Name

groff_man_style – GNU *roff* man page tutorial and style guide

Synopsis

groff –man [*option* . . .] [*file* . . .]
groff –m man [*option* . . .] [*file* . . .]

Description

The GNU implementation of the *man* macro package is part of the *groff* document formatting system. It is used to produce manual pages (“man pages”) like the one you are reading.

This document presents the macros thematically; for those needing only a quick reference, the following table lists them alphabetically, with cross references to appropriate subsections below.

Macro	Meaning	Subsection
.B	Bold	Font style macros
.BI	Bold, italic alternating	Font style macros
.BR	Bold, roman alternating	Font style macros
.EE	Example end	Document structure macros
.EX	Example begin	Document structure macros
.I	Italic	Font style macros
.IB	Italic, bold alternating	Font style macros
.IP	Indented paragraph	Paragraphing macros
.IR	Italic, roman alternating	Font style macros
.LP	Begin paragraph	Paragraphing macros
.ME	Mail-to end	Hyperlink macros
.MR	Man page cross reference	Hyperlink macros
.MT	Mail-to start	Hyperlink macros
.P	Begin paragraph	Paragraphing macros
.PP	Begin paragraph	Paragraphing macros
.RB	Roman, bold alternating	Font style macros
.RE	Relative inset end	Document structure macros
.RI	Roman, italic alternating	Font style macros
.RS	Relative inset start	Document structure macros
.SB	Small bold	Font style macros
.SH	Section heading	Document structure macros
.SM	Small	Font style macros
.SS	Subsection heading	Document structure macros
.SY	Synopsis start	Command synopsis macros
.TH	Title heading	Document structure macros
.TP	Tagged paragraph	Paragraphing macros
.TQ	Supplemental paragraph tag	Paragraphing macros
.UE	URI end	Hyperlink macros
.UR	URI start	Hyperlink macros
.YS	Synopsis end	Command synopsis macros

We discuss other macros (**.AT**, **.DT**, **.HP**, **.OP**, **.PD**, and **.UC**) in subsection “Deprecated features” below.

Throughout Unix documentation, a manual entry is referred to simply as a “man page”, regardless of its length, without gendered implication, and irrespective of the macro package selected for its composition.

Man pages should be encoded using Unicode basic Latin code points exclusively, and employ the Unix line-ending convention (U+000A only).

Fundamental concepts

groff is a programming system for typesetting: we thus often use the verb “to set” in the sense “to typeset”. The formatter *troff*(1) collects words from the input and *fills* output lines with as many as will fit. *Words* are separated by spaces and newlines. A transition to a new output line is called a *break.* When formatted, a word may be broken at hyphens, at **\%** or **\:** escape sequences (see subsection “Portability” below), or at predetermined locations if automatic hyphenation is enabled (see the **−rHY** option in section "Options" below). An output line may be supplemented with *inter-sentence space,* and then optionally *adjusted* with more space to a consistent line length (see the **−dAD** option). *roff*(7) details these processes.

An input line that starts with a dot (.) or neutral apostrophe (') is a *control line.* To call a macro, put its name after a dot on a control line. We refer to macros in this document using this leading dot. Some macros interpret *arguments,* words that follow the macro name. A newline, unless escaped (see subsection "Portability" below), marks the end of the macro call. An input line consisting of a dot followed by a newline is called the *empty request;* it does nothing. *Text lines* are input lines that are not control lines.

We describe below several *man* macros that plant one-line *input traps:* the next input line that directly produces formatted output is treated specially. For *man* documents that follow the advice in section "Portability" below, this means that control lines using the empty request and uncommented input lines ending with an escaped newline do not spring the trap; anything else does (but see the **.TP** macro description).

Macro reference preliminaries

A tagged paragraph describes each macro. We present coupled pairs together, as with **.EX** and **.EE**.

Optional macro arguments are indicated by surrounding them with square brackets. If a macro accepts multiple arguments, those containing space characters must be double-quoted to be interpreted correctly. An empty macro argument can be specified with a pair of double-quotes (""), but the *man* package is designed such that this should seldom be necessary. See section "Notes" below for examples of cases where better alternatives to empty arguments in macro calls are available. Most macro arguments will be formatted as text in the output; exceptions are noted.

Document structure macros

Document structure macros organize a man page's content. All of them break the output line. **.TH** (title heading) identifies the document as a man page and configures the page headers and footers. Section headings (**.SH**), one of which is mandatory and many of which are conventionally expected, facilitate location of material by the reader and aid the man page writer to discuss all essential aspects of the topic. Subsection headings (**.SS**) are optional and permit sections that grow long to develop in a controlled way. Many technical discussions benefit from examples; lengthy ones, especially those reflecting multiple lines of input to or output from the system, are usefully bracketed by **.EX** and **.EE**. When none of the foregoing meets a structural demand, use **.RS**/**.RE** to inset a region within a (sub)section.

.TH *topic section* [*footer-middle*] [*footer-inside*] [*header-middle*]

Determine the contents of the page header and footer. *roff* systems refer to these collectively as "titles". The subject of the man page is *topic* and the section of the manual to which it belongs is *section.* This use of "section" has nothing to do with the section headings otherwise discussed in this page; it arises from the organizational scheme of printed and bound Unix manuals. See *man*(1) or *intro*(1) for the manual sectioning applicable to your system. *topic* and *section* are positioned together at the left and right in the header (with *section* in parentheses immediately appended to *topic*). *footer-middle* is centered in the footer. The arrangement of the rest of the footer depends on whether double-sided layout is enabled with the option **−rD1**. When disabled (the default), *footer-inside* is positioned at the bottom left. Otherwise, *footer-inside* appears at the bottom left on recto (odd-numbered) pages, and at the bottom right on verso (even-numbered) pages. The outside footer is the page number, except in the continuous-rendering mode enabled by the option **−rcR=1**, in which case it is the *topic* and *section,* as in the header. *header-middle* is centered in the header. If *section* is an integer between 1 and 9 (inclusive), there is no need to specify *header-middle; an.tmac* will supply text for it. The macro package may also abbreviate *topic* and *footer-inside* with ellipses (**...**) if they would overrun the space available in the header and footer, respectively. For HTML output, headers and footers are suppressed.

Additionally, this macro breaks the page, resetting the number to 1 (unless the **−rC1** option is given). This feature is intended only for formatting multiple *man* documents in sequence.

A valid *man* document calls **.TH** once, early in the file, prior to any other macro calls.

By convention, *footer-middle* is the date of the most recent modification to the man page source document, and *footer-inside* is the name and version or release of the project providing it.

.SH [*heading-text*]

Set *heading-text* as a section heading. If no argument is given, a one-line input trap is planted; text on the next line becomes *heading-text.* The left margin is reset to zero to set the heading text in bold (or the font specified by the string **HF**), and, on typesetting devices, slightly larger than the base type size. If the heading font ***[HF]** is bold, use of an italic style in *heading-text* is mapped to the bold-italic style if available in the font family. The inset level is reset to 1, setting the left margin to the value of the **IN** register. Text after *heading-text* is set as an ordinary paragraph (**.P**).

