NAME

mandoc - format manual pages

SYNOPSIS

mandoc [**-ac**] [**-I os**=*name*] [**-K** *encoding*] [**-mdoc** | **-man**] [**-O** *options*] [**-T** *output*] [**-W** *level*] [*file* ...]

DESCRIPTION

The **mandoc** utility formats manual pages for display.

By default, **mandoc** reads mdoc(7) or man(7) text from stdin and produces **-T locale** output.

The options are as follows:

-a If the standard output is a terminal device and **-c** is not specified, use less(1) to paginate the output, just like man(1) would.

-c Copy the formatted manual pages to the standard output without using less(1) to paginate them. This is the default. It can be specified to override **-a**.

-I os=*name*
Override the default operating system *name* for the mdoc(7) **Os** and for the man(7) **TH** macro.

-K *encoding*
Specify the input encoding. The supported *encoding* arguments are **us-ascii**, **iso-8859-1**, and **utf-8**. If not specified, autodetection uses the first match in the following list:

1. If the first three bytes of the input file are the UTF-8 byte order mark (BOM, 0xefbbbf), input is interpreted as **utf-8**.

2. If the first or second line of the input file matches the **emacs** mode line format

   .\" -*- [...;] coding: *encoding*; -*-

   then input is interpreted according to *encoding*.

3. If the first non-ASCII byte in the file introduces a valid UTF-8 sequence, input is interpreted as **utf-8**.

4. Otherwise, input is interpreted as **iso-8859-1**.

-mdoc | **-man**
With **-mdoc**, all input files are interpreted as mdoc(7). With **-man**, all input files are interpreted as man(7). By default, the input language is automatically detected for each file: if the first macro is **Dd** or **Dt**, the mdoc(7) parser is used; otherwise, the man(7) parser is used. With other arguments, **-m** is silently ignored.

-O *options*
Comma-separated output options. See the descriptions of the individual output formats for supported *options*.

-T *output*
Select the output format. Supported values for the *output* argument are **ascii**, **html**, the default of **locale**, **man**, **markdown**, **pdf**, **ps**, **tree**, and **utf8**.

The special **-T lint** mode only parses the input and produces no output. It implies **-W all** and redirects parser messages, which usually appear on standard error output, to standard output.

-W *level*
Specify the minimum message *level* to be reported on the standard error output and to affect the exit status. The *level* can be **base**, **style**, **warning**, **error**, or **unsupp**. The **base** level automatically derives the operating system from the contents of the **Os** macro, from the **-Ios** command line option, or from the uname(3) return value. The levels **openbsd** and **netbsd** are variants of **base** that bypass autodetection and request validation of base system conventions for a particular operating system. The level **all** is an alias for **base**. By default, **mandoc** is silent. See *EXIT STATUS* and *DIAGNOSTICS* for details.

The special option **-W stop** tells **mandoc** to exit after parsing a file that causes warnings or errors of at least the requested level. No formatted output will be produced from that file. If both a *level* and **stop** are requested, they can be joined with a comma, for example **-W error**,**stop**.

file Read from the given input file. If multiple files are specified, they are processed in the given order. If unspecified, **mandoc** reads from standard input.

The options **-fhklw** are also supported and are documented in man(1). In **-f** and **-k** mode, **mandoc** also supports the options **-CMmOSs** described in the apropos(1) manual. The options **-fkl** are mutually exclusive and override each other.

ASCII Output

Use **-T ascii** to force text output in 7-bit ASCII character encoding documented in the ascii(7) manual

page, ignoring the locale(1) set in the environment.

Font styles are applied by using back-spaced encoding such that an underlined character ‘c’ is rendered as ‘_\[bs]c’, where ‘\[bs]’ is the back-space character number 8. Emboldened characters are rendered as ‘c\[bs]c’. This markup is typically converted to appropriate terminal sequences by the pager or ul(1). To remove the markup, pipe the output to col(1) **-b** instead.

The special characters documented in mandoc_char(7) are rendered best-effort in an ASCII equivalent. In particular, opening and closing ‘single quotes’ are represented as characters number 0x60 and 0x27, respectively, which agrees with all ASCII standards from 1965 to the latest revision (2012) and which matches the traditional way in which roff(7) formatters represent single quotes in ASCII output. This correct ASCII rendering may look strange with modern Unicode-compatible fonts because contrary to ASCII, Unicode uses the code point U+0060 for the grave accent only, never for an opening quote.

The following **-O** arguments are accepted:

indent=*indent*
: The left margin for normal text is set to *indent* blank characters instead of the default of five for mdoc(7) and seven for man(7). Increasing this is not recommended; it may result in degraded formatting, for example overfull lines or ugly line breaks. When output is to a pager on a terminal that is less than 66 columns wide, the default is reduced to three columns.

mdoc
: Format man(7) input files in mdoc(7) output style. This prints the operating system name rather than the page title on the right side of the footer line, and it implies **-O indent**=5. One useful application is for checking that **-T man** output formats in the same way as the mdoc(7) source it was generated from.

tag[=*term*]
: If the formatted manual page is opened in a pager, go to the definition of the *term* rather than showing the manual page from the beginning. If no *term* is specified, reuse the first command line argument that is not a *section* number. If that argument is in apropos(1) *key=val* format, only the *val* is used rather than the argument as a whole. This is useful for commands like ‘man -akO tag Ic=ulimit’ to search for a keyword and jump right to its definition in the matching manual pages.

width=*width*
: The output width is set to *width* instead of the default of 78. When output is to a pager on a terminal that is less than 79 columns wide, the default is reduced to one less than the terminal width. In any case, lines that are output in literal mode are never wrapped and may exceed the output width.

HTML Output

Output produced by **-T html** conforms to HTML5 using optional self-closing tags. Default styles use only CSS1. Equations rendered from eqn(7) blocks use MathML.

