Name

gropdf − *groff* output driver for Portable Document Format

Synopsis

gropdf [**−dels**] [**−F** *font-directory*] [**−I** *inclusion-directory*] [**−p** *paper-format*] [**−u** [*cmap-file*]] [**−y** *foundry*] [*file* ...]

gropdf −−help

gropdf −v
gropdf −−version

Description

The GNU *roff* PDF output driver translates the output of *troff*(1) into Portable Document Format. Normally, *gropdf* is invoked by *groff*(1) when the latter is given the "**−T pdf**" option. (In this installation, **ps** is the default output device.) Use *groff*'s **−P** option to pass any options shown above to *gropdf*. If no *file* arguments are given, or if *file* is "−", *gropdf* reads the standard input stream. Output is written to the standard output stream.

See section "Font installation" below for a guide to installing fonts for *gropdf*.

Options

−−help displays a usage message, while **−v** and **−−version** show version information; all exit afterward.

−d Include debug information as comments within the PDF. Also produces an uncompressed PDF.

−e Forces *gropdf* to embed *all* fonts (even the 14 base PDF fonts).

−F *dir* Prepend directory *dir*/dev*name* to the search path for font, and device description files; *name* is the name of the device, usually **pdf**.

−I *dir* Search the directory *dir* for files named in **\X'pdf: pdfpic'** device control commands. **−I** may be specified more than once; each *dir* is searched in the given order. To search the current working directory before others, add "**−I .**" at the desired place; it is otherwise searched last.

−l Orient the document in landscape format.

−p *paper-format*
Set the physical dimensions of the output medium. This overrides the **papersize**, **paperlength**, and **paperwidth** directives in the *DESC* file; it accepts the same arguments as the **papersize** directive. See *groff_font*(5) for details.

−s Append a comment line to end of PDF showing statistics, i.e. number of pages in document. Ghostscript's **ps2pdf** complains about this line if it is included, but works anyway.

−u [*cmap-file*]
gropdf normally includes a ToUnicode CMap with any font created using *text.enc* as the encoding file, this makes it easier to search for words which contain ligatures. You can include your own CMap by specifying a *cmap-file* or have no CMap at all by omitting the argument.

−y *foundry*
Set the foundry to use for selecting fonts of the same name.

Usage

The input to *gropdf* must be in the format output by *troff*(1). This is described in *groff_out*(5). In addition, the device and font description files for the device used must meet certain requirements: The resolution must be an integer multiple of 72 times the **sizescale**. The **pdf** device uses a resolution of 72000 and a sizescale of 1000.

The device description file must contain a valid paper format; see *groff_font*(5). *gropdf* uses the same Type 1 Adobe PostScript fonts as the **grops** device driver. Although the PDF Standard allows the use of other font types (like TrueType) this implementation only accepts the Type 1 PostScript font. Fewer Type 1 fonts are supported natively in PDF documents than the standard 35 fonts supported by **grops** and all PostScript printers, but all the fonts are available since any which aren't supported natively are automatically embedded in the PDF.

gropdf supports the concept of foundries, that is different versions of basically the same font. During install a *Foundry* file controls where fonts are found and builds *groff* fonts from the files it discovers on your system.

Each font description file must contain a command

internalname *psname*

which says that the PostScript name of the font is *psname*. Lines starting with **#** and blank lines are ignored. The code for each character given in the font file must correspond to the code in the default encoding for the font. This code can be used with the **\N** escape sequence in **troff** to select the character, even if the character does not have a *groff* name. Every character in the font file must exist in the PostScript font, and the widths given in the font file must match the widths used in the PostScript font.

Note that *gropdf* is currently only able to display the first 256 glyphs in any font. This restriction will be lifted in a later version.

gropdf can automatically include the downloadable fonts necessary to print the document. Fonts may be in PFA or PFB format.

Any downloadable fonts which should, when required, be included by *gropdf* must be listed in the file */usr/local/share/groff/1.23.0/font/devpdf/download*; this should consist of lines of the form

foundry font filename

where *foundry* is the foundry name or blank for the default foundry. *font* is the PostScript name of the font, and *filename* is the name of the file containing the font; lines beginning with **#** and blank lines are ignored; fields must be separated by tabs (spaces are **not** allowed); *filename* is searched for using the same mechanism that is used for *groff* font metric files. The *download* file itself is also sought using this mechanism. Foundry names are usually a single character (such as 'U' for the URW foundry) or empty for the default foundry. This default uses the same fonts as *ghostscript* uses when it embeds fonts in a PDF file.