The content of *heading-text* and ordering of sections follows a set of common practices, as has much of the layout of material within sections. For example, a section called "Name" or "NAME" must exist, must be the first section after the **.TH** call, and must contain only text of the form

topic[**,** *another-topic*]... **\−** *summary-description*

for a man page to be properly indexed. See *man*(7) for the conventions prevailing on your system.

.SS [*subheading-text*]

Set *subheading-text* as a subsection heading indented between a section heading and an ordinary paragraph (**.P**). If no argument is given, a one-line input trap is planted; text on the next line becomes *subheading-text.* The left margin is reset to the value of the **SN** register to set the heading text in bold (or the font specified by the string **HF**). If the heading font ***[HF]** is bold, use of an italic style in *subheading-text* is mapped to the bold-italic style if available in the font family. The inset level is reset to 1, setting the left margin to the value of the **IN** register. Text after *subheading-text* is set as an ordinary paragraph (**.P**).

.EX

.EE Begin and end example. After **.EX**, filling is disabled and a constant-width (monospaced) font is selected. Calling **.EE** enables filling and restores the previous font.

Example regions are useful for formatting code, shell sessions, and text file contents. An example region is not a "literal mode" of any sort: special character escape sequences must still be used to produce correct glyphs for **'**, **−**, ****, **^**, **`**, and **~**, and sentence endings are still detected and additional inter-sentence space applied. If the amount of additional inter-sentence spacing is altered, the rendering of, for instance, regular expressions using **.** or **?** followed by multiple spaces can change. Use the dummy character escape sequence **\&** before the spaces.

These macros are extensions introduced in Ninth Edition Research Unix. Systems running that *troff*, or those from Documenter's Workbench, Heirloom Doctools, or Plan 9 *troff* support them. To be certain your page will be portable to systems that do not, copy their definitions from the *an−ext.tmac* file of a *groff* installation.

.RS [*inset-amount*]

Start a new relative inset level. The position of the left margin is saved, then moved right by *inset-amount,* if specified, and by the amount of the **IN** register otherwise. Calls to **.RS** can be nested; each increments by 1 the inset level used by **.RE**. The level prior to any **.RS** calls is 1.

.RE [*level*]

End a relative inset. The left margin corresponding to inset level *level* is restored. If no argument is given, the inset level is reduced by 1.

Paragraphing macros

An ordinary paragraph (**.P**) like this one is set without a first-line indentation at the current left margin. In man pages and other technical literature, definition lists are frequently encountered; these can be set as "tagged paragraphs", which have one (**.TP**) or more (**.TQ**) leading tags followed by a paragraph that has an additional indentation. The indented paragraph (**.IP**) macro is useful to continue the indented content of a narrative started with **.TP**, or to present an itemized or ordered list. All of these macros break the output line. If another paragraph macro has occurred since the previous **.SH** or **.SS**, they (except for **.TQ**) follow the break with a default amount of vertical space, which can be changed by the deprecated **.PD** macro; see subsection "Horizontal and vertical spacing" below. They also reset the type size and font style to defaults (**.TQ** again excepted); see subsection "Font style macros" below.

.P
.LP
.PP Begin a new paragraph; these macros are synonymous. The indentation is reset to the default value; the left margin, as affected by **.RS** and **.RE**, is not.

.TP [*indentation*]
Set a paragraph with a leading tag, and the remainder of the paragraph indented. A one-line input trap is planted; text on the next line, which can be formatted with a macro, becomes the tag, which is placed at the current left margin. The tag can be extended with the **\c** escape sequence. Subsequent text is indented by *indentation,* if specified, and by the amount of the **IN** register otherwise. If the tag is not as wide as the indentation, the paragraph starts on the same line as the tag, at the applicable indentation, and continues on the following lines. Otherwise, the descriptive part of the paragraph begins on the line following the tag.

The line containing the tag can include a macro call, for instance to set the tag in bold with **.B**. **.TP** was used to write the first paragraph of this description of **.TP**, and **.IP** the subsequent one.

.TQ Set an additional tag for a paragraph tagged with **.TP**. An input trap is planted as with **.TP**.

This macro is a GNU extension not defined on systems running AT&T, Plan 9, or Solaris *troff*; see *an−ext.tmac* in section “Files” below.

The descriptions of **.P**, **.LP**, and **.PP** above were written using **.TP** and **.TQ**.

.IP [*tag*] [*indentation*]
Set an indented paragraph with an optional tag. The *tag* and *indentation* arguments, if present, are handled as with **.TP**, with the exception that the *tag* argument to **.IP** cannot include a macro call.

Two convenient uses for **.IP** are

(1) to start a new paragraph with the same indentation as an immediately preceding **.IP** or **.TP** paragraph, if no *indentation* argument is given; and

(2) to set a paragraph with a short *tag* that is not semantically important, such as a bullet (•)—obtained with the **\(bu** special character escape sequence—or list enumerator, as seen in this very paragraph.

Command synopsis macros

.SY and **.YS** aid you to construct a command synopsis that has the classical Unix appearance. They break the output line.

These macros are GNU extensions not defined on systems running AT&T, Plan 9, or Solaris *troff*; see *an−ext.tmac* in section “Files” below.

.SY *command*
Begin synopsis. A new paragraph begins at the left margin (as with **.P**) unless **.SY** has already been called without a corresponding **.YS**, in which case only a break is performed. Adjustment and automatic hyphenation are disabled. *command* is set in bold. If a break is required, lines after the first are indented by the width of *command* plus a space.

.YS End synopsis. Indentation, adjustment, and hyphenation are restored to their previous states.

Multiple **.SY/.YS** blocks can be specified, for instance to distinguish differing modes of operation of a complex command like *tar*(1); each will be vertically separated as paragraphs are.

.SY can be repeated before **.YS** to indicate synonymous ways of invoking a particular mode of operation.

groff’s own command-line interface serves to illustrate most of the specimens of synopsis syntax one is likely to encounter.

```
.SY groff
.RB [ \-abcCeEgGijklNpRsStUVXzZ ]
.RB [ \-d\~\c
.IR cs ]
.RB [ \-d\~\c
.IB name =\c
.IR string ]
.RB [ \-D\~\c
.IR enc ]
```
(and so on similarly)
```
.RI [ file\~ .\|.\|.]
.YS
.
.
.SY groff
.B \-h
.
.SY groff
.B \-\-help
.YS
.
.
.SY groff
.B \-v
.RI [ option\~ .\|.\|.\&]
.RI [ file\~ .\|.\|.]
.
.SY groff
.B \-\-version
.RI [ option\~ .\|.\|.\&]
.RI [ file\~ .\|.\|.]
.YS
```

produces the following output.

