3-Heights™ PDF Optimizer Shell

Version 4.8

PDF-TOOLS.COM
Premium PDF Technology

Contents

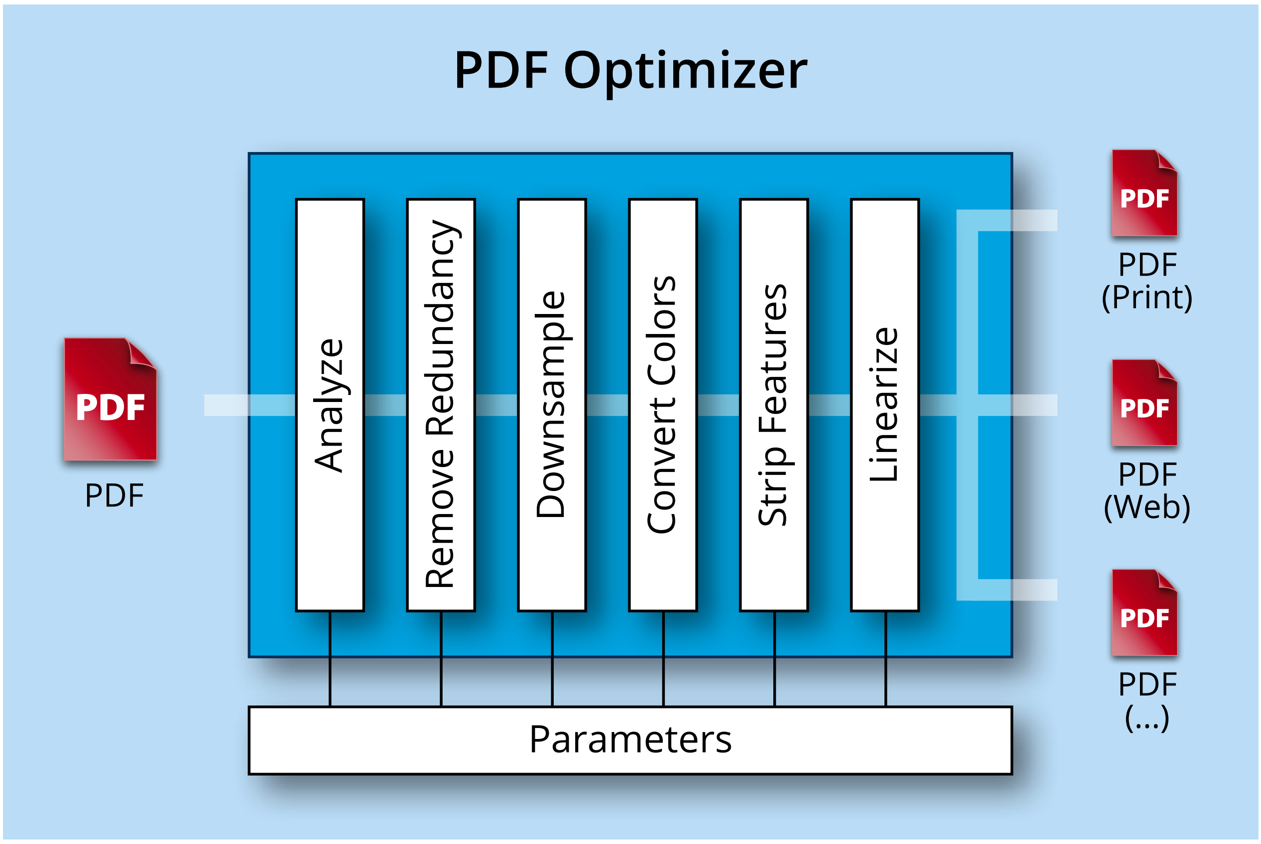
1 Introduction

1.1 Description

The 3-Heights™ PDF Optimizer Shell optimizes PDF files to enable their use as high resolution files for printing or, with less resolution, for electronic document exchange or space-saving document archiving.

Many processes produce very large PDF files that are not suitable for electronic document exchange. Users are then tempted to convert the PDF documents into other formats, but this only makes the situation even worse. The correct approach, and the easiest, is to optimize large PDF documents.

This process optimizes fonts and images to the best possible size and quality. It also removes redundant document content and "linearizes" PDF documents to enable fast web display.



1.2 Functions

The use of the latest compression algorithms enables the tool to reduce the memory space requirements for images or lessen their resolution, remove redundant and alternative information, optimize fonts through summarization or subsetting, convert colors and linearize the PDF.

1.2.1 Features

- Optimization for Electronic Document Exchange, Web Publishing and Archiving
- Customized compression of bi-tonal, monochrome and color images
- Define image resolution in dots per inch
- Define threshold value for down-sampling

- Set the quality index of lossy compression
- Automatically select best compression type for images
- Perform mixed raster content (MRC) optimization for images
- Remove invisible parts of images
- Linearization (fast web display)
- Compile and subset fonts
- Read encrypted input files
- Encrypt and set access authorization for the output file
- Process memory-resident files
- Removal of:
  - Redundant objects
  - Obsolete objects stemming from previous changes to the file
  - Embedded standard fonts (e.g. Courier, Arial, Times)
  - Embedded, non-symbolic fonts
  - Unnecessary file information
  - Article threads
  - Alternative images
  - Metadata
  - Page piece information
  - Document structure tree including markup
  - Miniature page preview images
  - Spider (web capture) information
- Remove or clear form fields and annotations

Optimize for Printing

- Color conversion (to RGB, CMYK or grayscale)
- Allow high print quality
- Set minimum PDF version of the output file

List and extract parameters

- Fonts and their properties
- Images and their properties
- Error Code
- Number of pages

1.2.2 Formats

Input Formats

- PDF 1.x (e.g. PDF 1.4, PDF 1.5.)

Target Formats

- PDF 1.x (e.g. PDF 1.4, PDF 1.5.)

1.2.3 Compliance

Standards ISO 32000 (PDF 1.7)

1.3 Operating Systems

The 3-Heights™ PDF Optimizer Shell is available for the following operating systems:

- Windows Vista, 7, 8, 8.1, 10 – 32 and 64 bit
- Windows Server 2008, 2008 R2, 2012, 2012 R2, 2016 – 32 and 64 bit
- HP-UX 11 and later PA-RISC2.0 – 32 bit
- HP-UX 11i and later ia64 (Itanium) – 64 bit
- IBM AIX 6.1 and later – 64 bit
- Linux 2.6 – 32 and 64 bit
- Oracle Solaris 2.8 and later, SPARC and Intel
- FreeBSD 4.7 and later (32 bit) or FreeBSD 9.3 and later (64 bit, on request)
- macOS 10.4 and later – 32 and 64 bit

2 Installation

2.1 Windows

The 3-Heights™ PDF Optimizer Shell comes as a ZIP archive containing various files including runtime binary executable code, files required for the developer, documentation and license terms.

1. Download the ZIP archive of the product from your download account at `https://www.pdf-tools.com`.
2. Unzip the file using a tool like WinZip available from WinZip Computing, Inc. at `http://www.winzip.com` to a directory on your hard disk where your program files reside (e.g. `C:\Program Files\PDF Tools AG`)
3. Check the appropriate option to preserve file paths (folder names). The unzip process now creates the following subdirectories:

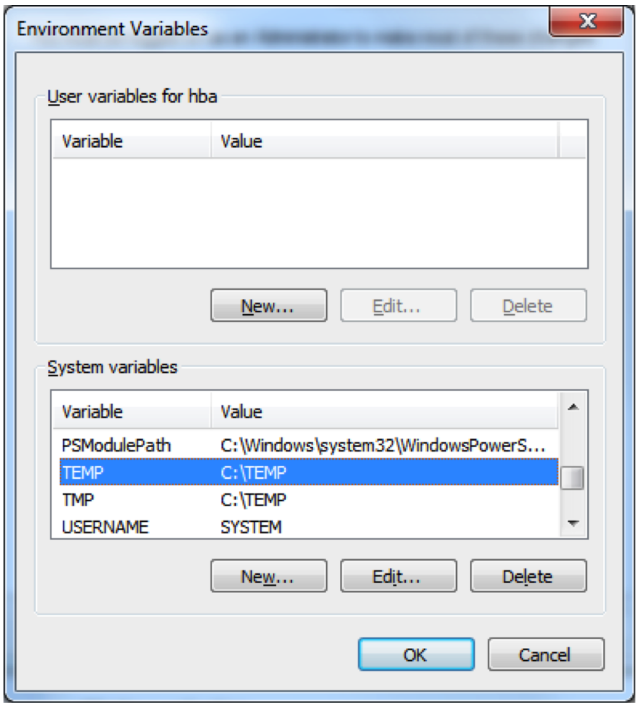
Subdirectory	Description
`bin`	Contains the runtime executable binary code.
`doc`	Contains documentation files.

   There is the option to download the software as MSI file, which makes the installation easier.
4. To easily use the 3-Heights™ PDF Optimizer Shell from a shell, the directory needs to be included in the "Path" environment variable.
5. Optionally register your license key using the License Management.
6. Make sure your platform meets the requirements regarding color spaces described in chapter Color Profiles.

2.1.1 How to set the Environment Variable "Path"

To set the environment variable "Path" on Windows, go to Start → Control Panel (classic view) → System → Advanced → Environment Variables.

Select "Path" and "Edit", then add the directory where `pdfoptimize.exe` is located to the "Path" variable. If the environment variable "Path" does not exist, create it.


Environment Variables

User variables for hba

Variable	Value

System variables

Variable	Value
PSModulePath	C:\Windows\system32\WindowsPowerS...
TEMP	C:\TEMP
TMP	C:\TEMP
USERNAME	SYSTEM

2.2 Unix

This section describes installation steps required on all Unix platforms, which includes Linux, macOS, Oracle Solaris, IBM AIX, HP-UX, FreeBSD and others.