In the default setup there are styles called **R**, **I**, **B**, and **BI** mounted at font positions 1 to 4. The fonts are grouped into families **A**, **BM**, **C**, **H**, **HN**, **N**, **P**, and **T** having members in each of these styles:

AR AvantGarde-Book
AI *AvantGarde-BookOblique*
AB **AvantGarde-Demi**
ABI ***AvantGarde-DemiOblique***
BMR Bookman-Light
BMI *Bookman-LightItalic*
BMB **Bookman-Demi**
BMBI ***Bookman-DemiItalic***
CR Courier
CI *Courier-Oblique*
CB **Courier-Bold**
CBI ***Courier-BoldOblique***
HR Helvetica
HI *Helvetica-Oblique*
HB **Helvetica-Bold**
HBI ***Helvetica-BoldOblique***
HNR Helvetica-Narrow
HNI *Helvetica-Narrow-Oblique*
HNB **Helvetica-Narrow-Bold**
HNBI ***Helvetica-Narrow-BoldOblique***
NR NewCenturySchlbk-Roman
NI *NewCenturySchlbk-Italic*
NB **NewCenturySchlbk-Bold**

NBI ***NewCenturySchlbk-BoldItalic***
PR Palatino-Roman
PI *Palatino-Italic*
PB **Palatino-Bold**
PBI ***Palatino-BoldItalic***
TR Times-Roman
TI *Times-Italic*
TB **Times-Bold**
TBI ***Times-BoldItalic***

There is also the following font which is not a member of a family:

ZCMI *ZapfChancery-MediumItalic*

There are also some special fonts called **S** for the PS Symbol font. The lower case greek characters are automatically slanted (to match the SymbolSlanted font (SS) available to PostScript). Zapf Dingbats is available as **ZD**; the "hand pointing left" glyph (**\[lh]**) is available since it has been defined using the **\X'pdf: xrev'** device control command, which reverses the direction of letters within words.

The default color for **\m** and **\M** is black.

gropdf understands some of the device control commands supported by *grops*(1).

\X'ps: invis'
Suppress output.

\X'ps: endinvis'
Stop suppressing output.

\X'ps: exec gsave currentpoint 2 copy translate *n* **rotate neg exch neg exch translate'**
where *n* is the angle of rotation. This is to support the **align** command in *pic*(1).

\X'ps: exec grestore'
Used by *pic*(1) to restore state after rotation.

\X'ps: exec *n* **setlinejoin'**
where *n* can be one of the following values.

0 = Miter join
1 = Round join
2 = Bevel join

\X'ps: exec *n* **setlinecap'**
where *n* can be one of the following values.

0 = Butt cap
1 = Round cap, and
2 = Projecting square cap

\X'ps: . . . **pdfmark'**
All the *pdfmark* macros installed by using *−m pdfmark* or *−m mspdf* (see documentation in *pdfmark.pdf*). A subset of these macros are installed automatically when you use **−Tpdf** so you should not need to use "**−m pdfmark**" to access most PDF functionality.

gropdf also supports a subset of the commands introduced in *present.tmac*. Specifically it supports:-

PAUSE
BLOCKS
BLOCKE

Which allows you to create presentation type PDFs. Many of the other commands are already available in other macro packages.

These commands are implemented with *groff* X commands:-

\X'ps: exec %%%%PAUSE'
The section before this is treated as a block and is introduced using the current **BLOCK** transition setting (see "**\X'pdf: transition'**" below). Equivalently, **.pdfpause** is available as a macro.

\X'ps: exec %%%%BEGINONCE'
Any text following this command (up to %%%%ENDONCE) is shown only once, the next %%%%PAUSE will remove it. If producing a non-presentation PDF, i.e. ignoring the pauses, see *GROPDF_NOSLIDE* below, this text is ignored.

\X'ps: exec %%%%ENDONCE'
This terminates the block defined by %%%%BEGINONCE. This pair of commands is what implements the .BLOCKS Once/.BLOCKE commands in *present.tmac*.

The *mom* macro package already integrates these extensions, so you can build slides with *mom*.

If you use *present.tmac* with *gropdf* there is no need to run the program *presentps*(1) since the output will already be a presentation PDF.

All other **ps:** tags are silently ignored.

One **\X** device control command used by the DVI driver is also recognised.

\X'papersize=*paper-format***'**
where the *paper-format* parameter is the same as that to the **papersize** directive. See *groff_font*(5). This means that you can alter the page size at will within the PDF file being created by *gropdf*. If you do want to change the paper format, it must be done before you start creating the page.

gropdf supports several more device control features using the **pdf:** tag. Some have counterpart *convenience macros* that take the same arguments and behave equivalently.