> **groff** [**−abcCeEgGijklNpRsStUVXzZ**] [**−d** *cs*] [**−d** *name*=*string*] [**−D** *enc*] [**−f** *fam*] [**−F** *dir*] [**−I** *dir*] [**−K** *enc*] [**−L** *arg*] [**−m** *name*] [**−M** *dir*] [**−n** *num*] [**−o** *list*] [**−P** *arg*] [**−r** *cn*] [**−r** *reg*=*expr*] [**−T** *dev*] [**−w** *name*] [**−W** *name*] [*file* . . .]
>
> **groff −h**
> **groff −−help**
>
> **groff −v** [*option* . . .] [*file* . . .]
> **groff −−version** [*option* . . .] [*file* . . .]

Several features of the above example are of note.

- The empty request (.), which does nothing, is used to vertically space the input file for readability by the document maintainer. Do not put blank (empty) lines in a man page source document.
- Command and option names are presented in **bold** to cue the user that they should be input literally.
- Option dashes are specified with the **\−** escape sequence; this is an important practice to make them clearly visible and to facilitate copy-and-paste from the rendered man page to a shell prompt or text file.
- Option arguments and command operands are presented in *italics* (but see subsection “Font style macros” below regarding terminals) to cue the user that they must be replaced with appropriate text.
- Symbols that are neither to be typed literally nor replaced at the user’s discretion appear in the roman style; brackets surround optional arguments, and an ellipsis indicates that the previous syntactical element may be repeated arbitrarily.
- The non-breaking adjustable space escape sequence **\~** is used to prevent the output line from being broken within the option brackets; see subsection “Portability” below.

- The output line continuation escape sequence **\c** is used with font style alternation macros to allow all three font styles to be set without (breakable) space among them; see subsection "Portability" below.
- The dummy character escape sequence **\&** follows the ellipsis when further text will follow after space on the output line, keeping its last period from being interpreted as the end of a sentence and causing additional inter-sentence space to be placed after it. See subsection "Portability" below.

Hyperlink macros

Man page cross references like *ls*(1) are best presented with **.MR**. Text may be hyperlinked to email addresses with **.MT/.ME** or other URIs with **.UR/.UE**. Hyperlinked text is supported on HTML and terminal output devices; terminals and pager programs must support ECMA-48 OSC 8 escape sequences (see *grotty*(1)). When device support is unavailable or disabled with the **U** register (see section "Options" below), **.MT** and **.UR** URIs are rendered between angle brackets after the linked text.

.MT, **.ME**, **.UR**, and **.UE** are GNU extensions not defined on systems running AT&T, Plan 9, or Solaris *troff*; see *an−ext.tmac* in section "Files" below. Plan 9 from User Space's *troff* implements **.MR**.

The arguments to **.MR**, **.MT**, and **.UR** should be prepared for typesetting since they can appear in the output. Use special character escape sequences to encode Unicode basic Latin characters where necessary, particularly the hyphen-minus. (See section "Portability" below.) URIs can be lengthy; rendering them can result in jarring adjustment or variations in line length, or *troff* warnings when a hyperlink is longer than an output line. The application of non-printing break point escape sequences **\:** after each slash (or series thereof), and before each dot (or series thereof) is recommended as a rule of thumb. The former practice avoids forcing a trailing slash in a URI onto a separate output line, and the latter helps the reader to avoid mistakenly interpreting a dot at the end of a line as a period (or multiple dots as an ellipsis). Thus,

```
.UR http://\:example\:.com/\:fb8afcfbaebc74e\:.cc
```

has several potential break points in the URI shown. Consider adding break points before or after at signs in email addresses, and question marks, ampersands, and number signs in HTTP(S) URIs. The formatter removes **\:** escape sequences from hyperlinks when supplying device control commands to output drivers.

.MR *topic manual-section* [*trailing-text*]

(since groff *1.23)* Set a man page cross reference as "*topic*(*manual-section*)". If *trailing-text* (typically punctuation) is specified, it follows the closing parenthesis without intervening space. Hyphenation is disabled while the cross reference is set. *topic* is set in the font specified by the **MF** string. The cross reference hyperlinks to a URI of the form "**man:***topic*(*manual-section*)".

```
The output driver
.MR grops 1
produces PostScript from
.I troff
output.
.
The Ghostscript program (\c
.MR gs 1 )
interprets PostScript and PDF.
```

.MT *address*
.ME [*trailing-text*]

Identify *address* as an RFC 6068 *addr-spec* for a "mailto:" URI with the text between the two macro calls as the link text. An argument to **.ME** is placed after the link text without intervening space. *address* may not be visible in the rendered document if hyperlinks are enabled and supported by the output driver. If they are not, *address* is set in angle brackets after the link text and before *trailing-text.* If hyperlinking is enabled but there is no link text, *address* is formatted and hyperlinked *without* angle brackets.

When rendered by *groff* to a PostScript device,

```
Contact
.MT fred\:.foonly@\:fubar\:.net
Fred Foonly
.ME
for more information.
```

displays as "Contact Fred Foonly ⟨fred.foonly@fubar.net⟩ for more information.".

.UR *uri*
.UE [*trailing-text*]
Identify *uri* as an RFC 3986 URI hyperlink with the text between the two macro calls as the link text. An argument to **.UE** is placed after the link text without intervening space. *uri* may not be visible in the rendered document if hyperlinks are enabled and supported by the output driver. If they are not, *uri* is set in angle brackets after the link text and before *trailing-text.* If hyperlinking is enabled but there is no link text, *uri* is formatted and hyperlinked *without* angle brackets.

When rendered by *groff* to a PostScript device,

```
The GNU Project of the Free Software Foundation
hosts the
.UR https://\:www\:.gnu\:.org/\:software/\:groff/
.I groff
home page
.UE .
```

displays as "The GNU Project of the Free Software Foundation hosts the *groff* home page ⟨https://www.gnu.org/software/groff/⟩.".

The hyperlinking of **.TP** paragraph tags with **.UR/.UE** and **.MT/.ME** is not yet supported; if attempted, the hyperlink will be typeset at the beginning of the indented paragraph even on hyperlink-supporting devices.

Font style macros

The *man* macro package is limited in its font styling options, offering only **bold** (**.B**), *italic* (**.I**), and roman. Italic text is usually set underscored instead on terminal devices. The **.SM** and **.SB** macros set text in roman or bold, respectively, at a smaller type size; these differ visually from regular-sized roman or bold text only on typesetting devices. It is often necessary to set text in different styles without intervening space. The macros **.BI**, **.BR**, **.IB**, **.IR**, **.RB**, and **.RI**, where "B", "I", and "R" indicate bold, italic, and roman, respectively, set their odd- and even-numbered arguments in alternating styles, with no space separating them.

Because font styles are presentational rather than semantic, conflicting traditions have arisen regarding which font styles should be used to mark file or path names, environment variables, and inlined literals.

The default type size and family for typesetting devices is 10-point Times, except on the **X75−12** and **X100−12** devices where the type size is 12 points. The default style is roman.

