Vim Command Summary

https://github.com/praful/vim-cheatsheet Updated: 2019/04/28

General pattern commands: [n] operator [m] movement

Movement Commands

Character	
`h, j, k, l`	Left, down, up, right (`3j` moves down 3 lines)
Text	
`w, W, b, B`	Next / previous word / WORD (lowercase `w`, `b`: letters, numbers, underscore make up word; uppercase `W`, `B`: separator is whitespace)
`e, E`	End of word/WORD
`ge, gE`	End of previous word/WORD
`), (`	Beginning of next / previous sentence
`}, {`	Beginning of next / previous paragraph
`]], [[`	Beginning of next / previous section (or function in C, Java, etc)
`%`	Move to matching brace, brackets, etc
Line	
`0, gm, $`	Beginning / middle / end of line
`g0, g$`	Beginning / end of screen line (wrap on)
`^, g_`	First / last character of line (ignore spaces)
`+, -`	First character of next / previous line
`n|`	Column n of current line
`H, M, L`	Top / middle / last line of screen
n`H`	*n* (number) of lines after top line
n`L`	*n* (number) of lines before last line
Screens	
`CTRL-F, CTRL-B`	Scroll forward / backward one screen
`CTRL-D, CTRL-U`	Scroll down / up one-half screen
`CTRL-E, CTRL-Y`	Scroll line up / down
`z <Enter> or zt`	Reposition line with cursor: to top of screen
`z. or zz`	Reposition line with cursor to middle of screen
`z- or zb`	Reposition line with cursor to bottom of screen
`CTRL-L`	Redraw screen (without scrolling)
`%`	Go to matching pair, eg (), {}, []
Searches	
`/`*pattern*	Search forward for *pattern*
`?`*pattern*	Search backward for *pattern*
`n, N`	Repeat last search in same / opposite direction
`/ , ?`	Repeat previous search forward / backward
`f`*x*	Search forward for character *x* in current line
`F`*x*	Search backward for character *x* in current line
`t`*x*	Search forward for character before *x* in current line
`T`*x*	Search backward for character after *x* in current line
`; , ,`	Repeat previous current-line search forward / backward
`*, #`	Move forward / back to string matching the one under cursor. Precede by `g` for embedded match.
`[i , [I`	(lower/uppercase `i`) In status bar, show first/all lines containing word under cursor
Line number	
`CTRL-G`	Display current line number
n`G or `*n*`gg or :n`	Move to line number *n*
`gg, G`	Move to first / last line in file
Bookmarks and positioning	
`m`*x*	Bookmark current position as *x* (uppercase mark eg `m`*X* set global mark, ie file and location)
`` `x `` (back tick)	Move cursor to *x*
`'x` (apostrophe)	Move to start of line containing *x*
`` `` `` (2 back ticks)	Return to position before most recent jump
`''` (2 apostrophes)	Like preceding, but return to start of line
`` `. `` (back tick period)	Move to last change
`` `[ , `] ``	Go to start, end of previously operated text
`CTRL-O, CTRL-I`	Move backwards and forwards in jumplist. See `:help jumplist`
`gi`	go to and insert text at last edit
`g; and g,`	Move backwards and forwards in changelist. See `:help changelist`
`:marks`	List active marks
`:jumps`	List jumps
`:reg x`	Show registers or register *x*