\X'pdf: pdfpic *file alignment width height line-length*'
Place an image of the specified *width* containing the PDF drawing from file *file* of desired *width* and *height* (if *height* is missing or zero then it is scaled proportionally). If *alignment* is **−L** the drawing is left-aligned. If it is **−C** or **−R** a *line-length* greater than the width of the drawing is required as well. If *width* is specified as zero then the width is scaled in proportion to the height.

\X'pdf: xrev'
Toggle the reversal of glyph direction. This feature works "letter by letter", that is, each letter in a word is reversed left-to-right, not the entire word. One application is the reversal of glyphs in the Zapf Dingbats font. To restore the normal glyph orientation, repeat the command.

\X'pdf: markstart */ANN-definition***'**
\X'pdf: markend'
Macros that support PDF bookmarks use these calls internally to start and stop (respectively) the placement of the bookmark's *hot spot;* the user will have called "**.pdfhref L**" with the text of the hot spot. Normally, these are never used except from within the *pdfmark* macros.

\X'pdf: marksuspend'
\X'pdf: markrestart'
If you use a page location trap to produce a header or footer, or otherwise interrupt a document's text, you need to use these commands if a PDF *hot spot* crosses a trap boundary; otherwise any text output by the trap will be marked as part of the hot spot. To prevent this error, place these device control commands or their corresponding convenience macros **.pdfmarksuspend** and **.pdfmarkrestart** at the start and end of the trap macro, respectively.

\X'pdf: pagename *name***'**
Assign the current page a *name*. All documents bear two default names, '**top**' and '**bottom**'. The convenience macro for this command is **.pdfpagename**.

\X'pdf: switchtopage *when name***'**
Normally each new page is appended to the end of the document, this command allows following pages to be inserted at a *'named'* position within the document (see pagename command above).

'*when'* can be either '*after*' or '*before*'. If it is omitted it defaults to '*before*'. It should be used at the end of the page before you want the switch to happen. This allows pages such as a TOC to be moved to elsewhere in the document, but more esoteric uses are possible. The convenience macro for this command is **.pdfswitchtopage**.

\X'pdf: transition *feature mode duration dimension motion direction scale bool***'**
where *feature* can be either SLIDE or BLOCK. When it is SLIDE the transition is used when a new slide is introduced to the screen, if BLOCK then this transition is used for the individual blocks which make up the slide.

mode is the transition type between slides:-

Split - Two lines sweep across the screen, revealing the new page. The lines may be either horizontal or vertical and may move inward from the edges of the page or outward from the center, as specified by the *dimension* and *motion* entries, respectively.
Blinds - Multiple lines, evenly spaced across the screen, synchronously sweep in the same direction to reveal the new page. The lines may be either horizontal or vertical, as specified by the *dimension* entry. Horizontal lines move downward; vertical lines move to the right.
Box - A rectangular box sweeps inward from the edges of the page or outward from the center, as specified by the *motion* entry, revealing the new page.
Wipe - A single line sweeps across the screen from one edge to the other in the direction specified by the *direction* entry, revealing the new page.
Dissolve - The old page dissolves gradually to reveal the new one.
Glitter - Similar to Dissolve, except that the effect sweeps across the page in a wide band moving from one side of the screen to the other in the direction specified by the *direction* entry.
R - The new page simply replaces the old one with no special transition effect; the *direction* entry shall be ignored.
Fly - (PDF 1.5) Changes are flown out or in (as specified by *motion*), in the direction specified by *direction*, to or from a location that is offscreen except when *direction* is **None**.
Push - (PDF 1.5) The old page slides off the screen while the new page slides in, pushing the old page out in the direction specified by *direction*.
Cover - (PDF 1.5) The new page slides on to the screen in the direction specified by *direction*, covering the old page.
Uncover - (PDF 1.5) The old page slides off the screen in the direction specified by *direction*, uncovering the new page in the direction specified by *direction*.
Fade - (PDF 1.5) The new page gradually becomes visible through the old one.

duration is the length of the transition in seconds (default 1).

dimension (Optional; **Split** and **Blinds** transition styles only) The dimension in which the specified transition effect shall occur: **H** Horizontal, or **V** Vertical.

motion (Optional; **Split**, **Box** and **Fly** transition styles only) The direction of motion for the specified transition effect: **I** Inward from the edges of the page, or **O** Outward from the center of the page.

direction (Optional; **Wipe**, **Glitter**, **Fly**, **Cover**, **Uncover** and **Push** transition styles only) The direction in which the specified transition effect shall moves, expressed in degrees counterclockwise starting from a left-to-right direction. If the value is a number, it shall be one of: **0** = Left to right, **90** = Bottom to top (Wipe only), **180** = Right to left (Wipe only), **270** = Top to bottom, **315** = Top-left to bottom-right (Glitter only) The value can be **None**, which is relevant only for the **Fly** transition when the value of *scale* is not 1.0.

scale (Optional; PDF 1.5; **Fly** transition style only) The starting or ending scale at which the changes shall be drawn. If *motion* specifies an inward transition, the scale of the changes drawn shall progress from *scale* to 1.0 over the course of the transition. If *motion* specifies an outward

transition, the scale of the changes drawn shall progress from 1.0 to *scale* over the course of the transition

bool (Optional; PDF 1.5; **Fly** transition style only) If **true**, the area that shall be flown in is rectangular and opaque.

