



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

\$ man groff.1

groff(1) General Commands Manual groff(1)

Name

groff - front end to the GNU roff document formatting system

Synopsis

```
groff [-abcCeEgGijKlNpRsStUVXzZ] [-d ctext] [-d string=text] [-D fallback-encoding]
      [-f font-family] [-F font-directory] [-I inclusion-directory] [-K input-encoding]
      [-L spooler-argument] [-m macro-package] [-M macro-directory] [-n page-number]
      [-o page-list] [-P postprocessor-argument] [-r cnumeric-expression]
      [-r register=numeric-expression] [-T output-device] [-w warning-category] [-W warning-
category] [file ...]
```

groff -h

groff --help

groff -v [option ...] [file ...]

groff --version [option ...] [file ...]

Description

groff is the primary front end to the GNU roff document formatting system. GNU roff is a typesetting system that reads plain text input files that include formatting commands to produce output in PostScript, PDF, HTML, DVI, or other formats, or for display to a terminal. Formatting commands can be low-level typesetting primitives, macros from a supplied package, or user-defined macros. All three approaches can be combined. If no file operands are specified, or if file is ?-?, groff reads the standard input stream.

A reimplement and extension of the typesetter from AT&T Unix, groff is present on most POSIX systems owing to its long association with Unix manuals (including man pages). It and

its predecessor are notable for their production of several best-selling software engineer? ing texts. groff is capable of producing typographically sophisticated documents while consuming minimal system resources.

The groff command orchestrates the execution of preprocessors, the transformation of input documents into a device-independent page description language, and the production of output from that language.

Options

-h and --help display a usage message and exit.

Because groff is intended to subsume most users' direct invocations of the troff(1) format, the two programs share a set of options. However, groff has some options that troff does not share, and others which groff interprets differently. At the same time, not all valid troff options can be given to groff.

groff-specific options

The following options either do not exist in GNU troff or are interpreted differently by groff.

-D enc Set fallback input encoding used by preconv(1) to enc; implies -k.

-e Run eqn(1) preprocessor.

-g Run grn(1) preprocessor.

-G Run grap(1) preprocessor; implies -p.

-I dir Works as troff's option (see below), but also implies -g and -s. It is passed to soelim(1) and the output driver, and grn is passed an -M option with dir as its argument.

-j Run chem(1) preprocessor; implies -p.

-k Run preconv(1) preprocessor. Refer to its man page for its behavior if neither of groff's -K or -D options is also specified.

-K enc Set input encoding used by preconv(1) to enc; implies -k.

-l Send the output to a spooler program for printing. The ?print? directive in the device description file specifies the default command to be used; see groff_font(5). If no such directive is present for the output device, output is piped to lpr(1). See options -L and -X.

-L arg Pass arg to the print spooler program. If multiple args are required, pass each with a separate -L option. groff does not prefix an option dash to arg before passing it to the spooler program.

- M Works as troff's option (see below), but is also passed to eqn(1), grap(1), and grn(1).
- N Prohibit newlines between eqn delimiters: pass -N to eqn(1).
- p Run pic(1) preprocessor.
- P arg Pass arg to the postprocessor. If multiple args are required, pass each with a separate -P option. groff does not prefix an option dash to arg before passing it to the postprocessor.
- R Run refer(1) preprocessor. No mechanism is provided for passing arguments to refer because most refer options have equivalent language elements that can be specified within the document.
- s Run soelim(1) preprocessor.
- S Operate in ?safer? mode; see -U below for its opposite. For security reasons, safer mode is enabled by default.
- t Run tbl(1) preprocessor.
- T dev Direct troff to format the input for the output device dev. groff then calls an output driver to convert troff's output to a form appropriate for dev; see subsection ?Output devices? below.
- U Operate in unsafe mode: pass the -U option to pic and troff.
- v
- version
Write version information for groff and all programs run by it to the standard output stream; that is, the given command line is processed in the usual way, passing -v to the formatter and any pre- or postprocessors invoked.
- V Output the pipeline that groff would run to the standard output stream, but do not execute it. If given more than once, groff both writes and runs the pipeline.
- X Use gxditview(1) instead of the usual postprocessor to (pre)view a document on an X11 display. Combining this option with -Tps uses the font metrics of the PostScript device, whereas the -TX75 and -TX100 options use the metrics of X11 fonts.
- Z Disable postprocessing. troff output will appear on the standard output stream (unless suppressed with -z); see groff_out(5) for a description of this format.