.B [*text*]
Set *text* in bold. If no argument is given, a one-line input trap is planted; text on the next line, which can be further formatted with a macro, is set in bold.

Use bold for literal portions of syntax synopses, for command-line options in running text, and for literals that are major topics of the subject under discussion; for example, this page uses bold for macro, string, and register names. In an **.EX/.EE** example of interactive I/O (such as a shell session), set only user input in bold.

.I [*text*] Set *text* in an italic or oblique face. If no argument is given, a one-line input trap is planted; text on the next line, which can be further formatted with a macro, is set in an italic or oblique face.

Use italics for file and path names, for environment variables, for C data types, for enumeration or preprocessor constants in C, for variant (user-replaceable) portions of syntax synopses, for the first occurrence (only) of a technical concept being introduced, for names of journals and of literary works longer than an article, and anywhere a parameter requiring replacement by the user is encountered. An exception involves variant text in a context already typeset in italics, such as file or path names with replaceable components; in such cases, follow the convention of mathematical typography: set the file or path name in italics as usual but use roman for the variant part (see **.IR** and **.RI** below), and italics again in running roman text when referring to the variant material.

.SM [*text*]
Set *text* one point smaller than the default type size on typesetting devices. If no argument is given, a one-line input trap is planted; text on the next line, which can be further formatted with a macro, is set smaller.

Note: terminals will render *text* at normal size instead. Do not rely upon **.SM** to communicate semantic information distinct from using roman style at normal size; it will be hidden from readers using such devices.

.SB [*text*]
Set *text* in bold and (on typesetting devices) one point smaller than the default type size. If no argument is given, a one-line input trap is planted; text on the next line, which can be further formatted with a macro, is set smaller and in bold. This macro is an extension introduced in SunOS 4.0.

Note: terminals will render *text* in bold at the normal size instead. Do not rely upon **.SB** to communicate semantic information distinct from using bold style at normal size; it will be hidden from readers using such devices.

Observe what is *not* prescribed for setting in bold or italics above: elements of "synopsis language" such as ellipses and brackets around options; proper names and adjectives; titles of anything other than major works of literature; identifiers for standards documents or technical reports such as CSTR #54, RFC 1918, Unicode 13.0, or POSIX.1-2017; acronyms; and occurrences after the first of a technical term.

Be frugal with italics for emphasis, and particularly with bold. Article titles and brief runs of literal text, such as references to individual characters or short strings, including section and subsection headings of man pages, are suitable objects for quotation; see the **\(lq**, **\(rq**, **\(oq**, and **\(cq** escape sequences in subsection "Portability" below.

Unlike the above font style macros, the font style alternation macros below set no input traps; they must be given arguments to have effect. Italic corrections are applied as appropriate. If a space is required within an argument, first consider whether the same result could be achieved with as much clarity by using single-style macros on separate input lines. When it cannot, double-quote an argument containing embedded space characters. Setting all three different styles within a word presents challenges; it is possible with the **\c** and/or **\f** escape sequences. See subsection "Portability" below for approaches.

.BI *bold-text italic-text* ...
Set each argument in bold and italics, alternately.

```
.BI −r  register = numeric−expression
```

.BR *bold-text roman-text* ...
Set each argument in bold and roman, alternately.

```
After
.B .NH
is called,
```

.IB *italic-text bold-text* ...
Set each argument in italics and bold, alternately.

```
In places where
.IB n th
is allowed,
```

.IR *italic-text roman-text* ...
Set each argument in italics and roman, alternately.

```
Use GNU
.IR pic 's
.B figname
command to change the name of the vbox.
```

.RB *roman-text bold-text* ...
Set each argument in roman and bold, alternately.

```
if
.I file
is
.RB \[lq] \- \[rq],
the standard input stream is read.
```

.RI *roman-text italic-text* ...
Set each argument in roman and italics, alternately.

```
.RI ( tpic
was a fork of AT&T
.I pic
by Tim Morgan of the University of California at Irvine
```

Horizontal and vertical spacing

The *indentation* argument accepted by **.IP**, **.TP**, and the deprecated **.HP** is a number plus an optional scaling unit, as is **.RS**'s *inset-amount*. If no scaling unit is given, the *man* package assumes "n"; that is, the width of a letter "n" in the font current when the macro is called (see section "Measurements" in *groff*(7)). An indentation specified in a call to **.IP**, **.TP**, or the deprecated **.HP** persists until (1) another of these macros is called with an *indentation* argument, or (2) **.SH**, **.SS**, or **.P** or its synonyms is called; these clear the indentation entirely.

The left margin used by ordinary paragraphs set with **.P** (and its synonyms) not within an **.RS/.RE** relative inset is 7.2n for typesetting devices and 7n for terminal devices (but see the **−rIN** option). Headers, footers (both set with **.TH**), and section headings (**.SH**) are set at the page offset (see *groff*(7)) and subsection headings (**.SS**) indented from it by 3n (but see the **−rSN** option).

It may be helpful to think of the left margin and indentation as related but distinct concepts; *groff*'s implementation of the *man* macro package tracks them separately. The left margin is manipulated by **.RS** and **.RE** (and by **.SH** and **.SS**, which reset it to the default). Indentation is controlled by the paragraphing macros (though, again, **.SH** and **.SS** reset it); it is imposed by the **.TP**, **.IP**, and deprecated **.HP** macros, and cancelled by **.P** and its synonyms. An extensive example follows.

This ordinary (**.P**) paragraph is not in a relative inset nor does it possess an indentation.

Now we have created a relative inset (in other words, moved the left margin) with **.RS** and started another ordinary paragraph with **.P**.

tag This tagged paragraph, set with **.TP**, is still within the **.RS** region, but lines after the first have a supplementary indentation that the tag lacks.

A paragraph like this one, set with **.IP**, will appear to the reader as also associated with the tag above, because **.IP** re-uses the previous paragraph's indentation unless given an argument to change it. This paragraph is affected both by the moved left margin (**.RS**) and indentation (**.IP**).

This table is affected both by the left margin and indentation.

• This indented paragraph has a bullet for a tag, making it more obvious that the left margin and indentation are distinct; only the former affects the tag, but both affect the text of the paragraph.

This ordinary (**.P**) paragraph resets the indentation, but the left margin is still inset.

This table is affected only by the left margin.

Finally, we have ended the relative inset by using **.RE**, which (because we used only one **.RS/.RE** pair) has reset the left margin to the default. This is an ordinary **.P** paragraph.

Resist the temptation to mock up tabular or multi-column output with tab characters or the indentation arguments to **.IP**, **.TP**, **.RS**, or the deprecated **.HP**; the result may not render comprehensibly on an output device you fail to check, or which is developed in the future. The table preprocessor *tbl*(1) can likely meet your needs.

Several macros insert vertical space: **.SH**, **.SS**, **.TP**, **.P** (and its synonyms), **.IP**, and the deprecated **.HP**. The default inter-section and inter-paragraph spacing is is 1v for terminal devices and 0.4v for typesetting devices ("v" is a unit of vertical distance, where 1v is the distance between adjacent text baselines in a single-spaced document). (The deprecated macro **.PD** can change this vertical spacing, but its use is discouraged.) Between **.EX** and **.EE** calls, the inter-paragraph spacing is 1v regardless of output device.

