



Linux Ubuntu 26.04 Manual Pages on command 'grops.1'

\$ man grops.1

grops(1) General Commands Manual grops(1)

Name

grops - groff output driver for PostScript

Synopsis

grops [-glm] [-b brokenness-flags] [-c num-copies] [-F font-directory] [-I inclusion-
directory] [-p paper-format] [-P prologue-file] [-w rule-thickness] [file ...]
grops --help
grops -v
grops --version

Description

The GNU roff PostScript output driver translates the output of troff(1) into PostScript. Normally, grops is invoked by groff(1) when the latter is given the `-T ps` option. (In this installation, ps is the default output device.) Use groff's `-P` option to pass any options shown above to grops. If no file arguments are given, or if file is `?-?`, grotty reads the standard input stream. Output is written to the standard output stream.

When called with multiple file arguments, grops doesn't produce a valid document structure (one conforming to the Document Structuring Conventions). To print such concatenated output, it is necessary to deactivate DSC handling in the printing program or previewer.

See section `?Font installation?` below for a guide to installing fonts for grops.

Options

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

`-b n` Work around problems with spoolers, previewers, and older printers. Normally, grops

produces output at PostScript LanguageLevel 2 that conforms to version 3.0 of the Document Structuring Conventions. Some software and devices can't handle such a data stream. The value of `n` determines what `grops` does to make its output acceptable to such consumers. If `n` is 0, `grops` employs no workarounds, which is the default; it can be changed by modifying the broken directive in `grops's` DESC file.

Add 1 to suppress generation of `%%BeginDocumentSetup` and `%%EndDocumentSetup` comments; this is needed for early versions of `TranScript` that get confused by anything between the `%%EndProlog` comment and the first `%%Page` comment.

Add 2 to omit lines in included files beginning with `%!`, which confuse Sun's pageview previewer.

Add 4 to omit lines in included files beginning with `%%Page`, `%%Trailer` and `%%End?Prolog`; this is needed for spoolers that don't understand `%%BeginDocument` and `%%End?Document` comments.

Add 8 to write `%!PS-Adobe-2.0` rather than `%!PS-Adobe-3.0` as the first line of the PostScript output; this is needed when using Sun's Newsprint with a printer that requires page reversal.

Add 16 to omit media size information (that is, output neither a `%%DocumentMedia` comment nor the `setpagedevice` PostScript command). This was the behavior of `groff` 1.18.1 and earlier; it is needed for older printers that don't understand PostScript LanguageLevel 2, and is also necessary if the output is further processed to produce an EPS file; see subsection "Escapsulated PostScript" below.

`-c n` Output `n` copies of each page.

`-F dir` Prepend directory `dir/devname` to the search path for font and device description and PostScript prologue files; `name` is the name of the device, usually `ps`.

`-g` Generate PostScript code to guess the page length. The guess is correct only if the imageable area is vertically centered on the page. This option allows you to generate documents that can be printed on both U.S. letter and A4 paper formats without change.

`-I dir` Search the directory `dir` for files named in `\X'ps: file'` and `\X'ps: import'` escape sequences. `-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` Use landscape orientation rather than portrait.

- m Turn on manual feed for the document.
- p fmt Set physical dimensions of output medium, overriding the papersize, paperlength, and paperwidth directives in the DESC file. fmt can be any argument accepted by the papersize directive; see groff_font(5).
- P prologue
Use the file prologue, sought in the groff font search path, as the PostScript prologue, overriding the default (see section ?Files? below) and the environment variable GROPS_PROLOGUE.
- w n Draw rules (lines) with a thickness of n thousandths of an em. The default thickness is 40 (0.04 em).

Usage

The input to grops must be in the format output by troff(1), described in groff_out(5). In addition, the device and font description files for the device used must meet certain requirements. The device resolution must be an integer multiple of 72 times the sizescale. The device description file must contain a valid paper format; see groff_font(5). Each font description file must contain a directive

```
internalname psname
```

which says that the PostScript name of the font is psname.