Transparent options

The following options are passed as-is to the formatter program troff(1) and described in more detail in its man page.

- a Generate a plain text approximation of the typeset output.
- b Write a backtrace to the standard error stream on each error or warning.
- c Start with color output disabled.
- C Enable AT&T troff compatibility mode; implies -c.
- d cs
- d name=string
Define string.
- E Inhibit troff error messages; implies -Ww.
- f fam Set default font family.
- F dir Search in directory dir for the selected output device's directory of device and font description files.
- i Process standard input after the specified input files.
- I dir Search dir for input files.
- m name
Process name.tmac before input files.
- M dir Search directory dir for macro files.
- n num Number the first page num.
- o list
Output only pages in list.
- r cnumeric-expression
- r register=numeric-expression
Define register.
- w name
- W name
Enable (-w) or inhibit (-W) emission of warnings in category name.
- z Suppress formatted device-independent output of troff.

Usage

The architecture of the GNU roff system follows that of other device-independent roff implementations, comprising preprocessors, macro packages, output drivers (or ?postprocessors?), a suite of utilities, and the formatter troff at its heart. See roff(7) for a survey of how a roff system works.

The front end programs available in the GNU roff system make it easier to use than traditional roffs that required the construction of pipelines or use of temporary files to carry

a source document from maintainable form to device-ready output. The discussion below summarizes the constituent parts of the GNU roff system. It complements roff(7) with groff-specific information.

Getting started

Those who prefer to learn by experimenting or are desirous of rapid feedback from the system may wish to start with a "Hello, world!" document.

```
$ echo "Hello, world!" | groff -Tascii | sed '/^$/d'
```

```
Hello, world!
```

We used a sed command only to eliminate the 65 blank lines that would otherwise flood the terminal screen. (roff systems were developed in the days of paper-based terminals with 66 lines to a page.)

Today's users may prefer output to a UTF-8-capable terminal.

```
$ echo "Hello, world!" | groff -Tutf8 | sed '/^$/d'
```

Producing PDF, HTML, or TeX's DVI is also straightforward. The hard part may be selecting a viewer program for the output.

```
$ echo "Hello, world!" | groff -Tpdf > hello.pdf
```

```
$ evince hello.pdf
```

```
$ echo "Hello, world!" | groff -Thtml > hello.html
```

```
$ firefox hello.html
```

```
$ echo "Hello, world!" | groff -Tdvi > hello.dvi
```

```
$ xdvi hello.html
```

Using groff as a REPL

Those with a programmer's bent may be pleased to know that they can use groff in a read-evaluate-print loop (REPL). Doing so can be handy to verify one's understanding of the formatter's behavior and/or the syntax it accepts. Turning on all warnings with -ww can aid this goal.

```
$ groff -ww -Tutf8
```

```
\# This is a comment. Let's define a register.
```

```
.nr a 1
```

```
\# Do integer arithmetic with operators evaluated left-to-right.
```

```
.nr b \n[a]+5/2
```

```
\# Let's get the result on the standard error stream.
```

```
.tm \n[b]
```

\# Now we'll define a string.

.ds name Leslie\" This is another form of comment.

.nr b (\n[a] + (7/2))

\# Center the next two text input lines.

.ce 2

Hi, *[name].

Your secret number is \n[b].

\# We will see that the division rounded toward zero.

It is

\# Here's an if-else control structure.

.ie (\n[b] % 2) odd.