2.2.1 All Unix Platforms

1. Unpack the archive in an installation directory, e.g. `/opt/pdf-tools.com/`
2. Copy or link the executable into one of the standard executable directories, e.g:

```
ln -s /opt/pdf-tools.com/bin/pdfoptimize /usr/bin
```

3. Verify that the GNU shared libraries required by the product are available on your system:
   - *On Linux*:

   ```
   ldd pdfoptimize
   ```

   - *On AIX*:

   ```
   dump -H pdfoptimize
   ```

   In case you have not installed the GNU shared libraries yet, proceed as follows:
   a. Go to `http://www.pdf-tools.com` and navigate to “Support” →“Resources”.
   b. Download the GNU shared libraries for your platform.
   c. Extract the archive and copy or link the libraries into your library directory, e.g `/usr/lib` or `/usr/lib64`.
   d. Verify that the GNU shared libraries required by the product are available on your system now.
4. Optionally register your license key using the Command Line License Manager Tool.
5. Make sure your platform meets the requirements regarding color spaces described in chapter Color Profiles.

2.3 Uninstall

If you used the MSI for the installation, go to Start → 3-Heights™ PDF Optimizer Shell. . . → Uninstall . . .

If you used the ZIP file: In order to uninstall the product undo all the steps done during installation, e.g. un-register using `regsvr32 -u`, delete all files, etc.

2.4 Color Profiles

The color conversion feature of the 3-Heights™ PDF Optimizer Shell uses color profiles by default.

For calibrated color spaces (such color spaces with an associated ICC color profile) the color conversion is well defined. For the conversion of uncalibrated device color spaces (DeviceGray, DeviceRGB, DeviceCMYK) however, the 3-Heights™ PDF Optimizer Shell requires apropriate color profiles. Therefore it is important, that the profiles are available and that they describe the colors of the device your input documents are intended for.

Note: When setting an alternative color management system such as Neugebauer, no color profiles are required.

If no color profiles are available, default profiles for both RGB and CMYK are generated on the fly by the 3-Heights™ PDF Optimizer Shell.

2.4.1 Default Color Profiles

If no particular color profiles are set default profiles are used. For device RGB colors a color profile named `"sRGB Color Space Profile.icm"` and for device CMYK a profile named `"USWebCoatedSWOP.icc"` are searched for in the following directories:

Windows

1. `%SystemRoot%\spool\drivers\color`
2. directory `Icc`, which must be a direct sub-directory of where the `pdfoptimize.exe` resides.

Linux and other Unixes

1. `$PDF_ICC_PATH` if the environment variable is defined
2. the current working directory

2.4.2 Get Other Color Profiles

Most systems have pre-installed color profiles available, for example on Windows at `%SystemRoot%\system32\spool\drivers\color\`. Color profiles can also be downloaded from the links provided in the directory `bin\Icc\` or from the following websites:

- http://www.pdf-tools.com/public/downloads/resources/colorprofiles.zip
- http://www.color.org/srgbprofiles.html
- https://www.adobe.com/support/downloads/iccprofiles/iccprofiles_win.html

2.5 Note about the Evaluation License

With the evaluation license the 3-Heights™ PDF Optimizer Shell automatically adds a watermark to the output files.

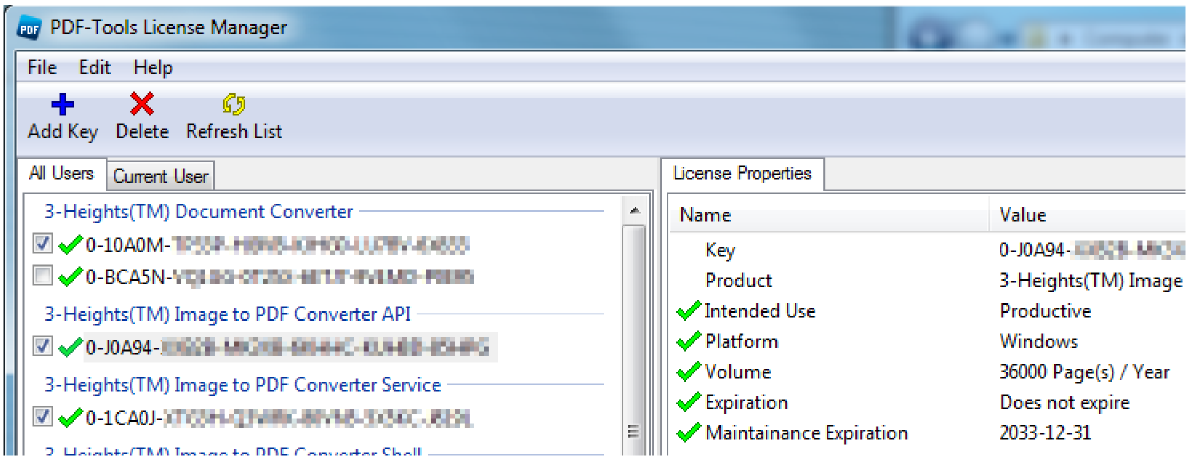
3 License Management

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

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

3.1 Graphical License Manager Tool

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



3-Heights(TM) Document Converter
- ☑ ✓ 0-10A0M-[redacted]
- ☐ ✓ 0-BCA5N-[redacted]

3-Heights(TM) Image to PDF Converter API
- ☑ ✓ 0-J0A94-[redacted]

3-Heights(TM) Image to PDF Converter Service
- ☑ ✓ 0-1CA0J-[redacted]

3-Heights(TM) Image to PDF Converter Shell

Name	Value
Key	0-J0A94-[redacted]
Product	3-Heights(TM) Image
✓ Intended Use	Productive
✓ Platform	Windows
✓ Volume	36000 Page(s) / Year
✓ Expiration	Does not expire
✓ Maintainance Expiration	2033-12-31

3.1.1 List all installed license keys

The license manager always shows a list of all installed license keys in the left pane of the window. This includes licenses of other PDF Tools products. The user can choose between:

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

3.1.2 Add and delete license keys

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

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

3.1.3 Display the properties of a license

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

3.1.4 Select between different license keys for a single product

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

3.2 Command Line License Manager Tool

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

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

```
licmgr
```

List all installed license keys:

```
licmgr list
```

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

Add and delete license keys:

Install new license key:

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

Delete old license key:

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

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

g	For all users
u	Current user

Select between different license keys for a single product:

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

3.3 License Key Storage

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

3.3.1 Windows

The license keys are stored in the registry:

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

3.3.2 macOS

The license keys are stored in the file system:

- `/Library/Application Support/PDF Tools AG` (for all users)

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

3.3.3 Unix/Linux

The license keys are stored in the file system:

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

Note: The user, group and permissions of those directories are set solely by the license manager tool. It may be necessary to change permissions to make the licenses readable for all users. Example:

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

4 Getting Started

The simplest command requires two parameters: The names of the PDF input and output files.

```
pdfoptimize input.pdf output.pdf
```

This command will generate an new PDF file with almost no optimization done at all. In order to perform an actual optimization, more parameters have to be specified in the form of options. E.g., in the following the "web" optimization profile is selected (see `-pr`):

```
pdfoptimize -pr web input.pdf output.pdf
```

All options start with a hyphen (-). Some options, as in the example above, require additional parameters. All options are documented in the Reference Manual.

4.1 Usage

By typing `pdfoptimize` without parameters, the usage, the version and a list of available options is returned.

4.2 Specify the Folder of the Output File

The output folder can simply be added in front of the output file name

```
pdfoptimize input.pdf myfolder\output.pdf
```

or absolute (Windows):

```
pdfoptimize input.pdf C:\myfolder\output.pdf
```

4.3 Processing All Files in a Folder

If you would like to process all files in a directory, it is required to use a variable to name the output files. Here is an example using the `for` command of the CMD shell (see also `for /?` for additional help) and the variable `%i`. It optimizes all `*.pdf` files in the current directory and saves them with the appendix `"_opt"`, in the same folder:

```
for %i in (*.pdf) do pdfoptimize -v -pr web %i %~ni_opt.pdf
```

If you would like to keep the file name, the output documents need to be created into another folder. The input file cannot be overwritten directly due to the fact that the optimization process reads from the input file, while it already writes to the output file.

```
for %i in (C:\in\*.pdf) do pdfoptimize -pr web %i C:\out\%~ni.pdf
```

When using variables in a batch file (`.bat`), variables have 2 leading `%` instead of just 1 like on the command line.

4.3.1 Windows Batch Sample

In a situation where all files in a directory need to be processed and the optimized file should have the same name as the original document, i.e. overwrite it, the following approach can be used.

- Make sure you really want this, the original file is lost in this process!
- Create the output files, either with a different name or in a different directory.
- Ensure the output files are created correctly. This can be done by verifying the return code (must be 0), or verify the document was created at all and is not empty.
- Delete the original file.
- Rename or copy back the new file to replace the original file.

The following sample does the steps described above. This sample does not ensure to always yield a correct result. Errors in the optimization or an abort of the process can still lead to loss of data. It is suggested to keep a backup of the original files.

```
@ECHO off
rem *************************************************************************
rem * This batch files optimizes all PDF files in the current directory     *
rem *                                                                        *
rem * The steps are as following:                                            *
rem *                                                                        *
rem * 1. Optimize all files in a folder. The optimized output files         *
rem *    have the temporary extension .tmp.                                  *
rem *                                                                        *
rem * 2. If the return code of the pdfoptimize is 0, and an output is       *
rem *    created, the optimization process is considered successful.        *
rem *                                                                        *
rem * 3. If successful, the original input file is deleted and the         *
rem *    .tmp file is renamed to .pdf.                                      *
rem *                                                                        *
rem *    If the process was not successful, the .tmp file is deleted        *
rem *    and the original file is left as is.                               *
rem *************************************************************************

IF EXIST *.tmp DEL /F /Q *.tmp
FOR %%i in (*.pdf) DO (
   SET name=%%~ni
   CALL :_Optimize
)
GOTO :EOF
rem *************************************************************************
:_Optimize
pdfoptimize -pr web "%name%.pdf" "%name%.tmp"
IF NOT %ERRORLEVEL%==0 (
   @ECHO ** Optimization process for %name%.pdf [error code %ERRORLEVEL%].
   IF EXIST "%name%.tmp" DEL /F /Q "%name%.tmp"
) ELSE (
   IF EXIST "%name%.tmp" (
      IF EXIST "%name%.pdf" (
         DEL /F /Q "%name%.pdf"
         IF NOT EXIST "%name%.pdf" (
           RENAME "%name%.tmp" "%name%.pdf"
```

```
            @ECHO ** Optimization successful for %name%.pdf.
          ) ELSE (
            DEL /F /Q "%name%.tmp"
            @ECHO ** Optimization failed for %name%.pdf [file locked].
          )
        )
    ) ELSE (
      @ECHO ** Optimization failed: %name%.pdf [error code %ERRORLEVEL%].
    )
)
GOTO :EOF
```

In order to optimize all files in all sub-folders, it's easiest to create a batch file that runs through all sub-folders and executes the batch file above.

