX="x1,y1,x2,y2"  
  TRIMBOX="x1,y1,x2,y2"  
  ARTBOX="x1,y1,x2,y2"  
  ROTATE=number>
```

Here are the options for the PAGEINFO tag:

<u>Option</u>	<u>Description</u>
PAGE=number	The page number the settings refer to.
MEDIABOX="x1,y1,x2,y2"	The rectangle defining the boundaries of the physical medium the page is intended to be displayed or printed. The values are in points (1/72 of an inch) or the unit setting.
CROPBOX="x1,y1,x2,y2"	The rectangle defining the clipped (cropped) area to display or print. The values are in points (1/72 of an inch) or the unit setting. Default is the media box setting.
BLEEDBOX="x1,y1,x2,y2"	The rectangle defining the clipped (cropped) area to display or print in a production environment. The values are in points (1/72 of an inch) or the unit setting. Default is the crop box setting.
TRIMBOX="x1,y1,x2,y2"	The rectangle defining the finished page after trimming. The values are in points (1/72 of an inch) or the unit setting. Default is the crop box setting.
ARTBOX="x1,y1,x2,y2"	The rectangle defining the extent of the page's meaningful content as intended by the page's creator. The values are in points (1/72 of an inch) or the unit setting. Default is the crop box setting.
ROTATE=number	The angle of rotation for the page. Valid values are 0, 90, 180 and 270.


```
<PAGELABEL  
  PAGE=number  
  STYLE=text  
  PREFIX="text"  
  START=number>
```

The PAGELABEL tags, if used, must be in order from lowest page number to highest. This refers to the PAGE=number option in the tag. Here are the options for the PAGELABEL tag:

<u>Option</u>	<u>Description</u>
PAGE=number	The starting page number the settings refer to. You don't need a separate PAGELABEL tag for each page but rather one for the first page in each section you want to apply the settings to. The settings remain in effect until the the end of the PDF or the next PAGELABEL tag is found.
STYLE=text	The numbering style to use. Specify one of the following: D Decimal Arabic numerals R Uppercase roman numerals r Lowercase roman numerals A Uppercase letters (A to Z for the first 26 pages, AA to ZZ for the next 26, and so on) a Lowercase letters (a to z for the first 26 pages, aa to zz for the next 26, and so on)
PREFIX="text"	Optional prefix to use for the pages (like "A-" for example).
START=number	The starting number to use for the numeric portion of the page number. For example, to start numbering page 1 at 1000 set START=1000 and PAGE=1. The default is the PAGE value.

For Example:

```
<PAGELABEL PAGE=1 STYLE="r" >  
<PAGELABEL PAGE=5 STYLE="D" >  
<PAGELABEL PAGE=80 PREFIX="A-" START=1 >
```

Pages 1-4 will be labeled as i, ii, iii and iv. Pages 5-79 will be numbered and starting on page 80 the numbering will start over with 1 and the text "A-" in front (A-1, A-2, etc.).

```
<SHADING  
  NAME="text"  
  PAGES="text"  
  COLOR1=color  
  COLOR2=color  
  COLOR3=color  
  COLOR4=color  
  COLOR5=color  
  COLORARY=text>
```

Used to define a gradient shading pattern for pages in the PDF. You may specify from two to five colors. Only one SHADING tag is allowed in the input file. The last one will be used if multiples are found.

<u>Parameter</u>	<u>Description</u>
NAME="text"	The name for the shading pattern. Must be unique within the document.
PAGES="text"	The list of pages to perform the shading on. Enter a comma separated list and/or use a - (minus sign) for a page range. May also use "odd" or "even". Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. For example, PAGES="10-300,oddrange" will only shade odd pages in the range of 10-300 (11, 13, ... 299). In comparison, PAGES="10-300,odd" will shade all odd pages and all pages within the range of 10-300. Leave PAGES out to include on all pages.
COLOR1=color	The starting color. Any valid color code.
COLOR2=color	Any valid color code.
COLOR3=color	Optional. Any valid color code.
COLOR4=color	Optional. Any valid color code.
COLOR5=color	Optional. Any valid color code.

<u>Parameter</u>	<u>Description</u>
COLORARY=text	<p>A comma separated list of 4 or 6 numbers. The default is 0,0,1,0. These represent the X_0, Y_0, X_1, Y_1 matrix coordinates for the shading pattern. A matrix of 0,0,1,0 goes from left to right from COLOR1 to COLOR_n. A matrix of 0,0,0,1 goes from top to bottom. You may use decimals or negative numbers as well to offset where the middle of the gradient lies.</p> <p>Use 6 numbers x_0, y_0, r_0, x_1, y_1, r_1 for a radial type shading, such as for a sphere. The numbers specify the centers and radii of the starting and ending circles, expressed in points. The radii r_0 and r_1 must both be greater than or equal to 0. If one radius is 0, the corresponding circle is treated as a point; if both are 0, nothing is painted. The x_0, y_0, x_1 and y_1 values should typically be between 0 and 1. Use $x_0=.5$, $y_0=.5$, $x_1=.5$ and $y_1=.5$ to center the circles in the middle.</p>

<UNITS
VALUE=*text***>**

Used to set the unit of measurement for tags that follow. Typically you'll place one UNITS tag at the top but you may use them through the file with different values. The UNITS tag affects all tags that follow it until another UNITS tag is found.

For example, to set centimeters use `<UNITS VALUE="cm">`. The default is "pt" for points.

<u>Parameter</u>	<u>Description</u>
VALUE=	Use one of the following: <u>PT</u> Points (1/72 of an inch) <u>IN</u> Inches <u>CM</u> Centimeters <u>MM</u> Millimeters

Color Files

The `-colorin/-colorout` or `setColorFileIn/setColorFileOut` methods allow you to modify text and line drawing colors in a PDF. The base file is created by using the `-colorout` option or `setColorFileOut` method. A set of tags will be created in the output file based on the various colors found in the PDF. You may then change the `NEW` setting to replace the `OLD` values in the PDF. The change can be a different color or a different color type (i.e. RGB to a CMYK). Note that images are not affected by this process.

Form fields may have colors specified for borders (code `BC`) or backgrounds (code `BG`). These entries will be in brackets with the code `BC` or `BG` in front.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `COLOR` or `COLORSPACE` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<COLOR OLD="1 0 0 rg" NEW="0 1 0 rg">
```

will change from the color red to green and

```
<COLORSPACE OBJ="15" NEW="pantone green" NAME="Cs3">
```

sets the color space `Cs3` to a new CMYK color.

```
<COLOR OLD="0 0 1 rg" NEW="0 0 1 rg" REND=3>
```

Turns all text in blue invisible.

```
<COLOR OLD="BC [1 0 0]" NEW="BC [0 1 0]">
```

Changes all form fields with a border color of red to green

```
<COLOR  
  OLD="text"  
  NEW="text"  
  REND=number>
```

Use this tag to replace colors in a PDF.

<u>Option</u>	<u>Description</u>
OLD="text"	The old color string. Note this is in PDF syntax. This will be 1 to 4 numbers, each ranging from 0 to 1, followed by an upper or lower case G, RG, K, SC or SCN. G and RG are for RGB colors (1 or 3 numbers), K is for CMYK (4 numbers) and SC and SCN are special color operators. G is for grayscale and is shorthand for writing the same 3 numbers followed by RG. For example, "0.25 G" is the same as "0.25 0.25 0.25 RG". Capital letters are for stroking operations and lower case for nonstroking.
NEW="text"	Set this value if you want to update the color to something else. Tags where the OLD and NEW value are the same (and do not contain the REND option) are ignored. Include a leading 0 in front of any decimal number - i.e. use 0.25, not .25. You can use this feature to change colors or change from one format (RGB) to another (CMYK). Note that the format of the colors are in PDF syntax and no checks are made to see if the changes are valid.
REND=number	The rendering mode for the text that is colored with OLD. This option will not work for form field border/background colors in brackets. Use to change the style of the text by passing one of the following numbers: 0 - default, fill the character 1 - stroke (or outline) the character only 2 - stroke and fill the character 3 - no stroke or fill (invisible)

```
<COLORSPACE
  OBJ="number"
  NEW="text">
  NAME="text">
  SEPARATION="text">
```

Use this tag to replace color spaces in a PDF. May not work on all PDFs depending on the structure and intent of the color space.

<u>Option</u>	<u>Description</u>
OBJ="number"	The object number in the PDF containing a color space. A color space is a mapping containing color information. In this case, the OBJ is simply the object containing a pointer to a more complex data structure. The -colorout or setColorFileOut method will export this value for you or you can set this to a "*" meaning all color spaces should be replaced.
NEW="text"	Set this value if you want to update the color to something else. Tags where the NEW value is blank or missing are ignored. This can be any valid RGB or CMYK color string. For example, "red", "#ff0000", "486" (Pantone), or "pantone green". See the color section for details on colors and a Pantone color chart.
NAME="text"	For reference only and is usually generated by the program creating the PDF. Exported by the -colorout or setColorFileOut method. For example, this might be set to something like "Cs8".
SEPARATION="text"	For reference only. Exported by the -colorout or setColorFileOut method when available in the source PDF. For example, this might be set to something like "PANTONE 486". This value is not changed when updating the PDF using -colorin or setColorFileIn.

Here's how to create a color file:

```
pdfmeld.exe input.pdf colors.txt -colorout
```

And here's how to load in any changes made to it:

```
pdfmeld.exe input.pdf output.pdf -colorin colors.txt
```

Keep the following in mind when updating your tagged file:

- No spaces between the option and = sign
- No spaces between = and the value

Text Files

The `-strin` option or `setStrFileIn` method allows you to enter text on a page. You either pass in the name of a file containing the text tags or you may pass in the tags directly. For larger text blocks you are better off using a file however.

You specify the X/Y coordinates and, optionally, other aspects of the text. The text can be a single line or multi-line. The text breaks or wraps based on where you have line breaks in your input file. You may also use this feature to place invisible text on pages to allow for searching on the text. The `-transparency` option or `setTransparency` method can be used on this text as well.

In addition, you can use the `PAGE` tag in this file to add blank pages anywhere in the output PDF. You can then place text on these pages using the `TEXT` tag.

You may use Unicode characters as well. Use the [FONTS](#) tag to embed a font containing Unicode characters (such as `arialuni.ttf` that comes with Windows). Unicode text must be formatted as UTF-8 or use the syntax `✏` where 9999 is the decimal Unicode value, `香` where 9999 is the hexadecimal Unicode value or `&#o9999;` where 9999 is the octal Unicode value. For example, `ا` is the same as `ا` and `&#o3047.`

You may also supply Unicode text with a codepage of `IDENTITY`, `932` (Japanese), `936` (GBK - Chinese), or `949` (Korean). Use the `SUBSET` and `UNICODE` options on the `FONT` tag. A font such as `arialuni.ttf` should be used in this case. The needed glyphs will be embedded in the PDF so it can be viewed on not just PC's or Mac's but devices such as an iPad as well.

You may use the following variables in your text. To include the text without converting to the value, place a backslash (the `\` character) in front of the `&` symbol.

<u>Variable</u>	<u>Description</u>
<code>&page;</code>	The page number from the current PDF. This is reset with each PDF in the input list.
<code>&runpage;</code>	The running page number. This is not reset with each PDF in the input list. The value starts at 1 and ends at the total number of pages in the output PDF.

<u>Variable</u>	<u>Description</u>
&totpages;	The total number of pages in the output PDF.
&filepages;	The total number of pages in the current input PDF.
&file;	The file name of the input PDF.
&pathfile;	The full path and file name of the input PDF.
&relfile;	The requested or relative path and file name of the input PDF.
&yy;	The year (2 digits).
&yyyy;	The year (4 digits).
&m;	The month.
&mm;	The month (2 digits).
&mon;	The month name (3 letter abbreviation).
&month;	The month name.
&d;	The day.
ⅆ	The day (2 digits).
&wd;	The day of the week (3 letter abbreviation).
&weekday;	The day of the week.
&h;	The hour.
&hh;	The hour (2 digits).
&hh24;	The hour (24 hour clock).
&mi;	The minutes (always 2 digits).
&ss;	The seconds (always 2 digits).
&am;	AM or PM.
<	<
>	>
&varname;	Replace "varname" with an environmental variable defined for the user running the software (for example, &username;).

Each tag, or command, starts with an angle bracket < and closes with >. The tag name comes directly after the opening <. Options are then listed, space separated, with an = sign between it and its value. Note this tag has an opening and closing tag. For example:

```
<UNITS VALUE="in">  
<TEXT X=2 Y=3 COLOR=red ALIGN="center">  
Sample text line 1  
Next line (page=&page;)  
</TEXT>
```

will print "Sample text line 1" with "Next line (page=1)" underneath.

Text Files

Use `` to bold a section of a text. Close with ``. Use `<I>` to italicize a section of text. Close with `</I>`. The text `<` can be used for a `<` symbol and `>` can be used for a `>` symbol.

Text Files

You may use the <UNITS> and the tags in this file as well. See the [Fonts File](#) section for information on the FONT tag. Here's an example with a Unicode FONT added in:

```
<UNITS VALUE="in">  
<FONT NAME="arial" SRC="arial.ttf" UNICODE>  
<TEXT X=2 Y=3 ALIGN="center" FACE="arial">  
Sample text line 1  
Next line  
Third line with Unicode character &#x05D0;  
</TEXT>
```

```
<PAGE
  INSERTPOS=number
  ADDBEFORE
  ADDAFTER
  WIDTH=number
  HEIGHT=number>
```

The PAGE tag is used to insert a blank page into the PDF.

Use INSERTPOS=-1 to insert a page at the end. You may have as many PAGE tags as you want.

<u>Option</u>	<u>Description</u>
INSERTPOS=number	The page number (or position) to insert the page.
ADDBEFORE	Used to specify insertion before the page number given with INSERTPOS.
ADDAFTER	Used to specify insertion after the page number given with INSERTPOS.
WIDTH=number	The added page width in points (where 1 point = 1/72 of an inch or based on current UNITS setting).
HEIGHT=number	The added page height in points (where 1 point = 1/72 of an inch or based on current UNITS setting).

<TEXT
X=number
Y=number
PAGEWIDTH
FROMRIGHT
FROMTOP
FACE="text"
SIZE=number
BARCODE="UPCA"
SUPP="text"
ALIGN="text"
ANGLE=number
ANCHOR=text
NOROTATE
IFROTATE=number
COLOR="text"
SCOLOR="text"
BGCOLOR="text"
SHADING="text"
TEXTSHADING="text"
BORDER=number
BORDERCOLOR="text"
PADDING=number
LINESPACE=number
COMP=number
REND=number
LINEWIDTH=number
IMG="text"
IMGSCALE=number
TILE
FILLIMG="text"
FILLIMGSCALE=number
FILLTILE
SHADOW=number
SHADOWX=number
SHADOWY=number
SHADOWCOLOR="text"
SHADOWREND=number
TRANSPARENCY=number
TRANSPMODE="text"
PAGES="text"
SKIP="number"
FIRST
NOTFIRST
RUNPAGE
BASELINE=Yes|No
GLYPHPOS

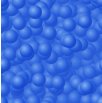

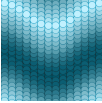
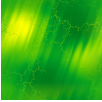



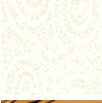

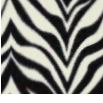
RTL>
</TEXT>

The TEXT tag is used to place text on a page or multiple pages. Place your text between the opening and closing TEXT tags.

<u>Option</u>	<u>Description</u>
X=number	Required. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the left page edge to place the text.
Y=number	Required. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the bottom page edge to place the baseline of the first line of text.
PAGEWIDTH	Uses the width of the page for center or right alignment. For example, specify PAGEWIDTH along with ALIGN="Center" to center the text in the middle of the page. The X value is used as the margin on the left and right when using with ALIGN="Right".
FROMRIGHT	Treats the X value as a distance from the right edge of the page instead of the left.
FROMTOP	Treats the Y value as a distance from the top edge of the page instead of the bottom.
FACE="text"	Set to one of the following: Courier Helvetica (Default) Times-Roman Courier-Bold Helvetica-Bold Times-Bold Courier-Oblique Helvetica-Oblique Times-Italic Courier-BoldOblique Helvetica-BoldOblique Times-BoldItalic Or use the FONTS tag option to embed your own font. Use the NAME value from the FONT tag for this option.
SIZE=number	The font point size. The default is 10.

<u>Option</u>	<u>Description</u>
BARCODE="UPCA"	Converts the value supplied into the correct character string for the barcode. May only be used with the UPCA TrueType barcode font from FyTek. You may use barcodes supplied by other companies but this option will not work with them. Different font vendors for barcodes will have their own coding scheme for character mappings. Instead, follow the directions supplied with the third-party font for formatting the string and pass that result as your text string. Only UPCA barcodes are currently supported so set BARCODE to the value "UPCA". Be sure to use the FONTS tag option to embed the barcode font. Reference the NAME value for the barcode font in the FACE option for the TEXT tag itself.
SUPP="text"	The optional 2 or 5 character supplemental value for UPCA barcodes.
ALIGN="text"	The alignment. Use "Center" or "Right". The default is "Left".
ANGLE=number	The counter-clockwise angle of rotation around the center of the text block. Enter a value from 0 to 360. Use a negative number from -360 to 0 for clockwise rotation. The default is 0.
ANCHOR=text	Used to specify a different anchor point for the rotation. The default is the center of the text block. Specify one of the following: LL for Lower-Left LR for Lower-Right UL for Upper-Left UR for Upper-Right For most cases you'll want to use LL as the value when overriding the default center point rotation. For example, if you have a string rotated 90 degrees with the default center point rotation, its X position will vary depending on the length of the string. To keep the X position fixed, use ANCHOR="LL".

<u>Option</u>	<u>Description</u>
NOROTATE	Prevents the text from rotating when there is a page rotation. Sometimes pages that appear to be oriented correctly when viewed actually have a rotation set on the page. When you place text on the page it then becomes rotated even though you didn't specify an ANGLE for it. This option will compensate for the page rotation and adjust the angle of the text. See the Page Layout Files section for information on exporting page information, including rotation, if you want to find out how the pages are arranged. Note that ANCHOR will be set to "UL" and BASEALIGN will be set to "No" when this option is used regardless of what value is passed in.
IFROTATE=number	Set to a value of 0, 90, 180 or 270. This is used to specify what pages the text should be printed on based on the current page's rotation value. You might use this with the NOROTATE option to provide a different position for the text based on how the page is rotated.
COLOR="text"	The color for the text.
SCOLOR="text"	The color when stroking the text (when REND=1 or REND=2).
BGCOLOR="text"	The background color for the text.
SHADING="text"	A shading pattern to use for the background. See the SHADING tag for details.
TEXTSHADING="text"	A shading pattern to use for the text itself. See the SHADING tag for details.
BORDER=number	The width of the border in points. Can be a decimal number (such as .5) for a thin border.
BORDERCOLOR="text"	The color for the border.
PADDING=number	The amount of padding to leave between the text and the border (also for background color) in points.
LINESPACE=number	The amount of space in points (1/72 of an inch) to drop down between text lines. The default is 2 + the font point size.
COMP=number	The percentage amount to horizontally compress the text by. Enter as a whole number (for example, use 80 for 80%). The default is 100.

<u>Option</u>	<u>Description</u>
REND=number	The render mode. The default is 0 (fill the character). Other options are: 1 - stroke (or outline) the character only 2 - stroke and fill the character 3 - no stroke or fill (invisible)
LINEWIDTH=number	The linewidth to use when stroking the text (when REND=1 or REND=2).
IMG="text"	Full path and name of an image to use as the background for the text. Images smaller than the text block are expanded to fit. You may also use one of the following for a predefined pattern:  Bubbles  Circuit  Dotwave  Emerald  Fiesta  Leaves  Linoleum  Maize  Tiger  Zebra

<u>Option</u>	<u>Description</u>
IMGSCALE=number	A scaling factor expressed as a percentage for the image. Default is 100 for 100% or no scaling. Values smaller than 100 will shrink the image while values greater than 100 will enlarge.
TILE	Used with the IMG option. When set, the background image is tiled rather than expanded to fit.
FILLIMG="text"	Full path and name of an image to use as the fill for the text characters. Images smaller than the text block are expanded to fit. You may also use one of the predefined patterns (see the IMG option for the list).
FILLIMGSCALE=number	A scaling factor expressed as a percentage for the image. Default is 100 for 100% or no scaling. Values smaller than 100 will shrink the image while values greater than 100 will enlarge. You can create interesting text effects by using the same image for IMG and FILLIMG but with different scaling factors.
FILLTILE	Used with the FILLIMG option. When set, the fill image is tiled rather than expanded to fit.
SHADOW=number	Draws a shadow effect for the text. Set to the transparency value you want for the shadow. Enter a value from 1 to 100. The lower the number, the lighter the shadow.
SHADOWX=number	The X-axis offset for the shadow. Enter a value in points (1 point = 1/72 of an inch). Can be a positive or negative value (or decimal values). General range is -5 to 5.
SHADOWY=number	The Y-axis offset for the shadow. Enter a value in points (1 point = 1/72 of an inch). Can be a positive or negative value (or decimal values). General range is -5 to 5.
SHADOWCOLOR="text"	The color for the shadow. Default is black.
SHADOWREND=number	The rendering mode for the shadow text. The default is 2. Use 0 for a sharper shadow or 1 for a hollow shadow.
TRANSPARENCY=number	Optional transparency for the text. Requires Acrobat or Reader 5.0 or higher to view the transparency. Enter a value from 1 to 100. The default without this option is 100 - the text is placed on top of the background PDF with none of the background showing through.