.el even.

\# This trick sets the page length to the current vertical

\# position, so that blank lines don't spew when we're done.

.pl \n[nl]u

<Control-D>

Hi, Leslie.

Your secret number is 4.

It is even.

Paper format

In GNU roff, the page dimensions for the formatter troff and for output devices are handled separately. In the formatter, requests are used to set the page length (.pl), page offset (or left margin, .po), and line length (.ll). The right margin is not explicitly configured; the combination of page offset and line length provides the information necessary to derive it. The papersize macro package, automatically loaded by troff, provides an interface for configuring page dimensions by convenient names, like ?letter? or ?A4?; see groff_tmac(5). The formatter's default in this installation is ?A4?.

It is up to each macro package to respect the page dimensions configured in this way. Some offer alternative mechanisms.

For each output device, the size of the output medium can be set in its DESC file. Most output drivers also recognize a command-line option -p to override the default dimensions and an option -l to use landscape orientation. See groff_font(5) for a description of the

papersize directive, which takes an argument of the same form as -p. The output driver's man page, such as grops(1), may also be helpful. groff uses the command-line option -P to pass options to output devices; for example, use the following for PostScript output on A4 paper in landscape orientation.

```
groff -Tps -dpaper=a4l -P-pa4 -P-l -ms foo.ms > foo.ps
```

Front end

The groff program is a wrapper around the troff(1) program. It allows one to specify preprocessors via command-line options and automatically runs the appropriate postprocessor for the selected output device. Doing so, the manual construction of pipelines or management of temporary files required of users of traditional roff(7) systems can be avoided. Use the grog(1) program to infer an appropriate groff command line to format a document.

Language

Input to a roff system is in plain text interleaved with control lines and escape sequences.

The combination constitutes a document in one of a family of languages we also call roff; see roff(7) for background. An overview of GNU roff language syntax and features, including lists of all supported escape sequences, requests, and predefined registers, can be found in groff(7). GNU roff extensions to the AT&T troff language, a common subset of roff dialects extant today, are detailed in groff_diff(7).

Preprocessors

A preprocessor interprets a domain-specific language that produces roff language output. Frequently, such input is confined to sections or regions of a roff input file (bracketed with macro calls specific to each preprocessor), which it replaces. Preprocessors therefore often interpret a subset of roff syntax along with their own language. GNU roff provides reimplementations of most preprocessors familiar to users of AT&T troff; these routinely have extended features and/or require GNU troff to format their output.

tbl	lays out tables;
eqn	typesets mathematics;
pic	draws diagrams;
refer	processes bibliographic references;
soelim	preprocesses ?sourced? input files;
grn	renders gremlin(1) diagrams;
chem	draws chemical structural formul? using pic;
gperl	populates groff registers and strings using perl(1);

glilypond embeds LilyPond sheet music; and

gpinyin eases Mandarin Chinese input using Hanyu Pinyin.

A preprocessor unique to GNU roff is preconv(1), which converts various input encodings to something GNU troff can understand. When used, it is run before any other preprocessors.

Most preprocessors enclose content between a pair of characteristic tokens. Such a token must occur at the beginning of an input line and use the dot control character. Spaces and tabs must not follow the control character or precede the end of the input line. Deviating from these rules defeats a token's recognition by the preprocessor. Tokens are generally preserved in preprocessor output and interpreted as macro calls subsequently by troff. The ideal preprocessor is not yet available in groff.

??

? preprocessor ? starting token ? ending token ?

??

? chem ? .cstart ? .cend ?

? eqn ? .EQ ? .EN ?

? grap ? .G1 ? .G2 ?

? grn ? .GS ? .GE ?

? ideal ? .IS ? .IE ?

? ? ? .IF ?

? pic ? .PS ? .PE ?

? ? ? .PF ?

? ? ? .PY ?

? refer ? .R1 ? .R2 ?

? tbl ? .TS ? .TE ?

??

