



**Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!**

### ***Rocky Enterprise Linux 9.2 Manual Pages on command 'Pod::Checker.3perl'***

#### ***\$ man Pod::Checker.3perl***

Pod::Checker(3perl) Perl Programmers Reference Guide Pod::Checker(3perl)

#### NAME

Pod::Checker - check pod documents for syntax errors

#### SYNOPSIS

```
use Pod::Checker;

$syntax_okay = podchecker($filepath, $outputpath, %options);

my $checker = Pod::Checker->new(%options);

$checker->parse_from_file($filepath, \*STDERR);
```

#### OPTIONS/ARGUMENTS

\$filepath is the input POD to read and \$outputpath is where to write POD syntax error messages. Either argument may be a scalar indicating a file-path, or else a reference to an open filehandle. If unspecified, the input-file it defaults to "`\*STDIN`", and the output-file defaults to "`\*STDERR`".

#### podchecker()

This function can take a hash of options:

`-warnings => val`

Turn warnings on/off. val is usually 1 for on, but higher values trigger additional warnings. See "Warnings".

`-quiet => val`

If "val" is true, do not print any errors/warnings.

#### DESCRIPTION

podchecker will perform syntax checking of Perl5 POD format documentation.

Curious/ambitious users are welcome to propose additional features they wish to see in

Pod::Checker and podchecker and verify that the checks are consistent with perlpod.

The following checks are currently performed:

- ? Unknown '=xxxx' commands, unknown 'X<...>' interior-sequences, and unterminated interior sequences.
- ? Check for proper balancing of "=begin" and "=end". The contents of such a block are generally ignored, i.e. no syntax checks are performed.
- ? Check for proper nesting and balancing of "=over", "=item" and "=back".
- ? Check for same nested interior-sequences (e.g. "L<...L<...>...>").
- ? Check for malformed or non-existing entities "E<...>".
- ? Check for correct syntax of hyperlinks "L<...>". See perlpod for details.
- ? Check for unresolved document-internal links. This check may also reveal misspelled links that seem to be internal links but should be links to something else.

## DIAGNOSTICS

### Errors

- ? empty =headn  
A heading ("=head1" or "=head2") without any text? That ain't no heading!
- ? =over on line N without closing =back
- ? You forgot a '=back' before '=headN'
- ? =over is the last thing in the document?!  
The "=over" command does not have a corresponding "=back" before the next heading ("=head1" or "=head2") or the end of the file.
- ? '=item' outside of any '=over'
- ? =back without =over  
An "=item" or "=back" command has been found outside a "=over"/"=back" block.
- ? Can't have a 0 in =over N  
You need to indent a strictly positive number of spaces, not 0.
- ? =over should be: '=over' or '=over positive\_number'  
Either have an argumentless =over, or have its argument a strictly positive number.
- ? =begin TARGET without matching =end TARGET  
A "=begin" command was found that has no matching =end command.
- ? =begin without a target?  
A "=begin" command was found that is not followed by the formatter specification.
- ? =end TARGET without matching =begin.

- A standalone "=end" command was found.
- ? '=end' without a target?
- '=end' directives need to have a target, just like =begin directives.
- ? '=end TARGET' is invalid.
- TARGET needs to be one word
- ? =end CONTENT doesn't match =begin TARGET
- CONTENT needs to match =begin's TARGET.
- ? =for without a target?
- There is no specification of the formatter after the "=for" command.
- ? unresolved internal link NAME
- The given link to NAME does not have a matching node in the current POD. This also happened when a single word node name is not enclosed in "".
- ? Unknown directive: CMD
- An invalid POD command has been found. Valid are "=head1", "=head2", "=head3", "=head4", "=over", "=item", "=back", "=begin", "=end", "=for", "=pod", "=cut"
- ? Deleting unknown formatting code SEQ
- An invalid markup command has been encountered. Valid are: "B<>", "C<>", "E<>", "F<>", "I<>", "L<>", "S<>", "X<>", "Z<>"
- ? Unterminated SEQ<> sequence
- An unclosed formatting code
- ? An E<...> surrounding strange content
- The STRING found cannot be interpreted as a character entity.
- ? An empty E<>
- ? An empty "L<>"
- ? An empty X<>
- There needs to be content inside E, L, and X formatting codes.
- ? Spurious text after =pod / =cut
- The commands "=pod" and "=cut" do not take any arguments.
- ? =back doesn't take any parameters, but you said =back ARGUMENT
- The "=back" command does not take any arguments.
- ? =pod directives shouldn't be over one line long! Ignoring all N lines of content
- Self explanatory
- ? =cut found outside a pod block.

