VIM CHEAT SHEET(S)

DREIPFUNDFLACHS – GITHUB.COM/DREIPFUNDFLACHS

1. Notation

A word is composed of letters, numbers and underscores, but no other characters. A Word is any text object separated by whitespaces.

Optional arguments or characters appear within square brackets, as in `:sp[lit]`, while required arguments or complements to operators appear within curly braces, as in `d{motion}`.

<C-x> stands for CTRL-X, <CR> for carriage return. The expression (IM) next to a command means that it must be executed within insert mode.

2. Marks

Command	Mnemonic	Description
`m{char}`	mark	mark current cursor location with *char*
`` `{char} ``	back(tick)	to cursor location marked by *char*
`'{char}`	—	to line marked by *char*
`` `` ``	back back	to before last jump
`` `. ``	—	to location of last change
`` `^ ``	—	to location of last insertion
`gi`	go and insert	insert at location of last insertion
`` `[ ``	—	to start of last change or yank
`` `] ``	—	to end of last change or yank
`` `< ``	—	to start of last visual selection
`` `> ``	—	to end of last visual selection
`%`	—	toggle between matching delimiters or tags

3. Registers and Macros

Command	Mnemonic	Description
`q{char}{edits}q`	quote unquote	record *edits* as a macro in register *char*
`q{Char}{further edits}q`	quote unquote	append *further edits* to macro in register *char*
`@{char}`	loop	execute macro stored in register *char*
`@@`	loop	execute macro invoked most recently

4. Registers

Command	Description
`"{char}`	store in or access register *char*
`"{Char}`	append to register *char*
`"_`	black hole register
`"+`	clipboard register
`"0`	yank register
`""`	unnamed register
`"=`	expression register
`"*`	primary register
`"%`	name of current file
`"#`	name of alternate file
`".`	last inserted text
`":`	last Ex command
`"/`	last search pattern

5. Motion

Command	Mnemonic	Description
`h`, ← or BACKSPACE	—	left one column
`j` or ↓	—	down one line
`k` or ↑	—	up one line
`l`, → or SPACE	—	right one column
+ or ENTER	1 more	to first character of next line
`-`	1 less	to first character of previous line
`^`	—	to first nonblank character of current line
n`|`	*n*-th column	to *n*-th column of current line
`(` or `)`	—	to beginning of current (next) sentence
`{` or `}`	—	to beginning of current (next) paragraph
`[[` or `]]`	—	to beginning of current (next) section
`0`	initial	cursor to beginning of line
`$`	—	cursor to end of line
`b` or `B`	back	to beginning of previous word (Word)
`w` or `W`	word	to beginning of next word (Word)
`e` or `E`	end	to end of current word (Word)
`ge` or `gE`	end	to end of previous word (Word)
`<C-l>`	as in shell	redraw screen
`<C-f>`	forward	scroll forward one screen
`<C-b>`	backward	scroll backward one screen
`<C-d>`	down	scroll down half screen
`<C-u>`	up	scroll up half screen
`M`	middle	to middle line on screen
`H`	home	to home position (top line of screen)
`L`	last	to last line on screen
`<<`	shift left	shift current line left by one shiftwidth
`>>`	shift right	shift current line right by one shiftwidth
`%`	—	from bracket (, [, { or < to matching partner
`/{pattern}` or `?{pattern}`	—	search forward (backward) for *pattern*
`*`	—	search forward for word under the cursor
`n` or `N`	next	repeat last search in same (opposite) direction
`f`*x* or `F`*x*	find	to next (previous) character *x* on current line
`t`*x* or `T`*x*	till, to	to just before (after) character *x* on current line
`;` or `,`	—	repeat find command in same (opposite) direction
n`G` or `:`*n*	go to *n*	to line number *n*
`gg`	go go	to first line of the file
`G`	Go	to last line of the file
`gj`, `gk`	—	down, up one display line
`g0`, `g$`	—	to beginning, end of display line
`g^`	—	to first nonblank character of display line