? glilypond ? .lilypond start ? .lilypond stop ?

? gperl ? .Perl start ? .Perl stop ?

? gpinyin ? .pinyin start ? .pinyin stop ?

??

Macro packages

Macro files are roff input files designed to produce no output themselves but instead ease the preparation of other roff documents. When a macro file is installed at a standard location and suitable for use by a general audience, it is termed a macro package.

Macro packages can be loaded prior to any roff input documents with the -m option. The GNU roff system implements most well-known macro packages for AT&T troff in a compatible way and extends them. These have one- or two-letter names arising from intense practices of naming economy in early Unix culture, a laconic approach that led to many of the packages being identified in general usage with the nroff and troff option letter used to invoke them, sometimes to punning effect, as with ?man? (short for ?manual?), and even with the option dash, as in the case of the s package, much better known as ms or even -ms.

Macro packages serve a variety of purposes. Some are ?full-service? packages, adopting responsibility for page layout among other fundamental tasks, and defining their own lexicon of macros for document composition; each such package stands alone and a given document can use at most one.

an is used to compose man pages in the format originating in Version 7 Unix (1979); see groff_man(7). It can be specified on the command line as -man.

doc is used to compose man pages in the format originating in 4.3BSD-Reno (1990); see groff_md(7). It can be specified on the command line as -mdoc.

e is the Berkeley general-purpose macro suite, developed as an alternative to AT&T's s; see groff_me(7). It can be specified on the command line as -me.

m implements the format used by the second-generation AT&T macro suite for general documents, a successor to s; see groff_mm(7). It can be specified on the command line as -mm.

om (invariably called ?mom?) is a modern package written by Peter Schaffter specifically for GNU roff. Consult the mom HTML manual for extensive documentation. She?for mom takes the female pronoun?can be specified on the command line as -mom.

s is the original AT&T general-purpose document format; see groff_ms(7). It can be specified on the command line as -ms.

Others are supplemental. For instance, andoc is a wrapper package specific to GNU roff that recognizes whether a document uses man or mdoc format and loads the corresponding macro package. It can be specified on the command line as -mandoc. A man(1) librarian program may use this macro file to delegate loading of the correct macro package; it is thus unnecessary for man itself to scan the contents of a document to decide the issue.

Many macro files augment the function of the full-service packages, or of roff documents that do not employ such a package?the latter are sometimes characterized as ?raw?. These auxiliary packages are described, along with details of macro file naming and placement, in

groff_tmac(5).

Formatters

The formatter, the program that interprets roff language input, is `troff(1)`. It provides the features of the AT&T `troff` and `nroff` programs as well as many extensions. The command-line option `-C` switches `troff` into compatibility mode, which tries to emulate AT&T `troff` as closely as is practical to enable the formatting of documents written for the older system. A shell script, `nroff(1)`, emulates the behavior of AT&T `nroff`. It attempts to correctly encode the output based on the locale, relieving the user of the need to specify an output device with the `-T` option and is therefore convenient for use with terminal output devices, described in the next subsection.

GNU `troff` generates output in a device-independent, but not device-agnostic, page description language detailed in `groff_out(5)`.

Output devices

`troff` output is formatted for a particular output device, typically specified by the `-T` option to the formatter or a front end. If neither this option nor the `GROFF_TYPESETTER` environment variable is used, the default output device is `ps`. An output device may be any of the following.

`ascii` for terminals using the ISO 646 1991:IRV character set and encoding, also known as US-ASCII.

`cp1047` for terminals using the IBM code page 1047 character set and encoding.

`dvi` for TeX DVI format.

`html`

`xhtml` for HTML and XHTML output, respectively.

`latin1` for terminals using the ISO Latin-1 (ISO 8859-1) character set and encoding.

`lbp` for Canon CaPSL printers (LBP-4 and LBP-8 series laser printers).

`lj4` for HP LaserJet4-compatible (or other PCL5-compatible) printers.

`pdf` for PDF output.

