c-support.vim VERSION 5.17 HOT KEYS

Key mappings for Vim with and without GUI. Plugin: http://vim.sourceforge.net

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

| | Comments |
|--------|--|
| [n]\cl | end-of-line comment (n,v,i) |
| [n]\cj | adjust end-of-line comment (n,v,i) |
| \cs | set end-of-line comment column (n) |
| [n]\C* | $code \Rightarrow comment /* */ (n,v)$ |
| [n]\CC | $code \Rightarrow comment // (n,v,o)$ |
| [n]\CO | $comment \Rightarrow code \qquad \qquad (n,v,o)$ |
| \cfr | frame comment (n,i) |
| \cfu | function comment (n,i) |
| \cme | method description (n,i) |
| \ccl | class description (n,i) |
| \cfdi | file description (implementation) (n,i) |
| \cfdh | file description (header) (n,i) |
| \ccs | C/C++-file sections (tab compl.) (n,i) |
| \chs | H-file sections (tab compl.) (n,i) |
| \ckc | keyword comment (tab compl.) (n,i) |
| \csc | special comment (tab compl.) (n,i) |
| \cd | date (n,v,i) |
| \ct | date & time (n,v,i) |
| [n]\CX | toggle comments: $C \leftrightarrow C++$ (n,v,i) |

| | | tements |
|-------|----------------------------|---------|
| \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) |
| \se | else { } | (n,v,i) |
| \sw | while | (n,i) |
| \swh | while { } | (n,v,i) |
| \ss | switch | (n,v,i) |
| \sc | case | (n,i) |
| \sb | { } | (n,v,i) |
| | P repi | ocessor |
| \ps | choose a Std. Lib. include | (n,i) |
| \pc | choose a C99 include | (n,i) |
| \p< | #include<> | (n,i) |
| \p" | #include"" | (n,i) |
| \pd | #define | (n,i) |
| \pu | #undef | (n,i) |
| \pif | #if #endif | (n,v,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) |
| \pe | #error | (n,i) |
| \pl | #line | (n,i) |
| \pp | #pragma | (n,i) |

