↖ Topic Index

🚧 Emacs support for Perl 🚧

Description	Keystroke	Function	Note
Perl Editing • **Activate Perl** ☛ • **Select major mode** ☛ • **PEL prefix key** • **Speedbar** • **Indentation** control ☛	Emacs provides two major modes for Perl: **perl-mode** (Emacs default, simpler mode) and **cperl-mode**. 📦 The **HaraldJoerg/cperl-mode** external package has support for new Perl language features. PEL activates Perl support with the **pel-use-perl** user-options. When turned on: • PEL provides supports all of them: **pel-perl-mode** selects the mode, using 📦 **HaraldJoerg/cperl-mode** by default because it has the best support for Perl. • ⚠️ After using **HaraldJoerg/cperl-mode** if you want to revert to Emacs' own **cperl-mode**, remove the cperl-mode.el* files from ~/.emacs.d/utils • The **<f11> SPC P** prefix is made available. In a perl buffer using either major-modes for Perl these commands are accessible via the **<f12>** key. • It activates the ability to activate minor modes for the perl major mode through the PEL **pel-perl-activates-minor-modes** user-option. • If **pel-use-speedbar** is on, speedbar support for Perl is activated. 🎛 Perl indentation is controlled by the following user-options: 🚧 • **perl-indent-level** sets the number of columns used for indentation. It defaults to 4. • PEL sets **tab-width** with the same value in Perl buffers so that manual indentation commands use the same number of columns to indent. • **pel-perl-use-tabs** controls whether hard tabs are used for indentation (nil by default). • PEL sets **indent-tabs-mode** with the value of **pel-perl-use-tabs** in perl buffers.		
Open this PDF file. See also: **Help/Info**	**<f11> SPC P <f1>** **<f12> <f1>**	(**pel-help-pdf** &optional OPEN-WEB-PAGE)	Open the **Pl - Perl** local PDF. If the prefix argument (like **C-u** or **M--**) is used, then it opens the remote GitHub hosted raw PDF instead. If the **pel-flip-help-pdf-arg** user-option is set it's the other way around.
Customize PEL Perl support	**<f11> SPC P <f2>** **<f12> <f2>**	(**pel-customize-pel** &optional OTHER-WINDOW)	Customize PEL Perl support. • If OTHER-WINDOW is non-nil (use **C-u**), display in another window.
Customize Emacs Perl support	**<f11> SPC P <f3>** **<f12> <f3>**	(**pel-customize-library** &optional OTHER-WINDOW)	Customize Emacs Perl support: perl, cperl, electricity. • If OTHER-WINDOW is non-nil (use **C-u**), display in another window.
Perl Doc	With Perl info and man files installed, Emacs **Help/Info** man support provides access to the Perl documentation available on your system. ☝ Inside Emacs **you can use man instead of perldoc**: for example, **man perlintro** provides the same information as given by the **perldoc perlintro** command. The Perl Documentation web page provides the complete information on-line: • Overview • Tutorials • Reference Manual • Operators • Functions • Variables Modules Utilities • History • Internals and C Language Interface • Language-specific Miscellaneous Platform-specific		
Comments	Perl comments use '#' at the beginning of the line.		
Toggle display of comments in buffer or active region	**<f11> ; ;**	(**hide/show-comments-toggle** &optional START END)	Toggle hiding/showing of comments in the active region or whole buffer. • If the region is active then toggle in the region. Otherwise, in the whole buffer. 📦 This requires the **hide-comnt.el** package (see **Comments**). PEL activates it when the **pel-use-hide-comnt** user option is **t**.
Perl base	These commands are provided by Emacs basic Perl support.		
Move to beginning of function	**C-M-a**	(**perl-beginning-of-function** &optional ARG)	Move backward to next beginning-of-function, or as far as possible. With argument, repeat that many times; negative args move forward.
