NAME

perldoc - Look up Perl documentation in Pod format.

SYNOPSIS

```
perldoc [-h] [-D] [-t] [-u] [-m] [-l] [-F]
[-i] [-V] [-T] [-r]
[-d destination_file]
[-o formatname]
[-M FormatterClassName]
[-w formatteroption:value]
[-n nroff-replacement]
[-X]
[-L language_code]
PageName|ModuleName|ProgramName|URL
```

Examples:

```
perldoc -f BuiltinFunction

perldoc -L it -f BuiltinFunction

perldoc -q FAQ Keyword

perldoc -L fr -q FAQ Keyword

perldoc -v PerlVariable

perldoc -a PerlAPI
```

See below for more description of the switches.

DESCRIPTION

perldoc(1) looks up a piece of documentation in .pod format that is embedded in the perl installation tree or in a perl script, and displays it via `groff -man | $PAGER`. (In addition, if running under HP-UX, `col -x` will be used.) This is primarily used for the documentation for the perl library modules.

Your system may also have man pages installed for those modules, in which case you can probably just use the *man(1)* command.

If you are looking for a table of contents to the Perl library modules documentation, see the perltoc(1) page.

OPTIONS

-h Prints out a brief **h**elp message.

-D **D**escribes search for the item in **d**etail.

-t Display docs using plain **t**ext converter, instead of nroff. This may be faster, but it probably won't look as nice.

-u Skip the real Pod formatting, and just show the raw Pod source (**U**nformatted)

-m *module*
Display the entire module: both code and unformatted pod documentation. This may be useful if the docs don't explain a function in the detail you need, and you'd like to inspect the code directly; perldoc(1) will find the file for you and simply hand it off for display.

-l Display on**l**y the file name of the module found.

-F Consider arguments as file names; no search in directories will be performed.

-f *perlfunc(1)* 5 The **-f** option followed by the name of a perl built-in function will extract the documentation of this function from perlfunc.

Example:

```
perldoc -f sprintf
```

-q *perlfaq-search-regexp*

The **-q** option takes a regular expression as an argument. It will search the **q**uestion headings in perlfaq[1-9] and print the entries matching the regular expression.

Example:

```
perldoc -q shuffle
```

-a *perlapifunc* 5

The **-a** option followed by the name of a perl api function will extract the documentation of this function from perlapi.

Example:

```
perldoc -a newHV
```

-v *perlvar(1)* 5 The **-v** option followed by the name of a Perl predefined variable will extract the documentation of this variable from perlvar.

Examples:

```
perldoc -v '$"'
perldoc -v @+
perldoc -v DATA
```

-T This specifies that the output is not to be sent to a pager, but is to be sent directly to STDOUT.

-d *destination-filename*

This specifies that the output is to be sent neither to a pager nor to STDOUT, but is to be saved to the specified filename. Example: `perldoc(1) -oLaTeX -dtextwrapdocs.tex Text::Wrap`

-o *output-formatname*

This specifies that you want Perldoc to try using a Pod-formatting class for the output format that you specify. For example: `-oman`. This is actually just a wrapper around the `-M` switch; using `-oformatname` just looks for a loadable class by adding that format name (with different capitalizations) to the end of different classname prefixes.

For example, `-oLaTeX` currently tries all of the following classes: Pod::Perldoc::ToLaTeX Pod::Perldoc::Tolatex Pod::Perldoc::ToLatex Pod::Perldoc::ToLATEX Pod::Simple::LaTeX Pod::Simple::latex Pod::Simple::Latex Pod::Simple::LATEX Pod::LaTeX Pod::latex Pod::Latex Pod::LATEX.

-M *module-name*

This specifies the module that you want to try using for formatting the pod. The class must at least provide a `parse_from_file` method. For example: `perldoc(1) -MPod::Perldoc::ToChecker`.

You can specify several classes to try by joining them with commas or semicolons, as in `-MTk::SuperPod;Tk::Pod`.