This command can be used by calling the macro **.pdftransition** using the parameters described above. Any of the parameters may be replaced with a "." which signifies the parameter retains its previous value, also any trailing missing parameters are ignored.

Note: not all PDF Readers support any or all these transitions.

\X'pdf: background *cmd left top right bottom weight***'**
\X'pdf: background off'
\X'pdf: background footnote *bottom***'**

produces a background rectangle on the page, where

cmd is the command, which can be any of "**page|fill|box**" in combination. Thus, "**pagefill**" would draw a rectangle which covers the whole current page size (in which case the rest of the parameters can be omitted because the box dimensions are taken from the current media size). "**boxfill**", on the other hand, requires the given dimensions to place the box. Including "**fill**" in the command will paint the rectangle with the current fill colour (as with **\M[]**) and including "**box**" will give the rectangle a border in the current stroke colour (as with **\m[]**).

cmd may also be "**off**" on its own, which will terminate drawing the current box. If you have specified a page colour with "**pagefill**", it is always the first box in the stack, and if you specify it again, it will replace the first entry. Be aware that the "**pagefill**" box renders the page opaque, so tools that "watermark" PDF pages are unlikely to be successful. To return the background to transparent, issue an "**off**" command with no other boxes open.

Finally, *cmd* may be "**footnote**" followed by a new value for *bottom*, which will be used for all open boxes on the current page. This is to allow room for footnote areas that grow while a page is processed (to accommodate multiple footnotes, for instance). (If the value is negative, it is used as an offset from the bottom of the page.)

left
top
right
bottom are the coordinates of the box. The *top* and *bottom* coordinates are the minimum and maximum for the box, since the actual start of the box is *groff*'s drawing position when you issue the command, and the bottom of the box is the point where you turn the box "**off**". The top and bottom coordinates are used only if the box drawing extends onto the next page; ordinarily, they would be set to the header and footer margins.

weight provides the line width for the border if "**box**" is included in the command.

The convenience macro for this escape sequence is **.pdfbackground**. An *sboxes* macro file is also available; see *groff_tmac*(5).

Macros

gropdf's support macros in *pdf.tmac* define the convenience macros described above. Some features have no direct device control command counterpart.

.pdfinfo /*field content* ...
Define PDF metadata. *field* may be be one of **Title**, **Author**, **Subject**, **Keywords**, or another datum supported by the PDF standard or your reader. *field* must be prefixed with a slash.

Importing graphics

gropdf supports only the inclusion of other PDF files for inline images. Such a PDF file may, however, contain any of the graphic formats supported by the PDF standard, such as JPEG/JFIF, PNG, and GIF. Any

application that outputs PDF can thus be used to prepare files for embedding in documents processed by *groff* and *gropdf*.

The PDF file you wish to insert must be a single page and the drawing must just fit inside the media size of the PDF file. In *inkscape*(1) or *gimp*(1), for example, make sure the canvas size just fits the image.

The PDF parser *gropdf* implements has not been rigorously tested with all applications that produce PDF. If you find a single-page PDF which fails to import properly, try processing it with the *pdftk*(1) program.

```
pdftk existing-file output new-file
```

You may find that *new-file* imports successfully.

TrueType and other font formats

gropdf does not yet support any font formats besides Adobe Type 1 (PFA or PFB).

Font installation

The following is a step-by-step font installation guide for *gropdf*.

- Convert your font to something *groff* understands. This is a PostScript Type 1 font in PFA or PFB format, together with an AFM file. A PFA file begins as follows.

  ```
  %!PS-AdobeFont-1.0:
  ```

  A PFB file contains this string as well, preceded by some non-printing bytes. In the following steps, we will consider the use of CTAN's BrushScriptX-Italic ⟨https://ctan.org/tex−archive/fonts/brushscr⟩ font in PFA format.

- Convert the AFM file to a *groff* font description file with the *afmtodit*(1) program. For instance,

  ```
  $ afmtodit BrushScriptX-Italic.afm text.map BSI
  ```

  converts the Adobe Font Metric file *BrushScriptX−Italic.afm* to the *groff* font description file *BSI*.

