VIM IDE Cheat Sheet

by piping via cheatography.com/61694/cs/15920/

Beginner Block

i	start insert
<esc>	escape to normal mode
:w	in normal mode, write to current file
:q	in normal mode, quit the editor

Description in BOLD means overwrite default configuration
[https://github.com/Piping/dotfiles] has my configuration

Notation: `<leader>m` means `<space>m`
`<C-a>` means `Ctrl + A` , `<S-x>` means `Shift + x`
`a/b` means press keyboard either a or b

Motions: command that moves the cursor

hjkl	left/down/up/right
f/F	search and move to next typed character
t/T	similar to `f/F`. but stop before character
0/$	start/end of current line
w/e	next start/end of word, `W/E` for word with punctuation
b/B	previous start of word `B` for word with punctuation
gg	go to line {count}, default first line
G	go to line {count}, default last line
-/<enter>	previous/next start of the line
H/M/L	cursor go to TOP/MIDDLE/BOTTOM of the screen
''	(single quote twice) previous location in jumplist
'{a-z}/`{a-z}	previous marked position using m{a-z}, E.g. ma `a
'.	To the position where the last change was made.
%	move to closing pair [], {}, ()
{1-9}+	a number type before motions, repeat {count} motion
:help motion	check more on documents inside vim

Vim Concept: Motions - command that moves the cursor, depends on current cursor position. Above list is not complete! It can be used with `OPERATOR` to efficiently editing text in Textual User Interface

Text-Object-Motion

ap	around a paragraph
iw	inner word
aw	aroud word
i"	inner double quote
a"	around double quote
:h iw	help for more

Operator - editing the text efficiently

d	delete/cut
c	change
y	copy/yank
~	reverse case
!	filter with external program, E.g. format doc
gu/gU	make lower/upper case
</>	indent left/right
=	filter with predefined `equalprg`
zf	fold the text

Two way to combine operator and selections:
1. <operator><motion> or <operator><text-object-motion> etc.
2. <selection from visual mode><operator>
E.g. `di"` delete the word inside double quotes
Press one operator twice operate on current line as the selection
E.g. `yy` copy the current line, `dd` cut/delete the current line

Paste && Registers

p	paste using unnamed register `"` literally, after cursor
P	paste before cursor
"0p	paste `0` register's content in normal mode
<C-R>"	double quote, insert text as if typed
<C-r>0	the one before current copy/cut text
<C-R>=	eval expression after = and paste
<C-R>/	the last search pattern
<C-R><C-R>"	equivalent to p in normal mode
:h i_ctrl-r	more registers

VIM Concept: Registers, used to store copy/cut text, the register can have single character names, `{0-9a-z"%#*+:.-=}`.

`p/P` is used in normal mode, other `Ctrl-R{register}` are used in insert mode and command line mode. (I reversed p/P since I feel p before cursor is more natural for modern IDE)

VIM IDE Cheat Sheet

by piping via cheatography.com/61694/cs/15920/

Page Movement

<C-e>/<C-y>	**move buffer down/up and keep cursor position**
zz	bring current line and cursor to center
zb	move current line and cursor to bottom screen
zt	move current line and cursor to top screen

Visual Mode Commands

o	switch between two ends of selection (anchor)
<c-v>	switch to visual block mode
V	switch to visual line mode
v	switch to visual mode
<leader>a	alignment, using plugin `easy-align`

Tabs

gt	go to next tab
gT	go to previous tab,**<leader>tt**
:tabnew	new tab, **<leader>t**
:tabclose	close current tab, **<leader>tc**

Frequent Used Custom Shortcuts

[p /]p	go for next/previous quickfix item
[<space> /]<space>	add newline before/after current line
[e /]e	move current line up/down, count applies

Windows (Split, Size Adjustment, Placement)

<C-w>=	equal size display all panels
<C-w>s	horizontal split
<C-w>v	vertical split
<C-w>H	put pane to absolute left, take full height
<C-w>L	put pane to absolute right, take full height
<C-w>J	put pane to absolute bottom, take full width
<C-w>K	put pane to absolute top, take full width
:set splitright	for vertical split, place new pane right
<C-w>hjkl	move to relative left/down/up/right pane

Insert Mode (My configuration)

<C-a>	Jump to the beginning of the line
<C-e>	Jump to the end of the line
<C-w>	Backward-Delete Word
<C-d>	Forward-Delete Word
<C-z>	backward move cursor one word
<C-x>	forward move cursor one word
<C-u>	delete/cut current line
<C-y>	paste/yank to current line
<C-k>	delete the rest line after cursor

Useful Utility Commands (Normal Mode)

.	dot command, repeat last `change`
J	Join the line below to current line
<C-a>	add {count} to number under cusor
<C-x>	subtract {count} to number under cusor
<C-R>=	calculator <C-R>=128/2, insert mode
@;	repeat last cmdline command, **<leader>:**

dot command `.` does not repeat command line commands, only changes that is defined by vim. E.g. invoked by operator `c` and followup inserted text.

CSCOPE MAPPING (My Configuration)

:cs add <path to cscope.out> <path to worksapce>

<leader>gs	search the C symbol under cursor
<leader>gg	search global defintion
<leader>gd	search funtions called by this function
<leader>gc	search who called this function
<leader>gt	search this string
<leader>ge	search this egrep pattern
<leader>gii	search for files include the filename under cursor
<leader>gi	search for files include the current file

Normal Commands (My Configuration)

\	equivalent to %, go to pairs
\\	**format and intent the file** E.g. clang-format
=	**go to end of line**
q	close current window
gf	use fzf to fuzzy search recuently used files
<leader>l	open file browser (default bundle)

Normal Commands (My Configuration) (cont)

<leader>m	open tagbar for current file
<leader><cr>	Disable highlights when you press
<leader>u	open the undotree
<leader>zz	Distraction-Free Display

Command Mode :

:%	{range}, equal to 1,$ (the entire file)
:! <external cmd>	range of text is being pipe to cmd to be replaced
:%! xxd	edit binary file
:%! xxd -r	save the file into binary form

fuzzy-search-panel (My configuration)

<c-f>	select all
<c-g>	deselect all
tab/shift-tab	select-deselect current line
<C-t>	open file in new tab
<C-x>	split horizontaly
<C-v>	split vertically
<esc>/<c-d><c-c>	quit

Special and Very Useful Windows -- Quickfix

:copen/:cclose	open/close Quickfix
:cn	go to next fix
:cp	go to previous fix

Quickfix typically used after `:make` command and `cscope`, the quickfix window contains the parsed result from `:make` that contains where complication error happen, and put cursor to exactly the file/line/column so user can just fix it!
I open quick fix with my shortcut and prefix a `:botright copen` to open it at the bottom of window

Commentary (External Plugin Default)

gcc	comment/uncomment current line
10gcc	comment next 10 lines
gcu	comment block
gcap	comment the paragrah
gc	comment selection (visual mode)

Code Folding

zf	fold selection text
zo	open selection text