So, create a batch file called `run.bat` and copy the upper code in it.

Then create another batch file called for example `runsub.bat` and add the code below:

```
@ECHO OFF
FOR %%r IN (.\) DO SET rootfolder=%%~pr
FOR /R %%s IN (.) DO (
   CD %%s
   CALL %rootfolder%run.bat
)
CD %rootfolder%
SET rootfolder=
```

Now copy the two batch files to the root folder (i.e. the folder from which every PDF file in every sub folder should be processed) and run the batch `runsub.bat.`

5 Optimization Process

5.1 How to Optimize PDF Documents

5.1.1 Identify Target Application Area

PDF documents are used in a wide variety of application areas, all having different requirements. As a very first step, one should precisely identify the targeted application area. A few typical fields of application are described briefly below. However, PDF documents can also be used in other ways or in combinations of the ones listed below.

Web

All documents related to the web should be kept small in file size. As a consequence they take less storage on the web-server and can be transferred quicker, resulting in shorter download times.

In order to reduce the file size as much as possible, all information that is not required for displaying the document without a visual loss can be removed. This may include:

- Down-sampling images (`-dt`, `-dr`)
- Clipping images to their visible parts (`-oc`)
- Applying compressions algorithms with high compression ratios (`-fb`, `-fc`, `-fi`)
- Collapsing redundant objects (`-or`)
- Removing unused resources (`-od`)
- Removing irrelevant information such as article threads, metadata, alternate images, document structure information, etc. (Strip the File)
- Merging and sub-setting embedded font programs (`-s` and `-m`)
- Depending on the PDF documents to be optimized, font programs of embedded standard fonts can even be removed (`-rs`).

Additionally, PDF documents can be linearized (`-ow`). This is a method of preparing a PDF file in way that pages can be accessed randomly via a PDF viewer web-browser plug-in, i.e. selected pages can be displayed before the whole file is downloaded. For this to work, the PDF viewer web-browser plug-in has to support correct interpretation of linearized PDF.

Documents which are intended to be displayed on a Computer display should be saved in ab RGB (red green blue) color space. RGB is the native form for any light-emitting device, such as computer monitor or television. An RGB image uses three channels and therefore takes up less space than a CMYK (cyan magenta yellow black) image which uses four channels. (`-c`)

Printing

For printing applications the file size is not the highest priority. More important is to have a document which prints in a predictable way. This means that correct fonts should be used, colors should look as expected, images should be high in resolution, etc.

For that reason no data from the original document that is used for a well-defined re-production should be removed or altered. Fonts should not be un-embedded, images should not be down-sampled. (Of course there are always exceptions).

For many printing applications it may be advantageous to convert images to the CMYK color space because this is primarily used in systems that reflect light (such as printed paper). (`-c`)

In certain documents, the same font is embedded multiple times. If, e.g., a PDF-producing software embeds the same font for each created page, then large multi-page documents contain many copies of a font program. Also, a document can contain a complete font program, of which only very few glyphs are used for display. In such situations, merging and sub-setting font programs can lead to faster printing. (`-m` and `-s`)

There are still further ways to decrease the file size:

- Clipping images to their visible parts (`-oc`)
- Compressing uncompressed images, e.g. with a lossless compression type (`-fb`, `-fc`, `-fi`, see also Supported Image Compression Types)
- Collapsing redundant objects (`-or`)
- Removing unused resources (`-od`)
- Removing irrelevant information for printing, such as thumbnails, article threads, document structure information, etc. (Strip the File)

Archiving

Archiving can have varying requisites, such as: Minimize the file size, maximize the reproducibility of the document, minimize the access time to find a specific archived document, etc.

The most common way for archiving a PDF is the PDF/A format, which is defined in the ISO Standard 19005. PDF/A requires fonts to be embedded, metadata to be included and prohibits certain features, like LZW or JPEG2000 compression or alternate images. The 3-Heights™ PDF Optimizer Shell does not create PDF/A compliant output but can be used, e.g., to reduce the file size prior to converting to PDF/A.

Scanned Documents

For certain types of scanned documents, MRC (mixed raster content) optimization can have a significant impact on file size while still preserve the visual appearance of the document. The 3-Heights™ PDF Optimizer Shell supports MRC, see also Mixed Raster Content (MRC) Optimization for Images.

Special Requirements

As an example for a specific requirement, the 3-Heights™ PDF Optimizer Shell supports the restriction of compression types for images in a document to a given list of types. (See `-ft`, `-fb`, `-fc`, and `-fi`.)

Another requirement may be to encrypt the resulting PDF and protect it by a user password and/or by an owner password. The 3-Heights™ PDF Optimizer Shell provides both by means of the options `-p`, `-u`, and `-o`.

PDF documents which mainly consist of scanned images to which an OCR (optical character recognition) layer is to be applied at a later time should be optimized in a way that the OCR process of the optimized document works as well as with the original. That means that image compression should either be lossless, or at least perceptually lossless. Perceptually lossless refers to a compression which is lossy, but its visual quality is high enough that neither the human eye nor an OCR engine can distinguish between original and optimized document. (See also Supported Image Compression Types.)

5.1.2 Using Optimization Profiles

The 3-Heights™ PDF Optimizer Shell provides the notion of "optimization profiles" to quickly arrive at a configuration suitable for many application areas. Once the application area is defined, the optimization profile that best matches the requirements can be identified. The configuration is done by setting this profile (`-pr`) followed by adjusting individual settings that differ from the profile.

```
pdfoptimize -pr web -dbt -1 input.pdf output.pdf
```

The actual profile settings for all profiles are in Profile Settings.

5.2 Optimizing Images

For the 3-Heights™ PDF Optimizer Shell the target compression type of an image is specified by giving a numerical value between `-1` and `10` to the options `-fb`, `-fc`, and `-fi`. Several values can be combined in a comma-separated list. (No spaces are allowed in the list.)

The values `1` to `8` directly correspond to PDF compression types, other values indicate a special behavior.

Value	Compression
`0`	No Compression (Raw)
`1`	DCT (JPEG)
`2`	Flate (ZIP)
`3`	LZW
`4`	CCITT Fax Group 3 (CCITT Fax Group 3 and 4)
`5`	CCITT Fax Group 3 2D (CCITT Fax Group 3 and 4)
`6`	CCITT Fax Group 4 (CCITT Fax Group 3 and 4)
`7`	JBIG2 (Supported in PDF 1.4 or later)
`8`	JPEG2000 (Supported in PDF 1.5 or later, not supported in PDF/A-1)
`9`	In contrast to the values 0-8, this is not a single compression format. Instead, this enables MRC optimization on color and monochrome images. (See Mixed Raster Content (MRC) Optimization for Images) Application area: Scanned documents.
`10`	In contrast to the values 0-8, this is not a single compression format. Instead, this tells 3-Heights™ PDF Optimizer Shell to use the same compression as the original input image.
`-1`	Exclude from processing

5.2.1 Supported Image Compression Types

In PDF, up to 8 different ways of compressing binary data are supported. (See also PDF Reference 1.7, Chapter 3.3 for more information on these types.)

No Compression (Raw)

Raw means no compression is applied.

DCT (JPEG)

Developer	Joint Photographic Experts Group committee
Version	PDF 1.2, PDF/A-1
Color depth	8, 24 bits per pixel
Compression type	Lossy
Compression algorithm	The image is broken up into blocks that are 8 by 8 samples. On each of these blocks and color channel a discrete cosine transformation (DCT) is applied and its coefficients are quantized. The visual quality of the resulting image depends on the loss of information defined by the step size of the quantization and on the image that is being compressed. The compression can be controlled via an image quality parameter—a value from 1 to 100 (default 75). Typical compression ratios are 15:1 (no perceptible loss of information) to 30:1.
Application area	Sampled continuous-tone pictures (photographs)

Flate (ZIP)

Developer	Flate compression is based on the public-domain zlib / deflate compression method.
Version	PDF 1.2, PDF/A-1
Color depth	1-8, 24 bits per pixel
Compression type	Lossless
Compression algorithm	A lossless data compression algorithm that uses a combination of the LZ77 algorithm and Huffman coding.
Application area	Images

LZW

Developer	Abraham Lempel, Jacob Ziv and Terry Welch Copyright based issues, which expired in most countries in 2003/2004, reduced the popularity of this compression. As one of its consequences it is not included in PDF/A standard.
Version	PDF 1.2
Color depth	2-8 bits per pixel
Compression type	Lossless
Compression algorithm	An indexed based compression that is also used in the GIF and TIFF image formats.
Application area	Gray-scale images, artificial images

CCITT Fax Group 3 and 4

Developer	International Telecommunications Union (ITU), formerly known as the Comité Consultatif International Téléphonique et Télégraphique
Version	PDF 1.0, PDF/A-1
Color depth	1 bit per pixel
Compression type	Lossless
Compression algorithm	**Group3** 1-dimensional version of the CCITT Group 3 Huffman encoding algorithm. **Group 3 2D** 2-dimensional version of the CCITT Group 3 Huffman encoding algorithm. **Group 4** An advanced version of a bi-tonal algorithm based on the CCITT Fax Group 3 2D compression.
Application area	Line-art image, bi-tonal, faxes

JBIG2

Developer	Joint Bi-Level Image Experts Group
Version	PDF 1.4, PDF/A-1
Color depth	1 bit per pixel

Compression type	Lossless
Compression algorithm	The image is broken down into individual symbols, which are stored in a table. A symbol is added to the table if it does not exist yet. If a matching symbol already exists, it is used as a reference. This algorithm works especially well for images with a lot of similar symbols such as scanned text or images that use patterns. Generally JBIG2 provides a better compression ratio than CCITT Group 3 or Group 4 compression. Typical compression ratios for text pages are 20:1 to 50:1.
Application area	Line-art image, bi-tonal

JPEG2000

Developer	Joint Photographic Experts Group committee
Version	PDF 1.5, PDF/A-2
Color depth	8, 24 bits per pixel
Compression type	Lossless if the image quality index is set to `100`. Lossy otherwise
Compression algorithm	JPEG2000 is a wavelet-based image compression standard. It was developed with the intention of superseding the original discrete cosine transform-based JPEG standard.
Application area	Sampled continuous-tone pictures (photographs)

5.2.2 Relevant Factors for the File Size

The size of an image is basically determined by four factors:

The pixel mass The total amount of pixels the image has. An image with a size of 600 by 800 pixels has 480'000 pixels total.