<u>Option</u>	<u>Description</u>
TRANSPMODE="text"	Optional transparency mode. The valid values are: Normal (Default) Multiply Screen Overlay Darken Lighten ColorDodge ColorBurn HardLight SoftLight Difference Exclusion Hue Saturation Color Luminosity
PAGES="text"	The list of pages to show the text on. Enter a comma separated list and/or use a - (minus sign) for a page range. May also use "odd" or "even". Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. For example, PAGES="10-300,oddrange" will only include odd pages in the range of 10-300 (11, 13, ... 299). In comparison, PAGES="10-300,odd" will pull all odd pages and all pages within the range of 10-300. Leave PAGES out to include on all pages. Note the page number refers to the input file page number. If you have two input files you are merging, the page numbering starts over at 1 for the second PDF. Use the RUNPAGE option if you want to use the output PDF page numbers instead.
SKIP="number"	Set to skip the number of pages specified. For example, to place a text watermark on every 3rd page, set SKIP=3. The text will then show up on pages 1, 4, 7, 11, and so on. The range specified by PAGES will still take effect so if you have PAGES=2-50 with SKIP=3, the pages for the text will be 2, 5, 8, and so on.
FIRST	Include on first page only (same as setting PAGES=1).

<u>Option</u>	<u>Description</u>
NOTFIRST	Include on any page except the first page. For example, if you set PAGES="odd" and don't want this text to show on the first page, set this option.
RUNPAGE	Uses the running page number for the PAGES option. In this case, the pages are numbered from 1 to n where n is the total number of pages of all PDFs being merged.
BASELINE=Yes No	If set to "No", the Y value (for text position) represents the top of the first line of text. If set to "Yes" (the default), the Y value represents the baseline of the first line of text. Setting to "No" has the effect of lowering the text by the point size. This is fixed as "No" when NOROTATE is used.
GLYPHPOS	Used when the text contains positioning (kerning) information. Your text must be entered in parenthesis with a number between each set of parenthesis. A positive number will move the current text position to the left while a negative number will move it to the right. That is, the value is subtracted from the current position to obtain the new position. The value is specified in 1000ths of a unit. See the example at the end of this section.
RTL	Used to specify the text should be converted to right-to-left reading order. May not handle all the details depending on your particular text string.

```
<IMG
  X=number
  Y=number
  SRC="text"
  FROMRIGHT
  FROMTOP
  ANGLE=number
  NOROTATE
  IFROTATE=number
  BORDER=number
  BORDERCOLOR="text"
  PADDING=number
  REND=number
  IMGSCALE=number
  TRANSPARENCY=number
  TRANSPMODE="text"
  PAGES="text"
  SKIP="number"
  FIRST
  NOTFIRST
  RUNPAGE>
```

The IMG tag is used to place an image on a page or multiple pages.

<u>Option</u>	<u>Description</u>
X=number	Required. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the left page edge to place the image.
Y=number	Required. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the bottom page edge to place the image.
SRC="text"	Full path and name of an image to use.
FROMRIGHT	Treats the X value as a distance from the right edge of the page instead of the left.
FROMTOP	Treats the Y value as a distance from the top edge of the page instead of the bottom.
ANGLE=number	The counter-clockwise angle of rotation around the center of the image. Enter a value from 0 to 360. Use a negative number from -360 to 0 for clockwise rotation. The default is 0.

<u>Option</u>	<u>Description</u>
ANCHOR=text	<p>Used to specify a different anchor point for the rotation. The default is the center of the image. Specify one of the following:</p> <ul style="list-style-type: none">LL for Lower-LeftLR for Lower-RightUL for Upper-LeftUR for Upper-Right <p>For most cases you'll want to use LL as the value when overriding the default center point rotation. For example, if you have an image rotated 90 degrees with the default center point rotation, its X position will vary depending on the width of the image. To keep the X position fixed, use ANCHOR="LL".</p>
NOROTATE	<p>Prevents the image from rotating when there is a page rotation. Sometimes pages that appear to be oriented correctly when viewed actually have a rotation set on the page. When you place an image on the page it then becomes rotated even though you didn't specify an ANGLE for it. This option will compensate for the page rotation and adjust the angle of the image. See the Page Layout Files section for information on exporting page information, including rotation, if you want to find out how the pages are arranged. Note that ANCHOR will be set to "UL" and BASEALIGN will be set to "No" when this option is used regardless of what value is passed in.</p>
IFROTATE=number	<p>Set to a value of 0, 90, 180 or 270. This is used to specify what pages the image should be displayed on based on the current page's rotation value. You might use this with the NOROTATE option to provide a different position for the image based on how the page is rotated.</p>
BORDER=number	<p>The width of the border in points. Can be a decimal number (such as .5) for a thin border.</p>
BORDERCOLOR="text"	<p>The color for the border.</p>
PADDING=number	<p>The amount of padding to leave between the image and the border in points.</p>

<u>Option</u>	<u>Description</u>
REND=number	The render mode. The default is 0 (fill the character). Other options are: 1 - stroke (or outline) the character only 2 - stroke and fill the character 3 - no stroke or fill (invisible)
IMGSCALE=number	A scaling factor expressed as a percentage for the image. Default is 100 for 100% or no scaling. Values smaller than 100 will shrink the image while values greater than 100 will enlarge.
TRANSPARENCY=number	Optional transparency for the image. Requires Acrobat or Reader 5.0 or higher to view the transparency. Enter a value from 1 to 100. The default without this option is 100 - the image is placed on top of the background PDF with none of the background showing through.
TRANSPMODE="text"	Optional transparency mode. The valid values are: Normal (Default) Multiply Screen Overlay Darken Lighten ColorDodge ColorBurn HardLight SoftLight Difference Exclusion Hue Saturation Color Luminosity

<u>Option</u>	<u>Description</u>
PAGES="text"	The list of pages to show the image on. Enter a comma separated list and/or use a - (minus sign) for a page range. May also use "odd" or "even". Use "oddrange" or "evenrange" along with a page range to only pull odd or even pages within that range. For example, PAGES="10-300,oddrange" will only include odd pages in the range of 10-300 (11, 13, ... 299). In comparison, PAGES="10-300,odd" will pull all odd pages and all pages within the range of 10-300. Leave PAGES out to include on all pages. Note the page number refers to the input file page number. If you have two input files you are merging, the page numbering starts over at 1 for the second PDF. Use the RUNPAGE option if you want to use the output PDF page numbers instead.
SKIP="number"	Set to skip the number of pages specified. For example, to place an image watermark on every 3rd page, set SKIP=3. The image will then show up on pages 1, 4, 7, 11, and so on. The range specified by PAGES will still take effect so if you have PAGES=2-50 with SKIP=3, the pages for the image will be 2, 5, 8, and so on.
FIRST	Include on first page only (same as setting PAGES=1).
NOTFIRST	Include on any page except the first page. For example, if you set PAGES="odd" and don't want this image to show on the first page, set this option.
RUNPAGE	Uses the running page number for the PAGES option. In this case, the pages are numbered from 1 to n where n is the total number of pages of all PDFs being merged.


```
<SHADING  
  NAME="text"  
  COLOR1=color  
  COLOR2=color  
  COLOR3=color  
  COLOR4=color  
  COLOR5=color  
  COLORARY=text>
```

Used to define a gradient shading pattern. The shading pattern can then be used for text or text background. You may specify from two to five colors.

<u>Parameter</u>	<u>Description</u>
NAME="text"	The name for the shading pattern. Must be unique within the document.
COLOR1=color	The starting color. Any valid color code.
COLOR2=color	Any valid color code.
COLOR3=color	Optional. Any valid color code.
COLOR4=color	Optional. Any valid color code.
COLOR5=color	Optional. Any valid color code.
COLORARY=text	A comma separated list of 4 or 6 numbers. The default is 0,0,1,0. These represent the X_0, Y_0, X_1, Y_1 matrix coordinates for the shading pattern. A matrix of 0,0,1,0 goes from left to right from COLOR1 to COLOR _n . A matrix of 0,0,0,1 goes from top to bottom. You may use decimals or negative numbers as well to offset where the middle of the gradient lies. Use 6 numbers x0, y0, r0, x1, y1, r1 for a radial type shading, such as for a sphere. The numbers specify the centers and radii of the starting and ending circles, expressed in points. The radii r0 and r1 must both be greater than or equal to 0. If one radius is 0, the corresponding circle is treated as a point; if both are 0, nothing is painted. The x0, y0, x1 and y1 values should typically be between 0 and 1. Use x0=.5, y0=.5, x1=.5 and y1=.5 to center the circles in the middle.

**<UNITS
VALUE=text>**

Used to set the unit of measurement for tags that follow. Typically you'll place one UNITS tag at the top but you may use them through the file with different values. The UNITS tag affects all tags that follow it until another UNITS tag is found.

For example, to set centimeters use <UNITS VALUE="cm">. The default is "pt" for points.

<u>Parameter</u>	<u>Description</u>
VALUE=text	Use one of the following: <u>PT</u> Points (1/72 of an inch) <u>IN</u> Inches <u>CM</u> Centimeters <u>MM</u> Millimeters

<MONTHS
SHORT=list
LONG=list>

The MONTHS tag allows a different set of month names and abbreviations to be used with the &mon and &month text variables. The default is the English spelling of the names if this tag is not used.

<u>Option</u>	<u>Description</u>
SHORT=list	A comma separated list of 12 month names going from January to December. This is used by the &mon variable.
LONG=list	A comma separated list of 12 month names going from January to December. This is used by the &month variable.

<MONTHS
SHORT=list
LONG=list>

The WEEKDAYS tag allows a different set of day of the week names and abbreviations to be used with the &wd and &weekday text variables. The default is the English spelling of the names if this tag is not used.

<u>Option</u>	<u>Description</u>
SHORT=list	A comma separated list of the 7 day of the week names going from Sunday to Saturday. This is used by the &wd variable.
LONG=list	A comma separated list of the 7 day of the week names going from Sunday to Saturday. This is used by the &weekday variable.

Here's a sample (including an added page after existing page #5):

The SRC value for the FONT tags below assume the font is located in your Windows font directory or your current directory. Be sure to fully qualify this path (i.e. "c:\myfonts\mycustom.ttf") if you have fonts residing elsewhere. <UNITS VALUE="in">

```
<SHADING NAME="blue" COLOR1=#FFFFFF
COLOR2=#9CCFFF COLORARY="0,0,1,0">
<PAGE INSERTPOS=5 ADDAFTER HEIGHT=8.5 WIDTH=11>
<FONT NAME="custom" SRC="myfont.ttf">
<TEXT X=1 Y=3 FACE="custom" PAGES="1-3,6" COLOR="green">
This is the first line of text.
This is another line.
</TEXT>
<TEXT X=2.5 Y=4 SIZE=15 FACE="helvetica" ALIGN="center"
BORDER=3 PADDING=2>
Here is some text to center.
Line 2
Last line of centered text.
</TEXT>
<TEXT X=3 Y=5 SIZE=12 FACE="helvetica" SHADING="blue">
Here is some shaded text.
Line 2
Last line of shaded text.
</TEXT>
<TEXT X=1 Y=1 REND=3>
This text is invisible.
</TEXT>
<TEXT X=1 Y=6 SIZE=18 GLYPHPOS>
(T)-20(his)-215(text)-220(uses glyph)-230(positioning.)
(A)20(11 text must go within parenthesis.)
(Use a slash in front of any parenthesis in the text.)
(For example: \ (555\ ) 555-1212)
</TEXT>
<FONT NAME="bar" SRC="upca075.ttf">
<TEXT X=3 Y=4 SIZE=30 FACE="bar" BARCODE="UPCA">
01234567890
</TEXT>
```

Here's how to load a text file:

```
pdfmeld.exe input.pdf output.pdf -strin mytext.txt
```

Rectangle Files

The `-rectin` option or `setRectFileIn` method allows you to draw rectangles on the page. A filled rectangle can be used to block out sections of a page when set to the page background color. Any text/images the rectangle covers remains in the document though it is blocked from view.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `RECT` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<RECT COLOR="red" X=72 Y=72 WIDTH=144 HEIGHT=288 PAGES=1,5-7>
```

is a tag to paint a rectangle in the lower left corner on pages 1, 5, 6 and 7.

```
<RECT
  X=number
  Y=number
  X2=number
  Y2=number
  WIDTH=number
  HEIGHT=number
  COLOR=text
  BORDER=number
  BORDERCOLOR=text
  SHADING="text"
  TRANSPARENCY=number
  TRANSPMODE="text"
  FROMRIGHT
  FROMTOP
  PAGES="text">
```

Used to draw a rectangle. This can be filled or outlined.

Note the `-addback` option or `setAddBack` method can be used to place the added rectangle(s) in the page background rather than foreground.

<u>Parameter</u>	<u>Description</u>
X=number	Required. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the left page edge for left side the rectangle.
Y=number	Required. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the bottom page edge for the bottom side of the rectangle.
X2=number	The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the left page edge for the right side of the rectangle. You can use the WIDTH option instead of this one. Any setting here will override the WIDTH setting. You may use a negative value to mean how far from the right page edge for the right side of the rectangle. For example, using X=36 X2=-36 means start a half-inch from the left edge and extend to a half-inch from the right side of the page.

Rectangle Files

<u>Parameter</u>	<u>Description</u>
Y2=number	The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the bottom page edge for the top side of the rectangle. You can use the HEIGHT option instead of this one. Any setting here will override the HEIGHT setting. You may use a negative value to mean how far from the top page edge for the top side of the rectangle. For example, using Y=36 Y2=-36 means start a half-inch from the bottom edge and extend to a half-inch from the top of the page.
WIDTH=number	The width (in points where 1 point = 1/72 of an inch or based on current UNITS setting) for the rectangle. Do not use along with parameter X2.
HEIGHT=number	The height (in points where 1 point = 1/72 of an inch or based on current UNITS setting) for the rectangle. Do not use along with parameter Y2.
COLOR=text	The fill color for the rectangle. This is the outline (rather than fill) color when SHADING is used. You may leave this option off when using SHADING if you don't want an outline. Leave this option off or set this option to "none" when you want to outline a rectangle only (no fill).
BORDER=number	The border width (if any) in points.
BORDERCOLOR=color	The color for the border.
SHADING="text"	A shading pattern to use for the background. See the SHADING tag for details.
TRANSPARENCY=number	Optional transparency for the rectangle. Requires Acrobat or Reader 5.0 or higher to view the transparency. Enter a value from 1 to 100. The default without this option is 100 - the rectangle is placed on top of the background PDF with none of the background showing through.

Rectangle Files

<u>Parameter</u>	<u>Description</u>
TRANSPMODE="text"	Optional transparency mode. The valid values are: Normal (Default) Multiply Screen Overlay Darken Lighten ColorDodge ColorBurn HardLight SoftLight Difference Exclusion Hue Saturation Color Luminosity
FROMRIGHT	Changes the meaning of the X value to be from the right of the page rather than the left. For example, if X=72 then setting FROMRIGHT will mean one inch from the right side of the page rather than one inch from the left.
FROMTOP	Changes the meaning of the Y value to be from the top of the page rather than the bottom. For example, if Y=72 then setting FROMTOP will mean one inch from the top of the page rather than one inch from the bottom.
PAGES="text"	The list of pages to show the rectagle on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.

<SHADING

NAME="text"
COLOR1=color
COLOR2=color
COLOR3=color
COLOR4=color
COLOR5=color
COLORARY=text>

Used to define a gradient shading pattern. The shading pattern can then be used for the rectangle background. You may specify from two to five colors.

<u>Parameter</u>	<u>Description</u>
NAME="text"	The name for the shading pattern. Must be unique within the document.
COLOR1=	The starting color. Any valid color code.
COLOR2=	Any valid color code.
COLOR3=	Optional. Any valid color code.
COLOR4=	Optional. Any valid color code.
COLOR5=	Optional. Any valid color code.
COLORARY=	A comma separated list of 4 or 6 numbers. The default is 0,0,1,0. These represent the X_0, Y_0, X_1, Y_1 matrix coordinates for the shading pattern. A matrix of 0,0,1,0 goes from left to right from COLOR1 to COLOR _n . A matrix of 0,0,0,1 goes from top to bottom. You may use decimals or negative numbers as well to offset where the middle of the gradient lies. Use 6 numbers x0, y0, r0, x1, y1, r1 for a radial type shading, such as for a sphere. The numbers specify the centers and radii of the starting and ending circles, expressed in points. The radii r0 and r1 must both be greater than or equal to 0. If one radius is 0, the corresponding circle is treated as a point; if both are 0, nothing is painted. The x0, y0, x1 and y1 values should typically be between 0 and 1. Use x0=.5, y0=.5, x1=.5 and y1=.5 to center the circles in the middle.

**<UNITS
VALUE=text>**

Used to set the unit of measurement for tags that follow. Typically you'll place one UNITS tag at the top but you may use them through the file with different values. The UNITS tag affects all tags that follow it until another UNITS tag is found.

For example, to set centimeters use <UNITS VALUE="cm">. The default is "pt" for points.

<u>Parameter</u>	<u>Description</u>
VALUE=text	Use one of the following: <u>PT</u> Points (1/72 of an inch) <u>IN</u> Inches <u>CM</u> Centimeters <u>MM</u> Millimeters

Rectangle Files

Here are some examples:

```
<SHADING NAME="redfade" COLOR1="#FF0000"  
  COLOR2="#FFFFFF" COLORARY="1,1,0,0">  
<RECT COLOR="#00C0F9" X=72 Y=72 WIDTH=150 HEIGHT=300>  
<RECT COLOR="blue" X=120 Y=540 WIDTH=72 HEIGHT=72  
  PAGES="2-5">  
<UNITS VALUE="in">  
<RECT COLOR="#FF0000" X=1 Y=1 X2=2 Y2=2 PAGES="4,8-10"  
SHADING="redfade">
```

Load the file like this:

```
pdfmeld.exe filein.pdf fileout.pdf -rectin filerect.txt
```

Image Files

(Note: `-imgout` and `setImgFileOut` are not available in [SE version](#))

The `-imgin/-imgout` or `setImgFileIn/setImgFileOut` methods allow you to modify images in a PDF. The base file is created by using the `-imgout` option or `setImgFileOut` method. A set of tags will be created in the output file based on the various images found in the PDF. You may then modify or add images by modifying or adding IMG tags. Image sources can be from JPEG, GIF, certain TIFFs or other PDFs.

In addition, you may use the MEDIA option of the IMG tag to add other media types to your PDF. For example, to place a Flash animation or movie on the page.

Also, see the `-imgdir` option or `setImgDir` method if you want to export images from the PDF.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always IMG in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<IMG NAME="obj_35_Img1" HEIGHT=100 WIDTH=350>
```

You can also export information on the location and size of images on each page. While the IMG tag shows physical images in the PDF, the IMGPOS tag is used to show on what page the image is used on, its position and size. A single image (IMG tag) may be used on multiple pages and positions. You can think of the IMG tag as the image objects and the IMGPOS as the instances of those objects or images.

Note the `-addback` option or `setAddBack` method can be used to place added image(s) in the page background rather than foreground.

```
<IMG
  NAME="text"
  HEIGHT=number
  WIDTH=number
  BYTES=number
  FILTER=text
  SRC="text"
  SRCNAME="text"
  X=number
  Y=number
  X2=number
  Y2=number
  SCALEX=number
  SCALEY=number
  SCALE=number
  MASK=text
  PAGESIZE
  BORDERWIDTH=number
  BORDEROUTER
  BORDERCOLOR=color
  DECORWIDTH=number
  DECORWIDTHHT=number
  DECORWIDTHHB=number
  DECORWIDTHHL=number
  DECORWIDTHHR=number
  DECORTILE
  TRANSPARENCY=number
  TRANSPMODE="text"
  PAGES="text"
  MEDIA="text">
```

The IMG tags are the individual images in the PDF. You can use the IMG tag to add your own images or other media to the PDF.

Use the MEDIA option to play an audio clip or movie on the page. For example, `` will place an MP3 file in the PDF on page 1. You can load this into an existing PDF with the following options:

```
-imgin "<IMG MEDIA='myfile.mp3' X=0 Y=0 X2=72 Y=36
PAGES=1>" -mediaplay o -mediarepeat
```

The `"-mediaplay o"` is to begin the clip when the PDF is opened. Note the user will receive a pop-up message when they open the PDF asking if they want to play the clip so they are not forced to play it.

See the IMGPOS tag for image instances.

Image Files

<u>Option</u>	<u>Description</u>
NAME="text"	This is generated when you create an image file from an existing PDF. It is a unique reference to the image and may or may not be named something that you can tell what image it is referring to. You only need the NAME option when you want to replace an image.
HEIGHT=number	The height in points of the image. Not necessary to pass this value with added images. This is only a reference for existing images. You can pass this value if you want to force a different height for an added image though.
WIDTH=number	The width in points of the image. Not necessary to pass this value with added images. This is only a reference for existing images. You can pass this value if you want to force a different width for an added image though.
BYTES=number	The size of the image in bytes. Not necessary to pass this value with added images. This is only a reference for existing images.
FILTER=text	The type of filter(s) for the image. Not necessary to pass this value with added images. This is only a reference for existing images.
SRC="text"	The full path and image name of an image you wish to use. Only use JPEGs at 72 DPI 256-color grayscale or 24-bit color, GIFs, or PNGs (alpha transparency in PNGs is not supported - the image will show without transparency). Note this may also be a PDF where you want to pull an image from. See the SRCNAME option when pulling from another PDF.
SRCNAME="text"	The NAME value that corresponds to the image from the PDF specified in the SRC parameter. Note you'll need to run pdfmeld on this PDF first so you can see what the image names are.
X=number	The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the left page edge for the left edge of the image to be placed.
Y=number	The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the bottom page edge for the bottom edge of the image to be placed.

