

VIM-PLUG-IN
perl-support.vim
 VERSION 5.3.2

HOT KEYS

Key mappings for Vim and gVim.

Plug-in: <http://vim.sourceforge.net>

Fritz Mehner (mehner.fritz@fh-swf.de)

(i) insert mode, (n) normal mode, (v) visual mode

<i>Perl</i>	
<code>\ft</code>	file tests (n,i)
<i>Help</i>	
<code>\h</code>	read perldoc for word under cursor (n,i)
<code>\hp</code>	help (plug-in) (n,i)
<i>Comments</i>	
<code>[n]\cl</code>	end-of-line comment (n, v, i)
<code>[n]\cj</code>	adjust end-of-line comments (n, v, i)
<code>\cs</code>	set end-of-line comment col. (n)
<code>[n]\cc</code>	code ↔ comment (n, v)
<code>\cb</code>	code block → comment (n, v)
<code>\cub</code>	uncomment code block (n)
<code>\cfr</code>	frame comment (n, i)
<code>\cfu</code>	function description (n, i)
<code>\cme</code>	method description (n, i)
<code>\chpl</code>	file header (.pl) (n)
<code>\chpm</code>	file header (.pm) (n)
<code>\cht</code>	file header (.t) (n)
<code>\chpo</code>	file header (.pod) (n)
<code>\cd</code>	date (n, i)
<code>\ct</code>	date & time (n, i)
<code>\ck</code>	keyword comments (n, i)
<code>\cma</code>	plug-in macros (n, i)

<i>Statements</i>	
<code>\sd</code>	do { } while (n, v, i)
<code>\sf</code>	for { } (n, v, i)
<code>\sfe</code>	foreach { } (n, v, i)
<code>\si</code>	if { } (n, v, i)
<code>\sie</code>	if { } else { } (n, v, i)
<code>\se</code>	else { } (n, v, i)
<code>\sei</code>	elsif { } (n, v, i)
<code>\su</code>	unless { } (n, v, i)
<code>\sue</code>	unless { } else { } (n, v, i)
<code>\st</code>	until { } (n, v, i)
<code>\sw</code>	while { } (n, v, i)
<code>\sg</code>	given { } (n, v, i)
<code>\swh</code>	when { } (n, v, i)
<i>Idioms</i>	
<code>\id</code>	my \$; (n, i)
<code>\ida</code>	my \$ = ; (n, i)
<code>\idd</code>	my (\$, \$); (n, i)
<code>\ia</code>	my @; (n, i)
<code>\iaa</code>	my @ = (, ,); (n, i)
<code>\ih</code>	my %; (n, i)
<code>\iha</code>	my % = (=>, =>,); (n, i)
<code>\ir</code>	my \$rgx_ = q//; (n, i)
<code>\im</code>	\$ =~ m//xm (n, i)
<code>\is</code>	\$ =~ s//xm (n, i)
<code>\it</code>	\$ =~ tr//xm (n, i)
<code>\isu</code>	subroutine (n, v, i)
<code>\ip</code>	print "... \n"; (n, i)
<code>\ii</code>	open input file (n, v, i)
<code>\io</code>	open output file (n, v, i)
<code>\ipi</code>	open pipe (n, v, i)
<i>Snippet</i>	
<code>\nr</code>	read code snippet (n, i)
<code>\nv</code>	view code snippet (n, i)
<code>\nw</code>	write code snippet (n, v, i)
<code>\ne</code>	edit code snippet (n, i)
<code>\ntl</code>	edit local templates (n, i)
<code>\ntr</code>	reread the templates (n, i)
<code>\nts</code>	choose template style (n, i)
<code>\njt</code>	insert jump tag (n, i)
<code>\nxs</code>	regex snippet template (n, i)