The color depth How many bits are required to describe 1 pixel? The table below gives the answer for different types of images. For example, an RGB image with 600 by 800 pixels requires therefore 600 x 800 x 3 bytes = 1.44 Mbytes in uncompressed format.

Color Space	Description	Bits/Pixel
Bi-tonal	Black and white	1
Indexed	Colors are stored in an index table which usually holds 2 to 256 entries, e.g. GIF.	2-8

Grayscale	Monochrome	8
Color RGB	Color using Red, Green, Blue	24
Color CMYK	Color using Cyan, Magenta, Yellow, Key (=black)	32

The compression type A compression algorithm can compress data (such as an image) to reduce its file size. Such an algorithm belongs to either of the following two classes:

Lossless The original image can be restored exactly.

Lossy The compression modifies the pixels. The original image cannot be restored from the compressed version. This is typically applied to photographic images where the human eye cannot distinguish whether the image was modified. The most common lossy compression is JPEG. The benefit of lossy compression is the higher compression ratio.

See also Supported Image Compression Types.

The content of the image The simpler the image, the better it compresses. For most compression algorithms a simple image (e.g. completely white) compresses much better than a complex image (e.g. a photo).

Examples:

CCITT Fax compression was designed to compress black text written on a white background. The algorithm was optimized under the assumption that a page contains more white pixels than black pixels. Therefore a bi-tonal image with a lot of black does generally not compress as well as in image with more white even if they have the same pixel mass.

JBIG2 compression searches for patterns, and uses them multiple times. For example in a scanned text document the same few dozen of characters are used over and over again. The algorithm is optimized to save frequent patterns more efficiently than rare ones.

5.2.3 Provided Features for Optimizing Images

The 3-Heights™ PDF Optimizer Shell offers the following possibilities to optimize images:

The pixel mass can be reduced. (It cannot be increased.) This is done by clipping (cropping) the image size to its visible extent and/or by reducing the image resolution.

The resolution defines how many pixels there are in given length of the image. The most common unit for resolution is DPI (dots per inch). If an image has a resolution of 200 DPI, it means when displayed at 100% zoom, there are 200 pixels for 1 inch of image. The higher the resolution, the “sharper” is the image. A monitor has usually a resolution of at least 96 DPI, a laser printer of at least 600 DPI. When the file size matters, a common resolution for color and grayscale images in PDF is 150 DPI (usually higher for bi-tonal).

The process of changing the amount of pixels an image has, is called re-sampling, or down-sampling when the result has less pixels than the original image.

In the 3-Heights™ PDF Optimizer Shell down-sampling is applied by setting a target resolution and a threshold resolution. The default values are 150 DPI for the target resolution and 225 DPI for the threshold resolution. This means every image that has a resolution of 225 DPI or higher is potentially down-sampled to 150 DPI. Of course, the threshold resolution can be set equal to the target resolution. However there are many cases where down-sampling by just a little bit has disadvantages. In particular, lossy images (e.g. JPEG compression) loose visual quality every time they are newly compressed. On top of that the compressed output can be larger than the input because artifacts introduced by the previous compression(s) are now considered as part of the image which needs to be compressed and lead to a worse compression even when the resolution is reduced. Per default, the 3-Heights™ PDF Optimizer Shell will, however, prevent such unnecessary re-sampling.

The color depth can be modified for color images. The color depth can be left unchanged, set to Grayscale (8 bit), RGB (24 bit) or CMYK (32 bit). It cannot be changed to black and white (1 bit).

> **Note:** In certain circumstances, the color depth of the image is not converted, e.g. if the resulting file size increases or if the image is pre-blended with a matte color.

The color complexity can be reduced. By "color complexity" we mean the following hierarchy of possible image pixel contents:

1. All pixels have the same color.
2. All pixels are either black or white.
3. All pixels are colored gray.
4. Pixels have differing colors.

By color complexity reduction we mean that images are converted to their lowest possible color complexity. E.g., a color image with only black and white pixels is converted to a bi-tonal image. Furthermore, an image with color complexity 1 (single color) is down-sampled to one pixel.

Color complexity reduction is also applied to masks and soft masks: Soft masks of complexity 2 (bi-tonal) are converted to masks. Masks and soft masks with complexity 1 (single color) whose color is such that the (soft) mask is opaque are removed.

> **Note:** Currently, color complexity reduction is only carried out for images that have a device color space (DeviceRGB, DeviceGray, or DeviceCMYK) or an indexed color space whose base color space is a device color space.

The compression can be setup independently for the following three image compression types:

Type	Description
Bi-tonal	Black and white images.
Indexed	Images with an indexed (also known as "paletted") color space.
Continuous	Color (RGB and CMYK) images and grayscale images.

Bi-tonal images usually contain text or black and white graphics, indexed images usually contain color graphics such as logos, while continuous images usually contain photographs.

For each of the above image types, several compression algorithms can be set. The 3-Heights™ PDF Optimizer Shell tries all the given compression algorithms and takes the one that yields the smallest file size. Note that the more compression algorithms are set, the longer the process of optimizing images will take.

Furthermore, a more conservative image processing strategy can be enabled. This strategy prevents all the compression trials if the image has neither been clipped nor down-sampled nor undergone a color-conversion. Hence, if the image has not been altered, then the original image from the input document is taken.

The content of the image cannot be changed directly. However changing the resolution or applying a lossy compression algorithm modifies the content of the image.

Note: Unless forcing of re-compression is enabled, the 3-Heights™ PDF Optimizer Shell never increases the file size of an image because it chooses the smallest among all tried compression algorithms and the original image in the input file. This means the 3-Heights™ PDF Optimizer Shell cannot be used to "uncompress" embedded images.

5.2.4 Mixed Raster Content (MRC) Optimization for Images

Some raster images—typically scanned documents—consist mainly of text, possibly in several colors and interspersed with some pictures. Such images are difficult to compress with one single compression type because of the diverse or even conflicting features of different parts of the image.

Note: There exists an optimization profile for MRC optimization. See Profile Settings.

MRC optimization is a way of breaking such images down into parts, such that each part is well suited for one type of a compression algorithm. With this approach, the resulting file size often can be reduced without significantly reducing the visual quality of the document.

Note:
- MRC optimization can only be enabled for continuous images, i.e. not for bitonal images and images with an indexed color space.
- MRC optimization may yield unexpected results, e.g. because the input image is not suitable for MRC. As another example, images in the original PDF may be stored as small slices, and MRC optimization fails because the 3-Heights™ PDF Optimizer Shell has no option to concatenate such image slices.
- A PDF that contains MRC-optimized images is not suited for optical character recognition (OCR) and image extraction.

In the 3-Heights™ PDF Optimizer Shell, MRC optimization works in three phases as explained below.

Phase 1: Cutting out Pictures

In this phase, the input image is analyzed and rectangular areas containing photographic features are detected. Each detected region is cut out and placed as a separate image in the resulting PDF.

Depending on the input image it is possible that this phase decides that the whole input image consists of one photographic region covering the whole image. In this case, the second phase (Phase 2: Separation into Layers) is omitted.

On the other hand, it is possible, that actual photographic regions present in the input image are not recognized correctly. This can happen for example if a photographic region contains parts with uniform color.

For the cut-out images, a compression type can be set.

Note: The resulting cut pictures are neither down-sampled nor color-converted.

Phase 2: Separation into Layers

For this second phase the image is not supposed to contain any photographic features. Instead, the image is assumed to consist of text and graphic, potentially with varying color.

Now, the whole image is separated into two layers, a foreground and a background layer. Additionally, a mask is created, which can be thought of as a bi-tonal image that is not displayed directly but tells for each pixel whether to show the foreground layer or the background layer.

Example:

Let the image consist of a yellow background with black paragraph text and a title text in red. Then the resulting background layer contains the yellow color only. The foreground layer contains the black text color where the paragraph text is located and the red text color where the title is located. In the mask, pixels for which the foreground layer should be displayed are set to 1, the others are set to 0. I.e. the mask contains 1's where the black and the red text is and 0's everywhere else.

In the resulting PDF the foreground layer, the background layer and the mask are stored as three images and thus are allowed to have different resolution and different compression types. Since all the detailed features have been moved to the mask, it makes sense to down-sample the foreground and background layers and use a low image quality. The mask on the other hand is usually stored with a lossless compression type optimized for text.

Phase 3: Reconstruction

In this phase the results of phase 1 (the cut-out images) and phase 2 (the layers and the mask) are used to synthesize the desired result. If in phase 1, a single photographic region covering the entire image is detected, then the original image is used and the reconstruction is finished. Otherwise, the reconstruction first places the background layer, followed by the foreground layer with the mask. Finally if any cut-images are found they are placed at their respective locations on top of the foreground layer.

5.3 Optimizing Fonts

Every text in a PDF document is written with a font. This font can either be embedded or not embedded in the resources of the PDF. Embedded means a font program is embedded that describes how glyphs are drawn. If a font is not embedded the application rendering the PDF (e.g. 3-Heights™ PDF Viewer or Adobe Acrobat) have to select a replacement font. Therefore the visual appearance of text written with an embedded font is determinable, whereas it is not when the font is not embedded.

A font program can be quite large. An embedded font which contains all WinAnsi characters has a size of about 20-100 Kbytes, if it contains a large Unicode range (e.g. Asian Characters) it can be several Mbytes, whereas an non embedded font requires much less.

This leads to the following ways to optimize fonts:

Remove the embedded font: Removing embedded fonts can reduce the file size of a document, particularly when the document contains many fonts. Removing fonts is best applied to (PDF-) standard fonts, such as Arial, Courier, Courier New, Helvetica, Times, Times New Roman. Removing fonts should not be applied to barcode fonts or fancy types.

> **Note:** PDF/A requires fonts to be embedded.

Subset fonts: Only keep the information in the font program that is required to render the characters that are actually used in text in this document. All unused characters are removed.