Image Files

<u>Option</u>	<u>Description</u>
X2=number	Optional. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the left page edge for the right edge of the image to be placed. Allows you to specify a bounding box of X, Y, X2, Y2 to place the added image. The default will simply be the width (after scaling) of the image.
Y2=number	Optional. The position (in points where 1 point = 1/72 of an inch or based on current UNITS setting) from the bottom page edge for the top edge of the image to be placed. Allows you to specify a bounding box of X, Y, X2, Y2 to place the added image. The default will simply be the height (after scaling) of the image.
SCALEX=number	The amount as a percent to scale the width of any added images by. For example, use 50 for 50%. Does not affect images that are replacing existing images in the PDF (where the NAME option is used).
SCALEY=number	The amount as a percent to scale the height of any added images by. For example, use 50 for 50%. Does not affect images that are replacing existing images in the PDF (where the NAME option is used).
SCALE=number	Shorthand option to set both SCALEX and SCALEY to the same value.

<u>Option</u>	<u>Description</u>
MASK=text	A set of numbers (separated by a space) representing the from and through color index numbers to make transparent. Normally, transparent GIFs will be handled without the need to use this option. You may use this with other images, such as JPEG, that do have a transparency stored in the image itself. The value is a set of 2 x n integers, min ₁ max ₁ ... min _n max _n , where n is the number of color components in the image's color space. In general, the values range from 0 to 255 (or 2 ^{BitsPerColorComponent} - 1 where BitsPerColorComponent is typically 8). The first number is the beginning color index number and the second is the ending color index. For grayscale images, you only need a single set of values such as MASK="240 255". For RGB images, you'll need to pass each RGB color component separately, for example MASK="230 245 100 128 40 52". The colors falling within the range provided are made transparent.
PAGESIZE	Size the image to the page.
BORDERWIDTH=number	The width of the border for the image. The image will be reduced in size to accommodate the border within the area specified. The image and the border will fall within the X/Y and X2/Y2 area. Use the BORDEROUTER option to place the border outside this area.
BORDEROUTER	Used to place the border outside of the area specified for the image rather than the default of inside. The image will first be placed and sized as if there were no border and then the border is placed outside of this area.
BORDERCOLOR=color	The color of the image border.

<u>Option</u>	<u>Description</u>
DECORWIDTH=number	The width of the border when using the image itself as a page border (or decoration). Note this is not a border for the image but rather a way to use the image as a border for a page. This allows you to place an image (of a picture frame, for example) on top of a PDF page but only cover the page edges and not the center. Normally when creating the border you'll want to use the PAGESIZE option or set the X2/Y2 values for the image. This option sets all edge widths at once (top, bottom, left and right). Use the options below to set each edge individually. The width is in points or the current unit setting.
DECORWIDTHT=number	Sets the top border width. See the DECORWIDTH option for details. The width is in points or the current unit setting.
DECORWIDTHB=number	Sets the bottom border width. See the DECORWIDTH option for details. The width is in points or the current unit setting.
DECORWIDTHL=number	Sets the left border width. See the DECORWIDTH option for details. The width is in points or the current unit setting.
DECORWIDTHR=number	Sets the right border width. See the DECORWIDTH option for details. The width is in points or the current unit setting.
DECORTILE	Tiles the border image. The image is repeated in the border area rather than sizing it to the width and height of the page.
TRANSPARENCY=number	Optional transparency for the image. Requires Acrobat or Reader 5.0 or higher to view the transparency. Enter a value from 1 to 100. The default without this option is 100 - the image is placed on top of the background PDF with none of the background showing through.

<u>Option</u>	<u>Description</u>
TRANSPMODE="text"	Optional transparency mode. The valid values are: Normal (Default) Multiply Screen Overlay Darken Lighten ColorDodge ColorBurn HardLight SoftLight Difference Exclusion Hue Saturation Color Luminosity
PAGES="text"	The list of pages to show the added image on. Leave this option out to include on all pages.
MEDIA="text"	A media file (such as a Flash animation file) to display at the location. Use the HEIGHT and WIDTH options in this case for the size of the media area. The SRC image is the static image to show until the media plays. You may leave the SRC option out if there is already an image or other indicator on the PDF page. Do not include the NAME option when using MEDIA.

Note the options X, Y, SCALEX, SCALEY, SCALE, and PAGES are only for added images (where the NAME option is not used).

Here's a set of sample image tags:

```
<IMG NAME="obj_21_Img1" HEIGHT=50 WIDTH=200>  
<IMG NAME="obj_35_Img2" HEIGHT=100 WIDTH=350  
  SRC="c:\images\modimage.jpg">  
<IMG NAME="obj_45_Img3" HEIGHT=200 WIDTH=150  
  SRC="c:\pdfs\test.pdf" SRCNAME="obj_84">  
<IMG SRC="c:\images\newimage.gif" X=200 Y=72  
  PAGES="1-3,7,9">
```

The first line shows an image that won't be changed - there is no SRC option so it is left alone.

The second image will be replaced by an existing JPEG image.

The third image will be replaced by an image, obj_84 in this case, pulled from an existing PDF called test.pdf.

Image Files

The last image is a new one that will appear 200 points from the left and 72 points from the bottom of pages 1, 2, 3, 7 and 9.

Here's how to create an image file:

```
pdfmeld.exe input.pdf images.txt -imgout
```

And here's how to load in any changes made to it:

```
pdfmeld.exe input.pdf output.pdf -imgin images.txt
```

Keep the following in mind when updating your tagged file:

- No spaces between the option and = sign
- No spaces between = and the value

```
<IMGPOS  
  NAME="text"  
  OBJ="text"  
  PAGE=number  
  ID=number  
  A=number  
  B=number  
  C=number  
  D=number  
  X=number  
  Y=number  
  REMOVE>
```

You can use the IMGPOS information to move, scale or prevent images from displaying. The IMGPOS tag represents the instances of the images from the IMG tag. Use the additional parameter -imgout "pos" or setImgFileOut("pos") to include this information in the output file. The information can also be used to determine where on a page an image is located. The UNITS tag (if used) does not apply to the values in this tag.

The matrix operation for the image is based on the following formula:

$$x' = x * A + y * C + X$$

$$y' = x * B + y * D + Y$$

The B and C values are typically 0 so what this means is move to the X/Y position on the page and display the image with a width of A and height of D. Modify A and/or D to stretch or shrink the image. Modify X and/or Y to move the image.

There may be other scaling or rotation commands issued that pertain to the image. In those instances the values given for the image location will be modified or offset by those actions. That's why the X and Y position, for example, are not guaranteed to be from the left and bottom edge of the page. The PDF must use a compression format that PDF Meld can work with (plain text, ASCII 85 or Zlib) in order to pull this information. If you run the option with this parameter and do not get any IMGPOS tags then likely the PDF is using some other compression algorithm (typically LZW).

<u>Option</u>	<u>Description</u>
NAME="text"	This is the object name identified in the PDF. Note that the image (the IMG tag) may be called by a different name on different pages. Use the OBJ value to refer back to a specific IMG tag if necessary.
OBJ="text"	The object number preceded by the text "obj_". This will match the NAME value (or the first portion of it) from the IMG tag. You can use this to reference what image(s) are used where.

Image Files

<u>Option</u>	<u>Description</u>
PAGE=number	The page number this IMGPOS tag was found on. This is the page that the image is displayed on. There may be multiple IMGPOS tags for the same image if it is displayed on multiple pages.
ID=number	A unique number generated for each IMGPOS tag. It is based on the order the image was found when parsing the input PDF. Do not modify this number as it is used when loading the file back in. You can rearrange the IMGPOS tags as this id will be used to determine the original order.
A=number	Generally the image width in points. The A, B, C and D values along with X and Y are used in a matrix operation when placing the image on the page. Any of the values can be positive or negative depending on any rotation or scaling applied to the image.
B=number	Generally this is 0. This will only be set if there is a rotation or skewing operation applied.
C=number	Generally this is 0. This will only be set if there is a rotation or skewing operation applied.
D=number	Generally the image height in points.
X=number	Generally the X position in points from the left edge of the page to place the image.
Y=number	Generally the Y position in points from the bottom edge of the page to place the image.
REMOVE	Add this text if you want to remove this image instance. If the image is displayed on more than one page or area it will still display at those locations (unless you have REMOVE set for those as well).

Image Files

```
<THUMB  
  SRC="text"  
  PAGE=number>
```

Use this tag to embed an image as a page thumbnail. It is not required that every page have a thumbnail image. In Acrobat Reader, the thumbnails will be generated automatically and the image used here will be ignored. The only use for this is when you want to embed your own thumbnails for another other application to use.

<u>Option</u>	<u>Description</u>
SRC="text"	The full path and image name of an image you wish to use. This should be a standard 72 DPI JPEG. The size should be no more than 106 by 106 pixels. In general, the thumbnail should be created at 1/8 scale of the actual page size.
PAGE=number	The page number this thumbnail image belongs with.

Image Files

<UNITS
VALUE=text>

Used to set the unit of measurement for tags that follow. Typically you'll place one UNITS tag at the top but you may use them through the file with different values. The UNITS tag affects all tags that follow it until another UNITS tag is found.

For example, to set centimeters use <UNITS VALUE="cm">. The default is "pt" for points.

<u>Parameter</u>	<u>Description</u>
VALUE=text	Use one of the following: <u>PT</u> Points (1/72 of an inch) <u>IN</u> Inches <u>CM</u> Centimeters <u>MM</u> Millimeters

Font Files

The `-fonts` option or `setFontFile` method allows you to embed TrueType, OpenType or Type 1 fonts in a PDF. The embedded fonts can be used for the page number (or other) string that you specify. Or when you want to embed a font that currently isn't embedded. Also, you can replace one font with another in a PDF. This is really only useful if the size of the characters are consistent with the font you're replacing. Otherwise you'll likely end up with text that's misaligned and flowing off the page and/or on top of itself.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `FONT` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:


```
<FONT SRC="c:\winnt\fonts\myfont.ttf" NAME="Myfont">
```

is a tag to embed the `myfont.ttf` font.

```
<FONT
  SRC="text"
  NAME="text"
  CODEPAGE=number
  BOM=text
  ENCODING=text
  UNICODE
  SUBSET
  REPLACE="text"
  REPLACEIFSUB>
```

Used to embed a custom font.

<u>Option</u>	<u>Description</u>
SRC="text"	The source file on disk for the font. Only TrueType, OpenType and Type 1 fonts are supported. For TrueType fonts specify the font file (ex. "c:\windows\fonts\myfont.ttf"). Also, for OpenType fonts specify the font file (ex. "c:\windows\fonts\myfont.otf"). For Type 1 fonts specify the file name without the extension (ex. "c:\windows\fonts\myfont"). Type 1 fonts have several different files associated with them and the software will handle locating the individual files.
NAME="text"	The name you wish to refer to this font as in the PDF.

<u>Option</u>	<u>Description</u>
CODEPAGE=number	<p>The codepage to use (1252 Windows Latin-1 is used by default). This option is valid only when embedding your own TrueType or OpenType font. Must be a codepage that is included in the font. Currently, the other codepages supported are:</p> <p>IDENTITY (None - text is Unicode)^{1,2} 737 (Greek)¹ 855 (Cyrillic)¹ 857 (Turkish)¹ 862 (Hebrew)¹ 866 (Cyrillic)¹ 869 (Greek)¹ 874 (Thai)¹ 932 (Japanese)¹ 936 (GBK - Chinese)¹ 949 (Korean)¹ 1250 (Central European) 1251 (Cyrillic) 1253 (Greek) 1254 (Turkish) 1255 (Hebrew) 1256 (Arabic)</p> <p>¹ Use the UNICODE option with these code pages and, typically, SUBSET ² All characters using this font, even those which can be represented by a single byte, must be double byte.</p> <p>Note you may also use the UNICODE option and pass text in UTF-8 format.</p>
BOM=text	<p>The Byte Order Mark for use with Unicode text and IDENTITY as the CODEPAGE value. Set this to the text "FFFE" if the Unicode text is in little-endian order. For example, if your text is the two-byte hex string D5 5C for the Korean symbol  then you do not need to use this option. If your string is 5C D5 then set BOM="FFFE".</p>

<u>Option</u>	<u>Description</u>
ENCODING=text	The encoding to use for the custom font. WinAnsiEncoding is used if not specified. This value is inserted directly into the PDF as typed so case is important. If you are not sure what value to use, leave this option out. The default should be fine for most cases. Possible values are WinAnsiEncoding, StandardEncoding, MacRomanEncoding or PDFDocEncoding.
UNICODE	Specifies that you will be using Unicode characters with this font. This allows you to enter text in Unicode for languages such as Hebrew and Arabic. The font must contain the Unicode character set you plan on using. See the Text Files section for more information.
SUBSET	Use this option to subset fonts with the UNICODE option and CODEPAGE set to IDENTITY, 737, 855, 857, 862, 866, 869, 874, 932, 936, or 949. This will reduce the PDF size by including only those glyphs from the font that are used in the PDF.
REPLACE="text"	Only use this option if you want to replace an existing font. Specify the name of an existing font in the PDF you want to replace. This name is case sensitive.
REPLACEIFSUB	Only make the replacement if the current font is a subsetted font.

Here's how to load a font file:

```
pdfmeld.exe input.pdf output.pdf -fonts fontlist.dat
```

Adding Custom Bookmarks

The `-bmin` option or `setBookmarkFile` method allows you to specify a file with the bookmark structure to use for the output PDF. Use the `-bmout` option or `setBookmarkOut` method to create a file of existing bookmarks in a PDF.

You may use the `UTF-8` option on the tag if the description contains UTF-8 characters. Or you may use Unicode (UTF-16). You'll need place two ASCII characters - 254 followed by 255 (FEFF in hex) - at the front of the string (the `DESCR` option) when using Unicode.

You may save the entire file in Unicode format and the program will perform the necessary conversions. You'll need to place two ASCII characters - 254 and 255 for big-endian or 255 and 254 for little-endian - at the front of your file. Also, each `BOOKMARK` tag must be closed with `/>` when the entire file is Unicode (rather than just `>`).

You may also use the syntax `✏` where 9999 is the decimal Unicode value, `香` where 9999 is the hexadecimal Unicode value or `&#o9999;` where 9999 is the octal Unicode value in the description (`DESCR` option). For example, `ا` is the same as `ا` and `&#o3047.` You must place the `UTF-8` option in the `BOOKMARK` tag when entering characters this way.

It is recommended you place the tag `<UNICODE />` at the top of your file if you are using UTF-8 or Unicode and the entire file is not Unicode. This tag will instruct the program to look for `/>` as the closing sequence to the `BOOKMARK` tag. This is to avoid closing the tag too soon if there are any `>` characters in the description text. All of your tags (including the `UNICODE` tag) must end with `/>` in this case.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `BOOKMARK` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<BOOKMARK LEVEL=2 CLOSED PAGE=5 DESCR="A Bookmark">
```

```
<BOOKMARK
  LEVEL=number
  CLOSED
  X="text"
  Y="text"
  COLOR="text"
  ITALIC
  BOLD
  PAGE=number
  URL=text
  DESCR="text"
  PAGEFMT="text"
  PAGENUM=number
  STARTPAGEOFFSET=number
  INITPAGEFMT="text"
  FDFFILE="text"
  PRINT[=text]
  JS="text"
  UTF-8>
```

Used to add a bookmark.

<u>Option</u>	<u>Description</u>
LEVEL=number	The bookmark level (1 is the highest). Lower numbers represent sub-entries for the level above.
CLOSED	Include this option for a closed (lower levels are not initially displayed) bookmark. Closed bookmarks show up with a + sign next to them in the viewer and the user must click on the + to expand the bookmark to show the levels below it.
X="text"	Optional. The position from the left edge of the page in points this bookmark is positioned at. Pass a numeric value or the text "null". Default is null.
Y="text"	Optional. The position from the top edge of the page in points this bookmark is positioned at. Pass a numeric value or the text "null". Default is 7920.
COLOR="text"	The bookmark color . Default is black.
ITALIC	Optional. Specify italic text for the bookmark.
BOLD	Optional. Specify bold text for the bookmark.

Bookmarks

<u>Option</u>	<u>Description</u>
PAGE=number	The page number this bookmark references. Leave this option off when using the URL option.
URL="text"	The URL for a web page to load rather than a link to a page in the current PDF document. You may also use "mailto:me@mymysite.com" syntax to open an email window. Leave this option off for a standard outline entry.
DESCR="text"	The actual text for the bookmark. This is what the user will see in the bookmark window. Use < for the < symbol and > for the > symbol. You may also use the syntax &#xHH where HH is the hex code for any character value from 0-255 you wish to use. You may supply big-endian Unicode (UTF-16) but place two ASCII characters - 254 followed by 255 (FEFF in hex) - at the front of the string.
PAGEFMT="text"	Similar to the -pagefmt option, this is the page format to start using on this page. You may use a blank string to clear out the page number format and thus turn off printing of the page number.
PAGENUM=number	The new starting page number to begin using. You would set this to 1, for example, if you are numbering your pages by sections and are starting a new section at this bookmark.
STARTPAGEOFFSET=number	The offset from the PAGE value specified for this bookmark where the PAGEFMT should start. For example, you might have a header page that starts each section and do not want to start numbering until the next page. In that case you would set this value to 2. The default value is 1 if this option is not used - that is, start the PAGEFMT setting at this page.

<u>Option</u>	<u>Description</u>
INITPAGEFMT="text"	When used with STARTPAGEOFFSET, this specifies the PAGEFMT to use until we reach the STARTPAGEOFFSET. For example, assume a bookmark with PAGE=5 is included with the options of PAGEFMT="%1 of %2", PAGENUM=1, STARTPAGEOFFSET=2 and INITPAGEFMT="". On page 5 of your PDF, no page numbering will show because INITPAGEFMT="" and STARTPAGEOFFSET is not reached. On page 6 (the second page in from the bookmark, which STARTPAGEOFFSET=2 is specifying) the PAGEFMT takes over along with PAGENUM=1. So page 6 would show "1 of x", page 7 would have "2 of x" and so on until the end of the PDF is reached or another bookmark containing new settings for these options is encountered.
FDFFILE="text"	Supply the path and file name of an FDF file to load when the bookmark is clicked. Leave this option off for a standard outline entry.
PRINT[=text]	The outline entry will be used as a print function rather than a link to a page in the current PDF document. Use PRINT by itself to simply print to the default printer. Use PRINT=Dialog to bring up the printer dialog box. Leave this option off for a standard outline entry.
JS="text"	Enter any valid Adobe JavaScript code you want executed when the bookmark is clicked. Leave this option off for a standard outline entry.
UTF-8	Use this when the DESCR value contains UTF-8 characters or you are passing Unicode in the form of ✏. The text will be converted to Unicode. You do not need the ASCII characters 254 followed by 255 at the beginning of the string in this case.

Here's an example of a bookmark structure with 3 levels. The upper levels are closed initially.

```
<BOOKMARK LEVEL=1 CLOSED PAGE=1 DESCR="Contents">  
<BOOKMARK LEVEL=2 PAGE=1 DESCR="Part I">
```


Bookmarks

```
<BOOKMARK LEVEL=3 PAGE=1 DESCR="Section I">
<BOOKMARK LEVEL=3 PAGE=3 DESCR="Section II">
<BOOKMARK LEVEL=3 PAGE=4 DESCR="Section III">
<BOOKMARK LEVEL=2 CLOSED PAGE=5 DESCR="Part II">
<BOOKMARK LEVEL=3 CLOSED PAGE=5 DESCR="Section I">
<BOOKMARK LEVEL=3 CLOSED PAGE=8 DESCR="Section II">
<BOOKMARK LEVEL=3 CLOSED PAGE=12 DESCR="Section III">
<BOOKMARK LEVEL=1 CLOSED PAGE=15 DESCR="Appendix">
<BOOKMARK LEVEL=2 CLOSED DESCR="More Info"
    URL="http://www.mysite.com">
<BOOKMARK LEVEL=2 CLOSED DESCR="Request Info"
    URL="mailto:me@mysite.com">
<BOOKMARK LEVEL=2 CLOSED DESCR="Print Report" PRINT>
```

The user will see two bookmarks ("Contents" and "Appendix") when the PDF is opened using the example above. Both will have a + sign next to them for expansion. Note that "Part I" is not marked as initially closed so all the detail under will be shown initially when the "Contents" bookmark is expanded.

Create a bookmark file from an existing PDF like this:

```
pdfmeld.exe input.pdf bm.dat -bmout
```

Load the bookmarks (assume they are in a file called bm.dat) and create the PDF by running:

```
pdfmeld.exe input.pdf output.pdf -bmin bm.dat
```

Here is an example using Unicode to create Chinese text:

```
<BOOKMARK LEVEL=1
    DESCR="&#20808;&#21069;&#24050;&#32463;&#21457;&#21
806;" UTF-8>
```

Note there are several ways of entering Unicode. This example uses the method of passing the decimal value of the Unicode character to show. You may also pass Unicode text directly.

Keep the following in mind when updating your tagged file:

- No spaces between the option and = sign
- No spaces between = and the value

Adding Custom Annotations

The `-annotes` option or `setAnnotesFile` method allows you to specify a file with the annotations to use for the output PDF. There are two types of annotations you may apply.

The first is a popup note. This type appears as a small yellow note icon on the page. Double click the note icon to open it up and read the contents. This allows you to place additional notes or text on the page without having it print. You may also use a stamp (custom image or text) instead of the note icon.

The second type is a web link. This type may or may not have a border, depending on how you format it. Clicking the link area on the page opens a browser to the specified location.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `A` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<A X1=200 Y1=300 TITLE="Passed" NOTE="Some text..." ANGLE=45
    LABEL="Approved" SIZE=20 LABELCOLOR=#FF0000 TRANSPARENCY=50 PAGES=1>
<A X1=72 Y1=72 X2=300 Y2=92 HREF="https://www.fytek.com" PAGES=3>
<A X1=140 Y1=300 X2=300 Y2=500 TITLE="Important" NOTE="Some text..." PAGES=5>
```

are tags to create a text stamp note on page 1, a link on page 3 to <https://www.fytek.com> and a note on page 5.

