NAME

Pod::Man - Convert POD data to formatted *roff input

SYNOPSIS

```
use Pod::Man;
my $parser = Pod::Man->new (release => $VERSION, section => 8);

# Read POD from STDIN and write to STDOUT.
$parser->parse_file (\*STDIN);

# Read POD from file.pod and write to file.1.
$parser->parse_from_file ('file.pod', 'file.1');
```

DESCRIPTION

Pod::Man is a module to convert documentation in the POD format (the preferred language for documenting Perl) into *roff input using the man macro set. The resulting *roff code is suitable for display on a terminal using *nroff(1)*, normally via *man(1)*, or printing using *troff(1)*. It is conventionally invoked using the driver script **pod2man**, but it can also be used directly.

As a derived class from Pod::Simple, Pod::Man supports the same methods and interfaces. See *Pod::Simple* for all the details.

new() can take options, in the form of key/value pairs that control the behavior of the parser. See below for details.

If no options are given, Pod::Man uses the name of the input file with any trailing `.pod`, `.pm`, or `.pl` stripped as the man page title, to section 1 unless the file ended in `.pm` in which case it defaults to section 3, to a centered title of "User Contributed Perl Documentation", to a centered footer of the Perl version it is run with, and to a left-hand footer of the modification date of its input (or the current date if given `STDIN` for input).

Pod::Man assumes that your *roff formatters have a fixed-width font named `CW`. If yours is called something else (like `CR`), use the `fixed` option to specify it. This generally only matters for troff output for printing. Similarly, you can set the fonts used for bold, italic, and bold italic fixed-width output.

Besides the obvious pod conversions, Pod::Man also takes care of formatting func(), func(3), and simple variable references like $foo or @bar so you don't have to use code escapes for them; complex expressions like `$fred{'stuff'}` will still need to be escaped, though. It also translates dashes that aren't used as hyphens into en dashes, makes long dashes--like this--into proper em dashes, fixes "paired quotes," makes C++ look right, puts a little space between double underscores, makes ALLCAPS a teeny bit smaller in **troff**, and escapes stuff that *roff treats as special so that you don't have to.

The recognized options to new() are as follows. All options take a single argument.

center

Sets the centered page header to use instead of "User Contributed Perl Documentation".

errors

How to report errors. `die` says to throw an exception on any POD formatting error. `stderr` says to report errors on standard error, but not to throw an exception. `pod` says to include a POD ERRORS section in the resulting documentation summarizing the errors. `none` ignores POD errors entirely, as much as possible.

The default is `output`.

date

Sets the left-hand footer. By default, the modification date of the input file will be used, or the current date if stat() can't find that file (the case if the input is from `STDIN`), and the date will be formatted as `YYYY-MM-DD`.

fixed

The fixed-width font to use for verbatim text and code. Defaults to `CW`. Some systems may want `CR` instead. Only matters for **troff** output.

fixedbold

Bold version of the fixed-width font. Defaults to `CB`. Only matters for **troff** output.

fixeditalic

Italic version of the fixed-width font (actually, something of a misnomer, since most fixed-width fonts only have an oblique version, not an italic version). Defaults to `CI`. Only matters for **troff** output.

fixedbolditalic

Bold italic (probably actually oblique) version of the fixed-width font. Pod::Man doesn't assume you have this, and defaults to `CB`. Some systems (such as Solaris) have this font available as `CX`. Only matters for **troff** output.

name

Set the name of the manual page. Without this option, the manual name is set to the uppercased base name of the file being converted unless the manual section is 3, in which case the path is parsed to see if it is a Perl module path. If it is, a path like `.../lib/Pod/Man.pm` is converted into a name like `Pod::Man`. This option, if given, overrides any automatic determination of the name.

nourls

Normally, L<> formatting codes with a URL but anchor text are formatted to show both the anchor text and the URL. In other words:

```
L<foo|http://example.com/>
```

is formatted as:

```
foo <http://example.com/>
```

This option, if set to a true value, suppresses the URL when anchor text is given, so this example would be formatted as just `foo`. This can produce less cluttered output in cases where the URLs are not particularly important.

quotes

Sets the quote marks used to surround C<> text. If the value is a single character, it is used as both the left and right quote; if it is two characters, the first character is used as the left quote and the second as the right quoted; and if it is four characters, the first two are used as the left quote and the second two as the right quote.

This may also be set to the special value `none`, in which case no quote marks are added around C<> text (but the font is still changed for troff output).

release

Set the centered footer. By default, this is the version of Perl you run Pod::Man under. Note that some system an macro sets assume that the centered footer will be a modification date and will prepend something like "Last modified: "; if this is the case, you may want to set `release` to the last modified date and `date` to the version number.

section

Set the section for the `.TH` macro. The standard section numbering convention is to use 1 for user commands, 2 for system calls, 3 for functions, 4 for devices, 5 for file formats, 6 for games, 7 for miscellaneous information, and 8 for administrator commands. There is a lot of variation here, however; some systems (like Solaris) use 4 for file formats, 5 for miscellaneous information, and 7 for devices. Still others use 1m instead of 8, or some mix of both. About the only section numbers that are reliably consistent are 1, 2, and 3.