The file */usr/share/misc/mandoc.css* documents style-sheet classes available for customising output. If a style-sheet is not specified with **-O style**, **-T html** defaults to simple output (via an embedded style-sheet) readable in any graphical or text-based web browser.

Non-ASCII characters are rendered as hexadecimal Unicode character references.

The following **-O** arguments are accepted:

fragment
: Omit the <!DOCTYPE> declaration and the <html>, <head>, and <body> elements and only emit the subtree below the <body> element. The **style** argument will be ignored. This is useful when embedding manual content within existing documents.

includes=*fmt*
: The string *fmt*, for example, *../src/%I.html*, is used as a template for linked header files (usually via the **In** macro). Instances of '%I' are replaced with the include filename. The default is not to present a hyperlink.

man=*fmt*[;*fmt*]
: The string *fmt*, for example, *../html%S/%N.%S.html*, is used as a template for linked manuals (usually via the **Xr** macro). Instances of '%N' and '%S' are replaced with the linked manual's name and section, respectively. If no section is included, section 1 is assumed. The default is not to present a hyperlink. If two formats are given and a file *%N.%S* exists in the current directory, the first format is used; otherwise, the second format is used.

style=*style.css*
: The file *style.css* is used for an external style-sheet. This must be a valid absolute or relative URI.

tag[=*term*]
: Same syntax and semantics as for *ASCII Output*. This is implemented by passing a **file://** URI ending in a fragment identifier to the pager rather than passing merely a file name. When using this argument, use a pager supporting such URIs, for example

```
MANPAGER='lynx -force_html' man -T html -O tag=MANPAGER man
MANPAGER='w3m -T text/html' man -T html -O tag=toc mandoc
```

Consequently, for HTML output, this argument does not work with more(1) or less(1). For example, 'MANPAGER=less man -T html -O tag=toc mandoc' does not work because less(1) does not support **file://** URIs.

toc If an input file contains at least two non-standard sections, print a table of contents near the beginning of the output.

Locale Output

By default, **mandoc** automatically selects UTF-8 or ASCII output according to the current locale(1). If any of the environment variables LC_ALL, LC_CTYPE, or LANG are set and the first one that is set selects the UTF-8 character encoding, it produces *UTF-8 Output*; otherwise, it falls back to *ASCII Output*. This output mode can also be selected explicitly with **-T locale**.

Man Output

Use **-T man** to translate mdoc(7) input into man(7) output format. This is useful for distributing manual sources to legacy systems lacking mdoc(7) formatters. Embedded eqn(7) and tbl(7) code is not supported.

If the input format of a file is man(7), the input is copied to the output. The parser is also run, and as usual, the **-W** level controls which *DIAGNOSTICS* are displayed before copying the input to the output.

Markdown Output

Use **-T markdown** to translate mdoc(7) input to the markdown format conforming to *John Gruber's 2004 specification*: **http://daringfireball.net/projects/markdown/syntax.text**. The output also almost conforms to the *CommonMark*: **http://commonmark.org/** specification.

The character set used for the markdown output is ASCII. Non-ASCII characters are encoded as HTML entities. Since that is not possible in literal font contexts, because these are rendered as code spans and code blocks in the markdown output, non-ASCII characters are transliterated to ASCII approximations in these contexts.

Markdown is a very weak markup language, so all semantic markup is lost, and even part of the presentational markup may be lost. Do not use this as an intermediate step in converting to HTML; instead, use **-T html** directly.

The man(7), tbl(7), and eqn(7) input languages are not supported by **-T markdown** output mode.

PDF Output

PDF-1.1 output may be generated by **-T pdf**. See *PostScript Output* for **-O** arguments and defaults.

PostScript Output

PostScript "Adobe-3.0" Level-2 pages may be generated by **-T ps**. Output pages default to letter sized and are rendered in the Times font family, 11-point. Margins are calculated as 1/9 the page length and width. Line-height is 1.4m.

Special characters are rendered as in *ASCII Output*.

The following **-O** arguments are accepted:

paper=*name*

The paper size *name* may be one of *a3*, *a4*, *a5*, *legal*, or *letter*. You may also manually specify dimensions as *NNxNN*, width by height in millimetres. If an unknown value is encountered, *letter* is used.

UTF-8 Output

Use **-T utf8** to force text output in UTF-8 multi-byte character encoding, ignoring the locale(1) settings in the environment. See *ASCII Output* regarding font styles and **-O** arguments.

On operating systems lacking locale or wide character support, and on those where the internal character representation is not UCS-4, **mandoc** always falls back to *ASCII Output*.

Syntax tree output

Use **-T tree** to show a human readable representation of the syntax tree. It is useful for debugging the source code of manual pages. The exact format is subject to change, so don't write parsers for it.

The first paragraph shows meta data found in the mdoc(7) prologue, on the man(7) **TH** line, or the fallbacks used.

In the tree dump, each output line shows one syntax tree node. Child nodes are indented with respect to their parent node. The columns are:

1. For macro nodes, the macro name; for text and tbl(7) nodes, the content. There is a special format for eqn(7) nodes.
2. Node type (text, elem, block, head, body, body-end, tail, tbl, eqn).
3. Flags:
   - An opening parenthesis if the node is an opening delimiter.
   - An asterisk if the node starts a new input line.
   - The input line number (starting at one).
   - A colon.
   - The input column number (starting at one).

- A closing parenthesis if the node is a closing delimiter.
- A full stop if the node ends a sentence.
- BROKEN if the node is a block broken by another block.
- NOSRC if the node is not in the input file, but automatically generated from macros.
- NOPRT if the node is not supposed to generate output for any output format.

The following **-O** argument is accepted:

noval Skip validation and show the unvalidated syntax tree. This can help to find out whether a given behaviour is caused by the parser or by the validator. Meta data is not available in this case.