The default unit is a point (1/72 of an inch) for the position of the annotations (the `X1`, `X2`, `Y1` and `Y2` values). You may use the `UNITS` tag to specify a different value. The `UNITS` tag must come before the `A` tag(s) that it is to be used for. The syntax is:

```
<UNITS VALUE="text">
```

Set the `VALUE` option to `pt` for points (the default), `in` for inches, `cm` for centimeters or `mm` for millimeters. In addition you may also set to an arbitrary value. The value you pass will be used as a multiplier for the number of points per unit. For example, passing a value of 72 will be the same as setting to "in" for inches. Setting to 28.35 is the same as "cm" ($72 / 2.54 = 28.35$). You have the flexibility to make up your own numbering scheme however.

Annotations

```
<A
  X1=number
  Y1=number
  X2=number
  Y2=number
  HREF="text"
  STRING="text"
  TITLE="text"
  NOTE="text"
  COLOR="text"
  SRC="text"
  LABEL="text"
  FACE="text"
  SIZE=number
  LABELCOLOR="text"
  LABELBGCOLOR="text"
  LABELBORDERCOLOR="text"
  NOPRINT
  NOVIEW
  ANGLE=number
  TRANSPARENCY=number
  TRANSPMODE="text"
  PAGES="text">
```

Used to add an annotation.

<u>Option</u>	<u>Description</u>
X1=number	The left X coordinate (in points where 1 point = 1/72 of an inch) of the rectangle for the annotation. The position 0,0 is the lower left corner of the page.
Y1=number	The lower Y coordinate (in points where 1 point = 1/72 of an inch) of the rectangle for the annotation.
X2=number	The right X coordinate (in points where 1 point = 1/72 of an inch) of the rectangle for the annotation.
Y2=number	The upper Y coordinate (in points where 1 point = 1/72 of an inch) of the rectangle for the annotation.
HREF="text"	The URL to link to (example, "https://www.fytek.com").

Annotations

<u>Option</u>	<u>Description</u>
STRING="text"	A text string to search for in the document to apply the link to. Use along with the HREF option. The location (X1, X2, Y1, and Y2) information is not needed as the location of the string will determine the placement of the link. You may specify -strcolor (or setStringColor) and -strcolormode (or setStringColorMode) to highlight the links with an underline or rectangle if desired. May not work on all PDFs.
TITLE="text"	The title for the open annotation when using a pop-up note (do not use with HREF).
NOTE="text"	The note text when using a pop-up note (do not use with HREF).
COLOR="text"	The pop-up note block color .
SRC="text"	Used to stamp an image on the PDF that allows for a pop-up note (do not use with HREF or LABEL). Set to the image file you want to use as the stamp (example, "c:\mypics\draft.jpg"). Use the TITLE and NOTE options along with this one. May also want to set the TRANSPARENCY value to allow some of the background to show through. Only use JPEGs at 72 DPI 256-color grayscale or 24-bit color, GIFs, or PNGs (alpha transparency in PNGs is not supported - the image will show without transparency) The X2 and Y2 values may be left off for images and will be based on the image size. If they are used, the image will be scaled to fit the size specified.

Annotations

<u>Option</u>	<u>Description</u>
LABEL="text"	Used to stamp text on the PDF that allows for a pop-up note (do not use with HREF or SRC). Set to the text string you want to use as the stamp (example, "Approved"). Put the text " " (without the quotes) in the string for a new line. You may also use a font tag such as "" in the label where x is the font size and c is the color in hex (red, green and blue components in that order - each 2 characters in hex from 00 to FF). The FONT tag, if used, must come at the beginning of a line, that is, directly after the . Use the TITLE and NOTE options along with this one. May also want to set the TRANSPARENCY value to allow some of the background to show through. The X2 and Y2 values may be left off for the label and will be based on the text size. If they are used, the text may be scaled.
FACE="text"	The text face to use. Either "Helv" for Helvetica (default), "Times" for Times-Roman or "Courier" for Courier.
SIZE=number	The font point size (where 1 point = 1/72 of an inch) for the height of the text specified with the LABEL option.
LABELCOLOR="text"	The label text color .
LABELBGCOLOR="text"	The label background color .
LABELBORDERCOLOR="text"	The label border color .
NOPRINT	Note or stamp is not printed.
NOVIEW	Note or stamp is not displayed but is printed if the NOPRINT option is not used.
ANGLE=number	The counter-clockwise angle of rotation for the stamp. Enter the number of degrees from 0 (default) to 359.

Annotations

<u>Option</u>	<u>Description</u>
TRANSPARENCY=number	Optional transparency for the stamp image or text. Requires Acrobat or Reader 5.0 or higher to view the transparency. Enter a value from 1 to 100. The default without this option is 100 - the image or text is placed on top of the background PDF with none of the background showing through.
TRANSPMODE="text"	Optional transparency mode. The valid values are: Normal (Default) Multiply Screen Overlay Darken Lighten ColorDodge ColorBurn HardLight SoftLight Difference Exclusion Hue Saturation Color Luminosity
PAGES="text"	The page numbers for the annotation. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.

Annotations

**<UNITS
VALUE=text>**

Used to set the unit of measurement for tags that follow. Typically you'll place one UNITS tag at the top but you may use them through the file with different values. The UNITS tag affects all tags that follow it until another UNITS tag is found.

For example, to set centimeters use <UNITS VALUE="cm">. The default is "pt" for points.

<u>Parameter</u>	<u>Description</u>
VALUE=text	Use one of the following: <u>PT</u> Points (1/72 of an inch) <u>IN</u> Inches <u>CM</u> Centimeters <u>MM</u> Millimeters

Overlay File

The `-overlayfile` option or `setOverlayFile` method allows you to specify a file containing text to search for in the base PDF to determine what background PDF page to use for the overlay. Use this along with `-overlay` or `setOverlay`.

For example, you might have a PDF that contains billing statements for multiple customers. Some customer statements might be one page while others span several pages. Assume your background PDF has two pages - one to use for the final page of each statement and one to use for any pages that are continued. You would setup this file to look for the text "continued on next page" (assuming that's the exact text used in your billing PDF). If the text is found, you want to use page 2 of the background PDF. Otherwise, you want to use page 1. This option would allow you to specify the background page to use.

The file layout consists of a set of tags containing a search string and the background PDF page number. There is a tag used to specify the default background page and one to specify text to match on. In the example above, the default would be page 1.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<DEFAULT PAGE=1>  
<SEARCH STR="continued on next page" PAGE=2>
```

are the tags that correspond to the example.

You may include as many `SEARCH` tags as you wish but only one `DEFAULT` tag should be specified. The searching is done in the order the `SEARCH` tags are found. Once a match is found, no further checking is done and the page specified is used. So, for example, if you have one `SEARCH` tag that has `STR="program"` followed by a `SEARCH` tag that has `STR="programming"`, only the first `SEARCH` will ever match since "program" is a substring of "programming".

Overlay File

<DEFAULT
PAGE=number>

Used to specify the default background page.

<u>Option</u>	<u>Description</u>
PAGE=number	Required. The page number of the background PDF to use when there are no string matches.

```
<SEARCH  
  STR="text"  
  PAGE=number  
  CASE>
```

Used to specify a text string to match on.

<u>Option</u>	<u>Description</u>
STR="text"	Required. The string to look for in the base PDF. This is not case sensitive. Note that text is not necessarily contiguous in a PDF and it may not be possible to perform a match. In these cases, try to find other text on the page that might work or shorten your search string. You may want to use the -textout option or setTextFileOut method so you can see what the program considers as the text strings on each page if you are having difficulty in making a match.
PAGE=number	Required. The page number to use from the background PDF when a match is found.
CASE	Specify this on the tag if you want the search to be case sensitive.

Text Detail Files

(Note: text detail options are not available in the [SE version](#))

The `-textdetailin` and `-textdetailout` options or `setTextDetailIn` and `setTextDetailOut` methods allow you to locate and export ASCII text from your PDF. Note that text is not necessarily contiguous in a PDF so these methods may not work on all types of PDFs. There is also a `setTextDetailOutXY` method and `-textdetailoutxy` option that allows you to focus on a small section of the page and only extract text from that section.

For example, you might have a PDF that contains a set of names and addresses. Assume each page has a different name and address in the header area and you want to extract that information to generate address labels. You would use `-textdetailout` or `setDetailFileOut` to determine what area on the page should be extracted. With that information, you create a small file with the desired layout. The file is then used as a template for `-textdetailin` or `setDetailFileIn`. Note the data will need to be in the same location throughout the PDF for this to work. Or use the `-json/setUseJSON` option to export the output in JSON format.

The file layouts are tag based and similar for both the export and import options. Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value.

There are two sets of X/Y string position values exported. The X, X2, Y, and Y2 values are the bounding box for text prior to any matrix operations. The TX, TX2, TY, and TY2 values are the bounding box for text after matrix operations. These values will be the same if the identity matrix is used.

For example:

```
<TEXTDETAIL PAGE=1 X=5 Y=10 STR="Some Text">
```

You would first use the detail out option or method to find out where text is located like this:

```
pdfmeld.exe input.pdf str.txt -textdetailout -pages 1
```

Text Detail Files

For example, you should see something like this in str.txt (some font information is not shown for readability):

```
<TEXTDETAIL PAGE=1 STRID=1 X=36 Y=638 STR="This is another line.">
```

Next, use the information in str.txt to setup your template. For example, to pull out address information, you could setup something like this:

```
"<TEXTDETAIL X=36 Y=746>" ,  
"<TEXTDETAIL X=36 Y=734>" ,  
"<TEXTDETAIL X=36 Y=722 CITY>" ,  
"<TEXTDETAIL X=36 Y=722 STATE>" ,  
"<TEXTDETAIL X=36 Y=722 ZIP>" ,  
"<TEXTDETAIL X=36 Y=710>"
```

The actual template layout would be all on one line; however it is shown here as with each TEXTDETAIL tag on a separate line for readability. Note the X/Y coordinates are taken from the export file (str.txt in this example). The TEXTDETAIL tags will be replaced with the text found at the location specified. The quotes are entered in the sample template so the text is quoted in the output. Once you have the template set, use it like this:

```
pdfmeld.exe input.pdf addr.txt -textdetailin templ.txt
```

The addr.txt file will then contain the following, based on this example:

```
"Jim Dandy", "123 Street", "Anywhere", "XX", "89765", "USA"  
"John Doe", "555 South St.", "Somecity", "XX", "42765", "USA"
```

and so on for all the addresses.

<TEXTDETAIL
PAGE=number
STRID=number
FONTID=text
FONTNAME=text
BASENAME=text
X=number
Y=number
X2=number
Y2=number
TX=number
TY=number
TX2=number
TY2=number
STR="text"
STRHEX="text"
START=number
LENGTH=number
CITY
STATE
ZIP>

Both the export and import methods use the TEXTDETAIL tag. Some options are for export (-textdetailout or setTextDetailOut) and others are for import (-textdetailin or setTextDetailIn).

<u>Option</u>	<u>Description</u>
PAGE=number	<i>(Export only)</i> The page number the text was found on.
STRID=number	<i>(Export only)</i> A unique number for a block of text at the same X/Y starting position.
FONTID=text	<i>(Export only)</i> The font code used by this text in the PDF.
FONTNAME=text	<i>(Export only)</i> The font name used by this text in the PDF. May be same as FONTID if not named.
BASENAME=text	<i>(Export only)</i> The basename or type of font.
X=number	Required. The X coordinate for the text. For export, this where the text was found. For import, this is the location to extract text from.
Y=number	Required. The Y coordinate for the text. For export, this where the text was found. For import, this is the location to extract text from.

Text Detail Files

<u>Option</u>	<u>Description</u>
X2=number	The ending X coordinate for the text.
Y2=number	The ending Y coordinate for the text.
TX=number	The X coordinate for the text after applying any matrix transformations.
TY=number	The Y coordinate for the text after applying any matrix transformations.
TX2=number	The ending X coordinate for the text after applying any matrix transformations.
TY2=number	The ending Y coordinate for the text after applying any matrix transformations.
STR="text"	<i>(Export only)</i> The text string found at the X/Y location.
STRHEX="text"	<i>(Export only)</i> The hex version of the text string found at the X/Y location.
START=number	<i>(Import only)</i> Optional. The starting character position (counting from 1) at the location specified to extract text from. The default is 1.
LENGTH=number	<i>(Import only)</i> Optional. The length or number of characters to extract starting from the START position. The default is the entire string.
CITY	<i>(Import only)</i> Optional. Attempts to extract the city when the string contains "City, ST Zip" format.
STATE	<i>(Import only)</i> Optional. Attempts to extract the state when the string contains "City, ST Zip" format.
ZIP	<i>(Import only)</i> Optional. Attempts to extract the zip when the string contains "City, ST Zip" format.

JSON format contains similar information to the text or text detail export. The text fragments are contained in the "details" structure within the page contents. If you want to make a modification to a string, change the "text" value within the "details".

Some fonts contain an alternate location for some of the glyphs. This means that a given character is stored in the font in a different location meaning you need to map the character position to the proper location. For example, a number 5 might be where the letter A is normally stored. There is a "differences" mapping that is included when exporting text details (with `-textdetailout` or `setTextDetailOut`) that can be used to map the characters. Or, add a new property called "textconv" and set the value you wish to have converted. Not every font includes this information in the PDF so this may not work in all situations.

Below is an example of the JSON formatted output using `-textout` or the method `setTextFileOut`. Note the "display" value is included only to give a better representation of the text fragments. Changing the "display" value will not update any text in the PDF if you load the changes against the PDF. You may include the entire file when making text changes or just an array of "text" and "id" structures that have changes. Use `-json "details"` (or `setUseJSON("details")`) to provide the "details" nodes.

```
[
  {
    "page" : "1",
    "contents" : [
      {
        "id" : 1,
        "details" : [
          {
            "text" : "\u0000i½",
            "id" : 1
          }
        ],
        "display" : "(copyrightserif)"
      },
      {
        "id" : 2,
        "details" : [
          {
            "text" : "Tes",
            "id" : 2
          },
          {
            "text" : "t",
            "id" : 3,
            "spacer" : "true"
          }
        ],
        "display" : "Test"
      }
    ]
  }
]
```

Text Detail Files

```
}  
]
```

Using `-textdetailout` or `setTextDetailOut` provides more details on the x/y positioning.

```
[  
  {  
    "page" : "1",  
    "contents" : [  
      {  
        "x2" : 443.8029,  
        "fontid" : "F1",  
        "fcolor" : "0,0,0",  
        "x" : 48.0897,  
        "fontname" : "AFDDKP+Symbol",  
        "details" : [  
          {  
            "x2" : 54.3861,  
            "tx" : 48.0897,  
            "ty2" : 70.5259,  
            "x" : 48.0897,  
            "tx2" : 54.3861,  
            "y" : 70.5259,  
            "text" : "\u0000i½",  
            "id" : 1,  
            "y2" : 62.5558,  
            "ty" : 62.5558  
          },  
          ...  
        ],  
        "size" : "7.970",  
        "display" : "(copyrightserif) Test String",  
        "y" : 70.5259,  
        "basename" : "AFDDKP+Symbol",  
        "id" : 1,  
        "y2" : 62.1573  
      }  
    ]  
  }  
]
```


Adding External Files

(Note: This feature is not available in [SE version](#))

The `-embed` option or `setEmbedFile` method allows you to bundle external files into the PDF. The end user can save or open these files in the appropriate application.

The input file in this case is a text file in a tag based (or HTML-like) layout. You may also pass in the tag(s) themselves as input to `-embed` or `setEmbedFile`. Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `EMBED` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<EMBED SRC="c:\myexcel.xls" DESCR="Financial Data">
```

is a tag to embed the file `myexcel.xls` with the description "Financial Data".

Here are the options for an `EMBED` tag:

<u>Option</u>	<u>Description</u>
<code>SRC="text"</code>	The full path and file to embed.
<code>DESCR="text"</code>	The description the end user will see for the file.
<code>MIME="text"</code>	The mime type for the file (such as <code>application/vnd.ms-excel</code> for Excel). You may omit this on Windows based systems and the software will attempt to determine the mime type from the system registry. Note this is the system registry from where the PDF is built and not the end user's machine.

Here are some other examples:

```
<EMBED SRC="c:\my audio\meeting.mp3"
```

```
  MIME="audio/mp3" DESCR="Audio from last meeting">
```

```
<EMBED SRC="c:\my files\detail.pdf" DESCR="Detailed information">
```

Users may view the list of attached files using Acrobat or Reader. Refer to the documentation for your version of Acrobat or Reader to find out how to see the list of attached documents.

Attach the files (assume the list is called `embed.dat`) and create the PDF by running:

```
pdfmeld.exe input.pdf output.pdf -embed embed.dat
```

URLs

The `-urls` option or `setURLFile` method allows you to modify existing URL targets in a tag based (or HTML-like) layout. The input file in this case is a text file containing the old and new URL values. Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `A` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<A OLDHREF="www.oldsite.com" NEWHREF="www.newsite.net">
```

is a tag to modify an existing URL. In this case, any URLs such as `http://www.oldsite.com/index.html` will be changed to `http://www.newsite.net/index.html`.

You may change any portion of the existing URL. The text shown on the page is not changed however, only the web link itself.

Here are the options for an `A` tag:

<u>Option</u>	<u>Description</u>
<code>OLDURL="text"</code>	The full or partial URL text to change.
<code>NEWURL="text"</code>	The replacement value that corresponds to the <code>OLDURL</code> value.

Here are some other examples:

```
<A OLDHREF="index.html" NEWHREF="index.php">  
<A OLDHREF="www.mysite.com" NEWHREF="apps.mysite.com">  
<A OLDHREF="http://www" NEWHREF="https://www">
```

Create your file then modify existing URLs by running:

```
pdfmeld.exe input.pdf output.pdf -urls urllist.dat
```

Data Files

(Note: This feature is not available in [SE version](#))

The `-data` option or `setData` method allows you to specify data for fields in a tag based (or HTML-like) layout. The input file in this case is a text file containing the PDF(s) to import along with the values for the fields. The concept is similar to using an FDF file where you have field names and values.

For the DLL, the `setDataPDF` and `setDataField` methods allow you to pass the same information without creating the tag based file. With these methods you pass only the options for the PDF and FDFFIELD tags and do not need an input file. The input is created by PDF Meld based on the data passed to these methods in the same order as they are called. That is, PDF Meld internally builds the input file described below dynamically as you make calls to `setDataPDF` and `setDataField`.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example `<PDF SRC="c:\pdf\input1.pdf">` is a PDF tag setting the option SRC to the value `c:\pdf\input1.pdf`.

Note that you must know what the names of the fields are in the target PDF and the types of values they take. One way to determine this information is by using the `-fdfout` option or `setFDFFileOut` method to create an output FDF file.

The file structure consists of two tags. First is the `<PDF>` tag. This tag tells the software to include a PDF into the output. Second is the `<FDFFIELD>` tag. This tag tells the software what field and value to plug into the PDF. You may have as many PDF tags as you want - each may reference a different PDF file or the same one. You may include the same PDF multiple times with different values for the same field names. You may use an XML formatted layout instead - see the end of this section for an example.

See the [fillable fields](#) section for information on adding fields to a PDF.

```
<PDF
  SRC="text"
  STATIC
  FLATTEN[="text"]
  NOPREFIX
  KEEP NAMES>
```

Used to specify the PDF that is to be filled.

<u>Option</u>	<u>Description</u>
SRC="text"	Required. The source PDF to include. For example, "c:\my pdfs\file1.pdf".
STATIC	Flattens or removes the form fields being set - others are left in the form. May use in conjunction with FLATTEN. May not work on every PDF depending on what rotations or page settings are in use.
FLATTEN[="text"]	Flattens or removes all form fields. Same as -flatten or setFieldsFlatten. Optionally pass in the text "sig" to not remove signature fields so they can be signed later.
NOPREFIX	Removes any prefix from field names. These are set depending on the software used to create your fillable PDF. Only use if your PDF contains one form set and there are no duplicate names. For example, a field called form1[0].#subform[0].TextField1[0] in the PDF can simply be referred to as TextField1 for the NAME option in the FDFFIELD tag.
KEEP NAMES	Set this option to keep the field names. Ordinarily, fields are renamed so as not to have duplicates when using more than one PDF.