`ps` for PostScript output.

`utf8` for terminals using the ISO 10646 (Unicode) character set in UTF-8 encoding.

`X75` for previewing with `gxditview` using 75 dpi resolution and a 10-point base type size.

`X75-12` for previewing with `gxditview` using 75 dpi resolution and a 12-point base type size.

X100 for previewing with `gxditview` using 100 dpi resolution and a 10-point base type size.

X100-12 for previewing with `gxditview` using 100 dpi resolution and a 12-point base type size.

Postprocessors

Any program that interprets the output of GNU troff is a postprocessor. The postprocessors provided by GNU roff are output drivers, which prepare a document for viewing or printing. Postprocessors for other purposes, such as page resequencing or statistical measurement of a document, are conceivable.

An output driver supports one or more output devices, each with its own device description file. A device determines its postprocessor with the `postpro` directive in its device description file; see `groff_font(5)`. The `-X` option overrides this selection, causing `gxditview` to serve as the output driver.

`grodvi(1)`

provides dvi.

`grohtml(1)`

provides html and xhtml.

`grolbp(1)`

provides lbp.

`grolj4(1)`

provides lj4.

`gropdf(1)`

provides pdf.

`grops(1)`

provides ps.

`grotty(1)`

provides ascii, cp1047, latin1, and utf8.

`gxditview(1)`

provides X75, X75-12, X100, and X100-12, and additionally can preview ps.

Utilities

GNU roff includes a suite of utilities.

`gdiffmk(1)`

marks differences between a pair of roff input files.

grog(1)

infers the groff command a document requires.

Several utilities prepare descriptions of fonts, enabling the formatter to use them when producing output for a given device.

addftinfo(1)

adds information to AT&T troff font description files to enable their use with GNU troff.

afmtodit(1)

creates font description files for PostScript Type 1 fonts.

pfbtops(1)

translates a PostScript Type 1 font in PFB (Printer Font Binary) format to PFA (Printer Font ASCII), which can then be interpreted by afmtodit.

hpftodit(1)

creates font description files for the HP LaserJet 4 family of printers.

tfmtodit(1)

creates font description files for the TeX DVI device.

xtotroff(1)

creates font description files for X Window System core fonts.

A trio of tools transform material constructed using roff preprocessor languages into graphical image files.

eqn2graph(1)

converts an eqn equation into a cropped image.

grap2graph(1)

converts a grap diagram into a cropped image.

pic2graph(1)

converts a pic diagram into a cropped image.

Another set of programs works with the bibliographic data files used by the refer(1) preprocessor.

indxbib(1)

makes inverted indices for bibliographic databases, speeding lookup operations on them.

lkbib(1)

searches the databases.

lookbib(1)

interactively searches the databases.

Exit status

groff exits with a failure status if there was a problem parsing its arguments and a successful status if either of the options `-h` or `--help` was specified. Otherwise, groff runs a pipeline to process its input; if all commands within the pipeline exit successfully, groff does likewise. If not, groff's exit status encodes a summary of problems encountered, setting bit 0 if a command exited with a failure status, bit 1 if a command was terminated with a signal, and bit 2 if a command could not be executed. (Thus, if all three misfortunes befall one's pipeline, groff would exit with status $2^0 + 2^1 + 2^2 = 1+2+4 = 7$.) To troubleshoot pipeline problems, you may wish to re-run the groff command with the `-V` option and break the reported pipeline down into separate stages, inspecting the exit status of and diagnostic messages emitted by each command.

Environment

Normally, the path separator in environment variables ending with `PATH` is the colon; this may vary depending on the operating system. For example, Windows uses a semicolon instead.

GROFF_BIN_PATH

This search path, followed by `PATH`, is used to locate commands executed by groff. If it is not set, the installation directory of the GNU roff executables, `/usr/bin`, is searched before `PATH`.