ENVIRONMENT

LC_CTYPE The character encoding locale(1). When *Locale Output* is selected, it decides whether to use ASCII or UTF-8 output format. It never affects the interpretation of input files.

MANPAGER Any non-empty value of the environment variable MANPAGER is used instead of the standard pagination program, less(1); see man(1) for details. Only used if **-a** or **-l** is specified.

PAGER Specifies the pagination program to use when MANPAGER is not defined. If neither PAGER nor MANPAGER is defined, less(1) is used. Only used if **-a** or **-l** is specified.

EXIT STATUS

The **mandoc** utility exits with one of the following values, controlled by the message *level* associated with the **-W** option:

0 No base system convention violations, style suggestions, warnings, or errors occurred, or those that did were ignored because they were lower than the requested *level*.

1 At least one base system convention violation or style suggestion occurred, but no warning or error, and **-W base** or **-W style** was specified.

2 At least one warning occurred, but no error, and **-W warning** or a lower *level* was requested.

3 At least one parsing error occurred, but no unsupported feature was encountered, and **-W error** or a lower *level* was requested.

4 At least one unsupported feature was encountered, and **-W unsupp** or a lower *level* was requested.

5 Invalid command line arguments were specified. No input files have been read.

6 An operating system error occurred, for example exhaustion of memory, file descriptors, or process table entries. Such errors may cause **mandoc** to exit at once, possibly in the middle of parsing or formatting a file.

Note that selecting **-T lint** output mode implies **-W all**.

EXAMPLES

To page manuals to the terminal:

```
$ mandoc -l mandoc.1 man.1 apropos.1 makewhatis.8
```

To produce HTML manuals with */usr/share/misc/mandoc.css* as the style-sheet:

```
$ mandoc -T html -O style=/usr/share/misc/mandoc.css mdoc.7 > mdoc.7.html
```

To check over a large set of manuals:

```
$ mandoc -T lint ‘find /usr/src -name \*\.[1-9]‘
```

To produce a series of PostScript manuals for A4 paper:

```
$ mandoc -T ps -O paper=a4 mdoc.7 man.7 > manuals.ps
```

Convert a modern mdoc(7) manual to the older man(7) format, for use on systems lacking an mdoc(7) parser:

```
$ mandoc -T man foo.mdoc > foo.man
```

DIAGNOSTICS

Messages displayed by **mandoc** follow this format:

mandoc: *file*:*line*:*column*: *level*: *message*: *macro arguments* (*os*)

The first three fields identify the *file* name, *line* number, and *column* number of the input file where the message was triggered. The line and column numbers start at 1. Both are omitted for messages referring to an input file as a whole. All *level* and *message* strings are explained below. The name of the *macro* triggering the message and its *arguments* are omitted where meaningless. The *os* operating system specifier is omitted for messages that are relevant for all operating systems. Fatal messages about invalid command line arguments or operating system errors, for example when memory is exhausted, may also omit the *file* and *level* fields.

Message levels have the following meanings:

syserr An operating system error occurred. There isn’t necessarily anything wrong with the input

files. Output may all the same be missing or incomplete.

badarg Invalid command line arguments were specified. No input files have been read and no output is produced.

unsupp An input file uses unsupported low-level roff(7) features. The output may be incomplete and/or misformatted, so using GNU troff instead of **mandoc** to process the file may be preferable.

error Indicates a risk of information loss or severe misformatting, in most cases caused by serious syntax errors.

warning Indicates a risk that the information shown or its formatting may mismatch the author's intent in minor ways. Additionally, syntax errors are classified at least as warnings, even if they do not usually cause misformatting.

style An input file uses dubious or discouraged style. This is not a complaint about the syntax, and probably neither formatting nor portability are in danger. While great care is taken to avoid false positives on the higher message levels, the **style** level tries to reduce the probability that issues go unnoticed, so it may occasionally issue bogus suggestions. Please use your good judgement to decide whether any particular **style** suggestion really justifies a change to the input file.

base A convention used in the base system of a specific operating system is not adhered to. These are not markup mistakes, and neither the quality of formatting nor portability are in danger. Messages of the **base** level are printed with the more intuitive **style** *level* tag.

Messages of the **base**, **style**, **warning**, **error**, and **unsupp** levels are hidden unless their level, or a lower level, is requested using a **-W** option or **-T lint** output mode.

As indicated below, all **base** and some **style** checks are only performed if a specific operating system name occurs in the arguments of the **-W** command line option, of the **Os** macro, of the **-Ios** command line option, or, if neither are present, in the return value of the uname(3) function.

Conventions for base system manuals

Mdocdate found

(mdoc, NetBSD) The **Dd** macro uses CVS **Mdocdate** keyword substitution, which is not supported by the NetBSD base system. Consider using the conventional "Month dd, yyyy" format instead.

Mdocdate missing

(mdoc, OpenBSD) The **Dd** macro does not use CVS **Mdocdate** keyword substitution, but using it is conventionally expected in the OpenBSD base system.

unknown architecture
(mdoc, OpenBSD, NetBSD) The third argument of the **Dt** macro does not match any of the architectures this operating system is running on.

operating system explicitly specified
(mdoc, OpenBSD, NetBSD) The **Os** macro has an argument. In the base system, it is conventionally left blank.

RCS id missing
(OpenBSD, NetBSD) The manual page lacks the comment line with the RCS identifier generated by CVS **OpenBSD** or **NetBSD** keyword substitution as conventionally used in these operating systems.