```
<FDFFIELD
  NAME="text"
  VALUE="text"
  SRC="text"
  KEEPSIZE
  SCALE=number
  SCALEX=number
  SCALEY=number
  FACE="text"
  SIZE=number
  COLOR="text"
  COMP=number>
```

Used to specify the field values.

<u>Option</u>	<u>Description</u>
NAME="text"	Required. The name of the field in the PDF.
VALUE="text"	The value for the corresponding field.
SRC="text"	The path and name of an image file to place as the background. For example, you might use this to place a scanned image of a signature in a field rather than text.
KEEPSIZE	By default, the image is expanded or contracted to fit the container of the field. This option keeps the image at its original height and width.
SCALE=number	A scaling factor expressed as a percentage for the image. Default is 100 for 100% or no scaling. Values smaller than 100 will shrink the image while values greater than 100 will enlarge.
SCALEX=number	A scaling factor expressed as a percentage for the image width. Default is 100 for 100% or no scaling. Values smaller than 100 will shrink the image width while values greater than 100 will enlarge.
SCALEY=number	A scaling factor expressed as a percentage for the image height. Default is 100 for 100% or no scaling. Values smaller than 100 will shrink the image height while values greater than 100 will enlarge.

Data Files

<u>Option</u>	<u>Description</u>
FACE="text"	<p>When using <i>STATIC</i> on the PDF tag Set to one of the following:</p> <ul style="list-style-type: none">CourierHelvetica (Default)Times-RomanCourier-BoldHelvetica-BoldTimes-BoldCourier-ObliqueHelvetica-ObliqueTimes-ItalicCourier-BoldObliqueHelvetica-BoldObliqueTimes-BoldItalic
SIZE=number	<p>When using <i>STATIC</i> on the PDF tag The font point size. The default is 10.</p>
COLOR="text"	<p>When using <i>STATIC</i> on the PDF tag The text color.</p>
COMP=number	<p>When using <i>STATIC</i> on the PDF tag The percentage amount to horizontally compress the text by. Enter as a whole number (for example, use 80 for 80%). The default is 100.</p>

```
<URL  
  OLDHREF="text"  
  NEWHREF="text"  
  NAME="text"  
>
```

Used to modify URLs in the PDF. This can be used to take a generic URL reference in the background PDF and make it specific to a particular customer or invoice, for example.

<u>Option</u>	<u>Description</u>
OLDHREF="text"	Required. The existing URL. For example, "https://www.mysite.com".
NEWHREF="text"	Required. The replacement URL. For example, "https://www.mysite.com?cust=123&invoice=555".
NAME="text"	The field name for the link. If you are flattening the form fields you will need to pass the field name so the link action is not removed with the rest of the fields. Otherwise the link will display but no longer work when clicked.

The FDFFIELD tags must appear after the PDF tag for the PDF they go with. For example, if you have a PDF with fields A, B and C then you first include a PDF tag followed by the FDFFIELD tags for these fields. Each tag must go on a separate line. Any FDFFIELD tags occurring before a PDF tag are global settings. You might use this if you're including the same PDF three times but there are some fields you always want set the same. You'd put the FDFFIELD tags for these common fields at the top with the common values.

Here's an example file with 2 common fields at the top:

```
# Sample data file
# First list the defaults
<FDFFIELD NAME="first_name" VALUE="Jane">
<FDFFIELD NAME="last_name" VALUE="Doe">

# Now pull in PDF files
<PDF SRC="c:\pdf\file1.pdf">
# Now set the fields
<FDFFIELD NAME="address" VALUE="123 Main">
<FDFFIELD NAME="state" VALUE="AK">

# Now pull in another PDF file
# This is a repeat of the first
<PDF SRC="c:\pdf\file1.pdf">
# Now set the fields
<FDFFIELD NAME="address" VALUE="555 North">
<FDFFIELD NAME="state" VALUE="AZ">

# Now pull in another PDF file
# Again, a repeat of the first
<PDF SRC="c:\pdf\file1.pdf">
# Note the global last_name is set to something
# different - only for this PDF
<FDFFIELD NAME="last_name" VALUE="Smith">
<FDFFIELD NAME="address" VALUE="987 East">
<FDFFIELD NAME="state" VALUE="MI">

# Now pull in a different PDF file
<PDF SRC="c:\pdf\file2.pdf">
<FDFFIELD NAME="gross" VALUE="92350.25">
<FDFFIELD NAME="net" VALUE="75000.00">
```

Note the above example will create a PDF containing 4 PDFs. Three of the four were based from the same PDF. The last_name value overridden in the 3rd PDF is only for that PDF. The value goes back to the global value once the next PDF tag is found.

Load the data (assume the tag file is called input.dat) and create the PDF by running:

```
pdfmeld.exe input.dat myfile.pdf -data
```


Keep the following in mind:

- Each tag must be on a separate line and not break across a line
- No spaces between the option and = sign
- No spaces between = and the value

To accomplish the same thing using setDataPDF and setDataField instead, you would call the methods as follows. Note the ordering of the setDataPDF vs. setDataField calls is important. The input file will be built internally in the same order as the methods are called.

```
Set PDF = CreateObject("pdf.Meld")

PDF.setOutFile "c:\temp\fileout.pdf"

PDF.setDataField "first_name", "Jane"
PDF.setDataField "last_name", "Doe"

PDF.setDataPDF "c:\pdf\file1.pdf"

PDF.setDataField "address", "123 Main"
PDF.setDataField "state", "AK"

PDF.setDataPDF "c:\pdf\file1.pdf"

PDF.setDataField "address", "555 North"
PDF.setDataField "state", "AZ"

PDF.setDataPDF "c:\pdf\file1.pdf"

PDF.setDataField "last_name", "Smith"
PDF.setDataField "address", "987 East"
PDF.setDataField "state", "MI"

PDF.setDataPDF "c:\pdf\file1.pdf"

PDF.setDataField "gross", "92350.25"
PDF.setDataField "net", "75000.00"

rslt = PDF.buildPDF
```

You may also specify your input in an XML format. Here's one way to structure the above example using XML:

```
<?xml version="1.0"?>
<fdldata>
  <first_name>Jane</first_name>
  <last_name>Doe</last_name>
```

Data Files

```
<PDF SRC="c:\pdf\file1.pdf">
  <address>123 Main</address>
  <state>AK</state>
</PDF>

<PDF SRC="c:\pdf\file1.pdf">
  <address>555 North</address>
  <state>AZ</state>
</PDF>

<PDF>
  <SRC>c:\pdf\file1.pdf</SRC>
  <last_name>Smith</last_name>
  <address>987 East</address>
  <state>MI</state>
</PDF>

<PDF SRC="c:\pdf\file2.pdf" gross="92350.25">
  <net>75000.00</net>
</PDF>
</fdldata>
```

The only required name is the SRC option on the PDF tag. Any SRC option with a value containing ".pdf" at the end will be used as the standard <PDF> tag. The SRC option can either be included with the PDF tag or appear separately in the PDF block. All other name/value settings will be used as the standard <FDFFIELD> tag.

SQL Queries

An SQL Query may be used to extract fillable field information from a data source. The query is placed in a file that is read during the build process or, with the DLL version, you can pass the query statement text using the `setSQL` method. Only one query statement is allowed so you may need to create a view to encapsulate all the information you need to extract. Note that field names are case-sensitive so you may need to place quotes around column names so they match the PDF field names. For example, select name "Name" from customers.

The output file can be a PDF file when you are expecting a single row or you may provide a directory name and one PDF will be created for each row processed. The output file name will be named the same as the input PDF along with a sequence number to make it unique.

You may optionally provide a unique identifier in your SQL with a column name of lower-case "key". This column will then be appended to the input file name when creating the name of the output file.

Use the `-sqlcomb` option or `setSQLComb` method to create a single PDF from the individual PDFs created when the SQL returns multiple rows. You may optionally provide a bookmark description in your SQL with a column name of lower-case "bm". This column will then be used as the bookmark in the single combined PDF. Use the `-bmkeep` option or `setBookmarkKeep` method as well in this case.

Use `-sqlcmds` on Unix to load the needed SQL library. Because of differences among Unix systems, the libraries may not be compatible so they are not loaded unless this option is used.

Database Connection

The database connection, specified with the `-sqldb` or `SetSQLDB` method, is used to set the parameters necessary for connecting to your database. Each type of connection has different options. In addition, you may specify the connection options on the [QUERY](#) tag. The `-sqldriver` option or `SetSQLDriver` method should be set to one of the below drivers (such as "Oracle"). Based on the driver below, determine what you should pass to `-sqldb` or `SetSQLDB` based on the description.

Oracle

Pass just the database or leave the option off and pass as part of the user such as "username@XE". Or, you may pass the host, SID and port as in "host=localhost;sid=XE;port=1521". Oracle version 8 through 11g should be able to connect.

mysql (Windows Only)

Specify the database name and the host IP address. For example, "database=mydb;host=localhost". If necessary, you may specify the port as well such as "database=mydb;host=localhost;port=3306". Both `mysql` and `mysqlPP` work against MySQL databases though `mysql` may give slightly better performance on Windows systems.

mysqlPP

Specify the database name and the host IP address. For example, "database=mydb;host=localhost". If necessary, you may specify the port as well such as "database=mydb;host=localhost;port=3306". Both `mysql` and `mysqlPP` work against MySQL databases though `mysql` may give slightly better performance on Windows systems.

ODBC (Windows Only)

Specify the DSN you wish to connect.

```
<QUERY
  NAME=text
  SQLDRIVER=text
  SQLDB=text
  USERID=text
  PASSWORD=text>
</QUERY>
```

Used to define a query. Place your SQL select statement between the opening and closing QUERY tags. Only the NAME is required. The connection options may be passed on the command line or via DLL methods. You may enter connection options on the query tag if you want to use different settings.

```
<QUERY NAME="getinfo" SQLDRIVER="Oracle"<BR>
  SQLDB="host=dev1.mysite.com;SID=dev1;port=1521"
  USERID="dev" PASSWORD="dev">
select id "key", name "Name", addr "Address"
  from customers
  where id = <QPARAM $custid>
</QUERY>
```

<u>Parameter</u>	<u>Description</u>
NAME=text	Provide a name for the query.
SQLDRIVER=text	The data source. This is a case-sensitive string. Entries with a * are only available for Windows operating systems. Valid values are: Oracle mysql* mysqlPP ODBC* mysql may give slightly better performance over mysqlPP on Windows systems. The Oracle client software will need to be installed when using Oracle.
SQLDB=text	The database schema or driver information. See the Database Connection section for details.
USERID=text	The user id (if any) for the database connection. For Oracle, you may also specify the password and/or schema in this field. For example, "user/pwd" or "user/pwd@prod".
PASSWORD=text	The password (if any) for the database connection.

<QPARAM variable>

Used to prevent SQL injection by passing parameters to the query during execution rather than placing directly in the SQL statement. The "variable" is the variable you want to use. For example:

```
<QUERY NAME="getCities">
select city, state, zip from cities
  where company = <QPARAM $comp>
</QUERY>
```

The \$comp variable would be passed using -sqlparams (or setSQLParams). Additionally, you can set variables using the environment variable RWQUERY_STRING. For example, set:
RWQUERY_STRING=\$comp=ABC Corp

on the command line then refer to the value as \$comp in the SQL. Be sure to use a \$ in front of the variable name in RWQUERY_STRING and in the code.

Fillable Fields

The `-fields` option or `setFieldsFile` method allows you to add fields in a tag based (or HTML-like) layout. The input file in this case is a text file containing the fields to add. Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (`INPUT` for most, `SELECT/OPTION` for choice) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<INPUT NAME="name1" TYPE="text"
  PAGES=1 X1=10 Y1=500 X2=200 Y2=420>
```

is a tag to create a text box.

These are the types of fields that can be added:

- Text or multiline text
- Checkbox
- Choice
- Radio
- Signature
- Button or submit

The default unit is a point (1/72 of an inch) for the position of the widgets (the `X1`, `X2`, `Y1` and `Y2` values). You may use the `UNITS` tag to specify a different value. The `UNITS` tag must come before the `INPUT` tag(s) that it is to be used for. The syntax is:

```
<UNITS VALUE="text">
```

Set the `VALUE` option to `pt` for points (the default), `in` for inches, `cm` for centimeters or `mm` for millimeters. In addition you may also set to an arbitrary value. The value you pass will be used as a multiplier for the number of points per unit. For example, passing a value of 72 will be the same as setting to `"in"` for inches. Setting to 28.35 is the same as `"cm"` ($72 / 2.54 = 28.35$). You have the flexibility to make up your own numbering scheme however.

Use the `-grid` or `setGrid` on the input PDF first to create a layout PDF with grid marks on the page. This will aid you in determining where to place the fields.

See the [data](#) section for information on populating fields.

See the [sample](#) section for a sample set of tags showing the various fields that can be added and some optional settings.

Fillable Fields

```
<INPUT
  TYPE="text"
  NAME="text"
  PAGES="text"
  X1=number
  X2=number
  Y1=number
  Y2=number
  ALIGN="text"
  FACE="text"
  SIZE=number
  RICHTEXT
  VALUE="text"
  READONLY
  TOOLTIP="text"
  HIDDEN
  NOPRINT
  NOVIEW
  BORDERCOLOR="text"
  BORDERSTYLE="text"
  BORDERDASH="number number"
  BORDERWIDTH=number
  BGCOLOR="text"
  FCOLOR="text">
```

Used to specify a single or multiline text field.

<u>Option</u>	<u>Description</u>
TYPE="text"	Required. The type of widget. Set to "Text" for a single line text box or "Multiline" for a multiline text box.
NAME="text"	Required. A unique identifier for the field. Can be a combination of letters and numbers. For example, "my name" or "address_line1". You may include fields multiple times with the same name (must be of the same TYPE) on the same or different pages. The value of all like-named fields will be updated whenever one of them changes.
PAGES="text"	The output page numbers to place the field on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.
X1=number	Required. The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the text box. The position 0,0 is the lower left corner of the page.

Fillable Fields

<u>Option</u>	<u>Description</u>
X2=number	Required. The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the text box.
Y1=number	Required. The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the text box.
Y2=number	Required. The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the text box.
ALIGN="text"	The text alignment. Set to C for Center, R for Right. Default is left alignment.
FACE="text"	The text face to use. Either "Helv" for Helvetica or "Times" for Times-Roman.
SIZE=number	The size of the font in points. Setting to 0 for multi-line text boxes will autosize the font to fit the text in the box.
RICHTEXT	Specifies the field should allow rich text.
VALUE="text"	The initial value of the text box. Use \n for a new line (2 characters - not an ASCII 10 line feed character) when using a multi-line text box.
READONLY	Makes the text box read only. Users may not interact with the widget.
TOOLTIP="text"	Text for popup information about the field.
HIDDEN	Field is not displayed but the value is sent when the submit button is pressed.
NOPRINT	Field is not printed.
NOVIEW	Field is not displayed but is printed if the NOPRINT option is not used.
BORDERCOLOR="text"	The border color .
BORDERSTYLE="text"	Type of border to draw. Options are: S - Solid (Default) D - Dashed B - Beveled I - Inset U - Underline

Fillable Fields

<u>Option</u>	<u>Description</u>
BORDERDASH= "number number"	Two numbers separated by a space. This is used when the BORDERSTYLE is set to D (Dashed). The first number is the number of points to draw on and the second is the number of points for the gap. Default is "3 3" when BORDERSTYLE is D and this option is not specified.
BORDERWIDTH=number	Width of the border in points (1 point = 1/72 of an inch). Default is 1.
BGCOLOR="text"	The background color .
FCOLOR="text"	The text color .

Fillable Fields

```
<INPUT  
  TYPE="text"  
  NAME="text"  
  PAGES="text"  
  X1=number  
  X2=number  
  Y1=number  
  Y2=number  
  SIZE=number  
  VALUE="text"  
  CHECKED  
  STYLE="text"  
  NOPRINT  
  READONLY  
  TOOLTIP="text"  
  BORDERCOLOR="text"  
  BORDERSTYLE="text"  
  BORDERDASH="number number"  
  BORDERWIDTH=number  
  BGCOLOR="text"  
  FCOLOR="text">
```

Used to specify a checkbox field.

<u>Option</u>	<u>Description</u>
TYPE="text"	Required. The type of widget. Set to "Checkbox".
NAME="text"	Required. A unique identifier for the field. Can be a combination of letters and numbers. For example, "my check" or "choice2". You may include fields multiple times with the same name (must be of the same TYPE) on the same or different pages. The value of all like-named fields will be updated whenever one of them changes.
PAGES="text"	The output page numbers to place the field on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.
X1=number	Required. The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the check box. The position 0,0 is the lower left corner of the page.
X2=number	Required. The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the check box.

Fillable Fields

<u>Option</u>	<u>Description</u>
Y1=number	Required. The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the check box.
Y2=number	Required. The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the check box.
SIZE=number	The size of the font in points.
VALUE="text"	The checked value of the check box. Default is "On".
CHECKED	Sets the initial state of the check box to on.
STYLE="text"	The character to use for the check mark. The default is ✓ if not specified and is the character 4. The character set used is the ZapfDingbats set so standard letters/numbers will show up as the corresponding symbol from that set.
NOPRINT	Field is not printed.
READONLY	Makes the check box read only. Users may not interact with the widget.
TOOLTIP="text"	Text for popup information about the field.
BORDERCOLOR="text"	The border color .
BORDERSTYLE="text"	Type of border to draw. Options are: S - Solid (Default) D - Dashed B - Beveled I - Inset U - Underline
BORDERDASH="number number"	Two numbers separated by a space. This is used when the BORDERSTYLE is set to D (Dashed). The first number is the number of points to draw on and the second is the number of points for the gap. Default is "3 3" when BORDERSTYLE is D and this option is not specified.
BORDERWIDTH=number	Width of the border in points (1 point = 1/72 of an inch). Default is 1.
BGCOLOR="text"	The background color .
FCOLOR="text"	The text color .

```
<SELECT  
  NAME="text"  
  PAGES="text"  
  X1=number  
  X2=number  
  Y1=number  
  Y2=number  
  ALIGN="text"  
  FACE="text"  
  SIZE=number  
  NOPRINT  
  READONLY  
  TOOLTIP="text"  
  BORDERCOLOR="text"  
  BORDERSTYLE="text"  
  BORDERDASH="number number"  
  BORDERWIDTH=number  
  BGCOLOR="text"  
  FCOLOR="text">  
</SELECT>
```

A choice widget is a combo box. The user can scroll through a drop-down list and select the appropriate item. The tag is SELECT rather than INPUT in this case. The individual options for the list have their own tag called OPTION. There must be a closing /SELECT tag after the options.

<u>Option</u>	<u>Description</u>
NAME="text"	Required. A unique identifier for the field. Can be a combination of letters and numbers. For example, "my list" or "list_1". You may include fields multiple times with the same name (must be of the same TYPE) on the same or different pages. The value of all like-named fields will be updated whenever one of them changes.
PAGES="text"	The output page numbers to place the field on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.
X1=number	Required. The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice. The position 0,0 is the lower left corner of the page.
X2=number	Required. The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice.

Fillable Fields

<u>Option</u>	<u>Description</u>
Y1=number	Required. The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice.
Y2=number	Required. The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice.
ALIGN="text"	The text alignment. Set to C for Center, R for Right. Default is left alignment.
FACE="text"	The text face to use. Either "Helv" for Helvetica or "Times" for Times-Roman.
SIZE=number	The size of the font in points. Setting to 0 will autosize the font to fit the text.
NOPRINT	Field is not printed.
READONLY	Makes the list read only. Users may not interact with the widget.
TOOLTIP="text"	Text for popup information about the field.
BORDERCOLOR="text"	The border color .
BORDERSTYLE="text"	Type of border to draw. Options are: S - Solid (Default) D - Dashed B - Beveled I - Inset U - Underline
BORDERDASH="number number"	Two numbers separated by a space. This is used when the BORDERSTYLE is set to D (Dashed). The first number is the number of points to draw on and the second is the number of points for the gap. Default is "3 3" when BORDERSTYLE is D and this option is not specified.
BORDERWIDTH=number	Width of the border in points (1 point = 1/72 of an inch). Default is 1.
BGCOLOR="text"	The background color .
FCOLOR="text"	The text color .

Fillable Fields

```
<OPTION  
  VALUE="text"  
  DESCR="text"  
  SELECTED>
```

The OPTION tag holds the values for the list. Each value or option is placed in its own tag.

<u>Option</u>	<u>Description</u>
VALUE="text"	The value associated with the description shown in the list.
DESCR="text"	Optional. The description to show in the list box. The VALUE is used if this is not specified.
SELECTED	Makes this entry the default.

```

<INPUT
  TYPE="text"
  NAME="text"
  PAGES="text"
  X1=number
  X2=number
  Y1=number
  Y2=number
  SIZE=number
  VALUE="text"
  CHECKED
  STYLE="text"
  NOPRINT
  READONLY
  TOOLTIP="text"
  BORDERCOLOR="text"
  BORDERSTYLE="text"
  BORDERDASH="number number"
  BORDERWIDTH=number
  BGCOLOR="text"
  FCOLOR="text">
    
```

A radio set allows only one of several choices to be selected. The NAME identifier is the same for all fields in a radio set while the VALUE is different for each choice. For example, if you want to allow 4 choices where at most 1 can be selected you would have three INPUT tags. Each input tag would have the same NAME (such as "region") and each would have a different VALUE (such as "north", "south", "east" and "west").

<u>Option</u>	<u>Description</u>
TYPE="text"	Required. The type of widget. Set to "Radio".
NAME="text"	Required. A unique identifier for the radio set. Can be a combination of letters and numbers. For example, "my check" or "choice2".
PAGES="text"	The output page numbers to place the field on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.
X1=number	Required. The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice. The position 0,0 is the lower left corner of the page.
X2=number	Required. The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice.

Fillable Fields

<u>Option</u>	<u>Description</u>
Y1=number	Required. The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice.
Y2=number	Required. The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the choice.
SIZE=number	The size of the font in points.
VALUE="text"	The value for the choice. Each field in the radio set must have a different value.
CHECKED	Sets the choice as initially on. Only one choice in the set should be marked as CHECKED.
STYLE="text"	The character to use for the selected choice. The default is ● if not specified and is the character l (lowercase L). The character set used is the ZapfDingbats set so standard letters/numbers will show up as the corresponding symbol from that set.
NOPRINT	Field is not printed.
READONLY	Makes the radio set read only. Users may not interact with the widget.
TOOLTIP="text"	Text for popup information about the field.
BORDERCOLOR="text"	The border color .
BORDERSTYLE="text"	Type of border to draw. Options are: S - Solid (Default) D - Dashed B - Beveled I - Inset U - Underline
BORDERDASH="number number"	Two numbers separated by a space. This is used when the BORDERSTYLE is set to D (Dashed). The first number is the number of points to draw on and the second is the number of points for the gap. Default is "3 3" when BORDERSTYLE is D and this option is not specified.
BORDERWIDTH=number	Width of the border in points (1 point = 1/72 of an inch). Default is 1.
BGCOLOR="text"	The background color .
FCOLOR="text"	The text color .