To end of function	**C-M-e**	(**perl-end-of-function** &optional ARG)	Move forward to next end-of-function. • The end of a function is found by moving forward from the beginning of one. • With argument, repeat that many times; negative args move backward.
Mark function	**C-M-h**	(**perl-mark-function**)	Put mark at end of Perl function, point at beginning
Indent	**C-M-q**	(**perl-indent-exp**)	Indent each line of the Perl grouping following point. ☝ You can also indent any are of Perl code by marking that area and using the tab key.
cperl-mode	PEL supports 2 **cperl-mode** implementations: Emacs' **cperl-mode** and 📦 **HaraldJoerg/cperl-mode** (more complete) activated by the **pel-use-perl** user-option.		
Show command info	**C-c C-h F**	(**cperl-info-on-command** COMMAND)	Show documentation for Perl command COMMAND in other window. • If perl-info buffer is shown in some frame, uses this frame. • Customized by setting variables '**cperl-shrink-wrap-info-frame**', '**cperl-max-help-size**'.
perldoc at point	**C-c C-h P**	(**cperl-perldoc-at-point**)	Run a 'perldoc' on the word around point to show information about that Perl word.
	C-c C-h p	(**cperl-perldoc** WORD &optional SECTION)	Prompt for for (default to word at point). Show information about the selected word with 'perldoc'.
Show Perl doc	**C-c C-h f**	(**cperl-info-on-current-command**)	Show documentation for Perl command at point in other window.
Help on symbol at point	**C-c C-h v**	(**cperl-get-help**)	Get one-line docs on the symbol at the point. • The data for these docs is a little bit obsolete and may be in fact longer than a line.
toggle Perl auto-help	**C-c C-h a**	(**cperl-toggle-autohelp**)	Toggle the state of Auto-Help on Perl constructs (put in the message area). • Delay of auto-help controlled by '**cperl-lazy-help-time**'.
Toggle auto-newline	**C-c C-a**	(**cperl-toggle-auto-newline**)	Toggle the state of '**cperl-auto-newline**'.
Toggle electric mode	**C-c C-e**	(**cperl-toggle-electric**)	Toggle the state of parentheses doubling in CPerl mode. When typing an opening parens character the closing one is automatically entered.
Toggle auto-fill mode	**C-c C-f** • **<f11> t f a** • **<f11> RET**	(**auto-fill-mode** &optional ARG)	Toggle automatic line breaking (Auto Fill mode). • With a prefix argument, enable Auto Fill mode if the prefix argument is positive, and disable it otherwise. • When Auto Fill mode is enabled, inserting a space at a column beyond 'current-fill-column' automatically breaks the line at a previous space.
Toggle keyword expansion	**C-c C-k**	(**cperl-toggle-abbrev**)	Toggle the state of automatic keyword expansion in CPerl mode.
Toggle space fix	**C-c C-w**	(**cperl-toggle-construct-fix**)	Toggle whether '**indent-region**'/'**indent-sexp**' fix whitespace too.
Find code for whitespace	**C-c C-b**	(**cperl-find-bad-style**)	Find places in the buffer where insertion of a whitespace may help. • Prompts user for insertion of spaces. Currently it is tuned to C and Perl syntax.
Spell-check here-docs	**C-c C-d**	(**cperl-here-doc-spell**)	Spell-check HERE-documents in the Perl buffer. • If a region is highlighted, restricts to the region.
Perl new line	**C-c C-j**	(**cperl-linefeed**)	Go to end of line, open a new line and indent appropriately. • If in POD, insert appropriate lines. This is a convenience replacement for typing carriage return. It places you in the next line with proper indentation, or if you type it inside the inline block of control construct, like foreach (@lines) {print; print} and you are on a boundary of a statement inside braces, it will transform the construct into a multiline and will place you into an appropriately indented blank line. • If you need a usual 'newline-and-indent' behavior, it is on **C-j**, see documentation on 'cperl-electric-linefeed'.