Editing Commands

Select	
`aw, iw, is, gn, is, as, a), at, a"`	`i` acts **i**nside object; `a` acts **a**round. a word / inner word / inner sentence / next search match / inner sentence / a sentence / a () block / a tag block, a quoted string. Use with visual mode and after operators, such as `c`, `d`, `y`, `>`, `g~`. Examples below.
`v, V, CTRL-V`	Select character / line / block (visual mode)
`gv`	Reselect previous selection
`o`	Exchange cursor position between start/end of selection
`vaw, vas, vap, vab, vaB`	Select current word / a sentence / a paragraph / a () block / a {} block. Can use other verbs, eg delete, yank to act on object.
Insert	
`i, a`	Insert text before / after cursor
`I, A`	Insert text at beginning / end of line
`o, O`	Open new line for text below / above cursor
Change	
`r, R`	Change character / overwrite characters
`c{`*motion*`}, cw, cc, 2cw, cG, caw`	Begin change / change to next word / line (stay in insert mode) / change to next two words / change to end of file / change word (when in middle)
`C`	Change to end of line
`C%`	Change text in next group
`R`	Type over characters
`s, S`	Delete character (=cl) / delete line and substitute text
`~`	Toggle case and move right.
`gu, gU{motion}`	Lowercase / uppercase
`g~w, guw, gUw`	Switch case to next word / lowercase to next word / uppercase to next word
`CTRL-A, CTRL-X`	Increment / decrement number under cursor
`.`	(period) Repeat last edit command
`u, CTRL-R, U`	Undo / redo / restore current line
`J , gJ`	Join two lines / join without space
Copy (yank), paste, delete	
`yw, y$, :%y`	Copy to next word / line / buffer.
`yy, 5yy`	Copy current line / next five lines
`"a`*yy*	Copy current line into named buffer *a*
`y'`*x*	Copy from mark *x* to cursor
`p, P`	Put yanked text after / before cursor
`]p , [p`	Like p/P with indent adjusted
`gp , gP`	Like p/P with cursor left after new text
`"a`*P*	Put text from buffer *a* before cursor
`"`*np*	Put text from delete buffer number *n* after cursor (for last nine deletions)
`x, X`	Delete character under/before cursor
`dw, dd, dfS, d^, d/`*pat*`, dn, dL, dG`	Delete to next word / line / upto S (`c` and `d` both cut but `c` stays in insert mode) / to beginning / to pattern / to next pattern / to end screen / to end file. `10dd` deletes 10 lines. `dwwP` transposes two words.
`D`	Delete to end of line
`daw, diW, di>, dgn`	Delete a word / inner WORD / inner block delimited by > / the next search match
`d'`*x*	Del from mark *x* to cursor
Insert mode	
`CTRL-R"`	Paste last yank/delete. Instead of ", use `%` (filename), `+` (clipboard)
`CTRL-A`	Repeat last insertion
`CTRL-D, CTRL-T`	Shift line left/right to previous shift width
`CTRL-E`	Insert character found just below cursor
`CTRL-H`	Delete previous char (=backspace)
`CTRL-I`	Insert a tab
`CTRL-K`	Begin insertion of multi-stroke char
`CTRL-N, CTRL-P`	Insert next / previous completion of the pattern to the left of the cursor
`CTRL-O `*command*	Execute one command eg dw
`CTRL-X + CTRL-L, F, O, K, T, J`	Show completions for current line / file / omni / dictionary / thesaurus / tags
`CTRL-U`	Delete from cursor to start of line
`CTRL-V`	Insert next char verbatim
`CTRL-W`	Delete previous word
`CTRL-Y`	Insert char found just above cursor
`CTRL-[`	End insert mode (=ESC)

Other Commands

Command line mode	
CTRL-B, CTRL-E	Beginning / end of line
CTRL-F	Open command history in new buffer (like q: in normal mode). Use CTRL-C to paste line in history or press *Enter* to run command.
CTRL-N, CTRL-P	Next / previous command in history
CTRL-H, CTRL-R, CTRL-U, CTRL-W	As insert mode
CTRL-V	Insert next non-digit literally. You can enter funky keys by pressing ctrl-v first, and then the keystroke. This is most useful in help, eg type :help ^v^t to get help for the keystroke ctrl-t. For this, though, you can also just type :help ctrl-t.
text UP-ARROW or *text* DOWN-ARROW	Searches history for previously entered commands starting with text
Shell and Shell Filtering	
:r !*command*	Read in output from *command* after current line
:!*command*	Run *command*, then return
!{*motion*}*command*	Send the text covered by *motion* to Unix *command*; replace with output
:n,m!*command*	Send lines *n*-*m* to *command*; replace with output
num!!*command*	Send *num* lines Unix *command*; replace with output
:!!	Repeat last system command
!!*command*	Pass current line only through *command*
!}*command*	Pass area from current line through end of paragraph through *command*
!G*command*	Pass area from current line through end of file through *command*
:%!*command*	Pass the entire current buffer through *command*
Code Reformatting	
==, =G	Fix line indent / indent current line to end of file
%!astyle or %!indent	Linux reformatting programs
gq{*motion*}	Re-do line wrapping intelligently
gqq	Rewrap current line
gqj	Rewrap current and following line
gq}	Rewrap from current line to end of paragraph
Buffers, Tabs, Windows	
:bnext, :bprev, :bdelete	Next / previous / delete buffer
:bufdo *command*	Execute command on all buffers eg command *bdelete* closes all open buffers
:sp *file* :vsp *file*	Split current window horizontally or vertically. With *file*, edit that file in the new window.
:new or CTRL-W n	Open a new windows
:tabnew	Open a new tab
:new *file*	Open *file* in new window
:clo, CTRL-W c	Close current window
:on, CTRL+W o	Make current window the only visible one
:qa	Close all buffers and exit