6. Undelimited Text Objects

Command	Mnemonic	Description
`iw`	inside word	current word
`aw`	around word	current word plus one space
`iW`	inside Word	current Word
`aW`	around Word	current Word plus one space
`is`	inside sentence	current sentence
`as`	around sentence	current sentence plus one space
`ip`	inside paragraph	current paragraph
`ap`	around paragraph	current paragraph plus one blank line

7. Delimited Text Objects

Command	Description
a) or ab	delimited by parentheses
a} or aB	delimited by brackets
a]>'"`	delimited by]>'"`
i) or ib	inside of parentheses
i} or iB	inside of brackets
i]>'"`	inside of]>'"`

8. Editing

Command	Mnemonic	Description
a	append, after	append text to current cursor position
A	Append, After	append text to end of current line
i	insert, in front	insert text at current cursor position
I	Insert, In front	insert text at the beginning of current line
~	toggle	change case (lower to upper or upper to lower)
x	x-out (as in typewriters)	delete current character
X	X-out (as in typewriters)	delete previous character
r	replace	replace current character by another character
R	Replace	enter overstrike mode (replace until Esc)
s	substitute	substitute current character by text
S	Substitute	substitute entire current line
d{motion}	delete	begin a deletion
dd	delete delete	delete entire current line
D	Delete	delete from current position to end of the line
c{motion}	change	begin a change
cc	change change	change entire current line
C	Change	change from current position to end of the line
p	paste or put	paste from buffer after cursor position
P	Paste or Put	paste from buffer before cursor position
gp	paste or put	paste after cursor, leave it at the end of new text
gP	Paste or Put	paste before cursor, leave it at the end of new text
y{motion}	yank	begin a yank
yy or Y	yank yank	copy entire line
.	dot, do it	repeat last editing command
u	undo	undo last edit
U	Undo	undo last change to line (if cursor has not moved)
R	redo, restore	restores an undone edit
o	open	insert blank line below, enter insert mode
O	Open	insert blank line above, enter insert mode
J	join	join current and next line
vipJ	unflow	unflow (unwrap) current paragraph
gq{motion}	—	reflow text
g~{motion}	toggle	toggle case
gu{motion}	—	make lowercase
gU{motion}	—	make uppercase
<{motion}	shift left	shift left by one shiftwidth
>{motion}	shift right	shift right by one shiftwidth
={motion}	—	autoindent
&	substitute	repeat substitution

9. Insert Mode

Command	Mnemonic	Description
<C-h>	as in shell	backspace
<C-w>	as in shell	delete back one word
<C-u>	as in shell	delete to start of line
<C-t>	tab	shift right by one shiftwidth
<C-d>	decrease	shift left by one shiftwidth
<C-o>	n*o*rmal	escapes insert mode for a single command
<C-r>{register}	register	paste from *register*
<C-r>=	—	access expression register (=)

10. Search

Command	Description
n	to next match (w.r.t. search direction)
N	to previous match (w.r.t. search direction)
/[pattern]<CR>	search forward for *pattern* (same if empty)
?[pattern]<CR>	search backward for *pattern* (same if empty)
/[pattern]/e<CR>	search for *pattern*, placing cursor at the end of match
:%s/[pattern]//gn	count number of matches for *pattern*
\v	all characters have special meaning except letters, digits, _
\V	only \, / (fwd search), ? (bkd search) have special meaning
\c	force case insensitivity
\C	force case sensitivity
<[pattern]>	set boundaries of *pattern* (needs \v)
\zs, \ze	set start (end) of matches
<C-r>{register}	paste contents of *register* into search field
<C-r><C-w>	complete using remainder of current match

11. Window Management

When resizing windows, it is often more convenient to use the mouse.

