Vim-Plugin
c-support.vim
Version 5.0.3

Hot keys

Key mappings for Vim without GUI.
All mappings also work for gVim.
Plugin: http://vim.sourceforge.net
Fritz Mehner (mehner@fh-swf.de)
October 2007

		Menu(s)
`\lcs`	Load Menus	(n & GUI only)
`\ucs`	Unload Menus	(n & GUI only)

		***H**elp*
`\h`	show plugin help	

		***C**omments*
`\cl`	end-of-line comment	(n,v,i)
`\cj`	adjust end-of-line comment	(n,v,i)
`\cs`	set end-of-line comment column	(n)
`\c*`	code ⇒ comment `/* */`	(n,v)
`\c/`	code ⇒ comment `//`	(n,v)
`\co`	comment ⇒ code	(n,v)
`\cfr`	frame comment	(n,i)
`\cfu`	function comment	(n,i)
`\cme`	method description	(n,i)
`\ccl`	class description	(n,i)
`\cd`	date	(n,i)
`\ct`	date & time	(n,i)

		***S**tatements*
`\sd`	`do { } while`	(n,v,i)
`\sf`	`for`	(n,i)
`\sfo`	`for { }`	(n,v,i)
`\si`	`if`	(n,i)
`\sif`	`if { }`	(n,v,i)
`\sie`	`if else`	(n,v,i)
`\sife`	`if { } else { }`	(n,v,i)
`\sw`	`while`	(n,i)
`\swh`	`while { }`	(n,v,i)
`\ss`	`switch`	(n,v,i)
`\sc`	`case`	(n,i)
`\s{`	`{ }`	(n,v,i)

		***P**reprocessor*
`\p<`	`#include<...>`	(n,i)
`\p"`	`#include"..."`	(n,i)
`\pd`	`#define`	(n,i)
`\pu`	`#undef`	(n,i)
`\pie`	`#if #else #endif`	(n,v,i)
`\pid`	`#ifdef #else #endif`	(n,v,i)
`\pin`	`#ifndef #else #endif`	(n,v,i)
`\pind`	`#ifndef #def #endif`	(n,v,i)
`\pi0`	`#if 0 #endif`	(n,v,i)
`\pr0`	remove `#if 0 #endif`	(n)

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

		***I**dioms*
`\if`	function	(n,v,i)
`\isf`	static function	(n,v,i)
`\im`	`main()`	(n,v,i)
`\i0`	`for( x=0; x<n; x+=1 )`	(n,v,i)
`\in`	`for( x=n-1; x>=0; x-=1 )`	(n,v,i)
`\ie`	`enum` + `typedef`	(n,v,i)
`\is`	`struct` + `typedef`	(n,v,i)
`\iu`	`union` + `typedef`	(n,v,i)
`\ip`	`printf()`	(n,i)
`\isc`	`scanf()`	(n,i)
`\ica`	`p=calloc()`	(n,i)
`\ima`	`p=malloc()`	(n,i)
`\isi`	`sizeof()`	(n,v,i)
`\ias`	`assert()`	(n,v,i)
`\ii`	open input file	(n,i)
`\io`	open output file	(n,i)

		***S**nippet*
`\nr`	read code snippet	(n)
`\nw`	write code snippet	(n,v)
`\ne`	edit code snippet	(n)
`\np`	pick up prototype	(n,v)
`\ni`	insert prototype(s)	(n)
`\nc`	clear prototype(s)	(n)
`\ns`	show prototype(s)	(n)

		C++
`\+m`	method implementation	(n,i)
`\+c`	class	(n,i)
`\+cn`	class (using `new`)	(n,i)
`\+tm`	template method implementation	(n,i)
`\+tc`	template class	(n,i)
`\+tcn`	template class (using `new`)	(n,i)
`\+tf`	template function	(n,i)
`\+ec`	error class	(n,i)
`\+tr`	`try` . . . `catch`	(n,v,i)
`\+ca`	`catch`	(n,v,i)
`\+c.`	`catch(...)`	(n,v,i)

		***R**un*
`\rc`	save and compile	(n)
`\rl`	link	(n)
`\rr`	run	(n)
`\ra`	set comand line arguments	(n)
`\rm`	run `make`	(n)
`\rg`	cmd. line arg. for `make`	(n)
`\rp`	run `splint`[1]	(n)
`\ri`	cmd. line arg. for `splint`	(n)
`\rk`	run `CodeCheck`[2]	(n)
`\re`	cmd. line arg. for `CodeCheck`	(n)
`\rd`	run `indent`	(n,v)
`\rh`	hardcopy buffer	(n,v)
`\rs`	show plugin settings	(n)
`\rx`	set xterm size	(n, only Unix & GUI)
`\ro`	change output destination	(n)
`\rt`	rebuild templates	(n)

[1] `splint` must be installed (`www.splint.org`).
[2] `CodeCheck` must be installed. CodeCheck^{TM} is a product of Abraxas Software, Inc.