GROFF_COMMAND_PREFIX

GNU roff can be configured at compile time to apply a prefix to the names of the programs it provides that had a counterpart in AT&T troff, so that name collisions are avoided at run time. The default prefix is empty.

When used, this prefix is conventionally the letter `g`?. For example, GNU troff would be installed as `gtroff`. Besides troff, the prefix applies to the formatter `nroff`; the preprocessors `eqn`, `grn`, `pic`, `refer`, `tbl`, and `soelim`; and the utilities `indxbib` and `lookbib`.

GROFF_ENCODING

The value of this variable is passed to the `preconv(1)` preprocessor's `-e` option to select the character encoding of input files. This variable's existence implies the groff option `-k`. If set but empty, groff calls `preconv` without an `-e` option. groff's `-K` option overrides `GROFF_ENCODING`.

GROFF_FONT_PATH

Seek the selected output device's directory of device and font description files in this list of directories. See troff(1) and groff_font(5).

GROFF_TMAC_PATH

Seek macro files in this list of directories. See troff(1) and groff_tmac(5).

GROFF_TMPDIR

Create temporary files in this directory. If not set, but the environment variable TMPDIR is set, temporary files are created there instead. On Windows systems, if neither of the foregoing are set, the environment variables TMP and TEMP (in that order) are checked also. Otherwise, temporary files are created in /tmp. The refer(1), grohtml(1), and grops(1) commands use temporary files.

GROFF_TYPESETTER

Set the default output device. If empty or not set, ps is used. The -T option overrides GROFF_TYPESETTER.

SOURCE_DATE_EPOCH

A time stamp (expressed as seconds since the Unix epoch) to use as the output creation time stamp in place of the current time. The time is converted to human-readable form using gmtime(3) and asctime(3) when the formatter starts up and stored in registers usable by documents and macro packages.

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.

Examples

roff systems are best known for formatting man pages. Once a man(1) librarian program has located a man page, it may execute a groff command much like the following.

```
groff -t -man -Tutf8 /usr/share/man/man1/groff.1
```

The librarian will also pipe the output through a pager, which might not interpret the SGR terminal escape sequences groff emits for boldface, underlining, or italics; see section ?Limitations? below.

To process a roff input file using the preprocessors tbl and pic and the me macro package in the way to which AT&T troff users were accustomed, one would type (or script) a pipeline.

```
pic foo.me | tbl | troff -me -Tutf8 | grotty
```

Using groff, this pipe can be shortened to an equivalent command.

```
groff -p -t -me -T utf8 foo.me
```

An even easier way to do this is to use `grog(1)` to guess the preprocessor and macro options and execute the result by using the command substitution feature of the shell.

```
$(grog -Tutf8 foo.me)
```

Each command-line option to a postprocessor must be specified with any required leading dashes `??` because `groff` passes the arguments as-is to the postprocessor; this permits arbitrary arguments to be transmitted. For example, to pass a title to the `gxditview` postprocessor, the shell commands

```
groff -X -P -title -P 'trial run' mydoc.t
```

and

```
groff -X -Z mydoc.t | gxditview -title 'trial run' -
```

are equivalent.

Limitations

When paging output for the `ascii`, `cp1047`, `latin1`, and `utf8` devices, programs like `more(1)` and `less(1)` may require command-line options to correctly handle some terminal escape sequences; see `grotty(1)`.

On EBCDIC hosts such as OS/390 Unix, the output devices `ascii` and `latin1` aren't available. Conversely, the output device `cp1047` is not available on systems based on the ISO 646 or ISO 8859 character encoding standards.

Installation directories

GNU `roff` installs files in varying locations depending on its compile-time configuration.

On this installation, the following locations are used.

`/etc/X11/app-defaults`

Application defaults directory for `gxditview(1)`.

`/usr/bin`

Directory containing `groff`'s executable commands.

`/usr/share/groff/1.23.0/eign`

List of common words for `indxbib(1)`.

`/usr/share/groff/1.23.0`

Directory for data files.