A font description file may also contain a directive

```
encoding enc-file
```

which says that the PostScript font should be reencoded using the encoding described in enc-file; this file should consist of a sequence of lines of the form

```
pschar code
```

where pschar is the PostScript name of the character, and code is its position in the encoding expressed as a decimal integer; valid values are in the range 0 to 255. Lines starting with # and blank lines are ignored. The code for each character given in the font description file must correspond to the code for the character in encoding file, or to the code in the default encoding for the font if the PostScript font is not to be reencoded. This code can be used with the \N escape sequence in troff to select the character, even if it does not have a groff glyph name. Every character in the font description file must exist in the PostScript font, and the widths given in the font description file must match the widths used in the PostScript font. grops assumes that a character with a groff name of space is blank (makes no marks on the page); it can make use of such a character to generate more ef?

efficient and compact PostScript output.

grops is able to display all glyphs in a PostScript font; it is not limited to 256 of them.

enc-file (or the default encoding if no encoding file is specified) just defines the order of glyphs for the first 256 characters; all other glyphs are accessed with additional encoding vectors which grops produces on the fly.

grops can embed fonts in a document that are necessary to render it; this is called ?downloading?. Such fonts must be in PFA format. Use pfbtops(1) to convert a Type 1 font in PFB format. Downloadable fonts must be listed in a download file containing lines of the form

```
psname file
```

where psname is the PostScript name of the font, and file is the name of the file containing it; lines beginning with # and blank lines are ignored; fields may be separated by tabs or spaces. file is sought using the same mechanism as that for groff font description files.

The download file itself is also sought using this mechanism; currently, only the first matching file found in the device and font description search path is used.

If the file containing a downloadable font or imported document conforms to the Adobe Document Structuring Conventions, then grops interprets any comments in the files sufficiently to ensure that its own output is conforming. It also supplies any needed font resources that are listed in the download file as well as any needed file resources. It is also able to handle inter-resource dependencies. For example, suppose that you have a downloadable font called Garamond, and also a downloadable font called Garamond-Outline which depends on Garamond (typically it would be defined to copy Garamond's font dictionary, and change the PaintType), then it is necessary for Garamond to appear before Garamond-Outline in the PostScript document. grops handles this automatically provided that the downloadable font file for Garamond-Outline indicates its dependence on Garamond by means of the Document Structuring Conventions, for example by beginning with the following lines.

```
%!PS-Adobe-3.0 Resource-Font

%%DocumentNeededResources: font Garamond

%%EndComments

%%IncludeResource: font Garamond
```

In this case, both Garamond and Garamond-Outline would need to be listed in the download file. A downloadable font should not include its own name in a %%DocumentSuppliedResources comment.

grops does not interpret %%DocumentFonts comments. The %%DocumentNeededResources,

%%DocumentSuppliedResources, %%IncludeResource, %%BeginResource, and %%EndResource comments (or possibly the old %%DocumentNeededFonts, %%DocumentSuppliedFonts, %%IncludeFont, %%BeginFont, and %%EndFont comments) should be used.

The default stroke and fill color is black. For colors defined in the ?rgb? color space, setrgbcolor is used; for ?cmy? and ?cmyk?, setcmykcolor; and for ?gray?, setgray. setcmyk? color is a PostScript LanguageLevel 2 command and thus not available on some older printers.

Typefaces

Styles called R, I, B, and BI mounted at font positions 1 to 4. Text fonts are grouped into families A, BM, C, H, HN, N, P, and T, each 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-DemItalic
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

Another text font is not a member of a family.

ZCMI ZapfChancery-MediumItalic

Special fonts include S, the PostScript Symbol font; ZD, Zapf Dingbats; SS (slanted symbol), which contains oblique forms of lowercase Greek letters derived from Symbol; EURO, which offers a Euro glyph for use with old devices lacking it; and ZDR, a reversed version of ZapfDingbats (with symbols flipped about the vertical axis). Most glyphs in these fonts are unnamed and must be accessed using \N. The last three are not standard PostScript fonts, but supplied by groff and therefore included in the default download file.