By default, section 1 will be used unless the file ends in `.pm` in which case section 3 will be selected.

stderr

Send error messages about invalid POD to standard error instead of appending a POD ERRORS section to the generated *roff output. This is equivalent to setting `errors` to `stderr` if `errors` is not already set. It is supported for backward compatibility.

utf8

By default, Pod::Man produces the most conservative possible *roff output to try to ensure that it will work with as many different *roff implementations as possible. Many *roff implementations cannot handle non-ASCII characters, so this means all non-ASCII characters are converted either to a *roff escape sequence that tries to create a properly accented character (at least for troff output) or to `X`.

If this option is set, Pod::Man will instead output UTF-8. If your *roff implementation can handle it, this is the best output format to use and avoids corruption of documents containing non-ASCII characters. However, be warned that *roff source with literal UTF-8 characters is not supported by many implementations and may even result in segfaults and other bad behavior.

Be aware that, when using this option, the input encoding of your POD source must be properly declared unless it is US-ASCII or Latin-1. POD input without an `=encoding` command will be assumed to be in Latin-1, and if it's actually in UTF-8, the output will be double-encoded. See *perlpod(1)* for more information on the `=encoding` command.

The standard Pod::Simple method parse_file() takes one argument naming the POD file to read from. By default, the output is sent to `STDOUT`, but this can be changed with the output_fd() method.

The standard Pod::Simple method parse_from_file() takes up to two arguments, the first being the input file to read POD from and the second being the file to write the formatted output to.

You can also call parse_lines() to parse an array of lines or parse_string_document() to parse a document already in memory. To put the output into a string instead of a file handle, call the output_string() method. See *Pod::Simple* for the specific details.

DIAGNOSTICS

roff font should be 1 or 2 chars, not "%s"

(F) You specified a *roff font (using `fixed`, `fixedbold`, etc.) that wasn't either one or two characters. Pod::Man doesn't support *roff fonts longer than two characters, although some *roff extensions do (the canonical versions of **nroff** and **troff** don't either).

Invalid errors setting "%s"

(F) The `errors` parameter to the constructor was set to an unknown value.

Invalid quote specification "%s"

(F) The quote specification given (the `quotes` option to the constructor) was invalid. A quote specification must be one, two, or four characters long.

POD document had syntax errors

(F) The POD document being formatted had syntax errors and the `errors` option was set to `die`.

BUGS

Encoding handling assumes that PerlIO is available and does not work properly if it isn't. The `utf8` option is therefore not supported unless Perl is built with PerlIO support.

There is currently no way to turn off the guesswork that tries to format unmarked text appropriately, and sometimes it isn't wanted (particularly when using POD to document something other than Perl). Most of the work toward fixing this has now been done, however, and all that's still needed is a user interface.

The NAME section should be recognized specially and index entries emitted for everything in that section. This would have to be deferred until the next section, since extraneous things in NAME tends to confuse various man page processors. Currently, no index entries are emitted for anything in NAME.

Pod::Man doesn't handle font names longer than two characters. Neither do most **troff** implementations, but GNU troff does as an extension. It would be nice to support as an option for those who want to use it.

The preamble added to each output file is rather verbose, and most of it is only necessary in the presence of non-ASCII characters. It would ideally be nice if all of those definitions were only output if needed, perhaps on the fly as the characters are used.

Pod::Man is excessively slow.

CAVEATS

If Pod::Man is given the `utf8` option, the encoding of its output file handle will be forced to UTF-8 if possible, overriding any existing encoding. This will be done even if the file handle is not created by Pod::Man and was passed in from outside. This maintains consistency regardless of PERL_UNICODE and other settings.

The handling of hyphens and em dashes is somewhat fragile, and one may get the wrong one under some circumstances. This should only matter for **troff** output.

When and whether to use small caps is somewhat tricky, and Pod::Man doesn't necessarily get it right.

Converting neutral double quotes to properly matched double quotes doesn't work unless there are no formatting codes between the quote marks. This only matters for troff output.

AUTHOR

Russ Allbery <rra@stanford.edu>, based *very* heavily on the original **pod2man** by Tom Christiansen <tchrist@mox.perl.com>. The modifications to work with Pod::Simple instead of Pod::Parser were originally contributed by Sean Burke (but I've since hacked them beyond recognition and all bugs are mine).

COPYRIGHT AND LICENSE

SEE ALSO

Pod::Simple, *perlpod(1)*, *pod2man(1)*, *nroff(1)*, *troff(1)*, *man(1)*, *man(7)*

Ossanna, Joseph F., and Brian W. Kernighan. "Troff User's Manual," Computing Science Technical Report No. 54, AT&T Bell Laboratories. This is the best documentation of standard **nroff** and **troff**. At the time of this writing, it's available at *http://www.cs.bell-labs.com/cm/cs/cstr.html*.

The man page documenting the man macro set may be *man(5)* instead of *man(7)* on your system. Also, please see *pod2man(1)* for extensive documentation on writing manual pages if you've not done it before and aren't familiar with the conventions.

The current version of this module is always available from its web site at *http://www.eyrie.org/~eagle/software/podlators/*. It is also part of the Perl core distribution as of 5.6.0.