Fillable Fields

```
<INPUT
  TYPE="text"
  NAME="text"
  PAGES="text"
  X1=number
  X2=number
  Y1=number
  Y2=number
  TOOLTIP="text"
  NOPRINT
  SIGNSSL="text"
  SIGNPKFILE="text"
  SIGNPEMFILE="text"
  SIGNTIMESTAMP="text"
  SIGNTS="text"
  SIGNDATE="text"
  SIGNREASON="text"
  SIGNIMG="text"
  SIGNKEEPRATIO
  SIGNIMGALIGN="text"
  SIGNSIZE=number
  SIGNSCRIPT
  SIGNSRC="text"
  SIGNCOLOR="text"
  SIGNBGCOLOR="text"
  BORDERCOLOR="text"
  BORDERSTYLE="text"
  BORDERDASH="number number"
  BORDERWIDTH=number
  BGCOLOR="text"
  FCOLOR="text">
```

A signature field can be used by Adobe Acrobat to digitally sign a PDF.

<u>Option</u>	<u>Description</u>
TYPE="text"	Required. The type of widget. Set to "Signature".
NAME="text"	Required. A unique identifier for the field. Can be a combination of letters and numbers. For example, "sign box" or "signature1".
PAGES="text"	The output page numbers to place the field on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.

Fillable Fields

<u>Option</u>	<u>Description</u>
X1=number	Required. The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature box. The position 0,0 is the lower left corner of the page.
X2=number	Required. The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature box.
Y1=number	Required. The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature box.
Y2=number	Required. The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature box.
TOOLTIP="text"	Text for popup information about the field.
NOPRINT	Field is not printed.
SIGNSSL="text"	Used when signing a signature field. The path and file name of the OpenSSL program. For example, "c:\openssl\bin\openssl.exe". See the Digital Signature section for details.
SIGNPKFILE="text"	Used when signing a signature field. The path and name of the private key file. For example, "c:\keys\mykey_pk.pem". See the Digital Signature section for details.
SIGNPEMFILE="text"	Used when signing a signature field. For example, "c:\keys\mykey.pem". The path and name of the signing certificate. See the Digital Signature section for details.
SIGNTIMESTAMP="text"	The path and name of your timestamp server if you want to include an embedded timestamp on the signature. Use "1" to use the default of https://freetsa.org/tsr . Pass your user name and password if your timestamp server requires it like this: "myuserid:mypassword@https://mytimestamp.org". Use the SIGNTS option with SIGNTIMESTAMP.
SIGNTS="text"	The path and name of the ts.exe (or just ts for Linux) executable that is used to perform the timestamping function. This is a separate download available here . Use this program for both Windows and Linux.

<u>Option</u>	<u>Description</u>
SIGNDATE="text"	<p>Use this option to force a particular date and time for the signing. Pass a comma separated string in the form "yyyy,mm,dd,hr,mi,ss,+ or -,offset hr,offset mi".</p> <p>yyyy = year mm = month ss = seconds hr = hour (optional) mi = minutes (optional) ss = seconds (optional) + or - = the time zone offset from UTC (optional) offset hr = the number of hours offset from UTC (optional) offset mi = the number of minutes offset from UTC (optional)</p> <p>Only the year, month and day are required. Any values not provided will be taken from the local computer settings. Be careful not to set a date before the start date of the signing certificate or in the future as the signature will not show as valid in Acrobat or Reader.</p>
SIGNREASON="text"	<p>Used when signing a signature field. Optional. The reason for signing the document. Default is "Attestation to the accuracy and integrity of this document". See the Digital Signature section for details.</p>
SIGNIMG="text"	<p>Used when signing a signature field. Optional. The path and name of an image to use for the signature. Set this option to the value "none" to not place any image in the signature field. See the Digital Signature section for details.</p>
SIGNKEEPRATIO	<p>Used when signing a signature field. Optional. Keep the image x/y scaling ratio when using an image with a signature field. See the Digital Signature section for details.</p>
SIGNIMGALIGN="text"	<p>Optional. The alignment for the image in the signature field when using SIGNKEEPRATIO. Set to "Left", "Center", or "Right". Default is "Left". See the Digital Signature section for details.</p>
SIGNSIZE=number	<p>Used when signing a signature field. Optional. The point size for the font in the text of the signature (or 0 for no text). Default is 12. See the Digital Signature section for details.</p>

<u>Option</u>	<u>Description</u>
SIGNSCRIPT	Used when signing a signature field. Optional. Use a script font that looks like this - <i>Jane Doe</i> - and place just the name from the certificate into the signature field. See the Digital Signature section for details.
SIGNSRC="text"	Used when signing a signature field. Optional. A TrueType font file to use for the signature text. See the Digital Signature section for details.
SIGNCOLOR="text"	Used when signing a signature field. Optional. The text color for the signature. See the Digital Signature section for details.
SIGNBGCOLOR="text"	Used when signing a signature field. Optional. The background fill color for the signature. See the Digital Signature section for details.
BORDERCOLOR="text"	The border color .
BORDERSTYLE="text"	Type of border to draw. Options are: S - Solid (Default) D - Dashed B - Beveled I - Inset U - Underline
BORDERDASH="number number"	Two numbers separated by a space. This is used when the BORDERSTYLE is set to D (Dashed). The first number is the number of points to draw on and the second is the number of points for the gap. Default is "3 3" when BORDERSTYLE is D and this option is not specified.
BORDERWIDTH=number	Width of the border in points (1 point = 1/72 of an inch). Default is 1. Set to "0" if you are signing a signature field and do not want the default border around the signature.
BGCOLOR="text"	The background color .
FCOLOR="text"	The text color .

```
<SIGN
  NAME="text"
  X1=number
  X2=number
  Y1=number
  Y2=number
  SIGNSSL="text"
  SIGNTIMESTAMP="text"
  SIGNTS="text"
  SIGNDATE="text"
  SIGNPKFILE="text"
  SIGNPEMFILE="text"
  SIGNREASON="text"
  SIGNIMG="text"
  SIGNKEEPRATIO
  SIGNIMGALIGN="text"
  SIGNSIZE=number
  SIGNSCRIPT
  SIGNSRC="text"
  SIGNCOLOR="text"
  SIGNBGCOLOR="text"
  BORDERWIDTH=number>
```

This tag is used on a PDF that already contains a signature field to be signed or to add and sign a new signature field. If the NAME is not passed then PDF Meld will look for the first open signature field to sign.

<u>Option</u>	<u>Description</u>
NAME="text"	The name of the field in the PDF to sign. This is optional and if not set then PDF Meld will look for the first open signature field to sign. You may also use ":new" or ":hide" to add and sign a new signature field when one doesn't exist in the PDF. This is similar to using an INPUT signature field except you don't need to pass the X/Y coordinates in this case. The ":hide" option does not place a visible signature on the page; it only shows in the signature pane. Only use ":new" or ":hide" on a PDF that doesn't already contain a signature field.
X1=number	Optional and only used when NAME begins ":new". The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature field. The position 0,0 is the lower left corner of the page.

Fillable Fields

<u>Option</u>	<u>Description</u>
X2=number	Optional and only used when NAME begins ":new". The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature field.
Y1=number	Optional and only used when NAME begins ":new". The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature field.
Y2=number	Optional and only used when NAME begins ":new". The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the signature field.
SIGNSSL="text"	The path and file name of the OpenSSL program. For example, "c:\openssl\bin\openssl.exe". See the Digital Signature section for details.
SIGNTIMESTAMP="text"	The path and name of your timestamp server if you want to include an embedded timestamp on the signature. Use "1" to use the default of https://freetsa.org/tsr . Pass your user name and password if your timestamp server requires it like this: "myuserid:mypassword@https://mytimestamp.org". Use the SIGNTS option with SIGNTIMESTAMP.
SIGNTS="text"	The path and name of the ts.exe (or just ts for Linux) executable that is used to perform the timestamping function. This is a separate download available here . Use this program for both Windows and Linux.

<u>Option</u>	<u>Description</u>
SIGNDATE="text"	<p>Use this option to force a particular date and time for the signing. Pass a comma separated string in the form "yyyy,mm,dd,hr,mi,ss,+ or -,offset hr,offset mi".</p> <p>yyyy = year mm = month ss = seconds hr = hour (optional) mi = minutes (optional) ss = seconds (optional) + or - = the time zone offset from UTC (optional) offset hr = the number of hours offset from UTC (optional) offset mi = the number of minutes offset from UTC (optional)</p> <p>Only the year, month and day are required. Any values not provided will be taken from the local computer settings. Be careful not to set a date before the start date of the signing certificate or in the future as the signature will not show as valid in Acrobat or Reader.</p>
SIGNPKFILE="text"	<p>The path and name of the private key file in PEM format. For example, "c:\keys\mykey_pk.pem".</p> <p>See the Digital Signature section for details.</p>
SIGNPEMFILE="text"	<p>The path and name of the signing certificate in PEM format. For example, "c:\keys\mykey.pem".</p> <p>See the Digital Signature section for details.</p>
SIGNREASON="text"	<p>Optional. The reason for signing the document. Default is "Attestation to the accuracy and integrity of this document". See the Digital Signature section for details.</p>
SIGNIMG="text"	<p>Optional. The path and name of an image to use for the signature. Set this option to the value "none" to not place any image in the signature field. See the Digital Signature section for details.</p>
SIGNKEEPRATIO	<p>Used when signing a signature field. Optional. Keep the image x/y scaling ratio when using an image with a signature field. See the Digital Signature section for details.</p>

Fillable Fields

<u>Option</u>	<u>Description</u>
SIGNIMGALIGN="text"	Optional. The alignment for the image in the signature field when using SIGNKEEPRATIO. Set to "Left", "Center", or "Right". Default is "Left". See the Digital Signature section for details.
SIGNSIZE=number	Used when signing a signature field. Optional. The point size for the font in the text of the signature (or 0 for no text). Default is 12. See the Digital Signature section for details.
SIGNSCRIPT	Used when signing a signature field. Optional. Use a script font that looks like this - <i>Jane Doe</i> - and place just the name from the certificate into the signature field. See the Digital Signature section for details.
SIGNSRC="text"	Used when signing a signature field. Optional. A TrueType font file to use for the signature text. See the Digital Signature section for details.
SIGNCOLOR="text"	Used when signing a signature field. Optional. The text color for the signature. See the Digital Signature section for details.
SIGNBGCOLOR="text"	Used when signing a signature field. Optional. The background fill color for the signature. See the Digital Signature section for details.
BORDERWIDTH=number	Optional. Set to "0" to not draw an a border around the signature field. Any other value or the absence of this option results in a thin outline around the field. See the Digital Signature section for details.

```
<INPUT  
  TYPE="text"  
  NAME="text"  
  PAGES="text"  
  X1=number  
  X2=number  
  Y1=number  
  Y2=number  
  SIZE=number  
  VALUE="text"  
  JS="text"  
  NOPRINT>
```

A button allows you to execute a javascript function in the PDF. Visit the Adobe website for the current javascript reference manual.

<u>Option</u>	<u>Description</u>
TYPE="text"	Required. Set to "button".
NAME="text"	Required. A unique identifier for the field. Can be a combination of letters and numbers. For example, "button1" or "print_button". You may include fields multiple times with the same name (must be of the same TYPE) on the same or different pages.
PAGES="text"	The output page numbers to place the button on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.
X1=number	Required. The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button. The position 0,0 is the lower left corner of the page.
X2=number	Required. The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button.
Y1=number	Required. The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button.
Y2=number	Required. The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button.
SIZE=number	The size of the font in points.
VALUE="text"	The label text for the button.

Fillable Fields

<u>Option</u>	<u>Description</u>
JS="text"	The javascript to execute. For example, set to "this.print()" to bring up the print dialog when clicked. You might use this to put a print button on each page in order to make it more noticeable than the print button in the toolbar. Or you can insert a formula based on other fields you've added to the PDF to perform a mathematical function.
NOPRINT	Do not print the button.

Fillable Fields

```
<INPUT
  TYPE="text"
  NAME="text"
  PAGES="text"
  X1=number
  X2=number
  Y1=number
  Y2=number
  SIZE=number
  VALUE="text"
  URL="text"
  NOPRINT>
```

A submit button allows the user to submit the form data to a web site for processing. The user must either have the PDF opened in a browser window or be using the full version of Acrobat for the button to function.

<u>Option</u>	<u>Description</u>
TYPE="text"	Required. Set to "Submit" for a submit button.
NAME="text"	Required. A unique identifier for the field. Can be a combination of letters and numbers. For example, "submit" or "submit_button". You may include fields multiple times with the same name (must be of the same TYPE) on the same or different pages.
PAGES="text"	The output page numbers to place the button on. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.
X1=number	Required. The left X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button. The position 0,0 is the lower left corner of the page.
X2=number	Required. The right X coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button.
Y1=number	Required. The top Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button.
Y2=number	Required. The lower Y coordinate (in points where 1 point = 1/72 of an inch or based on current UNITS setting) of the button.
SIZE=number	The size of the font in points.
VALUE="text"	The label text for the button.
URL="text"	The URL to submit to. Should be a page that is expecting the form fields and will process them. For example, URL="http://www.mysite.com/showresults.cgi".

Fillable Fields

<u>Option</u>	<u>Description</u>
NOPRINT	Do not print the button.

Fillable Fields

<UNITS
VALUE=text>

Used to set the unit of measurement for tags that follow. Typically you'll place one UNITS tag at the top but you may use them through the file with different values. The UNITS tag affects all tags that follow it until another UNITS tag is found.

For example, to set centimeters use <UNITS VALUE="cm">. The default is "pt" for points.

<u>Parameter</u>	<u>Description</u>
VALUE=text	Use one of the following: <u>PT</u> Points (1/72 of an inch) <u>IN</u> Inches <u>CM</u> Centimeters <u>MM</u> Millimeters

Sample

Here's an example file showing each of the fields:

```
<INPUT TYPE="radio" NAME="radioset"  
  PAGES=1 BORDERSTYLE=Inset BORDERWIDTH=1  
  X1=72 Y1=700 X2=92 Y2=720  
  VALUE=1 BORDERCOLOR=#999999 SIZE=12>  
  
<INPUT TYPE="radio" NAME="radioset"  
  PAGES=1 BORDERSTYLE=Inset BORDERWIDTH=1 CHECKED  
  X1=102 Y1=700 X2=122 Y2=720  
  VALUE=2 BORDERCOLOR=#999999 SIZE=12>  
  
<INPUT TYPE="radio" NAME="radioset"  
  PAGES=1 BORDERSTYLE=Inset BORDERWIDTH=1  
  X1=132 Y1=700 X2=152 Y2=720  
  VALUE=3 BORDERCOLOR=#999999 SIZE=12>  
  
<INPUT TYPE="text" NAME="first_name"  
  PAGES=1 BORDERSTYLE=U  
  X1=72 Y1=600 X2=216 Y2=584  
  VALUE="Jane" BORDERCOLOR=black SIZE=12>  
  
<INPUT TYPE="text" NAME="last_name"  
  PAGES=1 BORDERSTYLE=U TOOLTIP="Type in your last name"  
  X1=72 Y1=580 X2=216 Y2=564  
  VALUE="Doe" BORDERCOLOR=black SIZE=12>  
  
<INPUT TYPE="checkbox" NAME="married"  
  PAGES=1 FCOLOR=#FFFFFF BGCOLOR=#3333CC  
  X1=220 Y1=580 X2=234 Y2=564  
  VALUE="On" BORDERCOLOR=black>  
  
<INPUT TYPE="multiline" NAME="comments"  
  PAGES=1 BORDERSTYLE=I BORDERWIDTH=3  
  BORDERCOLOR=#999999  
  X1=72 Y1=560 X2=288 Y2=460 FCOLOR=#009900  
  VALUE="Enter\nComments\nHere" SIZE=0>  
  
<INPUT TYPE="signature" NAME="sig"  
  PAGES=1  
  X1=72 Y1=450 X2=172 Y2=380  
  BORDERCOLOR=black>  
  
<SELECT NAME="state"  
  PAGES=1 FCOLOR=#FFFFFF BGCOLOR=#33CC33  
  X1=72 Y1=300 X2=216 Y2=286  
  BORDERCOLOR=black>
```

Fillable Fields

```
<OPTION VALUE="AK" DESCR="Alaska">  
<OPTION VALUE="MI" DESCR="Michigan" SELECTED>  
<OPTION VALUE="OH" DESCR="Ohio">  
<OPTION VALUE="WY" DESCR="Wyoming">  
</SELECT>
```

Add the fields (assume the field file is called fields.dat) and create the PDF
by running:
pdfmeld.exe myfile.pdf output.pdf -fields fields.dat

Printer Configuration Files

The `-printcfgin` option or `setPrintCfgIn` method allows you to store printer configuration information in your PDF. You first create a file that holds the printer and, optionally, the device and port to print to along with a page range. For example, you might want pages 1 to 3 to print from tray 1 with blue paper and pages 4 and after to print from tray 2 with white paper. You then store the file within the PDF using `-printcfgin` or `setPrintCfgIn`.

You must setup each printer and tray you want to print to as a separate printer in Windows. During printing, the PDF will be split and sent to the requested printer. This means you cannot simply take the PDF from one system to another unless the printers are named the same. You can, however, export the printer configuration information and modify it. Also, you may pass in a different printer configuration file to use instead.

You use the `-printcomp` option or `setPrintComp` method once the printer information has been added. In addition, you can pass the name of a local printer configuration file to use with `-printcomp` or `setPrintComp`. This overrides the settings from the PDF.

The printer configuration file is a tag based file. Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `PRINT` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:
`<PRINT PRINTER="Accounting Printer" PAGES="1,2">`
`<PRINT PRINTER="Shipping Printer" PAGES="3--1">`
Will print pages 1 and 2 to a printer called "Accounting Printer" and pages 3 on will print on the "Shipping Printer".

While you can embed printer configuration information with PDF Meld for any platform, the PDFs may only be printed this way from Windows. This method may not work on all Windows systems depending on your configuration so be sure to test it before adding to your PDFs.

Printer Configuration Files

```
<PRINT
  PRINTER="text"
  DEVICE="text"
  PORT="text"
  COPIES=number
  PAGES="text">
```

Used to define a printer configuration.

<u>Option</u>	<u>Description</u>
PRINTER="text"	The printer to use, for example "\\server\printer". Or, use the word "default" to send to the default printer.
DEVICE="text"	Optional. The device to print to, for example "HP LaserJet".
PORT="text"	Optional. The port to use, for example "lpt1:".
COPIES=number	Optional. The number of copies to print. Default is 1.
PAGES="text"	The list of pages to print. Enter a comma separated list and/or use a - (minus sign) for a page range. Leave this option out to include on all pages.

Layer Files

The `-layersin/-layersout` or `setLayerFileIn/setLayerFileOut` methods allow you to remove layers in a PDF. The base file is created by using the `-layersout` option or `setLayerFileOut` method. A set of tags will be created in the output file based on the various layers found in the PDF. You may then add a `REMOVE` option to remove a layer from the output PDF.

Note that if the layers are not specified in the input PDF in such a way to encapsulate the proper settings you may wind up with a PDF that has errors. This is because the markup for the layer is removed entirely from the output PDF as if it was never there. The result is a "flattened" (as far as the layers are concerned) PDF with the remaining layers merged.

Each tag, or command, starts with an angle bracket `<` and closes with `>`. The tag name (always `LAYER` in this case) comes directly after the opening `<`. Options are then listed, space separated, with an `=` sign between it and its value. For example:

```
<LAYER DESC="Layer 1" NAME="M1">  
<LAYER DESC="Layer 2" NAME="M2" REMOVE>  
will remove the layer called "Layer 2" from the PDF.
```

Layer Files

```
<LAYER  
  DESCR="text"  
  NAME="text"  
  REMOVE>
```

Use this tag to remove layers from a PDF.

<u>Option</u>	<u>Description</u>
DESCR="text"	The description for the layer. This is what the user sees as the name of the layer when viewing the PDF and selecting the "layers" pane.
NAME="text"	The internal name of the layer used in the PDF. This is the name setting the program will search for when removing layers so do not change the name.
REMOVE	Add this option to remove the layer from the PDF.

Digital Signatures

Reasons for Using

Digital signatures provide a way to sign a PDF electronically in order to authenticate its contents. The person receiving the signed PDF can be assured the version they are viewing is authentic and has not been tampered with. Any changes to the document after the signature is applied will be noted in the signature pane in Reader or Acrobat so you know what, if anything, was modified. Another use is to self-sign a PDF contract or quote you send to a prospect or client. If they accept and send back the original contract you can verify if any modifications have been made to the version you signed by clicking the signature field to make sure nothing was altered.

You may create a PDF with more than one signature field to sign. In this situation you may have a contract that both you and your client will sign. This means you should have your final copy together before applying the first signature. Note that PDF Meld allows form field value changes to be made when signing the PDF as the only allowable change after the first signature is placed. Use the `-fdfin` option or `setFDFFileIn` method to load in any form field changes. You should strip out any fields whose value is not changing so those fields do not show as modified in the signature pane. This allows you to modify a field's value at the time you sign even if the PDF has already been signed by someone else. However, your changes will be highlighted in the signature pane of Reader so the first person signing the PDF is able to see what alterations were made between signatures. Reader will also allow you to view the PDF with the values it had at the time each signature was applied. You may also use the `-flattensig` option or `setFieldsFlatten("sig")` method to flatten form fields when signing.

You will need to pass in the user password using the `-u` option or `setUser` method if you are signing an encrypted PDF containing a user password. The PDF has a user password if you are prompted to enter a password before Reader or Acrobat opens the PDF. Note that the owner or user password can open the PDF, however it is the user password that will need to be passed into PDF Meld in order to sign it. PDFs containing only an owner password do not need to pass anything special in order to sign.

Requirements for Signing

PDF Meld uses an open source program called OpenSSL that is available for most Unix and Windows installations. If you do not have OpenSSL installed, you'll need to install it first before you can digitally sign documents. Most Unix systems will likely have it - if you're not sure, try

typing OpenSSL at a shell prompt and if it comes back with a prompt that looks like OpenSSL> then it is installed. Windows binaries are available here: <https://www.slproweb.com/products/Win32OpenSSL.html> or, if the link is not available, search for "openssl windows binary" in your favorite search engine.

