VIM and its secrets Cheat Sheet

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

Beginner Block

i	start insert
<esc> or <C-[>	escape to normal mode
:w [<filename>]	in normal mode, write to current file
:q	in normal mode, quit the editor

[https://github.com/Piping/dotfiles] has my configuration

Notation: `<leader>` key can be defined via `let mapleader = "\ <space>"`

`<C-a>` means `Ctrl + A` , `<S-x>` means `Shift + x`

Cursor Motions: keystokes that move 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
^/g_	start and end of current line with non whitespace
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
g; g,	cycle through change positions

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

Operator - commands that edit 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 first then selection
2. visual selection first then does operation

Example:

E.g. `di"` delete the word inside double quotes or equivalently `vi"d`.

E.g. `yy` copy the current line, `dd` cut/delete the current line (**Press one operator twice operate on current line as the selection**)

Text-Object-Motion : Level Up Cursor Motion

ap	around a paragraph
iw	inner word
aw	aroud word
i"	inner double quote
a"	around double quote
//	last search result, follow after operator
//0	search-offset to that whole line
:h iw	help for more

Paste && Registers

p	paste after cursor using content from register `"`
P	paste before cursor or cursor line
"0p	paste `0` register's content in normal mode
:register	show content of a list of registers
: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"%#*+:.-=}`.

By **piping**

cheatography.com/piping/

Published 13th June, 2018.

Last updated 15th July, 2020.

Cheatography

VIM and its secrets 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 cursor line to center
zb	bring current cursor line to bottom screen
zt	bring current cursor line 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

Tabs

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

File Navigation

Ctrl + 6 / Ctrl + ^	jump to previous opened file in current window
gf	open file using the text under the cursor
:tabe <filename>	open file in the new tab
:e <filename>	open file in the current window
:ls	list current opened buffers(files)
:buf <number>	open selected buffer(file) in the current window

Start Insert Mode in various way

a	insert after the cursor (append)
i	insert in front of the cursor
I (Caps i)	Insert at the beginning of the line
A	Insert at the end of the line
s	remove current character and enter insert mode
R	enter insert mode with REPLACE semantic
c <motions>	delete selected text and enter insert mode
gi	go to last edited place and enter insert mode

Windows (Split, Size Adjustment, Placement)

<C-w>=	equal size display all panels
<C-w>s	horizontal split
<C-w>v	vertical split
<C-W>p	go to last accessed window
<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
:vertical split <filen-ame>	vertical split buffer/file
<C-w>hjkl	move to cursor to relative left/down/up/right pane

Insert Mode (Emacs Style Single Line Editing)

<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-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
@;	repeat last cmdline command
<leader>j	break current line and move trailing part one line above

dot command `.` repeat text changes that is defined by vim. E.g. invoked by operator `c` and followup inserted text.

Cheatography

VIM and its secrets Cheat Sheet

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

Ctrl+R (Using Registers in Insert/Command Mode)

<C-R>/	put last search string
<C-R>=	calculator <C-R>=128/2, insert mode
<C-R>"	put last copied text
<C-R>0	put second to last copied text

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>ca	add cscope.out <worksapce path>
<leader>gs	search the C symbol under cursor
<leader>gd	search global defintion
<leader>gc	search who called this function
<leader>ge	search this string as egrep pattern
<leader>gf	search for this file under cursor
<leader>gi	search for files that include the current file
<leader>ga	search assignment to this variable

cscope is the most common tool for developing c projects. (look up symbol, definition, locate caller/callee of functions, etc). cscope interface is built-in feature for most vim distribution. To generate cscope database, first use `cscope -Rbq` in your project directory.

Normal Commands (My Configuration)

q	**close current window**
<cr>	equivalent to :noh Remove Search Highlights
<leader>m	open tagbar for current file
<leader>l	toggle line number display
<leader>z	open current file in a new tab to "ZOOM"

Command Mode :

:%	{range}, equal to 1,$ (the entire file)
:{range}! <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
Ctrl-f	open cmdline editing windows

Special and Very Useful Windows -- Quickfix

:copen/:cclose	open/close Quickfix
:cn	go to next fix
:cp	go to previous fix
:make all -j	build the code & report on quickfix list

Quickfix typically used after `:make` command and can be configured to work with `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

Fold The Content

:set foldmethod=	available values: syntax, indent, manual
:set foldlevel=	0,1,2,3,.., applied when value is changed
za	toggle folding at the cursor position
zf	Operator to create a fold (manual mode only)
zo	open selection text
zc	close the fold

Vim Diff Mode

:diffsplit <[filename]>	split the window with diff mode on
:diffoff	turn off diff mode, (include diff highlights)
:set diff!	only switch the diff color highlight
do	diff obtain the change from the other side
dp	diff put change to the other side

`windo diffoff` can turn off all diffmode windows in current tab

Commentary (External Plugin)

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