Style suggestions

legacy man(7) date format
(mdoc) The **Dd** macro uses the legacy man(7) date format "yyyy-dd-mm". Consider using the conventional mdoc(7) date format "Month dd, yyyy" instead.

normalizing date format to: ...
(mdoc, man) The **Dd** or **TH** macro provides an abbreviated month name or a day number with a leading zero. In the formatted output, the month name is written out in full and the leading zero is omitted.

lower case character in document title
(mdoc, man) The title is still used as given in the **Dt** or **TH** macro.

duplicate RCS id
A single manual page contains two copies of the RCS identifier for the same operating system. Consider deleting the later instance and moving the first one up to the top of the page.

possible typo in section name
(mdoc) Fuzzy string matching revealed that the argument of an **Sh** macro is similar, but not identical to a standard section name.

unterminated quoted argument
(roff) Macro arguments can be enclosed in double quote characters such that space characters and macro names contained in the quoted argument need not be escaped. The closing quote of the last argument of a macro can be omitted. However, omitting it is not recommended because it makes the code harder to read.

useless macro
(mdoc) A **Bt**, **Tn**, or **Ud** macro was found. Simply delete it: it serves no useful purpose.

consider using OS macro
(mdoc) A string was found in plain text or in a **Bx** macro that could be represented using **Ox**, **Nx**, **Fx**, or **Dx**.

errnos out of order
(mdoc, NetBSD) The **Er** items in a **Bl** list are not in alphabetical order.

duplicate errno
(mdoc, NetBSD) A **Bl** list contains two consecutive **It** entries describing the same **Er** number.

referenced manual not found
(mdoc) An **Xr** macro references a manual page that was not found. When running with **-W base**, the search is restricted to the base system, by default to */usr/share/man*:*/usr/X11R6/man*. This path can be configured at compile time using the MANPATH_BASE preprocessor macro. When running with **-W style**, the search is done along the full search path as described in the man(1) manual page, respecting the **-m** and **-M** command line options, the MANPATH environment variable, the man.conf(5) file and falling back to the default of */usr/share/man*:*/usr/X11R6/man*:*/usr/local/man*, also configurable at compile time using the MANPATH_DEFAULT preprocessor macro.

trailing delimiter
(mdoc) The last argument of an **Ex**, **Fo**, **Nd**, **Nm**, **Os**, **Sh**, **Ss**, **St**, or **Sx** macro ends with a trailing delimiter. This is usually bad style and often indicates typos. Most likely, the delimiter can be removed.

no blank before trailing delimiter
(mdoc) The last argument of a macro that supports trailing delimiter arguments is longer than one byte and ends with a trailing delimiter. Consider inserting a blank such that the delimiter becomes a separate argument, thus moving it out of the scope of the macro.

fill mode already enabled, skipping
(man) A **fi** request occurs even though the document is still in fill mode, or already switched back to fill mode. It has no effect.

fill mode already disabled, skipping
(man) An **nf** request occurs even though the document already switched to no-fill mode and did not switch back to fill mode yet. It has no effect.

input text line longer than 80 bytes

Consider breaking the input text line at one of the blank characters before column 80.

verbatim "--", maybe consider using \(em
(mdoc) Even though the ASCII output device renders an em-dash as "--", that is not a good way to write it in an input file because it renders poorly on all other output devices.

function name without markup
(mdoc) A word followed by an empty pair of parentheses occurs on a text line. Consider using an **Fn** or **Xr** macro.

whitespace at end of input line
(mdoc, man, roff) Whitespace at the end of input lines is almost never semantically significant -- but in the odd case where it might be, it is extremely confusing when reviewing and maintaining documents.

bad comment style
(roff) Comment lines start with a dot, a backslash, and a double-quote character. The **mandoc** utility treats the line as a comment line even without the backslash, but leaving out the backslash might not be portable.

Warnings related to the document prologue

missing manual title, using UNTITLED
(mdoc) A **Dt** macro has no arguments, or there is no **Dt** macro before the first non-prologue macro.

missing manual title, using ""
(man) There is no **TH** macro, or it has no arguments.

missing manual section, using ""
(mdoc, man) A **Dt** or **TH** macro lacks the mandatory section argument.

unknown manual section
(mdoc) The section number in a **Dt** line is invalid, but still used.

filename/section mismatch
(mdoc, man) The name of the input file being processed is known and its file name extension starts with a non-zero digit, but the **Dt** or **TH** macro contains a *section* argument that starts with a different non-zero digit. The *section* argument is used as provided anyway. Consider checking whether the file name or the argument need a correction.

missing date, using ""
(mdoc, man) The document was parsed as mdoc(7) and it has no **Dd** macro, or the **Dd** macro has no

arguments or only empty arguments; or the document was parsed as man(7) and it has no **TH** macro, or the **TH** macro has less than three arguments or its third argument is empty.

cannot parse date, using it verbatim
(mdoc, man) The date given in a **Dd** or **TH** macro does not follow the conventional format.

date in the future, using it anyway
(mdoc, man) The date given in a **Dd** or **TH** macro is more than a day ahead of the current system time(3).

missing Os macro, using ""
(mdoc) The default or current system is not shown in this case.

late prologue macro
(mdoc) A **Dd** or **Os** macro occurs after some non-prologue macro, but still takes effect.

prologue macros out of order
(mdoc) The prologue macros are not given in the conventional order **Dd**, **Dt**, **Os**. All three macros are used even when given in another order.

Warnings regarding document structure

.so is fragile, better use ln(1)
(roff) Including files only works when the parser program runs with the correct current working directory.

no document body
(mdoc, man) The document body contains neither text nor macros. An empty document is shown, consisting only of a header and a footer line.

content before first section header
(mdoc, man) Some macros or text precede the first **Sh** or **SH** section header. The offending macros and text are parsed and added to the top level of the syntax tree, outside any section block.

first section is not NAME
(mdoc) The argument of the first **Sh** macro is not ‘NAME’. This may confuse makewhatis(8) and apropos(1).

NAME section without Nm before Nd
(mdoc) The NAME section does not contain any **Nm** child macro before the first **Nd** macro.