A '=cut' directive found in the middle of non-POD

? Invalid =encoding syntax: CONTENT

Syntax error in =encoding directive

## Warnings

These may not necessarily cause trouble, but indicate mediocre style.

? nested commands CMD<...CMD<...>...>

Two nested identical markup commands have been found. Generally this does not make sense.

? multiple occurrences (N) of link target name

The POD file has some "=item" and/or "=head" commands that have the same text. Potential hyperlinks to such a text cannot be unique then. This warning is printed only with warning level greater than one.

? line containing nothing but whitespace in paragraph

There is some whitespace on a seemingly empty line. POD is very sensitive to such things, so this is flagged. vi users switch on the list option to avoid this problem.

? =item has no contents

There is a list "=item" that has no text contents. You probably want to delete empty items.

? You can't have =items (as at line N) unless the first thing after the =over is an =item

A list introduced by "=over" starts with a text or verbatim paragraph, but continues with "=item"s. Move the non-item paragraph out of the "=over"/"=back" block.

? Expected '=item EXPECTED VALUE'

? Expected '=item \*'

? Possible =item type mismatch: 'x' found leading a supposed definition =item

A list started with e.g. a bullet-like "=item" and continued with a numbered one. This is obviously inconsistent. For most translators the type of the first "=item" determines the type of the list.

? You have '=item x' instead of the expected '=item N'

Erroneous numbering of =item numbers; they need to ascend consecutively.

? Unknown E content in E<CONTENT>

A character entity was found that does not belong to the standard ISO set or the POD specials "verbar" and "sol". Currently, this warning only appears if a character

entity was found that does not have a Unicode character. This should be fixed to adhere to the original warning.

? empty `=over/=back` block

The list opened with `"=over"` does not contain anything.

? empty section in previous paragraph

The previous section (introduced by a `"=head"` command) does not contain any valid content. This usually indicates that something is missing. Note: A `"=head1"` followed immediately by `"=head2"` does not trigger this warning.

? Verbatim paragraph in NAME section

The NAME section (`"=head1 NAME"`) should consist of a single paragraph with the script/module name, followed by a dash ``-'` and a very short description of what the thing is good for.

? `=headn` without preceding higher level

For example if there is a `"=head2"` in the POD file prior to a `"=head1"`.

? A non-empty `Z<>`

The `"Z<>"` sequence is supposed to be empty. Caveat: this issue is detected in `Pod::Simple` and will be flagged as an ERROR by any client code; any contents of `"Z<...>"` will be disregarded, anyway.

## Hyperlinks

There are some warnings with respect to malformed hyperlinks:

? ignoring leading/trailing whitespace in link

There is whitespace at the beginning or the end of the contents of `L<...>`.

? alternative text/node `'%s'` contains non-escaped `|` or `/`

The characters `"|"` and `"/"` are special in the `L<...>` context. Although the hyperlink parser does its best to determine which `"/"` is text and which is a delimiter in case of doubt, one ought to escape these literal characters like this:

```
/  E<sol>
```

```
|  E<verbar>
```

Note that the line number of the error/warning may refer to the line number of the start of the paragraph in which the error/warning exists, not the line number that the error/warning is on. This bug is present in errors/warnings related to formatting codes.

This should be fixed.

podchecker returns the number of POD syntax errors found or -1 if there were no POD commands at all found in the file.

## EXAMPLES

See "SYNOPSIS"

## SCRIPTS

The podchecker script that comes with this distribution is a lean wrapper around this module. See the online manual with

```
podchecker -help
```

```
podchecker -man
```

## INTERFACE

While checking, this module collects document properties, e.g. the nodes for hyperlinks ("=headX", "=item") and index entries ("X<>"). POD translators can use this feature to syntax-check and get the nodes in a first pass before actually starting to convert. This is expensive in terms of execution time, but allows for very robust conversions.