The following section deals with using OpenSSL to create your signature files. While you don't need to be an expert at digital certificates, you should be comfortable running commands from the DOS prompt. This is a process you will probably only run once to set your certificates. Once you have them you simply supply them to the PDF Meld for signing so this section is not something you will need to do each time you want to sign a PDF.

First you'll need a certificate to sign PDFs with. You may purchase them from security companies on-line or use OpenSSL to create your own. There are 2 files you'll need to sign with and there is a step-by-step process below on how to create them.

mykey_pk.pem - your private signing key
mykey.pem - your public certificate

There are two types of file formats for certificate files. One is PEM which is a text file and DER which is binary. The names of your files may be different but the point is you'll need a private key in pem format and a certificate in binary form.

Covering all the commands of OpenSSL is beyond the scope of this document. We'll just be covering the basics to get a certificate setup. There are many websites you may visit to explain other uses and options for OpenSSL if you are interested. Installing OpenSSL should only take a few minutes depending on your internet connection.

Setting up OpenSSL

The first step is to create a configuration file for OpenSSL if you don't have one already. You only need to do this once and you may place it in the directory you installed OpenSSL into. Here's a sample openssl.cnf file to get you started if you need one. This file is also embedded in this PDF so you can download it from this document rather than cut & paste.

```
#
# SSlEay example configuration file.
# This is mostly being used for generation of certificate requests.
#
RANDFILE                = .rnd
#####
[ ca ]
default_ca              = CA_default          # The default ca section
#####
[ CA_default ]
dir                    = demoCA              # Where everything is kept
```

```
certs           = $dir\certs           # Where the issued certs are kept
crl_dir         = $dir\crl             # Where the issued crl are kept
database        = $dir\index.txt       # database index file.
new_certs_dir   = $dir\newcerts        # default place for new certs.
certificate      = $dir\cacert.pem     # The CA certificate
serial          = $dir\serial          # The current serial number
crl             = $dir\crl.pem         # The current CRL
private_key     = $dir\private\cakey.pem # The private key
RANDFILE        = $dir\private\private.rnd # private random number file
x509_extensions = x509v3_extensions    # The extentions to add to the cert
default_days    = 365                  # how long to certify for
default_crl_days = 30                  # how long before next CRL
default_md      = md5                  # which md to use.
preserve        = no                   # keep passed DN ordering
# A few difference way of specifying how similar the request should look
# For type CA, the listed attributes must be the same, and the optional
# and supplied fields are just that :-)
policy          = policy_match
# For the CA policy
[ policy_match ]
countryName     = optional
stateOrProvinceName = optional
organizationName = optional
organizationalUnitName = optional
commonName      = supplied
emailAddress    = optional
# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName     = optional
stateOrProvinceName = optional
localityName    = optional
organizationName = optional
organizationalUnitName = optional
commonName      = supplied
emailAddress    = optional
#####
[ req ]
default_bits    = 1024
default_keyfile = privkey.pem
distinguished_name = req_distinguished_name
attributes      = req_attributes
[ req_distinguished_name ]
countryName     = Country Name (2 letter code)
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
localityName    = Locality Name (eg, city)
0.organizationName = Organization Name (eg, company)
organizationalUnitName = Organizational Unit Name (eg, section)
commonName      = Common Name (eg, your website's domain name)
commonName_max  = 64
emailAddress    = Email Address
emailAddress_max = 40
[ req_attributes ]
```

```
challengePassword          = A challenge password
challengePassword_min      = 4
challengePassword_max      = 20
[ x509v3_extensions ]
# under ASN.1, the 0 bit would be encoded as 80
nsCertType                 = 0x40
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName
#nsCertSequence
#nsCertExt
#nsDataType
```

The end user will not need to do anything special to use certificates you create but they will not be trusted certificates. They have the option to trust your certificate, if they wish, but they do not have to. In either case, Reader will report whether or not the document has been modified since it was signed.

Creating Self-Signed Certificates

Now that OpenSSL is setup, here are the steps to create a self-signed certificate. Note that there are a variety of security companies that sell self-signed certificates. However, we'll use OpenSSL here to show you how to create your own in just a few short steps.

1. Open a DOS window or a shell in Linux/Unix.
2. Be sure your PATH environment variable contains the executable for OpenSSL. This will be the directory you installed it into. If not set, you can type this at the DOS prompt:
`path=%path%;c:\(openssl-directory)`
Where the "(openssl-directory)" is replaced with the directory containing the binary openssl.exe program. This should be the directory you installed the program into along with the path of \bin at the end of that.
3. Create the public and private key files by running the following:
`openssl req -x509 -new -config openssl.cnf -days 365 -out mykey.pem -keyout mykey_enc.pem -newkey rsa:2048`
The file mykey_end.pem is the encrypted private key. You may set the number of days for expiration to whatever you want. In the example, we've used 1 year but you may set for whatever you like. This is just the expiration for the certificate. Be sure to put the full path to openssl.cnf in the line above if it is not in your current directory. The -newkey rsa:2048 (or rsa:4096) is optional if you want to create larger encryption keys than the 1024 default size. Now you should have mykey_enc.pem and mykey.pem on your system.

Digital Signatures

4. Convert the encrypted private key to an RSA private key file:
`openssl rsa -in mykey_enc.pem -out mykey_pk.pem`
The file `mykey_pk.pem` is the private key you'll use for the `SIGNPKFILE` option. Be sure to put the full path to `openssl.cnf` in the line above if it is not in your current directory.

Passing Signature Information to PDF Meld

Now that you have the files, use them on the command line or in a tagged input file. On the command line, `-signpkfile` would be set to `mykey_pk.pem` and `-signpemfile` would be `mykey.pem` using the example above. You do not have to pass the signing password to the program.

Here is a sample input field that can be loaded with the `-fields` option or `setFieldsFile` method. This will add a signature field to a PDF and sign the field at the same time:

```
<INPUT TYPE="signature" NAME="sig"
  PAGES=1
  X1=72 Y1=450 X2=272 Y2=380
  SIGNSSL="c:\openssl\bin\openssl.exe"
  SIGNPKFILE="c:\openssl\mykey_pk.pem"
  SIGNPEMFILE="c:\openssl\mykey.pem"
  SIGNREASON="Approving this document"
  SIGNIMG="c:\images\sigimage.jpg">
```

Here is a sample using the `SIGN` option instead. This signs an existing signature field in a PDF. If there is only one signature field available you may leave off the `NAME` option:

```
<SIGN NAME="sig"
  SIGNSSL="c:\openssl\bin\openssl.exe"
  SIGNPKFILE="c:\openssl\mykey_pk.pem"
  SIGNPEMFILE="c:\openssl\mykey.pem"
  SIGNREASON="Approving this document"
  SIGNIMG="c:\images\sigimage.jpg">
```

Or you may use the `-signssl`, `-signpkfile`, `-signpemfile`, etc. PDF Meld options or `setSign` method to sign an existing signature without creating a separate file with the `SIGN` tag.

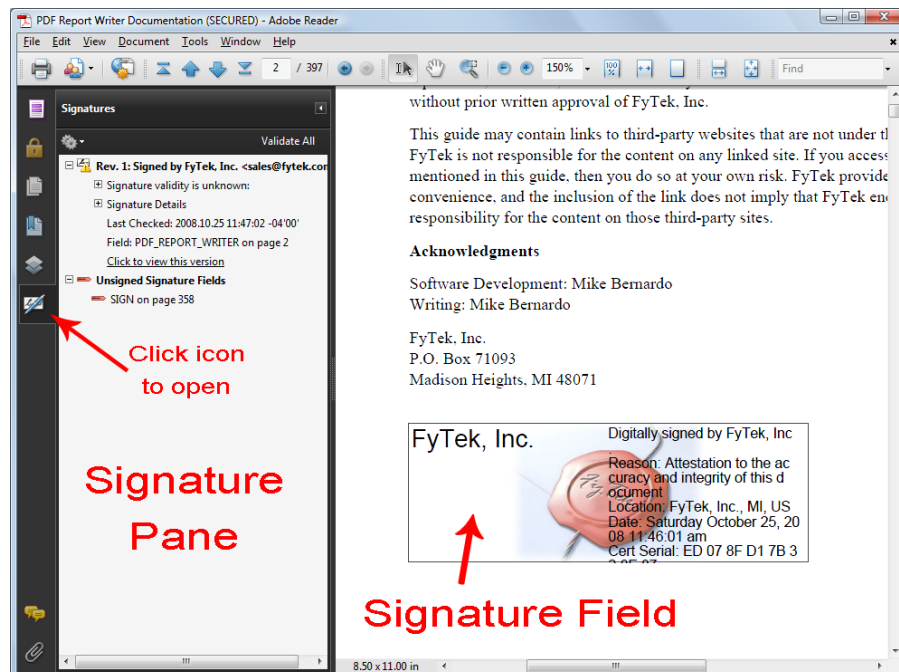
Another way to pass this information in is to use an FDF file. In this case, pass the values on a line in the file similar to this:

```
<</T (sign_field) /V ()  
  /TU (DIGITAL SIGNATURE)  
  /FT /Sig  
  /SIGNSSL (c:\\openssl\\bin\\openssl.exe)  
  /SIGNPKFILE (c:\\openssl\\mykey_pk.pem)  
  /SIGNPEMFILE (c:\\openssl\\mykey.pem)  
  /SIGNREASON (Approval) >>
```

This method allows you to set values for other fields in the document at the time of signing. Note the double backslashes \\ above are necessary for a valid FDF file or you may convert the values to hex and use angle brackets for the values instead - <78797A313233> is the same as (xyz123). You may use all of the SIGNSSL, etc. options available in the INPUT or SIGN tag. Use the -fdfout option or setFDFFileOut method to first export the fields in the PDF if necessary. Modify the signature as shown above - only one signature may be applied at a time so you'll need to run the process multiple times for multiple signings. Then, once the FDF file has been updated, use the -fdfin option or setFDFFileIn method to load in the modified field values along with the signature. The options for SIGNSSL, etc. must be entered in uppercase when using them in an FDF file.

Trusting Certificates

You will see something similar to the following after you sign a document for the first time.



Note the icon with the yellow warning icon in the signature pane. This is because the certificate has not yet been trusted by Reader. Once you have trusted the certificate the icon will change and all future signatures in PDFs with this certificate will be recognized. There are several ways to do this.

The first method is to export an FDF file using PDF Meld at the time you sign a PDF. You only need to create the FDF file once for any given certificate so you don't need to do this everytime. Use the `-certfile` option or `setCertFile` method to create the FDF file. This file contains just your certificate that users can open in Acrobat or Reader and allow it use the certificate as a trusted root. You can email this file or make it available on a website. There is no private information stored in it.

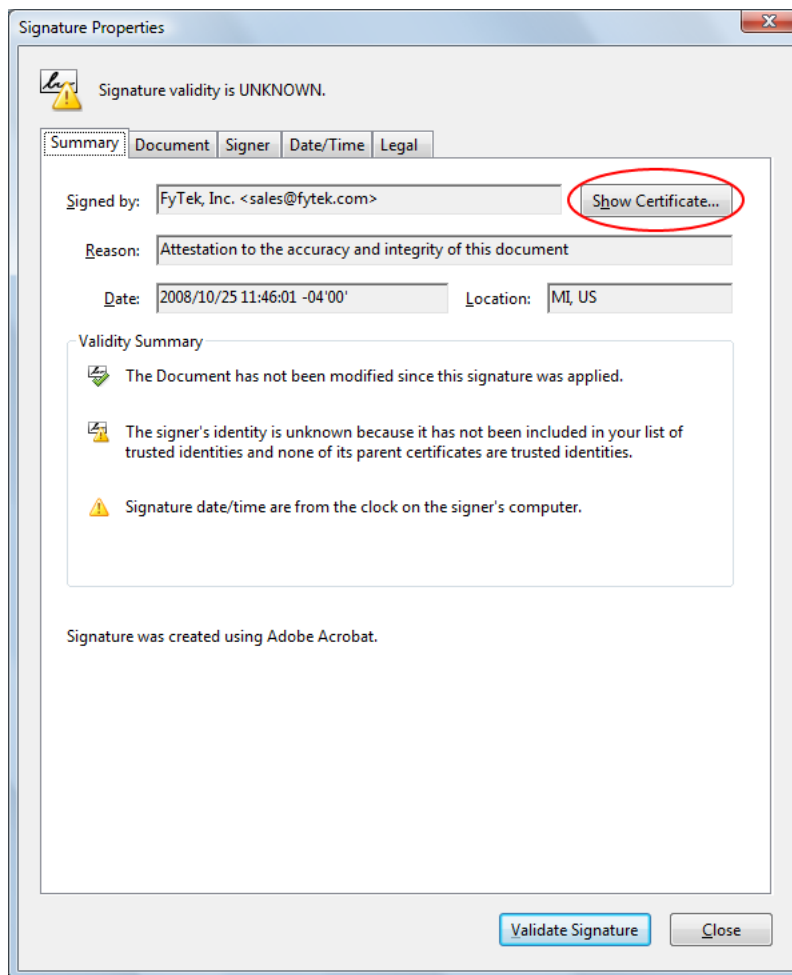
The second method is to export the certificate from the output PDF as an FDF file and send it to the intended user. You are sending just a certificate for the user to load into Acrobat or Reader in this case, not a PDF. Also, you only need to export a given certificate one time so once you create the file you can supply the same one to end users as often as you like and it will be good not just for this PDF but for any that you sign with this certificate. To do this, open the PDF with the newly added signature. Click on the signature to open a dialog box then click the "Signature Properties..." button to continue. Next, click the "Show Certificate..." button to continue. Screen shots of these dialog windows are shown in the next section. From the certificate screen you will see a button labeled "Export...". Click the "Export..." button to step through the process of exporting the certificate.

By default you should have Acrobat FDF Data Exchange as the format for the certificate, which is what you want. Click the "Next" button and supply the requested data when prompted to create the file. Once you are finished you should have a file with an extension of `.fdf` you can supply to end users containing your certificate. They open this file using Acrobat or Reader and, rather than opening a PDF, are presented with a dialog box where they can step through the process of trusting the certificate. In this way you can email, place on a website, or otherwise provide the certificate in advance rather than have the user trust the certificate directly from the PDF which involves a warning dialog when attempted.

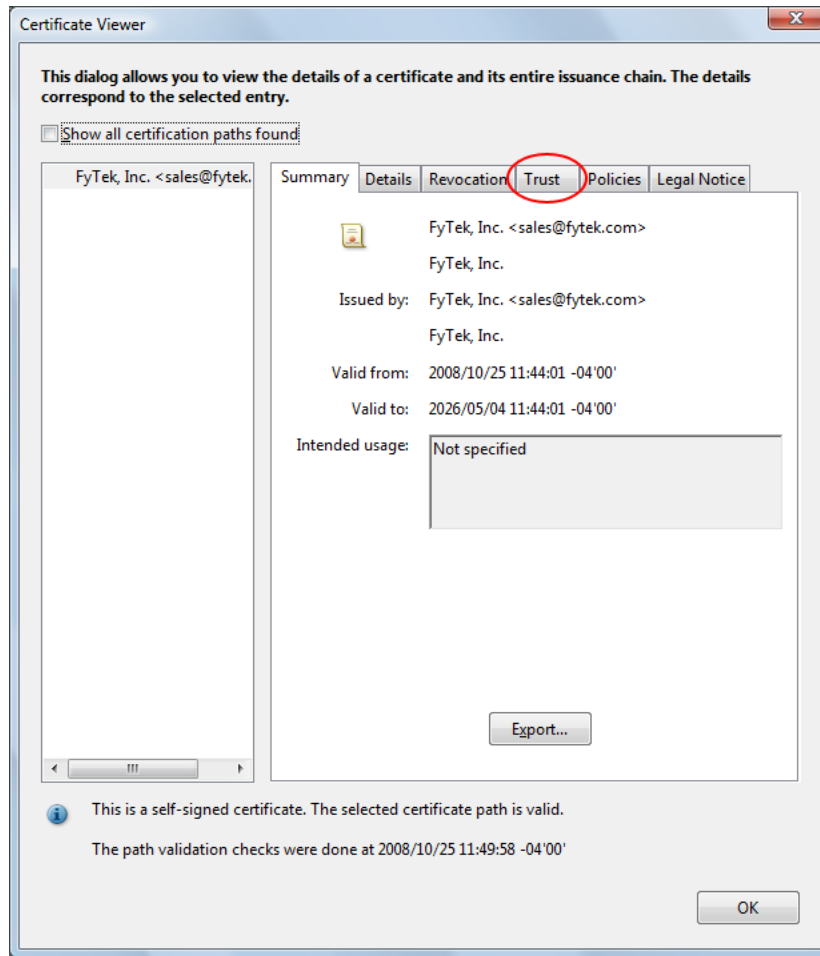
The third method is to trust the certificate directly from the PDF. You can use this method to trust certificates you created or when you are sure of the source of the PDF. The first step is to click on the signature field to bring up the dialog box shown. Your dialog boxes may differ slightly in options depending on the version of Adobe Reader you are using. These examples use Adobe Reader version 9. Note this document is signed so you can follow the steps below for this PDF if you like.



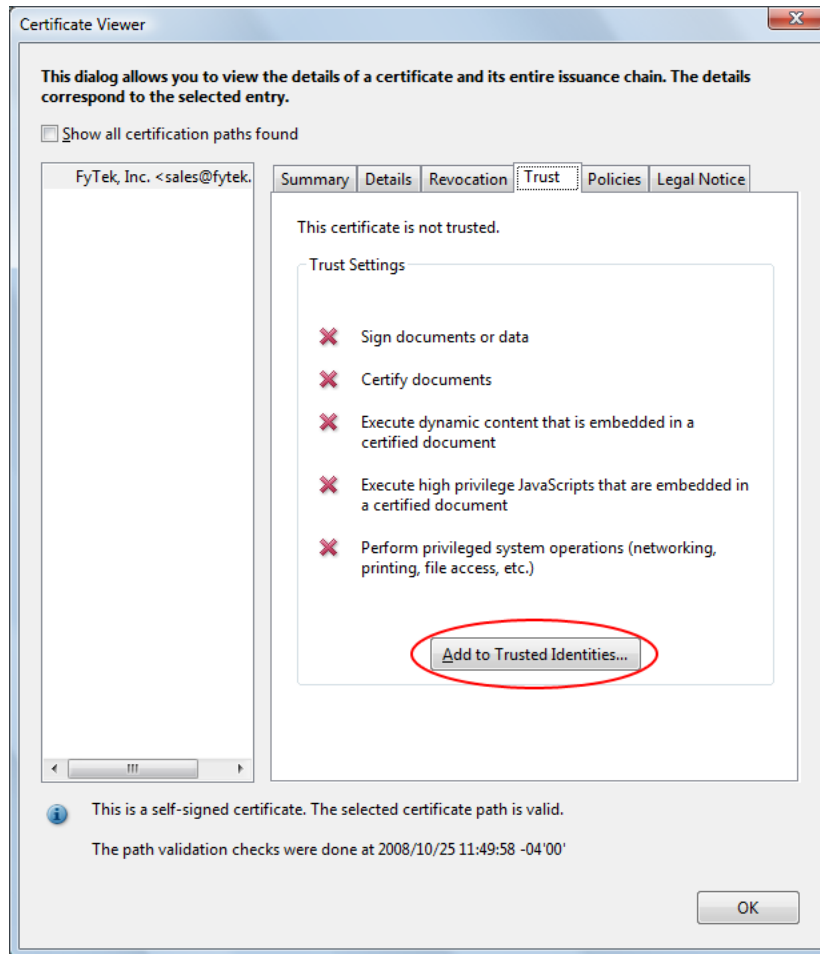
This is the dialog box that appears once you click the signature field. Click the "Signature Properties..." button to continue.



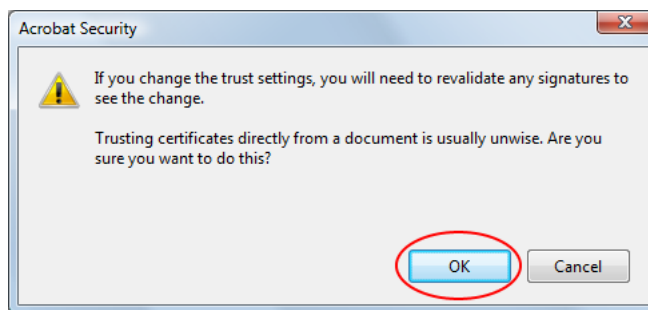
This is the signature property dialog for the certificate. Across the top of the dialog area is a set of tabs you can click on to view various information. For now, click the "Show Certificate..." button to continue.