Device control commands

groff recognizes device control commands produced by the \X escape sequence, but interprets only those that begin with a ?ps:? tag.

\X'ps: exec code'

Execute the arbitrary PostScript commands code. The PostScript currentpoint is set to the groff drawing position when the \X escape sequence is interpreted before executing code. The origin is at the top left corner of the page; x coordinates increase to the right, and y coordinates down the page. A procedure u is defined that converts groff basic units to the coordinate system in effect (provided the user doesn't change the scale). For example,

```
.nr x 1i
```

```
\X'ps: exec \nx u 0 rlineto stroke'
```

draws a horizontal line one inch long. code may make changes to the graphics state, but any changes persist only to the end of the page. A dictionary containing the definitions specified by the def and mdef commands is on top of the dictionary stack. If your code adds definitions to this dictionary, you should allocate space for them using ?\X'ps: mdef n'?. Any definitions persist only until the end of the page. If

you use the \Y escape sequence with an argument that names a macro, code can extend over multiple lines. For example,

```
.nr x 1i
.de y
ps: exec
\nx u 0 rlineto
stroke
..
\Yy
```

is another way to draw a horizontal line one inch long. The single backslash before `?nx??` the only reason to use a register while defining the macro `?y??` is to convert a user-specified dimension `?1i?` to groff basic units which are in turn converted to PostScript units with the `u` procedure.

`grops` wraps user-specified PostScript code into a dictionary, nothing more. In particular, it doesn't start and end the inserted code with `save` and `restore`, respectively. This must be supplied by the user, if necessary.

`\X'ps: file name'`

This is the same as the `exec` command except that the PostScript code is read from file name.

`\X'ps: def code'`

Place a PostScript definition contained in code in the prologue. There should be at most one definition per `\X` command. Long definitions can be split over several `\X` commands; all the code arguments are simply joined together separated by newlines. The definitions are placed in a dictionary which is automatically pushed on the dictionary stack when an `exec` command is executed. If you use the `\Y` escape sequence with an argument that names a macro, code can extend over multiple lines.

`\X'ps: mdef n code'`

Like `def`, except that code may contain up to `n` definitions. `grops` needs to know how many definitions code contains so that it can create an appropriately sized PostScript dictionary to contain them.

`\X'ps: import file llx lly urx ury width [height]'`

Import a PostScript graphic from file. The arguments `llx`, `lly`, `urx`, and `ury` give the bounding box of the graphic in the default PostScript coordinate system. They should

all be integers: `llx` and `lly` are the x and y coordinates of the lower left corner of the graphic; `urx` and `ury` are the x and y coordinates of the upper right corner of the graphic; `width` and `height` are integers that give the desired width and height in groff basic units of the graphic.

The graphic is scaled so that it has this width and height and translated so that the lower left corner of the graphic is located at the position associated with `\X` command. If the height argument is omitted it is scaled uniformly in the x and y axes so that it has the specified width.

The contents of the `\X` command are not interpreted by troff, so vertical space for the graphic is not automatically added, and the width and height arguments are not allowed to have attached scaling indicators.

If the PostScript file complies with the Adobe Document Structuring Conventions and contains a `%%BoundingBox` comment, then the bounding box can be automatically extracted from within groff input by using the `psbb` request.

See `groff_tmac(5)` for a description of the `PSPIC` macro which provides a convenient high-level interface for inclusion of PostScript graphics.

`\X'ps: invis'`

`\X'ps: endinvis'`

No output is generated for text and drawing commands that are bracketed with these `\X` commands. These commands are intended for use when output from troff is previewed before being processed with grops; if the previewer is unable to display certain characters or other constructs, then other substitute characters or constructs can be used for previewing by bracketing them with these `\X` commands.