| \nr | | | Snippet |
|---|--------|--|----------------|
| \ne edit code snippet (n,i) n \nf pick up function prototype (n,v,i) n \np pick up method prototype (n,v,i) \ni insert prototype(s) (n,i) \nc clear prototype(s) (n,i) \ns show prototype(s) (n,i) \nt edit local templates (n,i) \nt edit global templates (n,i) \nt reread the templates (n,i) \nt reread the templates (n,i) \nt function (n,v,i) \is static function (n,v,i) \im main() (n,v,i) \in for(x=0; x <n; (n,v,i)="" (x="n-1;" \for="" \in="" x="" x+="1)" ="">=0; x-=1) (n,v,i) \is struct + typedef (n,v,i) \iu union + typedef (n,v,i) \iu printf() (n,i) \is scanf() (n,i) \in p=malloc() (n,i) \in p=realloc() (n,i) \is sizeof() (n,v,i) \ia open input file (n,v,i) \if fscanf (n,v,i) \if fscanf (n,i)</n;> | \nr | read code snippet | (n,i) |
| pick up function prototype (n,v,i) | ∖nw | write code snippet | (n,v,i) |
| | \ne | edit code snippet | (n,i) |
| | [n]\nf | pick up function prototype | (n,v,i) |
| \ni | [n]\np | | |
| \nc | [n]\nm | 1 | (n,v,i) |
| \ns | \ni | insert prototype(s) | (n,i) |
| \ntl edit local templates (n,i) \ntg edit global templates (n,i) \ntr reread the templates (n,i) \nts change templates style (n,i) Idioms \if function (n,v,i) \isf static function (n,v,i) \im main() (n,v,i) \im main() (n,v,i) \in for(x=0; x <n; (n,v,i)="")="" \in="" for(="" x="" x+="1">=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) \ire p=realloc() (n,i) \ire p=realloc() (n,i) \ire p=realloc() (n,i) \ire p=realloc() (n,v,i) \iii open input file (n,v,i) \ii open output file (n,v,i) \if fscanf (n,v,i)</n;> | \nc | clear prototype(s) | |
| \ntg edit global templates | \ns | show prototype(s) | (n,i) |
| \ntr | \ntl | | (n,i) |
| \nts | \ntg | edit global templates ¹ | (n,i) |
| Idioms | \ntr | reread the templates | (n,i) |
| \if function (n,v,i) \isf static function (n,v,i) \im main() (n,v,i) [n]\i0 for(x=0; x <n; (n,v,i)="")="" [n]\in="" for(="" x="" x+="1">=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) \ina p=malloc() (n,i) \ire p=realloc() (n,i) \ire p=realloc() (n,v,i) \iis sizeof() (n,v,i) \ii open input file (n,v,i) \ii open output file (n,v,i) \if fscanf (n,i)</n;> | \nts | change templates style | (n,i) |
| \isf static function (n,v,i) \im main() (n,v,i) \[[n]\io for(x=0; x <n; (n,v,i)="")="" [n]\io="" \[="" \]="" for(="" x="" x+="1">=0; x-=1) (n,v,i) \] \[ie enum + typedef (n,v,i) \] \[\io struct + typedef (n,v,i) \] \[\io printf() (n,i) \] \[\io scanf() (n,i) \] \[\io peralloc() (n,i) \] \[\io p=realloc() (n,i) \] \[\io p=realloc() (n,i) \] \[\io sizeof() (n,v,i) \] \[\io open output file (n,v,i) \] \[\io open output file (n,v,i) \] \[\io fscanf (n,i) \]</n;> | | | I dioms |
| \im main() (n,v,i) [n]\io for(x=0; x <n; (n,v,i)="")="" [n]\in="" for(="" x="" x+="1">=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) \ire p=realloc() (n,i) \ire p=realloc() (n,i) \iii sizeof() (n,v,i) \iii open input file (n,v,i) \iio open output file (n,v,i) \ifs fscanf (n,i)</n;> | \if | function | (n,v,i) |
| for(x=0; x <n;)<="" td="" x+="1"><td>\isf</td><td>static function</td><td>(n,v,i)</td></n;> | \isf | static function | (n,v,i) |
| for(x=n-1; x>=0; x-=1) (n,v,i) | \im | main() | (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) \ire p=realloc() (n,i) \isi sizeof() (n,v,i) \isi sizeof() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \ifs fscanf (n,i) | [n]\i0 | for(x=0; x <n;)<="" td="" x+="1"><td>(n,v,i)</td></n;> | (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) \ire p=realloc() (n,i) \isi sizeof() (n,v,i) \ias assert() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \ifs fscanf (n,i) | [n]\in | for(x=n-1; x>=0; x-=1 |) (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) \ire p=realloc() (n,i) \isi sizeof() (n,v,i) \ia assert() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \ifs fscanf (n,i) | \ie | enum + typedef | (n,v,i) |
| \ip printf() (n,i) \isc scanf() (n,i) \ica p=calloc() (n,i) \ima p=malloc() (n,i) \ire p=realloc() (n,v,i) \isi sizeof() (n,v,i) \ia assert() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \ifs fscanf (n,i) | ∖is | struct + typedef | (n,v,i) |
| \isc scanf() (n,i) \ica p=calloc() (n,i) \ima p=malloc() (n,i) \ire p=realloc() (n,i) \isi sizeof() (n,v,i) \ias assert() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \if fscanf (n,i) | \iu | union + typedef | (n,v,i) |
| \ica | \ip | <pre>printf()</pre> | (n,i) |
| \ima p=malloc() (n,i) \ire p=realloc() (n,v,i) \isi sizeof() (n,v,i) \ias assert() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \ifs fscanf (n,i) | \isc | scanf() | (n,i) |
| \ire p=realloc() (n,i) \isi sizeof() (n,v,i) \ias assert() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \ifs fscanf (n,i) | ∖ica | p=calloc() | (n,i) |
| \isi sizeof() (n,v,i) \ias assert() (n,v,i) \ii open input file (n,v,i) \io open output file (n,v,i) \ifs fscanf (n,i) | \ima | p=malloc() | (n,i) |
| \iasassert()(n,v,i)\iiopen input file(n,v,i)\ioopen output file(n,v,i)\ifsfscanf(n,i) | \ire | p=realloc() | (n,i) |
| \iiopen input file (n,v,i) \ioopen output file (n,v,i) \ifsfscanf (n,i) | ∖isi | sizeof() | (n,v,i) |
| \ioopen output file (n,v,i) \ifsfscanf (n,i) | ∖ias | assert() | (n,v,i) |
| \ifs fscanf (n,i) | \ii | open input file | (n,v,i) |
| | | | (n,v,i) |
| \ifp fprintf (n,i) | - | | (n,i) |
| = 1 * \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ | \ifp | fprintf | (n,i) |