Description	Keystroke	Function	Note
Narrows to here-doc	`C-c C-n`	(**cperl-narrow-to-here-doc** &optional POS)	Narrows editing region to the HERE-DOC at POS. POS defaults to the point.
Spell-check POD documentation	`C-c C-p`	(**cperl-pod-spell** &optional DO-HERES)	Spell-check POD documentation. • If invoked with prefix argument, will do HERE-DOCs instead. • If a region is highlighted, restricts to the region.
Refactor: if (A) {B}' ➥ 'B if A;'	`C-c C-t`	(**cperl-invert-if-unless**)	Change 'if (A) {B}' into 'B if A;' etc (or visa versa) if possible. • If the cursor is not on the leading keyword of the BLOCK flavor of construct, will assume it is the STATEMENT flavor, so will try to find the appropriate statement modifier.
Move to next interpolated	`C-c C-v`	(**cperl-next-interpolated-REx** &optional SKIP BEG LIMIT)	Move point to next REx which has interpolated parts. • SKIP is a list of possible types to skip, BEG and LIMIT are the starting point and the limit of search (default to point and end of buffer). • SKIP may be a number, then it behaves as list of numbers up to SKIP; this semantic may be used as a numeric argument. • Types are 0 for / $rex /o (interpolated once), 1 for /$rex/ (if $rex is a result of qr//, this is not a performance hit), t for the rest
	`C-c C-x`	(**cperl-next-interpolated-REx-0**)	Move point to next REx which has interpolated parts without //o.
	`C-c C-y`	(**cperl-next-interpolated-REx-1**)	Move point to next REx which has interpolated parts without //o. • Skips REexes consisting of one interpolated variable. • Note that skipped REexen are not performance hits.
Insert matching parens **(toggle with C-c C-e)**	• `(` • `<` • `[`	(**cperl-electric-paren** ARG)	Insert an opening parenthesis or a matching pair of parentheses. • Controlled by '**cperl-electric-parens**' and '**cperl-hairy**' user-options.
	• `)` • `]`	(**cperl-electric-rparen** ARG)	Insert a matching pair of parentheses if marking is active. • If not, or if we are not at the end of marking range, would self-insert. • Controlled by '**cperl-electric-parens**' and '**cperl-hairy**' user-options.
Insert : and indent	`:`	(**cperl-electric-terminator** ARG)	Insert character and correct line's indentation.
Insert ; and indent	`;`	(**cperl-electric-semi** ARG)	Insert character and correct line's indentation.
Insert { and indent	`{`	(**cperl-electric-lbrace** ARG &optional END)	Insert character, correct line's indentation, correct quoting by space.
Insert } and indent	`}`	(cperl-electric-brace ARG &optional ONLY-BEFORE)	Insert character and correct line's indentation. • If ONLY-BEFORE and '**cperl-auto-newline**', will insert newline before the place (even in empty line), but not after. If after ")" and the inserted char is "{", insert extra newline before only if '**cperl-extra-newline-before-brace**'.
Deleted and possibly untabify	`DEL`	(**cperl-electric-backspace** ARG)	Backspace, or remove whitespace around the point inserted by an electric key. • Will untabify if '**cperl-electric-backspace-untabify**' is non-nil.
Indent Perl code	`TAB`	(**cperl-indent-command** &optional WHOLE-EXP)	Indent current line as Perl code, or in some cases insert a tab character. • If '**cperl-tab-always-indent**' is non-nil (the default), always indent current line. Otherwise, indent the current line only if point is at the left margin or in the line's indentation; otherwise insert a tab. • A numeric argument, regardless of its value, means indent rigidly all the lines of the expression starting after point so that this line becomes properly indented. The relative indentation among the lines of the expression are preserved.
Newline and indent	`C-j`	(**newline-and-indent**)	Insert a newline, then indent according to major mode. • Indentation is done using the value of 'indent-line-function'. In programming language modes, this is the same as TAB.
Indent	`C-M-q`	(**cperl-indent-exp**)	Simple variant of indentation of continued-sexp. • Will not indent comment if it starts at '**comment-indent**' or looks like continuation of the comment on the previous line. • If '**cperl-indent-region-fix-constructs**', will improve spacing on conditional/loop constructs.
Indent	`C-M-\`	(**cperl-indent-region** START END)	Simple variant of indentation of region in CPerl mode. • Should be slow. Will not indent comment if it starts at '**comment-indent**' or looks like continuation of the comment on the previous line. • Indents all the lines whose first character is between START and END inclusive. • If '**cperl-indent-region-fix-constructs**', will improve spacing on conditional/loop constructs.
Lineup	`C-M-|`	(**cperl-lineup** BEG END &optional STEP MINSHIFT)	Lineup construction in a region. • Beginning of region should be at the start of a construction. • All first occurrences of this construction in the lines that are partially contained in the region are lined up at the same column. • MINSHIFT is the minimal amount of space to insert before the construction. • STEP is the tabwidth to position constructions. • If STEP is nil, 'cperl-lineup-step' will be used (or 'cperl-indent-level', if 'cperl-lineup-step' is nil). • Will not move the position at the start to the left.

Emacs & Perl — References

Document	Notes
Perl @ Wikipedia	
perl.org	
Learn Perl @ perl.org	• Perl Intro - a quick introduction to Perl • Online Perl books • Beginning Perl
Is Perl still relevant? **Most probably.**	• What makes Perl relevant in 2022? @ Stackoverflow blog. • Perl is dying quick. Could be extinct by 2023. The HFT Guy, 2019. (which includes several invalid points). • My point is that Perl was popular, there's a lot of Perl code still being used and it's worth knowing and being able to write and edit Perl code (which was certainly not the first programming language as state by the article). But anyway, the post represents a point of view (and has many people commenting on it).
Perl File name extensions	.plx, .pls, .pl, .pm, .xs, .t, .pod, .cgi, .psgi
Perl programs	• Perl command line options
• **perl**	Perl language interpreter
• **perlbug / perlthanks**	Describes how to submit bug report on Perl
• **perldoc**	Print Perl Documentation, looking it up in the .pod format embedded in perl installation. Support following options: • -f : built-in functions • -q : FAQ keyword search • -v : variable • -a perl API

Document	Notes
• **perlivp**	• Perl Installation Verification Procedure : checks Perl installation. • Part of **perl-level** package.