NAME section without description
(mdoc) The NAME section lacks the mandatory **Nd** child macro.

description not at the end of NAME
(mdoc) The NAME section does contain an **Nd** child macro, but other content follows it.

bad NAME section content
(mdoc) The NAME section contains plain text or macros other than **Nm** and **Nd**.

missing comma before name
(mdoc) The NAME section contains an **Nm** macro that is neither the first one nor preceded by a comma.

missing description line, using ""
(mdoc) The **Nd** macro lacks the required argument. The title line of the manual will end after the dash.

description line outside NAME section
(mdoc) An **Nd** macro appears outside the NAME section. The arguments are printed anyway and the following text is used for apropos(1), but none of that behaviour is portable.

sections out of conventional order
(mdoc) A standard section occurs after another section it usually precedes. All section titles are used as given, and the order of sections is not changed.

duplicate section title
(mdoc) The same standard section title occurs more than once.

unexpected section
(mdoc) A standard section header occurs in a section of the manual where it normally isn’t useful.

cross reference to self
(mdoc) An **Xr** macro refers to a name and section matching the section of the present manual page and a name mentioned in an **Nm** macro in the NAME or SYNOPSIS section, or in an **Fn** or **Fo** macro in the SYNOPSIS. Consider using **Nm** or **Fn** instead of **Xr**.

unusual Xr order
(mdoc) In the SEE ALSO section, an **Xr** macro with a lower section number follows one with a higher number, or two **Xr** macros referring to the same section are out of alphabetical order.

unusual Xr punctuation
(mdoc) In the SEE ALSO section, punctuation between two **Xr** macros differs from a single comma, or

there is trailing punctuation after the last **Xr** macro.

AUTHORS section without An macro
(mdoc) An AUTHORS sections contains no **An** macros, or only empty ones. Probably, there are author names lacking markup.

Warnings related to macros and nesting

obsolete macro
(mdoc) See the mdoc(7) manual for replacements.

macro neither callable nor escaped
(mdoc) The name of a macro that is not callable appears on a macro line. It is printed verbatim. If the intention is to call it, move it to its own input line; otherwise, escape it by prepending ‘\&’.

skipping paragraph macro
In mdoc(7) documents, this happens
- at the beginning and end of sections and subsections
- right before non-compact lists and displays
- at the end of items in non-column, non-compact lists
- and for multiple consecutive paragraph macros.

In man(7) documents, it happens
- for empty **P**, **PP**, and **LP** macros
- for **IP** macros having neither head nor body arguments
- for **br** or **sp** right after **SH** or **SS**

moving paragraph macro out of list
(mdoc) A list item in a **Bl** list contains a trailing paragraph macro. The paragraph macro is moved after the end of the list.

skipping no-space macro
(mdoc) An input line begins with an **Ns** macro, or the next argument after an **Ns** macro is an isolated closing delimiter. The macro is ignored.

blocks badly nested
(mdoc) If two blocks intersect, one should completely contain the other. Otherwise, rendered output is likely to look strange in any output format, and rendering in SGML-based output formats is likely to be outright wrong because such languages do not support badly nested blocks at all. Typical examples of badly nested blocks are "**Ao Bo Ac Bc**" and "**Ao Bq Ac**". In these examples, **Ac** breaks **Bo** and **Bq**, respectively.

nested displays are not portable
(mdoc) A **Bd**, **D1**, or **Dl** display occurs nested inside another **Bd** display. This works with **mandoc**, but fails with most other implementations.

moving content out of list
(mdoc) A **Bl** list block contains text or macros before the first **It** macro. The offending children are moved before the beginning of the list.

first macro on line
Inside a **Bl -column** list, a **Ta** macro occurs as the first macro on a line, which is not portable.

line scope broken
(man) While parsing the next-line scope of the previous macro, another macro is found that prematurely terminates the previous one. The previous, interrupted macro is deleted from the parse tree.

Warnings related to missing arguments

skipping empty request
(roff, eqn) The macro name is missing from a macro definition request, or an eqn(7) control statement or operation keyword lacks its required argument.

conditional request controls empty scope
(roff) A conditional request is only useful if any of the following follows it on the same logical input line:

- The ‘\{’ keyword to open a multi-line scope.
- A request or macro or some text, resulting in a single-line scope.
- The immediate end of the logical line without any intervening whitespace, resulting in next-line scope.

Here, a conditional request is followed by trailing whitespace only, and there is no other content on its logical input line. Note that it doesn’t matter whether the logical input line is split across multiple physical input lines using ‘\’ line continuation characters. This is one of the rare cases where trailing whitespace is syntactically significant. The conditional request controls a scope containing whitespace only, so it is unlikely to have a significant effect, except that it may control a following **el** clause.

skipping empty macro
(mdoc) The indicated macro has no arguments and hence no effect.

empty block
(mdoc, man) A **Bd**, **Bk**, **Bl**, **D1**, **Dl**, **MT**, **RS**, or **UR** block contains nothing in its body and will produce no output.

empty argument, using 0n
(mdoc) The required width is missing after **Bd** or **Bl -offset** or **-width**.

missing display type, using -ragged
(mdoc) The **Bd** macro is invoked without the required display type.

list type is not the first argument
(mdoc) In a **Bl** macro, at least one other argument precedes the type argument. The **mandoc** utility copes with any argument order, but some other mdoc(7) implementations do not.

missing -width in -tag list, using 8n
(mdoc) Every **Bl** macro having the **-tag** argument requires **-width**, too.

missing utility name, using ""
(mdoc) The **Ex -std** macro is called without an argument before **Nm** has first been called with an argument.

missing function name, using ""
(mdoc) The **Fo** macro is called without an argument. No function name is printed.

empty head in list item
(mdoc) In a **Bl -diag**, **-hang**, **-inset**, **-ohang**, or **-tag** list, an **It** macro lacks the required argument. The item head is left empty.