¹ system-wide installation only

| | | C++ |
|--------|----------------------------------|---------------|
| \+co | cout << << endl; | (n,i) |
| \+" | << "" | (n,i) |
| \+c | class | (n,i) |
| \+ps | #include<> STL | (n,i) |
| \+pc | #include <c> C</c> | (n,i) |
| \+cn | class (using new) | (n,i) |
| \+ci | class implementation | (n,i) |
| \+cni | class (using new) implementation | (n,i) |
| \+mi | method implementation | (n,i) |
| \+ai | accessor implementation | (n,i) |
| \+tc | template class | (n,i) |
| \+tcn | template class (using new) | (n,i) |
| \+tci | template class implementation | (n,i) |
| \+tcni | template class (using new) impl. | (n,i) |
| \+tmi | template method implementation | (n,i) |
| \+tai | template accessor implementation | n (n,i) |
| \+tf | template function | (n,i) |
| \+ec | error class | (n,i) |
| \+tr | trycatch | (n,v,i) |
| \+ca | catch | (n,v,i) |
| \+c. | catch() | (n,v,i) |
| | H elp and N | <i>1</i> enus |
| \hm | show manual | (n,i) |
| \hp | help (c-support) | (n,i) |
| \lcs | load Menus (n & GI | JI only) |
| \ucs | unload Menus (n & GI | JI only) |

| | | Run | |
|--------|--------------------------------------|---------|--|
| \rc | save and compile | (n,i) | |
| \rl | link | (n,i) | |
| \rr | run | (n,i) | |
| \ra | set comand line arguments | (n,i) | |
| \rm | run make ¹ | (n,i) | |
| \rmc | run make clean ¹ | (n,i) | |
| \rcm | choose a makefile ¹ | (n,i) | |
| \rme | executable to run ¹ | (n,i) | |
| \rma | cmd. line arg. for make ¹ | (n,i) | |
| \rp | run splint ² | (n,i) | |
| \rpa | cmd. line arg. for splint | (n,i) | |
| \rk | run CodeCheck ³ | (n,i) | |
| \rka | cmd. line arg. for CodeCheck | (n,i) | |
| \rd | run indent | (n,i) | |
| [n]\rh | hardcopy buffer | (n,i,v) | |
| \rs | show plugin settings | (n,i) | |
| \rx | set xterm size (n,i, only Unix & | GUI) | |
| \ro | change output destination | (n,i) | |
| | Additional Mappings | | |

| typing | expansion | | |
|--------------|----------------------------|----|-----|
| /* | /* */ | | (i) |
| /* | /* (multiline) marked text | */ | (v) |
| /* <cr></cr> | /* | | (i) |
| | * | | |
| | */ | | |
| { <cr></cr> | { | | (i) |
| | 1 | | |
| | } | | |
| { <cr></cr> | { | | (v) |
| | (multiline) marked text | | |
| | } | | |

Ex commands:

CFileSection C/C++–file sections (same as \ccs)

HFileSection H-file sections (same as \chs)

KeywordComment keyword comment (same as \ckc)

SpecialComment special comment (same as \csc)

IncludeStdLibrary standard library includes (same as \ps)

IncludeC99Library C99 includes (same as \pc)

IncludeCppLibrary STL includes (same as \+ps)

IncludeCppCLibrary C includes (same as \+pc)

CStyle C99 include (same as \nts)

Use tab expansion to show the items to choose from.

¹ also working for filetype **make**

www.splint.org

CodeCheckTM is a product of Abraxas Software,