For example, `gxditview` is not able to display a proper `\[em]` character because the standard X11 fonts do not provide it; this problem can be overcome by executing the following request

```
.char \[em] \X'ps: invis'\nZ\v'-.25m'h'.05m'D'l .9m 0'h'.05m"\nX'ps: endinvis'\[em]
```

In this case, `gxditview` is unable to display the `\[em]` character and draws the line, whereas `grops` prints the `\[em]` character and ignores the line (this code is already in file `Xps.tmac`, which is loaded if a document intended for grops is previewed with `gxditview`).

If a PostScript procedure BPhook has been defined via a `?ps: def?` or `?ps: mdef?` device control command, it is executed at the beginning of every page (before anything is drawn or written by groff). For example, to underlay the page contents with the word `?DRAFT?` in light gray, you might use

```
.de XX
ps: def
/BPhook
{ gsave .9 setgray clippath pathbbox exch 2 copy
  .5 mul exch .5 mul translate atan rotate pop pop
  /NewCenturySchlbk-Roman findfont 200 scalefont setfont
  (DRAFT) dup stringwidth pop -.5 mul -70 moveto show
  grestore }
def
..
.devicem XX
```

Or, to cause lines and polygons to be drawn with square linecaps and mitered linejoins instead of the round linecaps and linejoins normally used by groff, use

```
.de XX
ps: def
/BPhook { 2 setlinecap 0 setlinejoin } def
..
.devicem XX
```

(square linecaps, as opposed to butt linecaps (`?0 setlinecap?`), give true corners in boxed tables even though the lines are drawn unconnected).

Encapsulated PostScript

groff itself doesn't emit bounding box information. The following script, `groff2eps`, produces an EPS file.

```
#!/bin/sh
groff -P-b16 "$1" > "$1".ps
gs -dNOPAUSE -sDEVICE=bbbox -- "$1".ps 2> "$1".bbbox
sed -e "/^%%Orientation/r $1.bbbox" \
  -e "/^%!PS-Adobe-3.0/s/$/ EPSF-3.0/" "$1".ps > "$1".eps
rm "$1".ps "$1".bbbox
```

You can then use `?groff2eps foo?` to convert file `foo` to `foo.eps`.

TrueType and other font formats

TrueType fonts can be used with `grops` if converted first to Type 42 format, a PostScript wrapper equivalent to the PFA format described in `pfbtops(1)`. Several methods exist to generate a Type 42 wrapper; some of them involve the use of a PostScript interpreter such as Ghostscript?see `gs(1)`.

One approach is to use FontForge, a font editor that can convert most outline font formats. Here's an example of using the Roboto Slab Serif font with `groff`. Several variables are used so that you can more easily adapt it into your own script.

```
MAP=/usr/share/groff/1.23.0/font/devps/generate/text.map
TTF=/usr/share/fonts/truetype/roboto/slab/RobotoSlab-Regular.ttf
BASE=$(basename "$TTF")
INT=${BASE%.ttf}
PFA=$INT.pfa
AFM=$INT.afm
GFN=RSR
DIR=$HOME/.local/groff/font
mkdir -p "$DIR"/devps
fontforge -lang=ff -c "Open(\"$TTF\");\
Generate(\"$DIR/devps/$PFA\");"
afmtodit "$DIR/devps/$AFM" "$MAP" "$DIR/devps/$GFN"
printf "$BASE\t$PFA\n" >> "$DIR/devps/download"
```

`fontforge` and `afmtodit` may generate warnings depending on the attributes of the font. The test procedure is simple.

```
printf ".ft RSR\nHello, world!\n" | groff -F "$DIR" > hello.ps
```

Once you're satisfied that the font works, you may want to generate any available related styles (for instance, Roboto Slab also has `?Bold?`, `?Light?`, and `?Thin?` styles) and set up `GROFF_FONT_PATH` in your environment to include the directory you keep the generated fonts in so that you don't have to use the `-F` option.

Font installation

The following is a step-by-step font installation guide for `grops`.

? Convert your font to something `groff` understands. This is a PostScript Type 1 font in PFA format or a PostScript Type 42 font, together with an AFM file. A PFA file begins as fol?

lows.