Merge fonts: A document can have the same font, or a subset of it, embedded multiple times. This commonly occurs when multiple input document, are merged into one large output document. The 3-Heights™ PDF Optimizer Shell Tool can merge these fonts into one font (if they can be merged).

6 Reference Manual

6.1 Profile Settings

The following table lists all settings for optimization all profiles. The profile labeled "(default)" contains the settings in effect when no profile is selected.

Note: Values in parentheses, although set, have no effect because images are excluded from processing or image down-sampling is disabled due to a threshold set to `-1`.

Profile Settings

	(default)	web	print	max	mrc
Compression Types for Bi-tonal Images (`-fb`):					
Exclude (`-1`)	✓				✓
CCITT Group 4 (6)		✓	✓	✓	
JBIG2 (7)		✓		✓	
Source (`10`)		✓	✓	✓	
Compression Types for Continuous Images (`-fc`):					
Exclude (`-1`)	✓				
JPEG (1)		✓	✓	✓	
JPEG2000 (8)		✓		✓	
MRC (9)					✓
Source (`10`)		✓	✓	✓	
Compression Types for Indexed Images (`-fi`):					
Exclude (`-1`)	✓				
Flate (2)		✓	✓	✓	

Profile Settings

	(default)	web	print	max	mrc
LZW (3)				✓	
Source (10)		✓	✓	✓	
Resolution for Bi-Tonal Images (Resolution Values per Image Type)	(200)[1]	200	(200)[1]	160	(200)[1]
Threshold for Bi-Tonal Images (Threshold Values per Image Type)	-1	280	-1	220	-1
Resolution for Monochrome Images (Resolution Values per Image Type)	(150)[1]	150	(150)[1]	130	(150)[1]
Threshold for Monochrome Images (Threshold Values per Image Type)	-1	210	-1	180	-1
Resolution for Color Images (Resolution Values per Image Type)	(150)[1]	150	(150)[1]	130	(150)[1]
Threshold for Color Images (Threshold Values per Image Type)	-1	210	-1	180	-1
Image Quality (`-q`)	75				
Color Conversion (`-c`)	0	1	2	0	0
Clip Images (`-oc`)		✓	✓	✓	✓
Reduce Color Complexity (`-rc`)		✓	✓	✓	
Force Re-Compression (`-ff`)					
Force Compression Types (`-ft`)					
Dithering Mode (`-h`)	`0` (Floyd Steinberg)				
MRC Layer Compression (`-ml`)	8 (JPEG2000)				
MRC Layer Resolution (`-mlr`)	70				
MRC Layer Quality (`-mm`)	6 (CCITT Fax Group 4)				

Profile Settings

	(default)	web	print	max	mrc
MRC Layer Quality (-mlq)	10				
MRC Cut Picture Compression (-mp)	1 (JPEG)				
Convert Fonts to CFF (-cff)		✓	✓	✓	
Merge Font Programs (-m)		✓	✓	✓	✓
Remove Standard Fonts (-rs)				✓	
Subset Font Programs (-s)		✓	✓	✓	✓
Optimize Resources (-od)		✓	✓	✓	✓
Linearize (-ow)		✓			
Remove Redundant Objects (-or)		✓	✓	✓	✓
Strip the File:					
Article threads (Strip the File)		✓	✓	✓	✓
Metadata (Strip the File)		✓		✓	
Piece info (Strip the File)		✓	✓	✓	✓
Document structure (Strip the File)		✓	✓	✓	✓
Thumbnails (Strip the File)		✓	✓	✓	✓
Spider (Strip the File)		✓	✓	✓	✓
Alternates (Strip the File)		✓		✓	
Alternates (Strip the File)				✓	
Other annotations (Strip the File)				✓	
Form fields (Strip the File)				✓	
Link Annotations (Strip the File)					

[1] These values, although set, have no effect because down-sampling of images is disabled.

6.2 Optimization Options

An option is a configuration string of the form

```
-‹option› ‹parameters›
```

where ‹option› is a 1 to 3-letter string that names the option and ‹parameters› are further configuration values given to this options. Many options don't support any ‹parameters›.

Options are provided on the command line after the command pdfoptimize. They define how the document should be optimized. The last two strings of the command line should always be the input and the output-file. (There is no output-file required when using any of the listing-options -li and -lf.)

Options are parsed from left to right, the last set value is applied. (An exception to this is setting a profile with -pr: Profiles are always applied first.)

Example: With the following options, the resolution for re-sampling of all raster image types (color, monochrome, bi-tonal) is first set to 100, then the monochrome resolution is set explicitly to 120.

```
pdfoptimize -dr 100 -dmr 120 input.pdf output.pdf
```

If in the above command the setting `-dmr 120` was set before `-dr 100`, it would not have any influence, since `-dr 100` applies to all compressions and therefore would overwrite the previous setting.

In the following all options are listed in alphabetical order.

6.2.1 -c Set the Color Conversion

Set the Color Conversion `-c ‹n›`

This option activates conversion of raster images from one color space into another, e.g. convert all RGB images to CMYK images.

This option does not have any impact on objects other than raster images that use color spaces, such as vector graphics or text. Color key masked images are not color converted. Pre-blended images can be converted from RGB to Grayscale.

Note: This option is affected when setting a profile (-pr).

The parameter ‹n› has the following meaning:

Color Conversions

‹n›	Conversion	Color Values
0	(default) Don't convert colors	
1	Convert to ICE sRGB colors	red, green, blue
2	Convert to CYMK color (using profiles)	cyan, yellow, magenta, key
3	Convert color images to grey scale	grey

Example: To convert all embedded color images that use the RGB color space to images of the CMYK color space, use the following command

```
pdfoptimize -c 2 input.pdf output.pdf
```

6.2.2 `-cff` Compress Type1 fonts (convert to CFF)

Compress Type1 fonts (convert to CFF) `-cff`

Convert embedded Type1 (PostScript) fonts to Type1C (Compact Font Format). This reduces the file size.

Note: This option is affected when setting a profile (`-pr`).

6.2.3 `-cms` Set the Color Management Engine

Set the Color Management Engine `-cms ‹engine›`

The transformation of colors from one color space to another is performed using a color management engine.

Supported engines are:

`none` The algorithms specified in the PDF reference are used. This results in the maximum possible contrast.

`neugebauer` The Neugebauer algorithm efficiently converts CMYK to RGB. It does not need any color profiles. The results, however, look similar to conversion using color profiles.

`lcms` (default): Use ICC color profiles. Default profiles are used for all unmanaged device color spaces as described in section Color Profiles.

`‹FileName›` When providing a file name, a configurable version of the Neugebauer algorithm is applied. The coefficients can be defined in the text file. The default Neugebauer coefficients are listed below (Red, Green, Blue; Color):

```
0.996078, 0.996078, 0.996078 ; White
0.000000, 0.686275, 0.937255 ; C
0.925490, 0.149020, 0.560784 ; M
1.000000, 0.949020, 0.066667 ; Y
0.215686, 0.203922, 0.207843 ; K
0.243137, 0.247059, 0.584314 ; CM
0.000000, 0.658824, 0.349020 ; CY
0.066667, 0.176471, 0.215686 ; CK
0.929412, 0.196078, 0.215686 ; MY
0.215686, 0.101961, 0.141176 ; MK
0.200000, 0.196078, 0.125490 ; YK
0.266667, 0.266667, 0.274510 ; CMY
0.133333, 0.098039, 0.160784 ; CMK
0.074510, 0.180392, 0.133333 ; CYK
0.215686, 0.121569, 0.113725 ; MYK
0.125490, 0.121569, 0.121569 ; CMYK
```

The Neugebauer algorithm mixes the colors based on the amount of color and the corresponding weighted coefficient. Altering the values for a pure color specifically changes the result for this pure color.

The color transition remains smooth.

Example: The following command selects the neugebauer color management engine.

```
pdfoptimize -cms neugebauer input.pdf output.pdf
```

6.2.4 Resolution Values per Image Type

DPI for Bi-Tonal Images `-dbr ‹dpi›`

DPI for Color Images `-dcr ‹dpi›`

DPI for Monochrome Images `-dmr ‹dpi›`

The target resolution values for down-sampling images can be set individually for different types of images.

Note: This option is affected when setting a profile (`-pr`).

Parameter:

`‹dpi›` The target resolution in DPI. The default values for `‹dpi›` are as follows:

Bi-Tonal Images	`200`
Color Images	`150`
Monochrome Images	`150`

6.2.5 Threshold Values per Image Type

DPI for bi-tonal images `-dbt ‹dpi›`

DPI for color images `-dmt ‹dpi›`

DPI for monochrome images `-dct ‹dpi›`

The threshold values above which down-sampling an image is activated can be set with these options.

Note: This option is affected when setting a profile (`-pr`).

Parameter:

`‹dpi›` The threshold in DPI. A value of `-1` indicates that all images of this type are excluded from down-sampling.

Default: `-1`.

6.2.6 `-dr` Resolution in DPI

Resolution in DPI `-dr ‹dpi›`

Set the target resolution after re-sampling in dots per inch (DPI). Only those images with a resolution value higher than the threshold value, which is set with option `-dt`, will be processed. The default target resolution is 150 DPI.

Pre-blended images, images with a color key mask, masks, and soft mask images are not re-sampled.

Note: This option is affected when setting a profile (`-pr`).

Example: In order to down-sample all raster images with a resolution greater than 150 DPI to 75 DPI, apply the following

```
pdfoptimize -dt 150 -dr 75 input.pdf output.pdf
```

6.2.7 `-dt` Threshold in DPI

Threshold in DPI `-dt ‹dpi›`

This option defines the minimum resolution an image must have to be optimized. The threshold value for down-sampling raster images is used in conjunction with the option `-dr`, which sets the actual target resolution for those down-sampled images.

The threshold resolution must be equal or higher than the target resolution. If the value is set to `-1`, down-sampling is turned off. This is the default.

Note: This option is affected when setting a profile (`-pr`).

Example: Select the "`web`" profile and down-sample all raster images with an original resolution higher or equal to 150 DPI to a new resolution of 75 DPI.

```
pdfoptimize -pr web -dt 150 -dr 75 input.pdf output.pdf
```

Example: Select the "`web`" profile but disable down-sampling of images.

```
pdfoptimize -pr web -dt -1 input.pdf output.pdf
```

If the size (in terms of bytes) of the re-sampled image is larger than its original size, the original image is kept instead.