-w *option:value* or **-w** *option*

This specifies an option to call the formatter **w**ith. For example, `-w textsize:15` will call `$formatter->textsize(15)` on the formatter object before it is used to format the object. For this to be valid, the formatter class must provide such a method, and the value you pass should be valid. (So if`textsize` e xpects an integer, and you do `-w textsize:big`, expect trouble.)

You can use `-w optionname` (without a value) as shorthand for `-w optionname:`*`TRUE`*. This is presumably useful in cases of on/off features like: `-w page_numbering`.

You can use an "=" instead of the ":", as in: `-w textsize=15`. This might be more (or less) convenient, depending on what shell you use.

-X Use an index if it is present. The **-X** option looks for an entry whose basename matches the name given on the command line in the file `$Config{archlib}/pod.idx`. The *pod.idx* file should contain fully qualified filenames, one per line.

-L *language_code*
This allows one to specify the *language code* for the desired language translation. If the `POD2::<language_code>` package isn't installed in your system, the switch is ignored. All available translation packages are to be found under the `POD2::` namespace. See POD2::IT (or POD2::FR to see how to create new localized `POD2::*` documentation packages and integrate them into Pod::Perldoc.

PageName|ModuleName|ProgramName|URL
The item you want to look up. Nested modules (such as `File::Basename` are specified either as `File::Basename` or `File/Basename`. You may also give a descriptive name of a page, such as `perlfunc(1)` For URLs, HTTP and HTTPS are the only kind currently supported.

For simple names like 'foo', when the normal search fails to find a matching page, a search with the "perl" prefix is tried as well. So "perldoc(1) intro" is enough to find/render "perlintro.pod".

-n *some-formatter*
Specify replacement for groff

-r Recursive search.

-i Ignore case.

-V Displays the version of perldoc(1) you're running.

SECURITY

Because **perldoc(1)** does not run properly tainted, and is known to have security issues, when run as the superuser it will attempt to drop privileges by setting the effective and real IDs to nobody's or nouser's account, or -2 if unavailable. If it cannot relinquish its privileges, it will not run.

ENVIRONMENT

Any switches in the `PERLDOC` environment variable will be used before the command line arguments.

Useful values for `PERLDOC` include `-oterm`, `-otext`, `-ortf`, `-oxml`, and so on, depending on what modules you have on hand; or the formatter class may be specified exactly with `-MPod::Perldoc::ToTerm` or the like.

`perldoc(1)` also searches directories specified by the `PERL5LIB` (or `PERLLIB` if `PERL5LIB` is not defined) and `PATH` environment variables. (The latter is so that embedded pods for executables, such as `perldoc(1)` itself, are available.)

In directories where either `Makefile.PL` or `Build.PL` exist, `perldoc(1)` will add `.` and `lib` first to its search path, and as long as you're not the superuser will add `blib` too. This is really helpful if you're working inside of a build directory and want to read through the docs even if you have a version of a module previously installed.

`perldoc(1)` will use, in order of preference, the pager defined in `PERLDOC_PAGER`, `MANPAGER`, or `PAGER` before trying to find a pager on its own. (`MANPAGER` is not used if `perldoc(1)` was told to display plain text or unformatted pod.)

When using perldoc(1) in it's `-m` mode (display module source code), `perldoc(1)` will attempt to use the pager set in `PERLDOC_SRC_PAGER`. A useful setting for this command is your favorite editor as in `/usr/bin/nano`. (Don't judge me.)

One useful value for `PERLDOC_PAGER` is `less -+C -E`.

Having PERLDOCDEBUG set to a positive integer will make perldoc(1) emit even more descriptive output than the `-D` switch does; the higher the number, the more it emits.

CHANGES

Up to 3.14_05, the switch **-v** was used to produce verbose messages of **perldoc(1)** operation, which is now enabled by **-D**.

SEE ALSO

perlpod(1), Pod::Perldoc

AUTHOR

Current maintainer: Mark Allen <mallen@cpan.org>

Past contributors are: brian d foy <bdfoy@cpan.org> Adriano R. Ferreira <ferreira@cpan.org>, Sean M. Burke <sburke@cpan.org>, Kenneth Albanowski <kjahds@kjahds.com>, Andy Dougherty <doughera@lafcol.lafayette.edu>, and many others.