Registers

Registers are described in section "Options" below. They can be set not only on the command line but in the site *man.local* file as well; see section "Files" below.

Strings

The following strings are defined for use in man pages. Others are supported for configuration of rendering parameters; see section "Options" below.

***R** interpolates a special character escape sequence for the "registered sign" glyph, **\(rg**, if available, and "(Reg.)" otherwise.

***S** interpolates an escape sequence setting the type size to the document default.

***(lq**
***(rq** interpolate special character escape sequences for left and right double-quotation marks, **\(lq** and **\(rq**, respectively.

***(Tm** interpolates a special character escape sequence for the "trade mark sign" glyph, **\(tm**, if available, and "(TM)" otherwise.

None of the above is necessary in a contemporary man page. ***S** is superfluous, since type size changes are invisible on terminal devices and macros that change it restore its original value afterward. Better alternatives exist for the rest; simply use the **\(rg**, **\(lq**, **\(rq**, and **\(tm** special character escape sequences directly. Unless a man page author is aiming for a pathological level of portability, such as the composition of pages for consumption on simulators of 1980s Unix systems (or Solaris *troff*, though even it supports **\(rg**), the above strings should be avoided.

Portability

It is wise to quote multi-word section and subsection headings; the **.SH** and **.SS** macros of *man*(7) implementations descended from Seventh Edition Unix supported six arguments at most. A similar restriction applied to the **.B**, **.I**, **.SM**, and font style alternation macros.

The two major syntactical categories for formatting control in the *roff* language are requests and escape sequences. Since the *man* macros are implemented in terms of *groff* requests and escape sequences, one can, in principle, supplement the functionality of *man* with these lower-level elements where necessary.

However, using raw *groff* requests (apart from the empty request "**.**") is likely to make your page render poorly when processed by other tools; many of these attempt to interpret page sources directly for conversion to HTML. Some requests make implicit assumptions about things like character and page sizes that may not hold in an HTML environment; also, many of these viewers don't interpret the full *groff* vocabulary, a problem that can lead to portions of your text being omitted or presented incomprehensibly.

For portability to modern viewers, it is best to write your page solely with the macros described in this page (except for the ones identified as deprecated, which should be avoided). The macros we have described as extensions (**.EX/.EE**, **.SY/.YS**, **.TQ**, **.UR/.UE**, **.MT/.ME**, **.MR**, and **.SB**) should be used with caution, as they may not be built in to some viewer that is important to your audience. See *an−ext.tmac* in section "Files" below.

Similar caveats apply to escape sequences. Some escape sequences are however required for correct typesetting even in man pages and usually do not cause portability problems. Several of these render glyphs corresponding to punctuation code points in the Unicode basic Latin range (U+0000–U+007F) that are handled specially in *roff* input; the escape sequences below must be used to render them correctly and portably when documenting material that uses them syntactically—namely, any of the set **' − \ ^ ` ~** (apostrophe, dash or minus, backslash, caret, grave accent, tilde).

\" Comment. Everything after the double-quote to the end of the input line is ignored. Whole-line comments should be placed immediately after the empty request ("**.**").

\*newline*

Join the next input line to the current one. Except for the update of the input line counter (used for diagnostic messages and related purposes), a series of lines ending in backslash-newline appears to *groff* as a single input line. Use this escape sequence to split excessively long input lines for document maintenance.

\% Control hyphenation. The location of this escape sequence within a word marks a hyphenation point, supplementing *groff*'s automatic hyphenation patterns. At the beginning of a word, it suppresses any hyphenation breaks within *except* those specified with **\%**.

\: Insert a non-printing break point. A word can break at such a point, but a hyphen glyph is not written to the output if it does. This escape sequence is an input word boundary, so the remainder of the word is subject to hyphenation as normal. You can use **\:** and **\%** in combination to control breaking of a file name or URI or to permit hyphenation only after certain explicit hyphens within a word. See subsection "Hyperlink macros" above for an example.

This escape sequence is a *groff* extension also supported by Heirloom Doctools *troff* 050915 (September 2005), *mandoc* 1.14.5 (2019-03-10), and *neatroff* (commit 399a4936, 2014-02-17), but not by Plan 9, Solaris, or Documenter's Workbench *troff*s.

\~ Adjustable non-breaking space. Use this escape sequence to prevent a break inside a short phrase or between a numerical quantity and its corresponding unit(s).

```
Before starting the motor,
set the output speed to\~1.
There are 1,024\~bytes in 1\~KiB.
CSTR\~#8 documents the B\~language.
```

This escape sequence is a *groff* extension also supported by Heirloom Doctools *troff* 050915 (September 2005), *mandoc* 1.9.5 (2009-09-21), *neatroff* (commit 1c6ab0f6e, 2016-09-13), and Plan 9 from User Space *troff* (commit 93f8143600, 2022-08-12), but not by Solaris or Documenter's Workbench *troff*s.

\& Dummy character. Insert at the beginning of an input line to prevent a dot or apostrophe from being interpreted as beginning a *roff* control line. Append to an end-of-sentence punctuation sequence to keep it from being recognized as such.

\| Thin space (one-sixth em on typesetters, zero-width on terminals); a non-breaking space. Used primarily in ellipses (".\|.\|.") to space the dots more pleasantly on typesetting devices like **dvi**, **pdf**, and **ps**.

\c End a text line without inserting space or attempting a break. Normally, if filling is enabled, the end of a text line is treated like a space; an output line *may* be broken there (if not, an adjustable space is inserted); if filling is disabled, the line *will* be broken there, as in **.EX**/**.EE** examples. The next line is interpreted as usual and can include a macro call (contrast with \newline). **\c** is useful when three font styles are needed in a single word, as in a command synopsis.

```
.RB [ \-\-stylesheet=\c
.IR name ]
```

It also helps when changing font styles in **.EX/.EE** examples, since they are not filled.

```
.EX
$ \c
.B groff \-T utf8 \-Z \c
.I file \c
.B | grotty \-i
.EE
```

Alternatively, and perhaps with better portability, the **\f** font selection escape sequence can be used; see below. Using **\c** to continue a **.TP** paragraph tag across multiple input lines will render incorrectly with *groff* 1.22.3, *mandoc* 1.14.1, older versions of these programs, and perhaps with some other formatters.

\e Format the current escape character on the output; widely used in man pages to render a backslash glyph. It works reliably as long as the ".ec" request is not used, which should never happen in man pages, and it is slightly more portable than the more explicit **\(rs** ("reverse solidus") special character escape sequence.