empty list item
(mdoc) In a **Bl -bullet**, **-dash**, **-enum**, or **-hyphen** list, an **It** block is empty. An empty list item is shown.

missing argument, using next line
(mdoc) An **It** macro in a **Bd -column** list has no arguments. While **mandoc** uses the text or macros of the following line, if any, for the cell, other formatters may misformat the list.

missing font type, using \fR
(mdoc) A **Bf** macro has no argument. It switches to the default font.

unknown font type, using \fR
(mdoc) The **Bf** argument is invalid. The default font is used instead.

nothing follows prefix
(mdoc) A **Pf** macro has no argument, or only one argument and no macro follows on the same input line. This defeats its purpose; in particular, spacing is not suppressed before the text or macros following on

the next input line.

empty reference block
(mdoc) An **Rs** macro is immediately followed by an **Re** macro on the next input line. Such an empty block does not produce any output.

missing section argument
(mdoc) An **Xr** macro lacks its second, section number argument. The first argument, i.e. the name, is printed, but without subsequent parentheses.

missing -std argument, adding it
(mdoc) An **Ex** or **Rv** macro lacks the required **-std** argument. The **mandoc** utility assumes **-std** even when it is not specified, but other implementations may not.

missing option string, using ""
(man) The **OP** macro is invoked without any argument. An empty pair of square brackets is shown.

missing resource identifier, using ""
(man) The **MT** or **UR** macro is invoked without any argument. An empty pair of angle brackets is shown.

missing eqn box, using ""
(eqn) A diacritic mark or a binary operator is found, but there is nothing to the left of it. An empty box is inserted.

Warnings related to bad macro arguments

duplicate argument
(mdoc) A **Bd** or **Bl** macro has more than one **-compact**, more than one **-offset**, or more than one **-width** argument. All but the last instances of these arguments are ignored.

skipping duplicate argument
(mdoc) An **An** macro has more than one **-split** or **-nosplit** argument. All but the first of these arguments are ignored.

skipping duplicate display type
(mdoc) A **Bd** macro has more than one type argument; the first one is used.

skipping duplicate list type
(mdoc) A **Bl** macro has more than one type argument; the first one is used.

skipping -width argument
(mdoc) A **Bl -column**, **-diag**, **-ohang**, **-inset**, or **-item** list has a **-width** argument. That has no effect.

wrong number of cells
In a line of a **Bl -column** list, the number of tabs or **Ta** macros is less than the number expected from the list header line or exceeds the expected number by more than one. Missing cells remain empty, and all cells exceeding the number of columns are joined into one single cell.

unknown AT&T UNIX version
(mdoc) An **At** macro has an invalid argument. It is used verbatim, with "AT&T UNIX " prefixed to it.

comma in function argument
(mdoc) An argument of an **Fa** or **Fn** macro contains a comma; it should probably be split into two arguments.

parenthesis in function name
(mdoc) The first argument of an **Fc** or **Fn** macro contains an opening or closing parenthesis; that's probably wrong, parentheses are added automatically.

unknown library name
(mdoc, not on OpenBSD) An **Lb** macro has an unknown name argument and will be rendered as "library "*name*"".

invalid content in Rs block
(mdoc) An **Rs** block contains plain text or non-% macros. The bogus content is left in the syntax tree. Formatting may be poor.

invalid Boolean argument
(mdoc) An **Sm** macro has an argument other than **on** or **off**. The invalid argument is moved out of the macro, which leaves the macro empty, causing it to toggle the spacing mode.

argument contains two font escapes
(roff) The second argument of a **char** request contains more than one font escape sequence. A wrong font may remain active after using the character.

unknown font, skipping request
(man, tbl) A roff(7) **ft** request or a tbl(7) **f** layout modifier has an unknown *font* argument.

odd number of characters in request
(roff) A **tr** request contains an odd number of characters. The last character is mapped to the blank

character.

Warnings related to plain text

blank line in fill mode, using .sp

(mdoc) The meaning of blank input lines is only well-defined in non-fill mode: In fill mode, line breaks of text input lines are not supposed to be significant. However, for compatibility with groff, blank lines in fill mode are formatted like **sp** requests. To request a paragraph break, use **Pp** instead of a blank line.

tab in filled text

(mdoc, man) The meaning of tab characters is only well-defined in non-fill mode: In fill mode, whitespace is not supposed to be significant on text input lines. As an implementation dependent choice, tab characters on text lines are passed through to the formatters in any case. Given that the text before the tab character will be filled, it is hard to predict which tab stop position the tab will advance to.

new sentence, new line

(mdoc) A new sentence starts in the middle of a text line. Start it on a new input line to help formatters produce correct spacing.

invalid escape sequence

(roff) An escape sequence has an invalid opening argument delimiter, lacks the closing argument delimiter, the argument is of an invalid form, or it is a character escape sequence with an invalid name. If the argument is incomplete, ***** and **\n** expand to an empty string, **\B** to the digit ‘0’, and **\w** to the length of the incomplete argument. All other invalid escape sequences are ignored.

undefined escape, printing literally

(roff) In an escape sequence, the first character right after the leading backslash is invalid. That character is printed literally, which is equivalent to ignoring the backslash.

undefined string, using ""

(roff) If a string is used without being defined before, its value is implicitly set to the empty string. However, defining strings explicitly before use keeps the code more readable.

Warnings related to tables

tbl line starts with span

(tbl) The first cell in a table layout line is a horizontal span (‘**s**’). Data provided for this cell is ignored, and nothing is printed in the cell.

tbl column starts with span

(tbl) The first line of a table layout specification requests a vertical span (‘^’). Data provided for this cell is ignored, and nothing is printed in the cell.

skipping vertical bar in tbl layout
(tbl) A table layout specification contains more than two consecutive vertical bars. A double bar is printed, all additional bars are discarded.