`/usr/dict/papers/Ind`

Default index for `lkbib(1)` and `refer(1)`.

`/usr/share/doc/groff-base`

Documentation directory.

`/usr/share/doc/groff-base/examples`

Example directory.

`/usr/share/groff/1.23.0/font`

Font directory.

`/usr/share/doc/groff-base/html`

HTML documentation directory.

`/usr/lib/font`

Legacy font directory.

`/usr/share/groff/site-font`

Local font directory.

`/usr/share/groff/site-tmac`

Local macro package (tmac file) directory.

`/usr/share/groff/1.23.0/tmac`

Macro package (tmac file) directory.

`/usr/share/groff/1.23.0/oldfont`

Font directory for compatibility with old versions of groff; see `grops(1)`.

`/usr/share/doc/groff-base/pdf`

PDF documentation directory.

groff macro directory

Most macro files supplied with GNU roff are stored in `/usr/share/groff/1.23.0/tmac` for the installation corresponding to this document. As a rule, multiple directories are searched for macro files; see `troff(1)`. For a catalog of macro files GNU roff provides, see `groff_tmac(5)`.

groff device and font description directory

Device and font description files supplied with GNU roff are stored in `/usr/share/groff/1.23.0/font` for the installation corresponding to this document. As a rule, multiple directories are searched for device and font description files; see `troff(1)`. For the formats of these files, see `groff_font(5)`.

Availability

Obtain links to groff releases for download, its source repository, discussion mailing lists, a support ticket tracker, and further information from the groff page of the GNU web site.

A free implementation of the grap preprocessor, written by Ted Faber, can be found at the [grap website](#). groff supports only this grap.

Authors

groff (both the front-end command and the overall system) was primarily written by James Clark. Contributors to this document include Clark, Trent A. Fisher, Werner Lemberg, Bernd Warken, and G. Branden Robinson.

See also

Groff: The GNU Implementation of troff, by Trent A. Fisher and Werner Lemberg, is the primary groff manual. You can browse it interactively with `?info groff?`.

Introduction, history, and further reading:

`roff(7)`

Viewer for groff (and AT&T device-independent troff) documents:

`gxditview(1)`

Preprocessors:

`chem(1)`, `eqn(1)`, `neqn(1)`, `glilypond(1)`, `grn(1)`, `preconv(1)`, `gperl(1)`, `pic(1)`,
`gpinyin(1)`, `refer(1)`, `soelim(1)`, `tbl(1)`

Macro packages and package-specific utilities:

`groff_hdtbl(7)`, `groff_man(7)`, `groff_man_style(7)`, `groff_mdoc(7)`, `groff_me(7)`,
`groff_mm(7)`, `groff_mmse(7)` (only in Swedish locales), `mmroff(1)`, `groff_mom(7)`,
`pdfmom(1)`, `groff_ms(7)`, `groff_rfc1345(7)`, `groff_trace(7)`, `groff_www(7)`

Bibliographic database management tools:

`indxbib(1)`, `lkbib(1)`, `lookbib(1)`

Language, conventions, and GNU extensions:

`groff(7)`, `groff_char(7)`, `groff_diff(7)`, `groff_font(5)`, `groff_tmac(5)`

Intermediate output language:

`groff_out(5)`

Formatter program:

`troff(1)`

Formatter wrappers:

`nroff(1)`, `pdfroff(1)`

Postprocessors for output devices:

`grodvi(1)`, `grohtml(1)`, `grolbp(1)`, `grolj4(1)`, `gropdf(1)`, `grops(1)`, `grotty(1)`

Font support utilities:

addftinfo(1), afmtodit(1), hpftodit(1), pfbtops(1), tfmtodit(1), xtetroff(1)

Graphics conversion utilities:

eqn2graph(1), grap2graph(1), pic2graph(1)

Difference-marking utility:

gdiffmk(1)

?groff guess? utility:

grog(1)

groff 1.23.0

2 December 2025

groff(1)