\fB, **\fI**, **\fR**, **\fP**
Switch to bold, italic, roman, or back to the previous style, respectively. Either **\f** or **\c** is needed when three different font styles are required in a word.

```
.RB [ \-\-reference\-dictionary=\fI\,name\/\fP ]

.RB [ \-\-reference\-dictionary=\c
.IR name ]
```

Style escape sequences may be more portable than **\c**. As shown above, it is up to you to account for italic corrections with "**\/**" and "**\,**", which are themselves GNU extensions, if desired and if supported by your implementation.

\fP reliably returns to the style in use immediately preceding the previous **\f** escape sequence only if no sectioning, paragraph, or style macro calls have intervened.

As long as at most two styles are needed in a word, style macros like **.B** and **.BI** usually result in more readable *roff* source than **\f** escape sequences do.

Several special characters are also widely portable. Except for **\–**, **\(em**, and **\(ga**, AT&T *troff* did not consistently define the characters listed below, but its descendants, like Plan 9 or Solaris *troff*, can be made to support them by defining them in font description files, making them aliases of existing glyphs if necessary; see *groff_font*(5).

\– Minus sign or basic Latin hyphen-minus. This escape sequence produces the Unix command-line option dash in the output. "**–**" is a hyphen in the *roff* language; some output devices replace it with U+2010 (hyphen) or similar.

\(aq Basic Latin neutral apostrophe. Some output devices format "`'`" as a right single quotation mark.

\(oq
\(cq Opening (left) and closing (right) single quotation marks. Use these for paired directional single quotes, 'like this'.

\(dq Basic Latin quotation mark (double quote). Use in macro calls to prevent '`"`' from being interpreted as beginning a quoted argument, or simply for readability.

```
.TP
.BI "split \(dq" text \(dq
```

\(lq
\(rq Left and right double quotation marks. Use these for paired directional double quotes, "like this".

\(em Em-dash. Use for an interruption—such as this one—in a sentence.

\(en En-dash. Use to separate the ends of a range, particularly between numbers; for example, "the digits 1–9".

\(ga Basic Latin grave accent. Some output devices format "` " as a left single quotation mark.

\(ha Basic Latin circumflex accent ("hat"). Some output devices format "^" as U+02C6 (modifier letter circumflex accent) or similar.

\(rs Reverse solidus (backslash). The backslash is the default escape character in the *roff* language, so it does not represent itself in output. Also see **\e** above.

\(ti Basic Latin tilde. Some output devices format "~" as U+02DC (small tilde) or similar.

For maximum portability, escape sequences and special characters not listed above are better avoided in man pages.

Hooks

Two macros, both GNU extensions, are called internally by the *groff man* package to format page headers and footers and can be redefined by the administrator in a site's *man.local* file (see section "Files" below). The presentation of **.TH** above describes the default headers and footers. Because these macros are hooks for *groff man* internals, man pages have no reason to call them. Such hook definitions will likely consist of ".sp" and ".tl" requests. They must also increase the page length with ".pl" requests in continuous rendering mode; **.PT** furthermore has the responsibility of emitting a PDF bookmark after writing the first page header in a document. Consult the existing implementations in *an.tmac* when drafting replacements.

.BT Set the page footer text ("bottom trap").

.PT Set the page header text ("page trap").

To remove a page header or footer entirely, define the appropriate macro as empty rather than deleting it.

Deprecated features

Use of the following in man pages for public distribution is discouraged.

.AT [*system* [*release*]]

Alter the footer for use with legacy AT&T man pages, overriding any definition of the *footer-inside* argument to **.TH**. This macro exists only to render man pages from historical systems.

system can be any of the following.

3	7th edition *(default)*
4	System III
5	System V

The optional *release* argument specifies the release number, as in "System V Release 3".

.DT Reset tab stops to the default (every 0.5i [inches]).

Use of this presentation-oriented macro is deprecated. It translates poorly to HTML, under which exact space control and tabulation are not readily available. Thus, information or distinctions that you use tab stops to express are likely to be lost. If you feel tempted to change the tab stops such that calling this macro later is desirable to restore them, you should probably be composing a table using *tbl*(1) instead.

.HP [*indentation*]

Set up a paragraph with a hanging left indentation. The *indentation* argument, if present, is handled as with **.TP**.

Use of this presentation-oriented macro is deprecated. A hanging indentation cannot be expressed naturally under HTML, and non-*roff*-based man page interpreters may treat **.HP** as an ordinary paragraph. Thus, information or distinctions you mean to express with indentation may be lost.

.OP *option-name* [*option-argument*]

Indicate an optional command parameter called *option-name*, which is set in bold. If the option takes an argument, specify *option-argument* using a noun, abbreviation, or hyphenated noun phrase. If present, *option-argument* is preceded by a space and set in italics. Square brackets in roman surround both arguments.

Use of this quasi-semantic macro, an extension originating in Documenter's Workbench *troff*, is deprecated. It cannot easily be used to annotate options that take optional arguments or options whose arguments have internal structure (such as a mixture of literal and variable components). One could work around these limitations with font selection escape sequences, but it is preferable to use font style alternation macros, which afford greater flexibility.

.PD [*vertical-space*]

Define the vertical space between paragraphs or (sub)sections. The optional argument *vertical-space* specifies the amount; the default scaling unit is "v". Without an argument, the spacing is reset to its default value; see subsection "Horizontal and vertical spacing" above.

Use of this presentation-oriented macro is deprecated. It translates poorly to HTML, under which exact control of inter-paragraph spacing is not readily available. Thus, information or distinctions that you use **.PD** to express are likely to be lost.

.UC [*version*]

Alter the footer for use with legacy BSD man pages, overriding any definition of the *footer-inside* argument to **.TH**. This macro exists only to render man pages from historical systems.

version can be any of the following.

3	3rd Berkeley Distribution *(default)*
4	4th Berkeley Distribution
5	4.2 Berkeley Distribution
6	4.3 Berkeley Distribution
7	4.4 Berkeley Distribution

History

M. Douglas McIlroy ⟨m.douglas.mcilroy@dartmouth.edu⟩ designed, implemented, and documented the AT&T *man* macros for Unix Version 7 (1979) and employed them to edit the first volume of its *Programmer's Manual*, a compilation of all man pages supplied by the system. That *man* supported the macros listed in this page not described as extensions, except **.P** and the deprecated **.AT** and **.UC**. The only strings defined were **R** and **S**; no registers were documented.

.UC appeared in 3BSD (1980). Unix System III (1980) introduced **.P** and exposed the registers **IN** and **LL**, which had been internal to Seventh Edition Unix *man*. PWB/UNIX 2.0 (1980) added the **Tm** string. 4BSD (1980) added **lq** and **rq** strings. SunOS 2.0 (1985) recognized **C**, **D**, **P**, and **X** registers. 4.3BSD (1986) added **.AT** and **.P**. Ninth Edition Research Unix (1986) introduced **.EX** and **.EE**. SunOS 4.0 (1988) added **.SB**.

The foregoing features were what James Clark implemented in early versions of *groff*. Later, *groff* 1.20 (2009) originated **.SY/.YS**, **.TQ**, **.MT/.ME**, and **.UR/.UE**. Plan 9 from User Space's *troff* introduced **.MR** in 2020.