Command	Mnemonic	Description
:sp[lit] [file] or <C-w>s	split	split window horizontally, loading *file*
:vsp[lit] [file] or <C-w>v	vertical split	split window vertically, loading *file*
:new [file] or <C-w>n	new	open *file* in new window
:clo[se] or <C-w>c	close	close current window
:on[ly] or <C-w>o	only	close all windows but current one
:tabe[dit] [file]	tab edit	open *file* in new tab
:tabc[lose]	tab close	close current tab
:tabo[nly]	tab only	close all tabs but current one
[n]gt or [n]<C-PgDn>	tab, next page	move to next tab page (or page number *n*)
[n]gT or [n]<C-PgUp>	Tab, previous page	move one (*n*) tab pages back
<C-w>w or <C-w><C-w>	window, word	cycle to next window
<C-w>W	previous window	cycle to previous window
<C-w>p	previous	move to previously accessed window
<C-w>c	close	close current window (tab)
<C-w>h, j, k, l	as in command mode	move to window left, below, above, or right
<C-w>H, J, K, L	as in command mode	move current window left, below, above, or right
<C-w>+ or <C-w>-	plus or minus	increase (decrease) height of window by 1 line
<C-w>> or <C-w><	plus or minus	increase (decrease) width of window by 1 column
<C-w>=	equal	equalize width and height of all windows
<C-w>_	move to bottom	maximize height of current window
<C-w>|	—	maximize width of current window
<C-w>T	tab	move current window to new tab

12. The :set Command

Command	Description
`:set {option}`	enable Boolean *option*; show value of non-Boolean *option*
`:set no{option}`	disable Boolean *option*
`:set {option}!`	toggle Boolean *option*
`:set {option}?`	show value of *option*
`:set {option}&`	set *option* to default value
`:set {option}={value}`	set non-Boolean *option* to *value*
`:setlocal {option}[={value}]`	set *option* (to specified *value*) within current buffer
`:set`	show modified options
`:set all`	show all options

13. Visual Mode

Command	Description
`v`	select text by character
`V`	select text by line
`<C-v>`	select text by block
`o`	go to other end of selected text
`gv`	repeat last selection

14. Spell-checker

Command	Description
`:set spell`	enable spell-checker
`:set spelling={language}`	set spell-checker to *language*
`]s`	to next misspelling
`[s`	to previous misspelling
`z=`	suggest corrections for current word
`zg`	add current word to spell file
`zw`	remove current word from spell file
`zug`	revert `zg` or `zw` for current word
`<C-x>s` or `<C-x><C-s>` (IM)	suggest corrections to previous misspelling

15. Ex Command Line

Command	Mnemonic	Description
`:[range]d(elete) [x]`	delete	delete specified lines into register *x*
`:[range]y(ank) [x]`	yank	yank specified lines into register *x*
`:[line]pu(t)[x]`	put	paste contents of register *x* after *line*
`:[range]co(py) {address}`	copy	copy specified lines to *adress*
`:[range]m(ove) {address}`	move	move specified lines to address
`:[range]j(oin)`	join	join specified lines
`:[range]norm(al) {cmds}`	normal	execute normal mode *commands* on each specified line
`$`	—	last line of the file
`%`	—	entire file
`0`	—	virtual 0th line
`.`	—	current line
`'<` or `'>`	—	start (end) of visual selection
`<Tab>` or `<S-Tab>`	—	autocomplete (resp. in the reverse direction)
`<C-d>`	—	list possible completions
`<C-r><C-w>`	word	insert word under the cursor at the prompt
`<C-p>` or `<C-n>`	previous/next	scroll backward (forward) through command history
`:shell`	shell	start a shell session
`:!{cmd}`	!	execute *command* with the shell
`:[range]!{filter}`	—	filter *range* through external program *filter*
`:read !{cmd}`	read	execute *command* and insert output below cursor
`:[range]write !{cmd}`	write	execute *command* with *range* as input
`:argdo {cmd}`	do in arglist	execute *command* across all buffers in the arglist

Characters	Commands	Description
gb	esc lbi{ esc ea} esc	Surrond a word by curly braces
Y	esc y $esc	Yank from current position to end of the line