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

A PFB file contains this string as well, preceded by some non-printing bytes. If your font is in PFB format, use groff's pfbtops(1) program to convert it to PFA. For TrueType and other font formats, we recommend fontforge, which can convert most outline font formats. A Type 42 font file begins as follows.

```
%!PS-TrueTypeFont
```

This is a wrapper format for TrueType fonts. Old PostScript printers might not support them (that is, they might not have a built-in TrueType font interpreter). In the following steps, we will consider the use of CTAN's BrushScriptX-Italic 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 the letters 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 devps 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 devps/download file so they can be located for embedding in PostScript files groff 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 groff is stored in the internalname field in the groff font description file.

(This name does not necessarily resemble the font's file name.) We add the following line

to download.

BrushScriptX-Italic?BrushScriptX-Italic.pfa

A tab character, depicted as `?` , separates the fields.

? Test the selection and embedding of the new font.

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

see hello.pdf

Old fonts

groff versions 1.19.2 and earlier contained descriptions of a slightly different set of the base 35 PostScript level 2 fonts defined by Adobe. The older set has 229 glyphs and a larger set of kerning pairs; the newer one has 314 glyphs and includes the Euro glyph. For backwards compatibility, these old font descriptions are also installed in the `/usr/share/groff/1.23.0/oldfont/devps` directory.

To use them, make sure that groff finds the fonts before the default system fonts (with the same names): either give groff the `-F` command-line option,

```
$ groff -Tps -P-F -P/usr/share/groff/1.23.0/oldfont ...
```

or add the directory to groff's font and device description search path environment variable,

```
$ GROFF_FONT_PATH=/usr/share/groff/1.23.0/oldfont \
groff -Tps ...
```

when the command runs.

Environment

GROFF_FONT_PATH

A list of directories in which to seek the selected output device's directory of device and font description files. See `troff(1)` and `groff_font(5)`.

GROPS_PROLOGUE

If this is set to `foo`, then groff uses the file `foo` (in the font path) instead of the default prologue file `prologue`. The option `-P` overrides this environment variable.

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 `gmtime(3)` and `asctime(3)`, and recorded in a PostScript comment.

TZ The time zone to use when converting the current time to human-readable form; see `tzset(3)`. If `SOURCE_DATE_EPOCH` is used, it is always converted to human-readable

form using UTC.

Files

`/usr/share/groff/1.23.0/font/devps/DESC`

describes the ps output device.

`/usr/share/groff/1.23.0/font/devps/F`

describes the font known as F on device ps.

`/usr/share/groff/1.23.0/font/devps/download`

lists fonts available for embedding within the PostScript document (or download to the device).

`/usr/share/groff/1.23.0/font/devps/prologue`

is the default PostScript prologue prefixed to every output file.

`/usr/share/groff/1.23.0/font/devps/text.enc`

describes the encoding scheme used by most PostScript Type 1 fonts; the encoding directive of font description files for the ps device refers to it.

`/usr/share/groff/1.23.0/tmac/ps.tmac`

defines macros for use with the ps output device. It is automatically loaded by troffrc when the ps output device is selected.

`/usr/share/groff/1.23.0/tmac/pspic.tmac`

defines the PSPIC macro for embedding images in a document; see `groff_tmac(5)`. It is automatically loaded by troffrc.

`/usr/share/groff/1.23.0/tmac/psold.tmac`

provides replacement glyphs for text fonts that lack complete coverage of the ISO Latin-1 character set; using it, groff can produce glyphs like eth (?) and thorn (?) that older PostScript printers do not natively support.

groffs creates temporary files using the template `?groffsXXXXXX?`; see `groff(1)` for details on their storage location.

See also

PostScript Language Document Structuring Conventions Specification

`afmtodit(1)`, `groff(1)`, `troff(1)`, `pfbtops(1)`, `groff_char(7)`, `groff_font(5)`, `groff_out(5)`,

`groff_tmac(5)`