Since v1.24 the Pod::Checker module uses only the poderror method to print errors and warnings. The summary output (e.g. "Pod syntax OK") has been dropped from the module and has been included in podchecker (the script). This allows users of Pod::Checker to control completely the output behavior. Users of podchecker (the script) get the well-known behavior.

v1.45 inherits from Pod::Simple as opposed to all previous versions inheriting from Pod::Parser. Do not use Pod::Simple's interface when using Pod::Checker unless it is documented somewhere on this page. I repeat, DO NOT USE POD::SIMPLE'S INTERFACE.

The following list documents the overrides to Pod::Simple, primarily to make Pod::Coverage

happy:

end\_B

end\_C

end\_Document

end\_F

end\_I

end\_L

end\_Para

end\_S

end\_X

end\_fcode  
end\_for  
end\_head  
end\_head1  
end\_head2  
end\_head3  
end\_head4  
end\_item  
end\_item\_bullet  
end\_item\_number  
end\_item\_text  
handle\_pod\_and\_cut  
handle\_text  
handle\_whiteline  
hyperlink  
scream  
start\_B  
start\_C  
start\_Data  
start\_F  
start\_I  
start\_L  
start\_Para  
start\_S  
start\_Verbatim  
start\_X  
start\_fcode  
start\_for  
start\_head  
start\_head1  
start\_head2  
start\_head3  
start\_head4

start\_item\_bullet

start\_item\_number

start\_item\_text

start\_over

start\_over\_block

start\_over\_bullet

start\_over\_empty

start\_over\_number

start\_over\_text

whine

"Pod::Checker->new( %options )"

Return a reference to a new Pod::Checker object that inherits from Pod::Simple and is used for calling the required methods later. The following options are recognized:

"-warnings => num"

Print warnings if "num" is true. The higher the value of "num", the more warnings are printed. Currently there are only levels 1 and 2.

"-quiet => num"

If "num" is true, do not print any errors/warnings. This is useful when Pod::Checker is used to munge POD code into plain text from within POD formatters.

"\$checker->poderror( @args )"

"\$checker->poderror( {%opts}, @args )"

Internal method for printing errors and warnings. If no options are given, simply prints "@\_". The following options are recognized and used to form the output:

-msg

A message to print prior to @args.

-line

The line number the error occurred in.

-file

The file (name) the error occurred in. Defaults to the name of the current file being processed.

-severity

The error level, should be 'WARNING' or 'ERROR'.

"\$checker->num\_errors()"

Set (if argument specified) and retrieve the number of errors found.

"\$checker->num\_warnings()"

Set (if argument specified) and retrieve the number of warnings found.

"\$checker->name()"

Set (if argument specified) and retrieve the canonical name of POD as found in the

"=head1 NAME" section.

"\$checker->node()"

Add (if argument specified) and retrieve the nodes (as defined by "=headX" and

"=item") of the current POD. The nodes are returned in the order of their occurrence.

They consist of plain text, each piece of whitespace is collapsed to a single blank.

"\$checker->idx()"

Add (if argument specified) and retrieve the index entries (as defined by "X<>") of

the current POD. They consist of plain text, each piece of whitespace is collapsed to

a single blank.

"\$checker->hyperlinks()"

Retrieve an array containing the hyperlinks to things outside the current POD (as

defined by "L<>").

Each is an instance of a class with the following methods:

line()

Returns the approximate line number in which the link was encountered

type()

Returns the type of the link; one of: "url" for things like "http://www.foo", "man"

for man pages, or "pod".

page()

Returns the linked-to page or url.

node()

Returns the anchor or node within the linked-to page, or an empty string ("") if none

appears in the link.

## AUTHOR

Please report bugs using <<http://rt.cpan.org>>.

Brad Appleton <[bradapp@enteract.com](mailto:bradapp@enteract.com)> (initial version), Marek Rouchal <[marekr@cpan.org](mailto:marekr@cpan.org)>,

Marc Green <[marcgreen@cpan.org](mailto:marcgreen@cpan.org)> (port to Pod::Simple) Ricardo Signes <[rjbs@cpan.org](mailto:rjbs@cpan.org)> (more

porting to Pod::Simple) Karl Williamson <[khw@cpan.org](mailto:khw@cpan.org)> (more porting to Pod::Simple)

Based on code for Pod::Text::pod2text() written by Tom Christiansen <tchrist@mox.perl.com>

perl v5.34.0

2023-11-23

Pod::Checker(3perl)