Options

The following *groff* options set registers (with **−r**) and strings (with **−d**) recognized and used by the *man* macro package. To ensure rendering consistent with output device capabilities and reader preferences, man pages should never manipulate them.

−dAD=*adjustment-mode*

Set line adjustment to *adjustment-mode,* which is typically "**b**" for adjustment to both margins (the default), or "**l**" for left alignment (ragged right margin). Any valid argument to *groff*'s ".ad" request may be used. See *groff*(7) for less-common choices.

–rcR=1
Enable continuous rendering. Output is not paginated; instead, one (potentially very long) page is produced. This is the default for terminal and HTML devices. Use **–rcR=0** to disable it on terminal devices; on HTML devices, it cannot be disabled.

–rC1 Number output pages consecutively, in strictly increasing sequence, rather than resetting the page number to 1 (or the value of register **P**) with each new *man* document.

–rCS=1
Set section headings (the argument(s) to **.SH**) in full capitals. This transformation is off by default because it discards case distinction information.

–rCT=1
Set the man page topic (the first argument to **.TH**) in full capitals in headers and footers. This transformation is off by default because it discards case distinction information.

–rD1 Enable double-sided layout, formatting footers for even and odd pages differently; see the description of **.TH** in subsection "Document structure macros" above.

–rFT=*footer-distance*
Set distance of the footer relative to the bottom of the page to *footer-distance;* this amount is always negative. At one half-inch above this location, the page text is broken before writing the footer. Ignored if continuous rendering is enabled. The default is –0.5i.

–dHF=*heading-font*
Set the font used for section and subsection headings; the default is "**B**" (bold style of the default family). Any valid argument to *groff*'s ".ft" request may be used. See *groff*(7).

–rHY=0
Disable automatic hyphenation. Normally, it is enabled (1). The hyphenation mode is determined by the *groff* locale; see section "Localization" of *groff*(7).

–rIN=*standard-indentation*
Set the amount of indentation used for ordinary paragraphs (**.P** and its synonyms) and the default indentation amount used by **.IP**, **.RS**, **.TP**, and the deprecated **.HP**. See subsection "Horizontal and vertical spacing" above for the default. For terminal devices, *standard-indentation* should always be an integer multiple of unit "n" to get consistent indentation.

–rLL=*line-length*
Set line length; the default is 78n for terminal devices and 6.5i for typesetting devices.

–rLT=*title-length*
Set the line length for titles. ("Titles" is the *roff* term for headers and footers.) By default, it is set to the line length (see **–rLL** above).

–dMF=*man-page-topic-font*
Set the font used for man page topics named in **.TH** and **.MR** calls; the default is "**I**" (italic style of the default family). Any valid argument to *groff*'s ".ft" request may be used. If the **MF** string ends in "I", it is assumed to be an oblique typeface, and italic corrections are applied before and after man page topics.

–rP*n* Start enumeration of pages at *n*. The default is 1.

–rS*type-size*
Use *type-size* for the document's body text; acceptable values are 10, 11, or 12 points. See subsection "Font style macros" above for the default.

–rSN=*subsection-indentation*
Set indentation of subsection headings to *subsection-indentation.* See subsection "Horizontal and vertical spacing" above for the default.

−rU1 Enable generation of URI hyperlinks in the *grohtml* and *grotty* output drivers. *grohtml* enables them by default; *grotty* does not, pending more widespread pager support for OSC 8 escape sequences. Use **−rU0** to disable hyperlinks; this will make the arguments to **MT** and **UR** calls visible in the document text produced by link-capable drivers.

−rX*p* Number successors of page *p* as *p*a, *p*b, *p*c, and so forth. The register tracking the suffixed page letter uses format "a" (see the ".af" request in *groff*(7)). For example, the option **−rX2** produces the following page numbers: 1, 2, 2a, 2b, . . . , 2aa, 2ab, and so on.

Files

/usr/local/share/groff/1.23.0/tmac/an.tmac
Most *man* macros are defined in this file. It also loads extensions from *an−ext.tmac* (see below).

/usr/local/share/groff/1.23.0/tmac/andoc.tmac
This brief *groff* program detects whether the *man* or *mdoc* macro package is being used by a document and loads the correct macro definitions, taking advantage of the fact that pages using them must call **.TH** or **.Dd**, respectively, before any other macros. A *man* program or user typing, for example, "**groff −mandoc page.1**", need not know which package the file *page.1* uses. Multiple man pages, in either format, can be handled; *andoc* reloads each macro package as necessary.

/usr/local/share/groff/1.23.0/tmac/an−ext.tmac
Except for **.SB**, definitions of macros described above as extensions are contained in this file; in some cases, they are simpler versions of definitions appearing in *an.tmac*, and are ignored if the formatter is GNU *troff*. They are written to be compatible with AT&T *troff* and permissively licensed—not copylefted. To reduce the risk of name space collisions, string and register names begin only with "**m**". We encourage man page authors who are concerned about portability to legacy Unix systems to copy these definitions into their pages, and maintainers of *troff* implementations or work-alike systems that format man pages to re-use them.

The definitions for these macros are read after a page calls **.TH**, so they will replace any macros of the same names preceding it in your file. If you use your own implementations of these macros, they must be defined after **.TH** is called to have any effect. Furthermore, it is wise to define such page-local macros (if at all) after the "Name" section to accommodate timid *makewhatis* or *mandb* implementations that may give up their scan for indexing material early.

/usr/local/share/groff/1.23.0/tmac/man.tmac
This is a wrapper that loads *an.tmac*.

/usr/local/share/groff/1.23.0/tmac/mandoc.tmac
This is a wrapper that loads *andoc.tmac*.

/usr/local/share/groff/site−tmac/man.local
Put site-local changes and customizations into this file.

```
.\" Use narrower indentation on terminals and similar.
.if n .nr IN 4n
.\" Put only one space after the end of a sentence.
.ss 12 0 \" See groff(7).
.\" Keep pages narrow even on wide terminals.
.if n .if \n[LL]>78n .nr LL 78n
.\" Ensure hyperlinks are enabled for terminals.
.nr U 1
```

On multi-user systems, it is more considerate to users whose preferences may differ from the administrator's to be less aggressive with such settings, or to permit their override with a user-specific *man.local* file. Place the requests below at the end of the site-local file to manifest courtesy.

```
.soquiet \V[XDG_CONFIG_HOME]/man.local
.soquiet \V[HOME]/.man.local
```

However, a security-sandboxed *man*(1) program may lack permission to open such files.

Notes

Some tips on troubleshooting your man pages follow.

• Some ASCII characters look funny or copy and paste wrong.

On devices with large glyph repertoires, like UTF-8-capable terminals and PDF, several keyboard glyphs are mapped to code points outside the Unicode basic Latin range because that usually results in better typography in the general case. When documenting GNU/Linux command or C language syntax, however, this translation is sometimes not desirable.