6.2.8 `-fb` Compression Types for Bi-tonal Images

Compression Types for Bi-tonal Images `-fb ‹compr›`

This option affects only bi-tonal (black and white) images. The option `-fb` is followed by a comma-separated list of numerical values (no spaces allowed in the list). The following values are possible:

Note: This option is affected when setting a profile (`-pr`).

Bi-Tonal Compression

Value	Compression Filter
`0`	RAW data
`2`	Flate (ZIP) compression
`3`	Lempel-Ziv-Welch (LZW) compression
`4`	CCITT Fax Group 3 compression
`5`	CCITT Fax Group 3 2D compression
`6`	(default) CCITT Fax Group 4 compression
`7`	JBIG2 compression
`10`	Take the compression type from the original image in the input PDF
`-1`	Exclude bi-tonal images from processing

Example: To let the 3-Heights™ PDF Optimizer Shell try CCITT Group 3 compression, JBIG2 compression and the compression that is used in the source image of the original file use the following command

```
pdfoptimize -fb 3,7,10 input.pdf output.pdf
```

The above command makes the 3-Heights™ PDF Optimizer Shell go through all bi-tonal images and processes each image individually as follows. All the given compression algorithms are executed. If the input image has a compression different from CCITT Fax Group 3 and JBIG2, then the compression of the input image is also executed. As a result, several candidate versions are obtained. Now a choice is made among all these versions (including the original image) based on the size in bytes. The smallest candidate is chosen and used in the output document.

6.2.9 `-fc` Compression Types for Color and Grayscale Images

Compression Types for Color and Grayscale Images `-fc ‹compr›`

This option affects normal color images (RGB and CMYK) as well as grayscale (monochrome) images. The option `-fc` is followed by a comma-separated list of numerical values (no spaces allowed in the list). The following values are possible:

Note: This option is affected when setting a profile (`-pr`).

Color/Monochrome Compression

Value	Compression Filter
`0`	RAW data
`1`	(default) DCT(JPEG) compression
`2`	Flate (ZIP) compression
`8`	JPEG2000 compression
`9`	Perform MRC optimization (See Mixed Raster Content (MRC) Optimization for Images)
`10`	Take the compression type from the original image in the input PDF
`-1`	Exclude continuous images from processing

Example: To let the 3-Heights™ PDF Optimizer Shell try JPEG compression, JPEG2000 compression and the compression that is used in the source image of the original file use the following command.

```
pdfoptimize -fc 1,8,10 input.pdf output.pdf
```

The above command makes the 3-Heights™ PDF Optimizer Shell go through all color and grayscale images and processes each image individually as follows. All the given compression algorithms are executed. If the input image has a compression different from JPEG and JPEG2000, then the compression of the input image is also executed. As a result, several candidate versions are obtained. Now a choice is made among all these versions and the original image based on the size in bytes. The smallest candidate is chosen and used in the output document.

6.2.10 `-ff` Force Re-Compression

Force Re-Compression `-ff`

If this option is set, then images are always re-compressed, i.e., the original image is never used as a candidate for inclusion in the output document. If not set (default), then images are only re-compressed if the resulting image is smaller than the original, i.e. occupies less bytes to store in the file.

Note: This option is affected when setting a profile (`-pr`).

6.2.11 `-fi` Compression Types for Indexed (Paletted) Images

Compression Types for Indexed (Paletted) Images `-fi ‹compr›`

This affects only images with an indexed color space. This type of color space is sometimes used for color graphics and logos. The option `-fc` is followed by a comma-separated list of numerical values (no spaces allowed in the list). The following values are possible:

Note: This option is affected when setting a profile (`-pr`).

Indexed Compression Types

Value	Compression Filter
0	RAW data
2	Flate (ZIP) compression
3	Lempel-Ziv-Welch (LZW) compression
10	Take the compression type from the original image in the input PDF

Example: To let the 3-Heights™ PDF Optimizer Shell try Flate compression and LZW compression use the following command:

```
pdfoptimize -fi 2,3 input.pdf output.pdf
```

The above command makes the 3-Heights™ PDF Optimizer Shell go through all images with indexed color space and processes each image individually as follows. All the given compression algorithms are executed. As a result, two candidate versions are obtained. Now a choice is made among these two versions and the original image based on the size in bytes. The smallest candidate is chosen and used in the output document.

6.2.12 -fn File Name

File Name `-fn ‹file.pdf›`

The intension of this option is to provide support for file names that start with a dash character and would therefore cause a parameter error.

The parameter after the option `-fn` is a file name. It can optionally also be used for file names not starting with a dash character.

Example:

```
pdfoptimize -fn -input.pdf output.pdf
```

6.2.13 -ft Force Compression Types

Force Compression Types `-ft`

If this option is set, then re-compression of images is forced if an image in the input PDF has a compression type that differs from the compression types given in `-fb`, `-fc`, or `-fi`. Use this option if you want to allow only the given compression types for images in the output PDF.

Note: This option is affected when setting a profile (`-pr`).

6.2.14 -fv Minimum PDF Version

Minimum PDF Version `-fv ‹1.x›`

This option allows to set the minimum PDF version of the created PDF output file. Supported values are 1.1 to 1.7. (PDF 1.4 corresponds to Acrobat 5, PDF 1.5 to Acrobat 6, etc.) There are three parameters that influence the version of the PDF output file:

- The value set using the option `-fv`.
- The PDF version of the input file.
- Other settings in the optimization (JBIG2 requires PDF 1.4, JPEG2000 requires PDF 1.5) The maximum of the three values above sets the PDF version in the output file.

Example: 1. Input PDF is version 1.5 and the following setting is used

```
pdfoptimize -fv 1.4 input.pdf output.pdf
```

The output file is PDF version 1.5.

Example: 2. Input PDF is version 1.4 or lower and the following setting is used

```
pdfoptimize -fv 1.4 input.pdf output.pdf
```

The output file is PDF version 1.4.

Example: Input PDF is version 1.3 and the following setting is used

```
pdfoptimize -fv 1.4 -fc 8 input.pdf output.pdf
```

If input.pdf contains color images to which JPEG2000 compression is applied, the output file will be version 1.5. Otherwise it will be version 1.4.

6.2.15 -h Dithering Mode for Bi-Tonal Images

Dithering Mode for Bi-Tonal Images `-h ‹mode›`

This option enables or disables dithering when down-sampling bi-tonal images.

Note: This option is affected when setting a profile (`-pr`).

‹mode›	Description
0	No dithering
1	Floyd-Steinberg dithering algorithm

Some bi-tonal images try to evoke the impression of different levels of gray by randomly setting pixels to black. If dithering is applied during down-sampling then the gray levels of such images are preserved better. If dithering is switched off then lines (e.g. text glyphs) are preserved better.

6.2.16 -id Set Value in the Document Information Dictionary

Set Value in the Document Information Dictionary `-id ‹key› ‹value›`

Set the value of an document information dictionary entry `‹key›`. Popular entries specified in the PDF Reference 1.7 are `"Title"`, `"Author"`, `"Subject"`, `"Creator"` (sometimes referred to as Application), and `"Producer"` (sometimes referred to as PDF Creator). If the entry already exists then the previous entry is overwritten. If the key corresponds to a standard metadata key then the XMP metadata is updated accordingly.

Example: Overwrite the default producer:

```
pdfoptimize -id Producer "MyProgram 1.2" input.pdf output.pdf
```

6.2.17 -lf List Fonts

List Fonts `-lf`

List all fonts and their properties.

List Fonts

Parameter	Description	Example
FontName	The name of the font. Subsetting-prefixes are not listed as name of the font.	“Arial-BoldMT”, “Verdana”
FontType	The Font Type	TrueType, Type1
Encoding	The encoding of the font, see examples.	Difference Encoding, IntrinsicEncoding, MacRomanEncoding, SymbolEncoding, WinAnsiEncoding
IsCID	Whether the font is a CID font (Character Identifier Font) or not.	CID, Non-CID

List Fonts

`IsEmbedded`	Whether the font has an embedded font program or not.	Embedded, Non-embedded
`IsSubsetted`	Whether a font program is subsetted or not. This value is only set for fonts, which have an embedded font program.	Subsetted, Non-Subsetted
`FileName`	The file name of the font program. This is the name under which the font is saved to file in case the option `-xf` is applied. For all non-embedded fonts, there is no file name available (N/A).	`fnt12.ttf`, `fnt2477.cff`, N/A

Example: The following command lists all fonts of a PDF document

```
pdfoptimize -lf input.pdf
FontName, FontType, Encoding, IsCID, IsEmbedded, IsSubsetted, Filename
"Arial-BoldMT", TrueType, MacRomanEncoding, Non-CID, Non-embedded, N/A,
"Arial-BlackItalic", TrueType, MacRomanEncoding, Non-CID, Non-embedded, N/A,
"Verdana", TrueType, WinAnsiEncoding, Non-CID, Embedded, Subsetted, fnt38.ttf
```

The first line in the above example is the actual command, the following lines list the output.

See also option `-xf` for extracting fonts.

6.2.18 `-li` List Images

List Images `-li`

List all images and their properties.

List Images

Parameter	Description	Example
ObjectNumber	The PDF object number	9
Width	The width of the image in pixel.	400
Height	The height of the image in pixel.	589
BitsPerComponent	The number of bits that are used to represent one component. This number is in most cases either 1 (bi-tonal) or 8 (RGB, CMYK, Gray).	8

List Images

ColorSpace	The color space of the image.	DeviceCMYK, DeviceRGB, DeviceGray, ICCBased, Indexed
Resolution	The resolution in dots per inch (DPI).	96
Filter	The compression filter.	DCTDecode, FlateDecode
ImageSize	The uncompressed image size.	706800
CompressedSize	The compressed image size.	28172
CompressionRatio	The ratio compressed image size divided by uncompressed images size. The smaller this value, the higher the compression.	3.99%
FileName	The file name of the image. This is the name under which the image is saved to file in case the option `-xi` is applied.	img9.tif

Example: The following command lists all images in the file input.pdf. In this case there is one image.

```
pdfoptimize -li input.pdf
ObjectNumber, Width, Height, BitsPerComponent, ColorSpace, Resolution,
Filter, ImageSize, CompressedSize, CompressionRatio, FileName
9, 400, 589, 8, ICCBased, 96, DCTDecode, 706800, 28172, 3.99%, img9.tif
```

See also option `-xf` for extracting fonts.