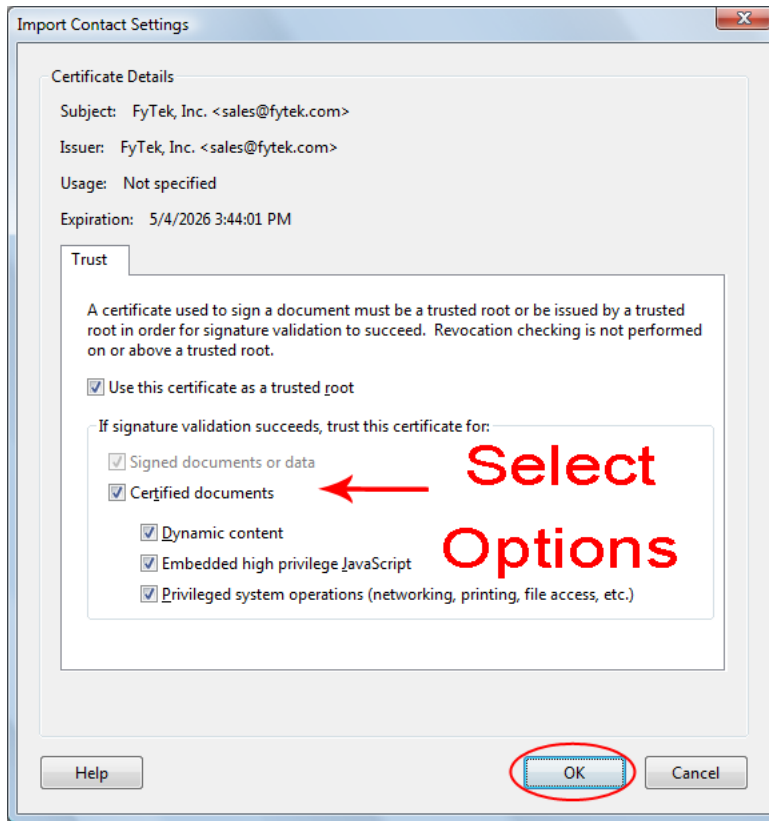
Another dialog box will open containing a set of tabs across the top. Click on the "Trust" tab.



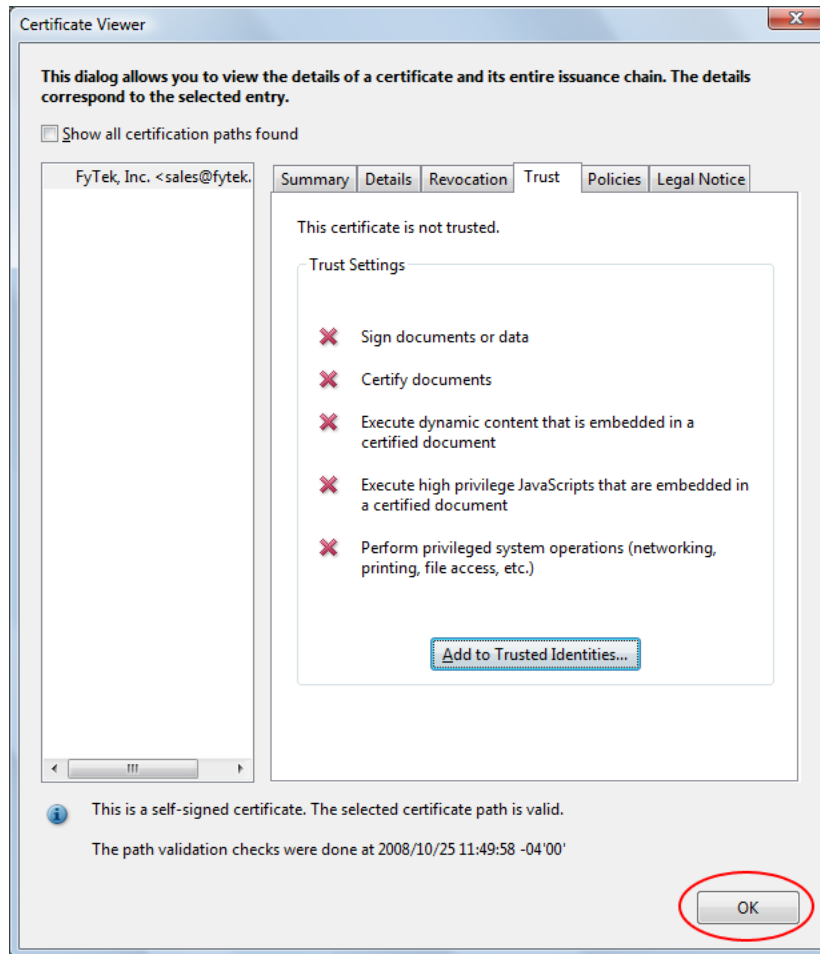
The trust tab shows what trusts you have enabled for the certificate. In this case, no trusts have been established. To trust this certificate, click the "Add to Trusted Identities..." button.



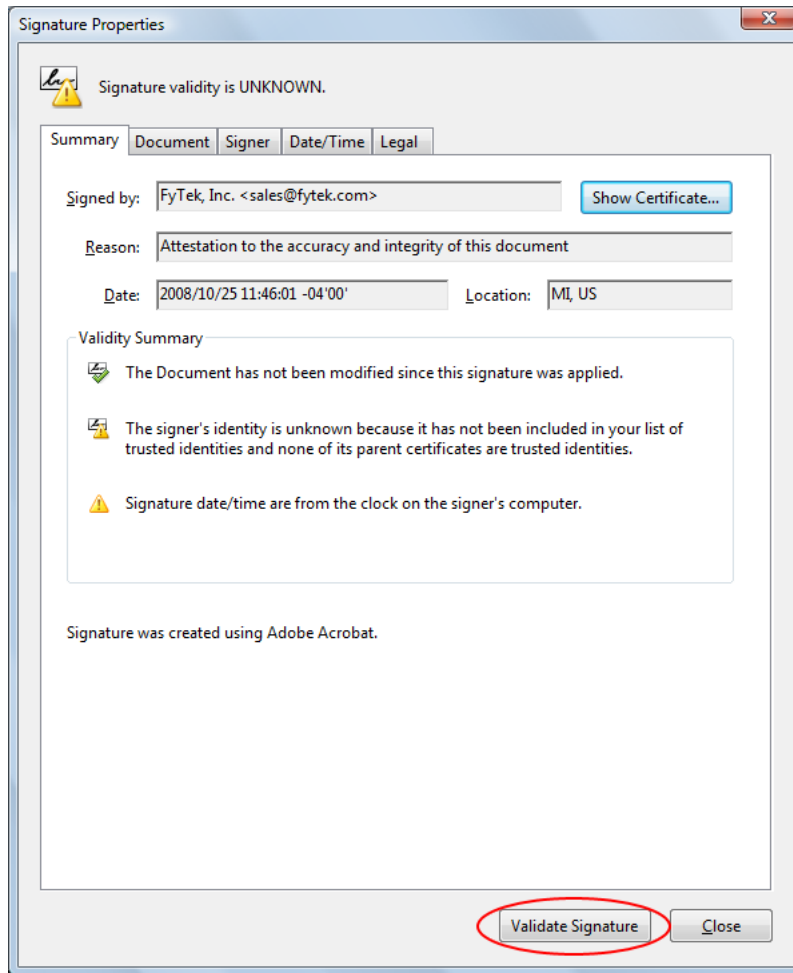
You will likely receive a warning box. Be sure to only trust certificates when you are certain of their source.



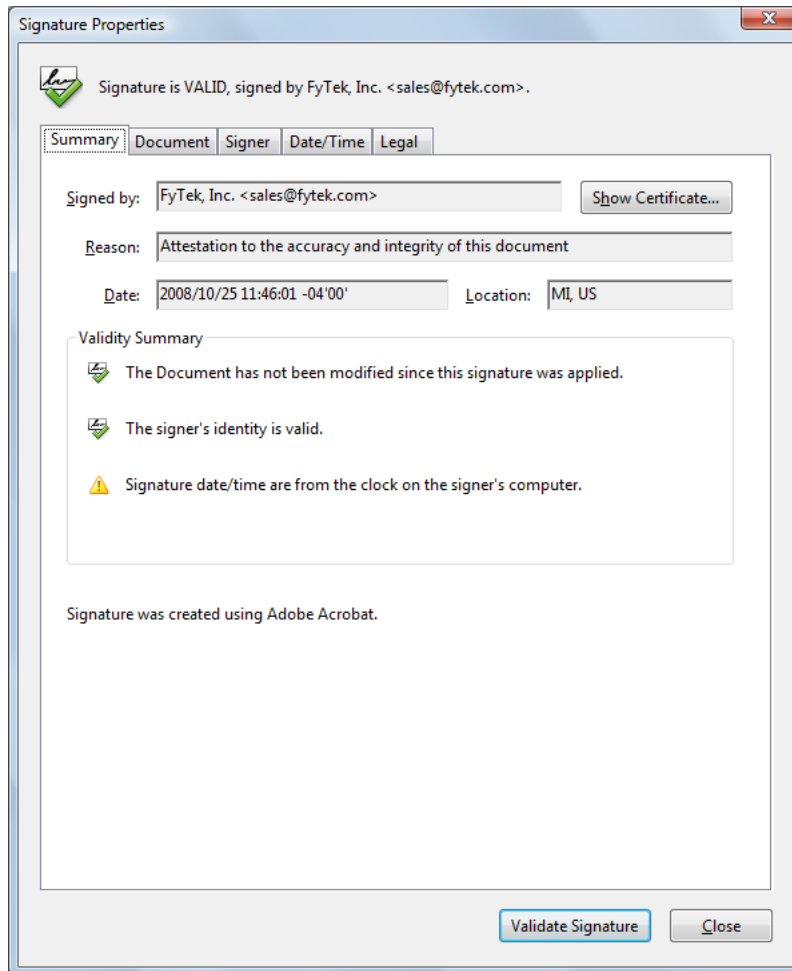
Select what items you want to trust the certificate for by clicking the checkboxes.



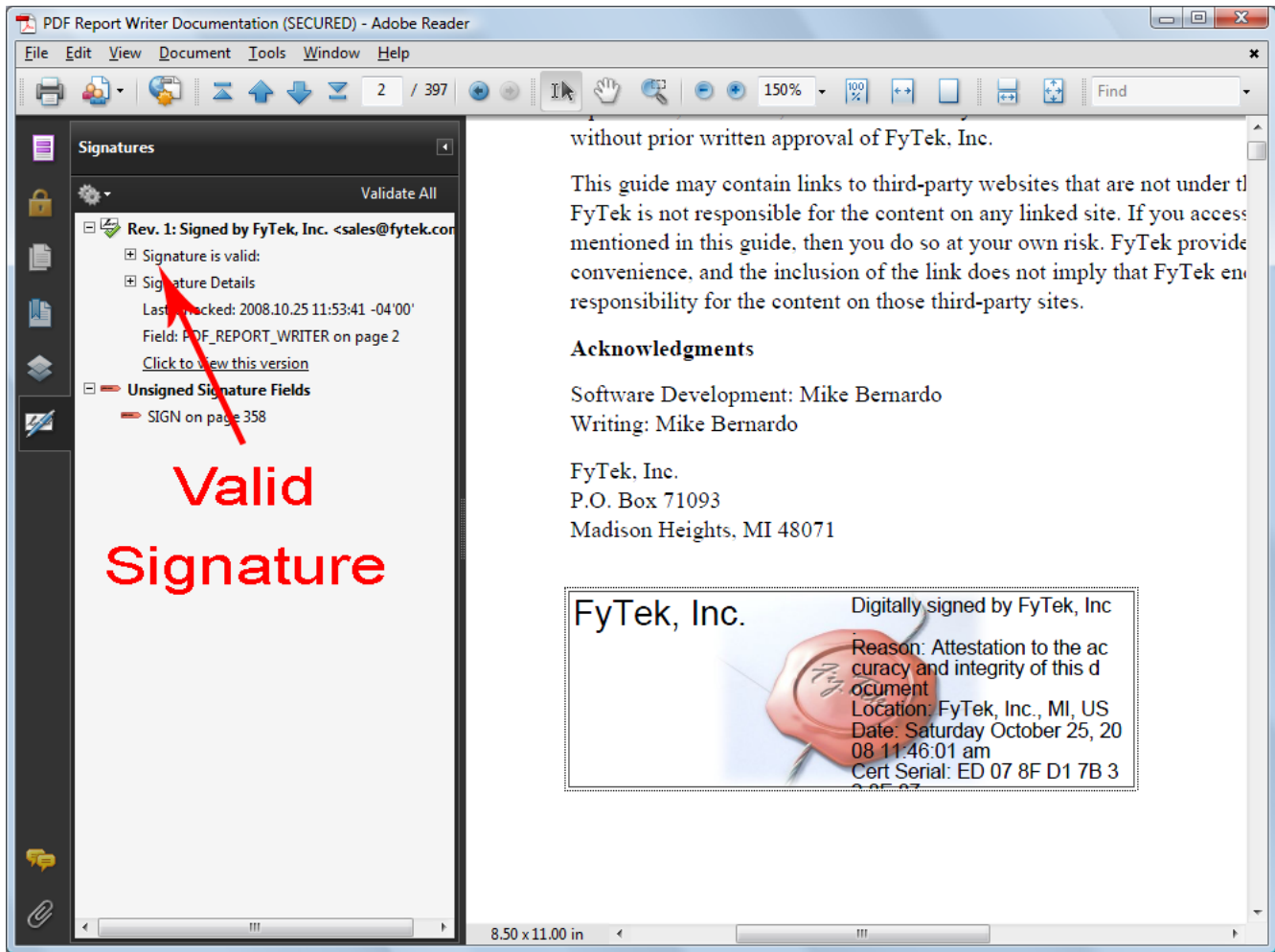
Click the "OK" button to continue. Note the red X's will remain until we revalidate the signatures.



Click the "Validate Signatures" button to validate the signature we just setup the trusts for.

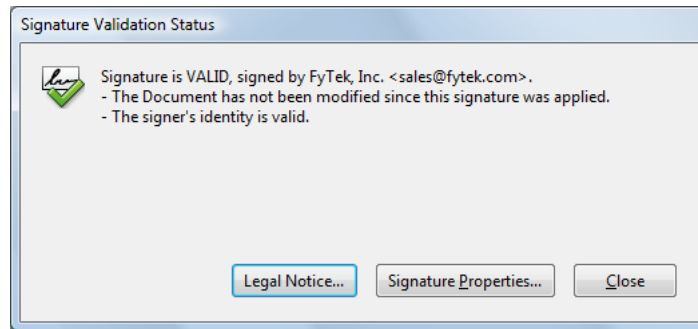


A green check icon now shows in the signature properties dialog.



A green check icon also shows in the signature pane. All future signings using this certificate will be trusted. The signature pane on the left will show what signings have taken place on the document and what signatures are open for signing. In this case there is one signature so far but an open signature box remains. You may also follow through on the dialog boxes by clicking the second signature (once signed) to view any changes to the document that happened between the time of the first and second signature.

Digital Signatures



This is what you will see now when you first click the signature field, assuming the signature is valid and the PDF has not been tampered with.

Overlaying PDFs

Overlaying of PDF pages (via the `setOverlay` method or `-overlay` option) involves low-level restructuring within PDFs. For most PDFs, these functions will provide the intended output. In some cases certain PDFs may not work correctly using the default algorithms for overlay. PDF Meld has a couple other options you may want to try when this happens.

The alternate method `setOverlay2` and option `-overlay2` are provided to perform a different type of overlay. These functions may work if `setOverlay` or `-overlay` are not working with your PDFs. Try `setOverlay2` or `-overlay2` if you are having difficulties with the overlay function. These may also work faster, depending on the PDFs involved.

There is another option in the event you are still not able to get the overlay to work with the above functions. The `setResRename` method or `-resrename` option may help. Use these along with the standard `setOverlay` or `-overlay` functions.

The DLL methods is:
`setResRename("x")`

The command line options is:
`-resrename "x"`

Set the "x" above to one of the following letters: "R", "S" or "F". Each one represents a different algorithm for updating the resources. One may allow you to perform the overlay. Do not use this option unless you are experiencing overlay problems. Note that in some cases it may not be possible to overlay pages due to the filters or certain compression algorithms used within the document.

Google Drive/OCR

The Google Drive options allow you include documents from Google Drive and/or save your PDF to Google Drive. Additionally, you may use this option to OCR a scanned image PDF. You may OCR with or without your own service account though it's recommended you use your own service account for security.

A service account is one that you set up through Google and provide it with the necessary privileges to access your documents. Following this introduction are some instructions on how to set up a service account. Once your service account is setup you request a JSON file containing the credentials needed to access your service account from other software, such as PDF Meld. Since your JSON file is plain text and may be used by anyone with access to it, there is a `-gdriveencfile` option you may use to encrypt this file so it is only readable by PDF Meld. To use it, pass in your JSON file with the `-gdrivejson` option and specify `-gdriveencfile` and provide a file name for the encrypted file. For example, `pdfmeld.exe filein.pdf fileout.pdf -gdrivejson "myfile.json" -gdriveencfile "myfile.enc"`. Then from now on, use the encrypted file to pass to `-gdrivejson` and leave off the `-gdriveencfile`. For example, `pdfmeld.exe filein.pdf fileout.pdf -gdrivejson "myfile.enc" -gdriveocrsave`.

Note that a service account is not the same as your personal Google Drive even if you setup the service account while signed in with your personal Gmail account. If you want to view the documents under your personal account you will need to share them with your email account. Use the `-gdriveshareemail`, `-gdrivesharetype` and `-gdriveshareaccess` options in this case. Set the options like this `-gdriveshareemail "youremail@gmail.com" -gdrivesharetype "user" -gdriveshareaccess "writer"`.

Another option is to create a folder under your account. Share that folder with your service account. Pass the folder id (which you can get if you go into the folder and look at the URL) with `-gdriveparents` option and the file will show up in the folder.

The `-gdriveocrsave` or `setGDriveOCRSave` method is used to OCR your output PDF. You will need an internet connection to transfer the output to Google Drive for OCR processing even if you are not using your own service account. Running an OCR option without providing Google Drive credentials uses a generic FyTek account and the file is not retained on Google Drive.

PDF Meld

Google Drive/OCR

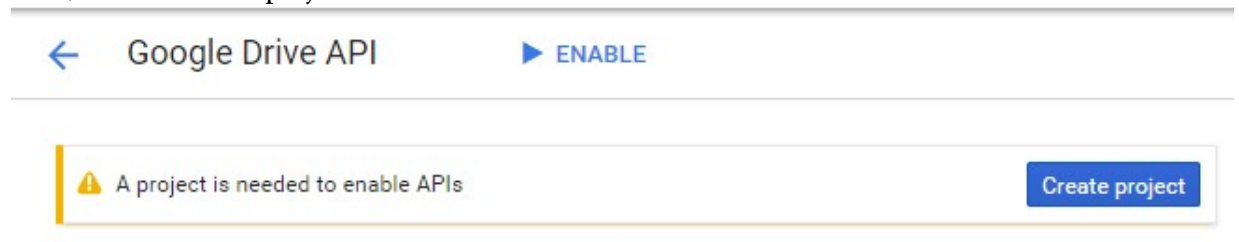
If you want to access a file from your Google Drive as an input to PDF Meld, make sure your service account has read access to the document. Include the `-gdrivejson` option or `setGDriveJSON` method to pass in your JSON formatted credentials by supplying the file name containing the credentials. Use the id of the document along with the https address to pass a Google Drive document as an input file. For a Google document/worksheet, use the format "https://www.googleapis.com/drive/v3/files/(fileid)/export?mimeType=application/pdf::ext=pdf" as your file name. For a PDF saved in Google Drive, use the format "https://www.googleapis.com/drive/v3/files/(fileid)::ext=pdf". Note the non-standard `::ext=pdf` option at the end for PDF Meld so it knows the content to expect is a PDF. To save your output to Google Drive be sure to also include `-gdrivesave` or `setGDriveSave`.

Below are the instructions for setting up a service account as of November 2016. Google may change this process at any time so you might need to make adjustments to the below instructions.

To set up a service account for yourself, start by going to this address to access the Google API page: <https://console.developers.google.com/apis/library>. Click on the "Drive API" link.



Next, click on Create project.



Name your project and click create.

New Project

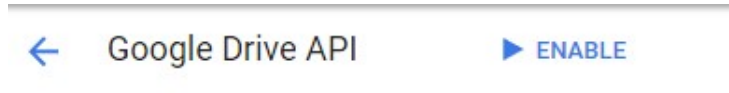
Project name ?

Your project ID will be liquid-idiom-147218 ? [Edit](#)

[Show advanced options...](#)

[CANCEL](#) [CREATE](#)

When the screen refreshes, click on Enable.



Next, go to Credentials.



 This API is enabled, but you can't use it in your project until you create credentials. Click "Go to Credentials" to do this now (strongly recommended). [Go to Credentials](#)

Click the link to create a service account.

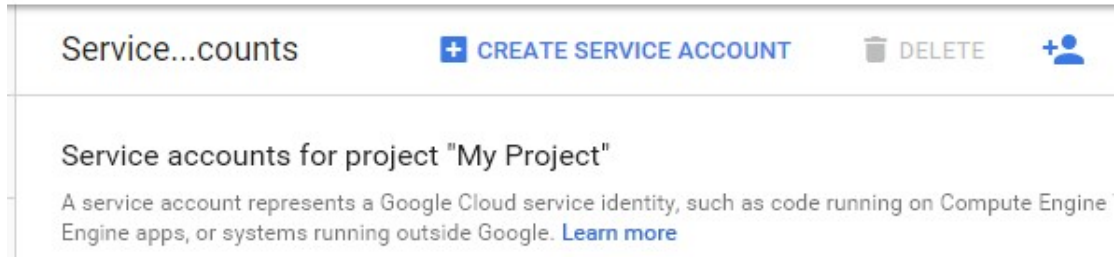
Add credentials to your project

1 Find out what kind of credentials you need

We'll help you set up the correct credentials
If you wish you can skip this step and create an [API key](#), [client ID](#), or [service account](#)

Which API are you using?

Click the link to create a service account.



Set the role as owner and provide a name. Export a JSON key.

Create service account

Service account name [?] Role [?]

Service account ID

Furnish a new private key
Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

JSON
Recommended

P12
For backward compatibility with code using the P12 format

Enable G Suite Domain-wide Delegation
Grants a client access to all users' data on a G Suite domain without manual authorization on their part. [Learn more](#)

CANCEL **CREATE**

Use the JSON key as the value for the option `-gdrivejson`. Note you may encrypt this key with the `-gdriveencfile` option. That will create a new file with same name and an `.enc` extension added. Then you pass this file instead to `-gdrivejson`.