CTRL-W j, k	Move cursor to next/previous window
CTRL-W h, l, t, b	Move cursor to left/right/top/bottom window
CTRL-W K, B, H, L	Move current window top/bottom/far left/far right
CTRL+W f CTRL+W gf	Open file at cursor in new windows / tab
CTRL+W r, R	Rotate windows down/up
CTRL+W +, -	Increase/decrease current window size
CTRL+W =	Make all windows the same height
Miscellaneous	
K	Look up word under cursor in help
z=,]s,[s, zg,zw	Suggest corrections (also insert mode CTRL-X s) / next/prev spelling error / add/remove word to exception list
qx, @x	Record typed character to *x* / run *x* (precede by *n* to run *n* times, eg 5@x).
q:	Command history
<, > {motion}	Shift text described by *motion* command left/right by one shift width, eg <} shifts paragraph left
<<, >>	One shiftwidth left / right; 3>> shifts three lines
:1,$> :1,$<	Move entire file 1 shiftwidth to the right / left
:%norm jdd	Delete every other line (example of using normal mode command on command line; powerful!).
:sort u / n	Sort, remove duplicates / Sort by number
:sh, :term	Start external shell / internal terminal
:so %	Source (read commands from) current file, eg vimrc after changes
:set path=path	Search path for files
:noh	Remove highlight on highlighted text
:helptags ~/vimfiles/doc	Install help file
:options	Display help for options and show current values (:set all dumps settings)
:verbose map *key*	Shows where *key* was mapped
:verb set *str*	Shows where *str* was set
:let g: :let v:	Show global/local variables
:set [no]spell	Disable/enable spellchecking
:set list	Display tabs and eol character
:set ft=html	Set filetype to html (use any ext.)
:mess	Show messages
:mksession	Save session. Use :source mysession.vim to restore. For views, use :mkview / :loadview
:tab help *index*	Open help in new tab for *index* (or whatever)
ex	
:s/old/new/g	Substitute in current line
:%s/old/new/g	Substitute in current file
:.+1,$s/old/new/g	Substitute in all lines after current. Can use line numbers (eg 3,45s/…) and marks (eg 'a,'bs/…)
:^,.s/old/new/g	Substitute from start of file to cursor
\v	Escapes regex Eg :%s/\(good|nice\)/awesome/g can be written as %s/\v(good|nice)/awesome/g
:&	Repeat last substitute command

:m	Move lines, eg :30,60m0 ($ to move to end)
:co or :t	Copy lines, eg :5,10t105
:.,/*pattern*/co$	Copy from current line through line containing *pattern* to end of file
:d	Delete lines, eg 3,10d
:g[!]/*pat*/*cmd*	Execute *cmd* on all lines containing *pat*. Use ! for all lines not containing pattern. eg g/Unix/p prints all lines containing Unix; g/Name:/s/tom/Tom/ change "tom" to "Tom" on all lines containing "Name:". :v is the same as :g!. See reference 1 for examples.
:center	Centre line

File Commands

:e *file*	Edit *file*
:e!	Return to version of current file at time of last write (save)
ZZ or :x or :wq	Write (save) and quit file
:w, :w!	Write (save) file / override protection
:30,60w *newfile*	Write lines 30 to 60 to *newfile*
:30,60w>>*file*	Append lines 30 to 60 to *file*
:w !sudo tee %	Write file using sudo
:q	Quit file
:q!	Quit file, overriding protection
Q	Quit vi and invoke ex
:n	Edit next file
:e#	Edit alternate file
%	Current filename
#	Alternate filename
:r *file*	Read in contents of file after cursor
:e .	Explore dir. Can change dot to path.
:e#	Returns to previous window
gf	Open file at cursor at same window

Links

1. Best of Vim tips: http://zzapper.co.uk/vimtips.html
2. The power and joy of Vim. Derek Wyatt: https://vimeo.com/15443936
3. Functions: https://vimrcfu.com
4. Plugins: https://vimawesome.com/