6.2.19 `-lk` Set License Key

Set License Key `-lk ‹key›`

Pass a license key to the application at runtime instead of using one that is installed on the system.

```
pdfoptimize -lk X-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX-XXXXX ...
```

This is required in an OEM scenario only.

6.2.20 `-m` Merge Embedded Font Programs

Merge Embedded Font Programs `-m`

Font programs can be merged, if they originate from the same font, e.g. they are of the same type, have the same name and encoding. Merging of Type1 (PostScript) and TrueType fonts is supported.

Note: This option is affected when setting a profile (`-pr`).

6.2.21 `-ml` Compression Type for MRC Layers

Compression Type for MRC Layers `-ml ‹compr›`

This option affects only MRC foreground and background layers. The option is followed by a single numerical value indicating the compression type to use for MRC foreground and background layers. For possible values see Color/Monochrome Compression. The default is 8 (JPEG2000 compression).

See also Mixed Raster Content (MRC) Optimization for Images.

Note: This option is affected when setting a profile (`-pr`).

6.2.22 `-mlq` Image Quality for MRC Layers

Image Quality for MRC Layers `-mlq ‹q›`

This option affects only MRC foreground and background layers. The option is followed by a numerical value between 0 and 100 to be used as the image quality for MRC foreground and background layers when using a lossy compression for these layers. The default is 10.

See also Mixed Raster Content (MRC) Optimization for Images.

Note: This option is affected when setting a profile (`-pr`).

6.2.23 `-mlr` Resolution in DPI for MRC Layers

Resolution in DPI for MRC Layers `-mlr ‹dpi›`

This option affects only MRC foreground and background layers. The option is followed by a numerical value that indicates the target resolution in DPI of MRC layers after down-sampling. The default is `70`.

See also Mixed Raster Content (MRC) Optimization for Images.

Note: This option is affected when setting a profile (`-pr`).

6.2.24 `-mm` Compression Type for the MRC Mask

Compression Type for the MRC Mask `-mm ‹compr›`

This option affects only MRC masks. The option is followed by a single numerical value indicating the compression type to use for MRC masks. For possible values see Bi-Tonal Compression. The default is 6 (CCITT Fax Group 4 compression).

Note: This option is affected when setting a profile (`-pr`).

See also Mixed Raster Content (MRC) Optimization for Images.

6.2.25 `-mp` Compression Type for MRC Cut-Out Pictures

Compression Type for MRC Cut-Out Pictures `-mp ‹compr›`

This option affects only cut-out images when doing MRC optimization. The option is followed by a single numerical value indicating the compression type to use for MRC cut-out pictures. For possible values see Color/Monochrome Compression. The default is 1 (JPEG compression).

Note: This option is affected when setting a profile (`-pr`).

See also Mixed Raster Content (MRC) Optimization for Images.

6.2.26 `-o` Owner Password

Owner Password `-o ‹owner›`

The owner password is required to change the security settings of the document. In order to apply permission flags, an owner password must be set. Permission flags are set with the switch `-p`.

Example: Encrypt a document and set the owner password to "owner".

```
pdfoptimize -o owner input.pdf output.pdf
```

6.2.27 `-oc` Clip Images

Clip Images `-oc`

Images in PDF documents can be clipped. This means that only part of the image is visible, whilst the rest is hidden. The option `-oc` detects these images, reduces their size the area that is actually displayed and replaces the original image by the reduced image. Pre-blended images are not clipped.

Note: This option is affected when setting a profile (`-pr`).

Setting `-oc` activates the `-od` option.

6.2.28 `-od` Optimize Resources

Optimize Resources `-od`

Optimize the resources of the PDF, such as images, color spaces, or fonts. If set, unused resources are removed. Also content streams are re-built.

Note: This option is affected when setting a profile (`-pr`).

6.2.29 `-ol` Linearize Only

Linearize Only `-ol`

Do not apply any optimizations, but linearize the file. This can be significantly faster than the option `-ow`. See `-ow` for more information.

When this option is set then all other options are disabled except `-o`, `-u`, `-pw`, and `-p`.

6.2.30 `-or` Remove Redundant Objects

Remove Redundant Objects `-or`

This option removes redundant PDF objects. I.e. it identifies duplicates objects in the input document and collapses them into single objects in the output document.

Note: This option is affected when setting a profile (`-pr`).

6.2.31 `-ow` Optimize for the Web

Optimize for the Web `-ow`

Linearize the PDF output file, i.e. optimize file for fast web access.

Note: This option is affected when setting a profile (`-pr`).

A linearized document has a slightly larger file size than a non-linearized file and provides the following main features:

- When a document is opened in a PDF viewer of a web browser, the first page can be viewed without downloading the entire PDF file. In contrast, a non-linearized PDF file must be downloaded completely before the first page can be displayed.
- When another page is requested by the user, that page is displayed as quickly as possible and incrementally as data arrives, without downloading the entire PDF file.

Note: In order to make use of a linearized PDF file, the PDF must reside as a “file” on the web-server. It must not be streamed.

6.2.32 `-p` Permission Flags

Permission Flags `-p ‹flags›`

This option sets the permission flags. It is only usable in combination with encrypted documents, i.e. an owner password must be set. By default all permissions are granted. The permissions that can be granted are listed below.

Permission Flags

`p`	allow printing (low resolution)
`m`	allow changing the document
`c`	allow content copying or extraction
`o`	allow commenting
`f`	allow filling of form fields
`s`	allow content extraction for accessibility
`a`	allow document assembly
`d`	allow high quality printing
`-1`	(default) allow everything (all permissions are granted)
`0`	allow nothing (no permissions are granted)

The value `0` cannot be combined with other flags. The value `-1` is the default, it cannot be set explicitly. In order to combine multiple permissions concatenate them to one string.

Example: The following command sets the owner password to "owner" and the permission flags to "allow printing in low resolution" and "allow form filling".

```
pdfoptimize -o owner -p pf input.pdf output.pdf
```

Example: "High quality printing" requires the standard printing flag to be set too.

```
pdfoptimize -o owner -p pd input.pdf output.pdf
```

For further information about the permission flags, see PDF Reference 1.7 Section 3.5.2.

6.2.33 `-pr` Set an Optimization Profile

Set an Optimization Profile `-pr ‹profile›`

With this option one of the predefined optimization profiles can be set. Section Profile Settings tabulates the configuration values for all profiles. If a profile is set then all the tabulated configuration parameters are set to their respective values. Configuration parameters not listed in this table are left unchanged.

Parameter:

`‹profile›` The profile. Currently, the following profiles can be selected:

web Optimization for the Internet.

print Optimization for print.

max Optimization for maximal memory size reduction.

mrc MRC (Mixed Raster Content) optimization of images. See Mixed Raster Content (MRC) Optimization for Images.

One way of quickly arriving at a specific setting is to set a profile and adapt individual configuration parameters.

Example: Use the "web" profile, but inhibit the down-sampling of bi-tonal images.

```
pdfoptimize -pr web -dbt -1 input.pdf output.pdf
```

6.2.34 -pw Read an Encrypted PDF File

Read an Encrypted PDF File `-pw ‹password›`

A PDF document that has a user password (the password to open the document) can only be processed when either the user or the owner password is provided. The password can be provided using the option -pw followed by the password.

Example: The input PDF document is encrypted with a user password. Either the user or the owner password of the input PDF is "mypassword". The command to process such an encrypted file is:

```
pdfoptimize -pw mypassword input.pdf output.pdf
```

When a PDF is encrypted with a user password and the password is not provided or is incorrect, the 3-Heights™ PDF Optimizer Shell cannot read and process the file. Instead it will generate the following error message:

```
Password wasn't correct.
```

6.2.35 -q Compression Quality

Compression Quality `-q ‹quality›`

Set the compression quality index for lossy compression methods. This option only applies to JPEG, JPEG2000 and JBIG2 images. A lower value results in a smaller file size but the images are of poorer visual quality. A higher value results in better visual quality, but also a larger file size.

Note: This option is affected when setting a profile (-pr).

The supported values range from 1 (lowest) to 100 (highest). The default is 75. For image compressions that support lossless compression (JPEG2000), a value of 100 corresponds to lossless compression, any other value represents lossy compression. For JBIG2, compression is always lossless irrespective of the quality index set. JPEG compression is always lossy.

Example: The following selects the "web" optimization profile and sets the quality index to 50. All images types which support the quality parameter are recompressed with this quality index.

```
pdfoptimize -pr web -q 50 input.pdf output.pdf
```

6.2.36 -rc Reduce Color Complexity of Images

Reduce Color Complexity of Images `-rc`

This option is used to enable color complexity reduction of images. (See also Provided Features for Optimizing Images.)

Note: This option is affected when setting a profile (`-pr`).

If enabled then images with device color spaces (DeviceRGB, DeviceCMYK, or DeviceGray) and indexed images with a device color space as base color space are analyzed and if possible converted as follows:

- An image with DeviceRGB or DeviceCMYK color space in which all pixels are gray is converted to a grayscale image with DeviceGray color space.
- An image that contains only black and white pixels is converted into a bitonal image.
- An image in which all the pixels have the same color is down-sampled to one pixel.

Furthermore, images' masks and soft masks are optimized as follows:

- A soft mask that contains only black and white pixels is converted to a mask.
- A (soft) mask that is opaque is removed.

6.2.37 -rf Remove Embedded Font Program

Remove Embedded Font Program `-rf ‹font›`

This option makes the 3-Heights™ PDF Optimizer Shell remove the embedded font program for the given `‹font›`. This option can be given several times to remove several font programs.

Warning: The output document may not display correctly on certain systems.

6.2.38 -rs Remove Embedded Standard Fonts

Remove Embedded Standard Fonts `-rs`

This option enables the removal of the font programs of all embedded standard fonts, such as Arial, Courier, CourierNew, Helvetica, Symbol, Times, TimesNewRoman and ZapfDingbats. (A complete list is given below.) The fonts are replaced with one of the 14 PDF Standard Fonts, all of which have no associated font program. Un-embedding a font decreases the file size.

Note: This option is affected when setting a profile (`-pr`).