  If you have a font family which provides regular upright (roman), bold, italic, and bold-italic styles, (where "italic" may be "oblique" or "slanted"), we recommend using **R**, **B**, **I**, and **BI**, respectively, as suffixes to the *groff* font family name to enable *groff*'s font family and style selection features. An example is *groff*'s built-in support for Times: the font family name is abbreviated as **T**, and the *groff* font names are therefore **TR**, **TB**, **TI**, and **TBI**. In our example, however, the BrushScriptX font is available in a single style only, italic.

- Install the *groff* font description file(s) in a *devpdf* subdirectory in the search path that *groff* uses for device and font file descriptions. See the *GROFF_FONT_PATH* entry in section "Environment" of *troff*(1) for the current value of the font search path. While *groff* doesn't directly use AFM files, it is a good idea to store them alongside its font description files.

- Register fonts in the *devpdf/download* file so they can be located for embedding in PDF files *gropdf* generates. Only the first *download* file encountered in the font search path is read. If in doubt, copy the default *download* file (see section "Files" below) to the first directory in the font search path and add your fonts there. The PostScript font name used by *gropdf* is stored in the **internalname** field in the *groff* font description file. (This name does not necessarily resemble the font's file name.) If the font in our example had originated from a foundry named **Z**, we would add the following line to *download*.

  ```
  Z→BrushScriptX-Italic→BrushScriptX-Italic.pfa
  ```

  A tab character, depicted as →, separates the fields. The default foundry has no name: its field is empty and entries corresponding to it start with a tab character, as will the one in our example.

- Test the selection and embedding of the new font.

  ```
  printf "\\f[BSI]Hello, world!\n" | groff -T pdf -P -e >hello.pdf
  see hello.pdf
  ```

Environment

GROFF_FONT_PATH

A list of directories in which to seek the selected output device's directory of device and font description files. If, in the *download* file, the font file has been specified with a full path, no directories are searched. See *troff*(1) and *groff_font*(5).

GROPDF_NOSLIDE
: If set and evaluates to a true value (to Perl), *gropdf* ignores commands specific to presentation PDFs, producing a normal PDF instead.

SOURCE_DATE_EPOCH
: A timestamp (expressed as seconds since the Unix epoch) to use as the output creation timestamp in place of the current time. The time is converted to human-readable form using Perl's *localtime()* function and recorded in a PDF comment.

TZ
: The time zone to use when converting the current time (or value of *SOURCE_DATE_EPOCH*) to human-readable form; see *tzset*(3).

Files

/usr/local/share/groff/1.23.0/font/devpdf/DESC
: describes the **pdf** output device.

*/usr/local/share/groff/1.23.0/font/devpdf/*F
: describes the font known as *F* on device **pdf**.

/usr/local/share/groff/1.23.0/font/devpdf/U−F
: describes the font from the URW foundry (versus the Adobe default) known as *F* on device **pdf**.

/usr/local/share/groff/1.23.0/font/devpdf/download
: lists fonts available for embedding within the PDF document (by analogy to the **ps** device's downloadable font support).

/usr/local/share/groff/1.23.0/font/devpdf/Foundry
: is a data file used by the *groff* build system to locate PostScript Type 1 fonts.

/usr/local/share/groff/1.23.0/font/devpdf/enc/text.enc
: describes the encoding scheme used by most PostScript Type 1 fonts; the **encoding** directive of font description files for the **pdf** device refers to it.

/usr/local/share/groff/1.23.0/tmac/pdf.tmac
: defines macros for use with the **pdf** output device. It is automatically loaded by *troffrc* when the **pdf** output device is selected.

/usr/local/share/groff/1.23.0/tmac/pdfpic.tmac
: defines the **PDFPIC** macro for embedding images in a document; see *groff_tmac*(5). It is automatically loaded by *troffrc*.

Authors

gropdf was written and is maintained by Deri James ⟨deri@chuzzlewit.myzen.co.uk⟩.

See also

/usr/local/share/doc/groff−1.23.0/sboxes/msboxes.ms
/usr/local/share/doc/groff−1.23.0/sboxes/msboxes.pdf
: "Using PDF boxes with *groff* and the *ms* macros", by Deri James.

present.tmac
: is part of *gpresent* ⟨https://bob.diertens.org/corner/useful/gpresent/⟩, a software package by Bob Diertens that works with *groff* to produce presentations ("foils", or "slide decks").

afmtodit(1), *groff*(1), *troff*(1), *groff_font*(5), *groff_out*(5)