Errors related to tables

non-alphabetic character in tbl options
(tbl) The table options line contains a character other than a letter, blank, or comma where the beginning of an option name is expected. The character is ignored.

skipping unknown tbl option
(tbl) The table options line contains a string of letters that does not match any known option name. The word is ignored.

missing tbl option argument
(tbl) A table option that requires an argument is not followed by an opening parenthesis, or the opening parenthesis is immediately followed by a closing parenthesis. The option is ignored.

wrong tbl option argument size
(tbl) A table option argument contains an invalid number of characters. Both the option and the argument are ignored.

empty tbl layout
(tbl) A table layout specification is completely empty, specifying zero lines and zero columns. As a fallback, a single left-justified column is used.

invalid character in tbl layout
(tbl) A table layout specification contains a character that can neither be interpreted as a layout key character nor as a layout modifier, or a modifier precedes the first key. The invalid character is discarded.

unmatched parenthesis in tbl layout
(tbl) A table layout specification contains an opening parenthesis, but no matching closing parenthesis. The rest of the input line, starting from the parenthesis, has no effect.

ignoring excessive spacing in tbl layout
(tbl) A spacing modifier in a table layout is unreasonably large. The default spacing of 3n is used instead.

tbl without any data cells
(tbl) A table does not contain any data cells. It will probably produce no output.

ignoring data in spanned tbl cell
(tbl) A table cell is marked as a horizontal span (‘**s**’) or vertical span (‘^’) in the table layout, but it contains data. The data is ignored.

ignoring extra tbl data cells
(tbl) A data line contains more cells than the corresponding layout line. The data in the extra cells is ignored.

data block open at end of tbl
(tbl) A data block is opened with **T{**, but never closed with a matching **T}**. The remaining data lines of the table are all put into one cell, and any remaining cells stay empty.

Errors related to roff, mdoc, and man code

duplicate prologue macro
(mdoc) One of the prologue macros occurs more than once. The last instance overrides all previous ones.

skipping late title macro
(mdoc) The **Dt** macro appears after the first non-prologue macro. Traditional formatters cannot handle this because they write the page header before parsing the document body. Even though this technical restriction does not apply to **mandoc**, traditional semantics is preserved. The late macro is discarded including its arguments.

input stack limit exceeded, infinite loop?
(roff) Explicit recursion limits are implemented for the following features, in order to prevent infinite loops:
- expansion of nested escape sequences including expansion of strings and number registers,
- expansion of nested user-defined macros,
- and **so** file inclusion.

When a limit is hit, the output is incorrect, typically losing some content, but the parser can continue.

skipping bad character
(mdoc, man, roff) The input file contains a byte that is not a printable ascii(7) character. The message mentions the character number. The offending byte is replaced with a question mark (‘?’). Consider editing the input file to replace the byte with an ASCII transliteration of the intended character.

skipping unknown macro
(mdoc, man, roff) The first identifier on a request or macro line is neither recognized as a roff(7) request, nor as a user-defined macro, nor, respectively, as an mdoc(7) or man(7) macro. It may be mistyped or unsupported. The request or macro is discarded including its arguments.

skipping request outside macro
(roff) A **shift** or **return** request occurs outside any macro definition and has no effect.

skipping insecure request
(roff) An input file attempted to run a shell command or to read or write an external file. Such attempts are denied for security reasons.

skipping item outside list
(mdoc, eqn) An **It** macro occurs outside any **Bl** list, or an eqn(7) **above** delimiter occurs outside any pile. It is discarded including its arguments.

skipping column outside column list
(mdoc) A **Ta** macro occurs outside any **Bl -column** block. It is discarded including its arguments.

skipping end of block that is not open
(mdoc, man, eqn, tbl, roff) Various syntax elements can only be used to explicitly close blocks that have previously been opened. An mdoc(7) block closing macro, a man(7) **ME**, **RE** or **UE** macro, an eqn(7) right delimiter or closing brace, or the end of an equation, table, or roff(7) conditional request is encountered but no matching block is open. The offending request or macro is discarded.

fewer RS blocks open, skipping
(man) The **RE** macro is invoked with an argument, but less than the specified number of **RS** blocks is open. The **RE** macro is discarded.

inserting missing end of block
(mdoc, tbl) Various mdoc(7) macros as well as tables require explicit closing by dedicated macros. A block that doesn't support bad nesting ends before all of its children are properly closed. The open child nodes are closed implicitly.

appending missing end of block
(mdoc, man, eqn, tbl, roff) At the end of the document, an explicit mdoc(7) block, a man(7) next-line scope or **MT**, **RS** or **UR** block, an equation, table, or roff(7) conditional or ignore block is still open. The open block is closed implicitly.

escaped character not allowed in a name
(roff) Macro, string and register identifiers consist of printable, non-whitespace ASCII characters. Escape sequences and characters and strings expressed in terms of them cannot form part of a name. The first argument of an **am**, **as**, **de**, **ds**, **nr**, or **rr** request, or any argument of an **rm** request, or the name of a request or user defined macro being called, is terminated by an escape sequence. In the cases of **as**, **ds**, and **nr**, the request has no effect at all. In the cases of **am**, **de**, **rr**, and **rm**, what was parsed up to this

point is used as the arguments to the request, and the rest of the input line is discarded including the escape sequence. When parsing for a request or a user-defined macro name to be called, only the escape sequence is discarded. The characters preceding it are used as the request or macro name, the characters following it are used as the arguments to the request or macro.

using macro argument outside macro
(roff) The escape sequence \$ occurs outside any macro definition and expands to the empty string.

argument number is not numeric
(roff) The argument of the escape sequence \$ is not a digit; the escape sequence expands to the empty string.