To get a "literal"…	…should be input.
`'`	`\(aq`
`-`	`\-`
`\`	`\(rs`
`^`	`\(ha`
`` ` ``	`\(ga`
`~`	`\(ti`

Additionally, if a neutral double quote (") is needed in a macro argument, you can use **\(dq** to get it. You should *not* use **\(aq** for an ordinary apostrophe (as in "can't") or **\-** for an ordinary hyphen (as in "word-aligned"). Review subsection "Portability" above.

• Do I ever need to use an empty macro argument ("")?

Probably not. When this seems necessary, often a shorter or clearer alternative is available.

Instead of…	…should be considered.
`.TP ""`	`.TP`
`.BI ""` *italic-text bold-text*	`.IB` *italic-text bold-text*
`.TH foo 1 "" "foo 1.2.3"`	`.TH foo 1` *yyyy-mm-dd* `"foo 1.2.3"`
`.IP "" 4n`	`.IP`

`.IP "" 4n`
paragraph
…
…
	`.RS 4n`
	`.P`
	paragraph
	`.RE`

`.B one two "" three`	`.B one two three`

In the title heading (**.TH**), the date of the page's last revision is more important than packaging information; it should not be omitted. Ideally, a page maintainer will keep both up to date.

.IP is sometimes ill-understood and misused, especially when no marker argument is supplied—an indentation argument is not required. By setting an explicit indentation, you may be overriding the reader's preference as set with the **−rIN** option. If your page renders adequately without one, use the simpler form. If you need to indent multiple (unmarked) paragraphs, consider setting an inset region with **.RS** and **.RE** instead.

In the last example, the empty argument does have a subtly different effect than its suggested replacement: the empty argument causes an additional space character to be interpolated between the arguments "two" and "three"—but it is a regular breaking space, so it can be discarded at the end of an output line. It is better not to be subtle, particularly with space, which can be overlooked in source and rendered forms.

• **.RS** doesn't indent relative to my indented paragraph.

The **.RS** macro sets the left margin; that is, the position at which an *ordinary* paragraph (**.P** and its synonyms) will be set. **.IP**, **.TP**, and the deprecated **.HP** use the same default indentation. If not given an argument, **.RS** moves the left margin by this same amount. To create an inset relative to an indented paragraph, call **.RS** repeatedly until an acceptable indentation is achieved, or give **.RS** an indentation argument that is at least as much as the paragraph's indentation amount relative to an adjacent **.P** paragraph. See subsection "Horizontal and vertical spacing" above for the values.

Another approach you can use with tagged paragraphs is to place an **.RS** call immediately after the paragraph tag; this will also force a break regardless of the width of the tag, which some authors prefer. Follow-up paragraphs under the tag can then be set with **.P** instead of **.IP**. Remember to use **.RE** to end the indented region before starting the next tagged paragraph (at the appropriate nesting level).

- **.RE** doesn't move the inset back to the expected level.
- warning: scaling unit invalid in context
- warning: register 'an−saved−margin*n*' not defined
- warning: register 'an−saved−prevailing−indent*n*' not defined

  The **.RS** macro takes an *indentation amount* as an argument; the **.RE** macro's argument is a specific *inset level.* **.RE 1** goes to the level before any **.RS** macros were called, **.RE 2** goes to the level of the first **.RS** call you made, and so forth. If you desire symmetry in your macro calls, simply issue one **.RE** without an argument for each **.RS** that precedes it.

  After calls to the **.SH** and **.SS** sectioning macros, all relative insets are cleared and calls to **.RE** have no effect until **.RS** is used again.

- Do I need to keep typing the indentation in a series of **.IP** calls?

  Not if you don't want to change it. Review subsection "Horizontal and vertical spacing" above.

Instead of. . .	. . .should be considered.
`.IP \(bu 4n`	`.IP \(bu 4n`
paragraph	*paragraph*
`.IP \(bu 4n`	`.IP \(bu`
another-paragraph	*another-paragraph*

- Why doesn't the package provide a string to insert an ellipsis?

  Examples of ellipsis usage are shown above, in subsection "Command synopsis macros". The idiomatic *roff* ellipsis is three dots (periods) with thin space escape sequences **\|** internally separating them. Since dots both begin control lines and are candidate end-of-sentence characters, however, it is sometimes necessary to prefix and/or suffix an ellipsis with the dummy character escape sequence **\&**. That fact stands even if a string is defined to contain the sequence; further, if the string ends with **\&**, end-of-sentence detection is defeated when you use the string at the end of an actual sentence. (Ending a sentence with an ellipsis is often poor style, but not always.) A hypothetical string **EL** that contained an ellipsis, but not the trailing dummy character **\&**, would then need to be suffixed with the latter when not ending a sentence.

Instead of. . .	. . .do this.
`.ds EL \&.|.|.`	`Arguments are`
`Arguments are`	`.IR src-file\~ .|.|.\&`
`.IR src-file\~ \*(EL\&`	`.IR dest-dir .`
`.IR dest-dir .`	

  The first column practices a false economy; the savings in typing is offset by the cost of obscuring even the suggestion of an ellipsis to a casual reader of the source document, and reduced portability to non-*roff* man page formatters that cannot handle string definitions.

  There is an ellipsis code point in Unicode, and some fonts have an ellipsis glyph, which some man pages have accessed in a non-portable way with the font-dependent **\N** escape sequence. We discourage the use of these; on terminals, they may crowd the dots into a half-width character cell, and will not render at all if the output device doesn't have the glyph. In syntax synopses, missing ellipses can cause great confusion. Dots and space are universally supported.

Authors

The initial GNU implementation of the *man* macro package was written by James Clark. Later, Werner Lemberg ⟨wl@gnu.org⟩ supplied the **S**, **LT**, and **cR** registers, the last a 4.3BSD-Reno *mdoc*(7) feature. Larry Kollar ⟨kollar@alltel.net⟩ added the **FT**, **HY**, and **SN** registers; the **HF** string; and the **PT** and **BT** macros. G. Branden Robinson ⟨g.branden.robinson@gmail.com⟩ implemented the **AD** and **MF** strings;

CS, **CT**, and **U** registers; and the **MR** macro. Except for **.SB**, the extension macros were written by Lemberg, Eric S. Raymond ⟨esr@thyrsus.com⟩, and Robinson.

This document was originally written for the Debian GNU/Linux system by Susan G. Kleinmann ⟨sgk@debian.org⟩. It was corrected and updated by Lemberg and Robinson. The extension macros were documented by Raymond and Robinson. Raymond also originated the portability section, to which Ingo Schwarze ⟨schwarze@usta.de⟩ contributed most of the material on escape sequences.

See also

tbl(1), *eqn*(1), and *refer*(1) are preprocessors used with man pages. *man*(1) describes the man page librarian on your system. *groff_mdoc*(7) details the *groff* version of the BSD-originated alternative macro package for man pages.

groff_man(7), *groff*(7), *groff_char*(7), *man*(7)