<i>Regular Expressions</i>	
<code>xpc</code>	POSIX classes (n, i)
<code>xup</code>	Unicode properties (n, i)
<code>xex</code>	extended Regex (n, i)
<code>xms</code>	metasymbols (n, i)
<code>[n]\xr</code>	pick up Regex (n, v)
<code>[n]\xs</code>	pick up string (n, v)
<code>\xf</code>	pick up flag(s) (n, v)
<code>\xm</code>	match (n)
<code>\xmm</code>	match multiple (Regex/target) (n)
<code>\xe</code>	explain Regex (n, v)
<i>Special Variables</i>	
<code>\vb</code>	basics (n, i)
<code>\ve</code>	errors (n, i)
<code>\vf</code>	files (n, i)
<code>\vid</code>	IDs (n, i)
<code>\vio</code>	IO (n, i)
<code>\vr</code>	regexp (n, i)
<code>\vs</code>	POSIX signals (n, i)
<code>\vue</code>	use English (n, i)
<i>POD</i>	
<code>\ppc</code>	pod, cut (n, i)
<code>\pfc</code>	for, cut (n, i)
<code>\pbh</code>	begin html, end (n, i)
<code>\pbm</code>	begin man, end (n, i)
<code>\pbt</code>	begin text, end (n, i)
<code>\ph1</code>	head1 (n, i)
<code>\ph2</code>	head2 (n, i)
<code>\ph3</code>	head3 (n, i)
<code>\pob</code>	over, back (n, i)
<code>\pi</code>	item (n, i)
<code>\pod</code>	run podchecker (n, i)
<code>\podh</code>	convert POD data to .html file (n, i)
<code>\podm</code>	Convert POD data to *roff input (n, i)
<code>\podt</code>	Convert POD data to ASCII text (n, i)
<code>\pm</code>	markup sequences (n, i)

<i>Profiling</i>		
<code>\rps</code>	run <code>SmallProf</code>	(n, i)
<code>\rpss</code>	sort <code>SmallProf</code> report	(n, i)
<code>\rpso</code>	open existing <code>SmallProf</code> results	(n, i)
<code>\rpf</code>	run <code>FastProf</code>	(n, i)
<code>\rpfs</code>	sort <code>FastProf</code> report	(n, i)
<code>\rpfo</code>	open existing <code>FastProf</code> results	(n, i)
<code>\rpn</code>	run <code>NYTProf</code>	(n, i)
<code>\rpns</code>	sort <code>NYTProf</code> report	(n, i)
<code>\rpno</code>	open existing <code>NYTProf</code> results	(n, i)
<code>\rpnh</code>	browse HTML files (<code>NYTProf</code>)	(n, i)
<i>Run</i>		
<code>\rr</code>	update file, run script	(n, i)
<code>\rs</code>	update file, check syntax	(n, i)
<code>\ra</code>	set command line arguments	(n, i)
<code>\rw</code>	set Perl cmd. line switches	(n, i)
<code>\re</code>	make script executable/not exec.	(n, i)
<code>\rd</code>	start debugger	(n, i)
<code>\ri</code>	show installed Perl modules	(n, i)
<code>\rg</code>	generate Perl module list	(n, i)
<code>\ry</code>	run <code>perltidy</code>	(n, v, i)
<code>\rpc</code>	run <code>perlcritic</code>	(n, i)
<code>\rpcs</code>	set <code>perlcritic</code> severity	(n, i)
<code>\rpcv</code>	set <code>perlcritic</code> verbosity	(n, i)
<code>\rpco</code>	set <code>perlcritic</code> options	(n, i)
<code>\rt</code>	save buffer with timestamp	(n, i)
<code>\rh</code>	hardcopy buffer	(n, v, i)
<code>\rk</code>	settings and hotkeys	(n, i)
<code>\rx</code>	set xterm size	(n, i <small>GUI only</small>)
<code>\ro</code>	change output destination	(n, i)
<i>Tool Box</i>		
<code>\rm</code>	run <code>make</code>	(n, i)
<code>\rcm</code>	choose makefile	(n, i)
<code>\rmc</code>	<code>make clean</code>	(n, i)
<code>\rma</code>	command line arguments for <code>make</code>	(n, i)

Run

Specify command line arguments for the script in the current buffer. Use tab expansion to choose a file or a directory.

`:PerlScriptArguments`

Specify command line switches for the Perl interpreter.

`:PerlSwitches`

Perlcritic

Ex commands for `perlcritic` (version 1.01+)

Use tab expansion to choose the severity or the verbosity.

`:CriticSeverity` 1 2 3 4 5
brutal cruel harsh stern gentle

`:CriticVerbosity` 1 ... 11

`:CriticOptions` option(s), see `perlcritic(1)`

Regular Expression Tester

Ex command for the regular expression tester. Set control character replacements for newline and tabulator used to display the results of a match, e.g.:

`:RegexSubstitutions` '\$~'

Profiling

The following ex commands can be used to sort a profiler report in the quickfix window.

Use tab expansion to choose the sort criterion or the file name.

For `Devel::SmallProf`

`:SmallProfSort` file-name|line-number|line-count|time|ctime

For `Devel::FastProf`

`:FastProfSort` file-name|line-number|time|line-count

For `Devel::NYTProf`

`:NYTProfCSV` Read a CSV-file.

`:NYTProfHTML` Read the HTML-reports with an external viewer (GUI only).

`:NYTProfSort` file-name|line-number|time|calls|time-call