NOT IMPLEMENTED: Bd -file
(mdoc) For security reasons, the **Bd** macro does not support the **-file** argument. By requesting the inclusion of a sensitive file, a malicious document might otherwise trick a privileged user into inadvertently displaying the file on the screen, revealing the file content to bystanders. The argument is ignored including the file name following it.

skipping display without arguments
(mdoc) A **Bd** block macro does not have any arguments. The block is discarded, and the block content is displayed in whatever mode was active before the block.

missing list type, using -item
(mdoc) A **Bl** macro fails to specify the list type.

argument is not numeric, using 1
(roff) The argument of a **ce** request is not a number.

argument is not a character
(roff) The first argument of a **char** request is neither a single ASCII character nor a single character escape sequence. The request is ignored including all its arguments.

missing manual name, using ""
(mdoc) The first call to **Nm**, or any call in the NAME section, lacks the required argument.

uname(3) system call failed, using UNKNOWN
(mdoc) The **Os** macro is called without arguments, and the uname(3) system call failed. As a workaround, **mandoc** can be compiled with **-DOSNAME="\"***string***\""**.

unknown standard specifier

(mdoc) An **St** macro has an unknown argument and is discarded.

skipping request without numeric argument
(roff, eqn) An **it** request or an eqn(7) **size** or **gsize** statement has a non-numeric or negative argument or no argument at all. The invalid request or statement is ignored.

excessive shift
(roff) The argument of a **shift** request is larger than the number of arguments of the macro that is currently being executed. All macro arguments are deleted and \n(.$ is set to zero.

NOT IMPLEMENTED: .so with absolute path or ".."
(roff) For security reasons, **mandoc** allows **so** file inclusion requests only with relative paths and only without ascending to any parent directory. By requesting the inclusion of a sensitive file, a malicious document might otherwise trick a privileged user into inadvertently displaying the file on the screen, revealing the file content to bystanders. **mandoc** only shows the path as it appears behind **so**.

.so request failed
(roff) Servicing a **so** request requires reading an external file, but the file could not be opened. **mandoc** only shows the path as it appears behind **so**.

skipping all arguments
(mdoc, man, eqn, roff) An mdoc(7) **Bt**, **Ed**, **Ef**, **Ek**, **El**, **Lp**, **Pp**, **Re**, **Rs**, or **Ud** macro, an **It** macro in a list that don’t support item heads, a man(7) **LP**, **P**, or **PP** macro, an eqn(7) **EQ** or **EN** macro, or a roff(7) **br**, **fi**, or **nf** request or ‘..’ block closing request is invoked with at least one argument. All arguments are ignored.

skipping excess arguments
(mdoc, man, roff) A macro or request is invoked with too many arguments:

- **Fo**, **MT**, **PD**, **RS**, **UR**, **ft**, or **sp** with more than one argument
- **An** with another argument after **-split** or **-nosplit**
- **RE** with more than one argument or with a non-integer argument
- **OP** or a request of the **de** family with more than two arguments
- **Dt** with more than three arguments
- **TH** with more than five arguments
- **Bd**, **Bk**, or **Bl** with invalid arguments

The excess arguments are ignored.

Unsupported features

input too large
(mdoc, man) Currently, **mandoc** cannot handle input files larger than its arbitrary size limit of 2^31 bytes

(2 Gigabytes). Since useful manuals are always small, this is not a problem in practice. Parsing is aborted as soon as the condition is detected.

unsupported control character
(roff) An ASCII control character supported by other roff(7) implementations but not by **mandoc** was found in an input file. It is replaced by a question mark.

unsupported escape sequence
(roff) An input file contains an escape sequence supported by GNU troff or Heirloom troff but not by **mandoc**, and it is likely that this will cause information loss or considerable misformatting.

unsupported roff request
(roff) An input file contains a roff(7) request supported by GNU troff or Heirloom troff but not by **mandoc**, and it is likely that this will cause information loss or considerable misformatting.

eqn delim option in tbl
(eqn, tbl) The options line of a table defines equation delimiters. Any equation source code contained in the table will be printed unformatted.

unsupported table layout modifier
(tbl) A table layout specification contains an '**m**' modifier. The modifier is discarded.

ignoring macro in table
(tbl, mdoc, man) A table contains an invocation of an mdoc(7) or man(7) macro or of an undefined macro. The macro is ignored, and its arguments are handled as if they were a text line.

skipping tbl in -Tman mode
(mdoc, tbl) An input file contains the **TS** macro. This message is only generated in **-T man** output mode, where tbl(7) input is not supported.

skipping eqn in -Tman mode
(mdoc, eqn) An input file contains the **EQ** macro. This message is only generated in **-T man** output mode, where eqn(7) input is not supported.

Bad command line arguments

bad command line argument
The argument following one of the **-IKMmOTW** command line options is invalid, or a *file* given as a command line argument cannot be opened.

duplicate command line argument

The **-I** command line option was specified twice.

option has a superfluous value
An argument to the **-O** option has a value but does not accept one.

missing option value
An argument to the **-O** option has no argument but requires one.

bad option value
An argument to the **-O indent** or **width** option has an invalid value.

duplicate option value
The same **-O** option is specified more than once.

no such tag
The **-O tag** option was specified but the tag was not found in any of the displayed manual pages.

-Tmarkdown unsupported for man(7) input
(man) The **-T markdown** option was specified but an input file uses the man(7) language. No output is produced for that input file.

SEE ALSO

apropos(1), man(1), eqn(7), man(7), mandoc_char(7), mdoc(7), roff(7), tbl(7)

HISTORY

The **mandoc** utility first appeared in OpenBSD 4.8. The option **-I** appeared in OpenBSD 5.2, and **-aCcfhKklMSsw** in OpenBSD 5.7.

AUTHORS

The **mandoc** utility was written by Kristaps Dzonsons <*kristaps@bsd.lv*> and is maintained by Ingo Schwarze <*schwarze@openbsd.org*>.