A PDF Viewer must be able to display these 14 PDF Standard Fonts correctly.
Therefore using this option usually should not visually alter the PDF when it is displayed.

Un-embedding the font works based on the font's Unicode information. I.e. the un-embedded font's characters are mapped to those of the original font with the same Unicode. Therefore, only fonts with Unicode information will be un-embedded by the 3-Heights™ PDF Optimizer Shell. However, if a font's Unicode information is not correct, un-embedding may lead to visual differences. Whether or not a font's Unicode information is correct can be verified by extracting text that uses the font. Suitable tools for this purpose are for instance the 3-Heights™ PDF Extract Tool or an interactive PDF viewer.

If the extracted text is meaningful, the font's Unicode information is correct and unembedding of the font will not lead to visual differences.

List of Candidate Font Base Names for Removing

- Arial
- Arial,Bold
- Arial,BoldItalic
- Arial,Italic
- Arial-Bold
- Arial-BoldItalic
- Arial-BoldItalicMT
- Arial-BoldMT
- Arial-Italic
- Arial-ItalicMT
- ArialMT
- Courier
- Courier,Bold
- Courier,BoldItalic
- Courier,BoldOblique
- Courier,Italic
- Courier,Oblique
- Courier-Bold
- Courier-BoldOblique
- Courier-Oblique
- CourierNew
- CourierNew,Bold
- CourierNew,BoldItalic
- CourierNew,Italic
- CourierNew-Bold
- CourierNew-BoldItalic
- CourierNew-Italic
- CourierNewPS-BoldItalicNT
- CourierNewPS-BoldMT
- CourierNewPS-ItalicMT
- CourierNewPSMT
- Helvetica
- Helvetica,Bold
- Helvetica,BoldItalic
- Helvetica,BoldOblique
- Helvetica,Italic
- Helvetica,Oblique
- Helvetica-Bold
- Helvetica-BoldItalic
- Helvetica-BoldOblique
- Helvetica-Italic
- Helvetica-Oblique
- Symbol
- SymbolMT
- Times,Bold
- Times,BoldItalic
- Times,Italic
- Times-Bold
- Times-BoldItalic
- Times-Italic
- Times-Roman
- TimesNewRoman
- TimesNewRoman,Bold
- TimesNewRoman,BoldItalic
- TimesNewRoman,Italic
- TimesNewRoman-Bold
- TimesNewRoman-BoldItalic
- TimesNewRoman-Italic
- TimesNewRomanPS
- TimesNewRomanPS-Bold
- TimesNewRomanPS-BoldItalic
- TimesNewRomanPS-BoldItalicMT
- TimesNewRomanPS-BoldMT
- TimesNewRomanPS-Italic
- TimesNewRomanPS-ItalicMT
- TimesNewRomanPSMT
- ZapfDingbats

6.2.39 `-s` Subset Fonts

Subset Fonts	`-s`

Embedded fonts can be subsetted. Subsetting refers to only storing those character glyphs of the font that are actually used. Unused character glyphs are removed. The advantage is that the file size can be reduced this way (in particular for Asian fonts).

The downside is that if text is to be edited, only the characters of the subsetted font can be used.

Note: This option is affected when setting a profile (`-pr`).

6.2.40 Strip the File

Remove certain elements of the PDF file. Please refer to the PDF Reference 1.7 for more details on these elements.

The following parts of a PDF can be stripped:

> **Note:** This option is affected when setting a profile (`-pr`).

Strip article threads `-sa`

Flatten and strip form fields and annotations `-sf`

Flatten and strip form fields `-sff`

Flatten and strip link annotations `-sfl`

Flatten and strip other annotations `-sfa`

Strip alternate images (variant representations of the base image) `-si`

Strip the document's output intents `-so`

Strip meta data `-sm`

Strip page piece info (private application data) `-sp`

Strip document structure tree (incl. markup) `-ss`

Strip embedded thumbnails `-st`

Strip spider (web capture) info `-sw`

Strip everything (all of the above) `-se`

Note that `-sf` implies `-sfa`, `-sff`, and `-sfl`. Likewise, `-se` implies all other strip options listed.

6.2.41 `-sfs` Flatten Appearances of Signature Fields

Flatten Appearances of Signature Fields `-sfs`

A signature in a PDF consist of two parts:

a. The invisible digital signature in the PDF.
b. The visual appearance that was attributed to the signature.

Part (a) can be used by a viewing application, to verify that a document has not changed since it has been signed and report this to the user. Part (b) is merely a "decorative" element on the page without further significance.

When optimizing a PDF, the PDF is altered and hence the digital signature is broken. Therefore, the 3-Heights™ PDF Optimizer Shell removes all signatures, including parts (a) and (b).

If the option `sfs` is set, then digital signatures (parts (a)) are still removed, but their visual appearances (parts (b)) are flattened. I.e. the latter are retained and drawn as non-editable graphic onto the page.

> **Note:** The resulting PDF can be misleading as it visually appears to be signed, but it has no digital signature and hence, a viewer application does not report any broken signature. In most cases, such a behavior is undesirable.

6.2.42 `-u` User Password

User Password `-u ‹user›`

Set the user password of the document. If a document which has a user password is opened for any purpose (such as viewing, printing, editing), either the user or the owner password must be provided.

Someone who knows the user password is able to open and read the document. Someone who knows the owner password is able to open, read and modify (e.g. change passwords) the document. A PDF document can have none, either, or both passwords.

Example: Encrypt a document with a user and an owner password.

```
pdfoptimize -u userpassword -o ownerpassword
```

6.2.43 `-v` Verbose Mode

Verbose Mode `-v`

This option enables the verbose mode. In the verbose mode, the individual steps performed by the 3-Heights™ PDF Optimizer Shell are displayed.

6.2.44 `-xf` Extract Fonts

Extract Fonts `-xf`

This option enables the extraction of embedded fonts into to files. Only embedded fonts can be extracted, non-embedded fonts are not affected. Be aware that due to copyright reasons, the extracts fonts are not installable. The generated files are stored in the current directory and are named as following:

- A TrueType font file is named: `fnt‹objno›.ttf`
- A Type 1 font file is named: `fnt‹objno›.pfb`
- A CFF font file is named: `fnt‹objno›.cff`

In the above, `‹objno›` corresponds to the object number of the font in the PDF document.

6.2.45 `-xi` Extract Images

Extract Images `-xi`

This option extracts the images from a PDF document and automatically stores them as TIFF or JPEG.

The images are stored in the current directory and are named as following:

- `img‹objno›.jpg` for images with JPEG compression
- `img‹objno›.tif` for any other type of image

In the above, `‹objno›` corresponds to the object number of the image in the PDF document. This number can also be retrieved with the option `-li`.

6.3 Return Codes

All return codes other than `0` indicate an error in the processing.

Return Codes

Value	Description
`0`	Success.
`1`	Couldn’t open input file.
`2`	PDF output file could not be created.
`3`	Error with given option, e.g. too many parameters.
`4`	Linearization failed.
`10`	License error, e.g. invalid license key.

7 Tips, Tricks and Troubleshooting

It is recommended to use an optimization profile `-pr` and adapt it to your needs.

7.1 The Output File is Still Too large

First and foremost it is important to understand what kind of content there is in the document. There is no point in trying to optimizing fonts when the document contains scanned images only. Document properties, such as embedded fonts and images can be listed using the corresponding listing functions (`-li`, `-lf`).

Second, it is not possible to compress a document arbitrarily without loss of information.

7.1.1 Images

- Try setting a lower threshold and a lower DPI for the images.

Example: Use the "`web`" profile and rescale all images with a DPI greater than 72 DPI to 50 DPI

```
pdfoptimize -pr web -dt 72 -dr 50 input.pdf output.pdf
```

- Try reducing the quality of the JPEG and JPEG2000 images by setting `-q`.

Example: Use the "`web`" profile and set the quality index to 60

```
pdfoptimize -pr web -q 60 input.pdf output.pdf
```

7.1.2 Fonts

- Apply sub-setting to fonts using `-s`. This means all glyphs of characters that are unused are removed from the font.
- Merge font programs using `-m`. Multiply occurring glyphs in compatible font programs are then merged into one glyph.
- Remove non-symbolic embedded fonts using `-rs`. Keep in mind that the appearance when rendering a PDF document with non-embedded non-PDF Standard Fonts is unpredictable:

Note: While sub-setting and merging font programs is enabled in most optimization profiles, removing standard fonts is not. (See Profile Settings.)

7.2 The Output File Is Larger Than the Input File

- The 3-Heights™ PDF Optimizer Shell also repairs corrupt documents to a certain extent. This means if relevant data is missing it is recovered. This could possibly lead to a larger file size.
- If linearization is applied, there is information added to the document. This information contains hints for the browser plug-in, and allows it to specifically download only those objects relevant for displaying a certain page. The linearization information can increase the file size by about 1 to 10%.
- The input file may contain compressed object streams. These are currently not re-constructed in the output document.

7.3 The Selected Compression Type is not Applied

- Not all compression types can be applied to all types of images. Check the tables Bi-Tonal Compression, Color/Monochrome Compression, and Indexed Compression Types.
- The optimization is only applied if it reduces the file size, therefore an image usually is not re-compressed with a new compression that uses more disc space than the original compression. This behavior can, however, be switched off with `-ff`.

7.4 The Output Document Is not Encrypted

In order to encrypt the output document, set an owner password using the switch `-o` and permission flags using the switch `-p`.

Example: Set the owner password to "mypassword" and do not grant any permissions:

```
pdfoptimize -o mypassword -p 0 input.pdf output.pdf
```

It is not possible to inherit the owner or user password or the permission flags from the input document.

8 Licensing, Copyright, and Contact

PDF Tools AG is a world leader in PDF (Portable Document Format) software, delivering reliable PDF products to international customers in all market segments.

PDF Tools AG provides server-based software products designed specifically for developers, integrators, consultants, customizing specialists and IT-departments. Thousands of companies worldwide use our products directly and hundreds of thousands of users benefit from the technology indirectly via a global network of OEM partners. The tools can be easily embedded into application programs and are available for a multitude of operating system platforms.

Licensing and Copyright

Contact

PDF Tools AG
Kasernenstrasse 1
8184 Bachenbülach
Switzerland
http://www.pdf-tools.com
pdfsales@pdf-tools.com