Vim Cheat Sheet

Basic Movement

`h l k j`	character left, right; line up, down
`b w`	word/token left, right
`ge e`	end of word/token left, right
`{ }`	beginning of previous, next paragraph
`( )`	beginning of previous, next sentence
`0 gm`	beginning, middle of line
`^ $`	first, last character of line
n`G` *n*`gg`	line *n*, default the last, first
n`%`	percentage *n* of the file *(n must be provided)*
n`|`	column *n* of current line
`%`	match of next brace, bracket, comment, `#define`
n`H` *n*`L`	line *n* from start, bottom of window
`M`	middle line of window

Insert & Replace

`i a`	insert before, after cursor
`I A`	insert at beginning, end of line
`gI`	insert text in first column
`o O`	open a new line below, above the current line
`r`*c*	replace character under cursor with *c*
`gr`*c*	like `r`, but without affecting layout
`R`	replace characters starting at the cursor
`gR`	like `R`, but without affecting layout
`c`*m*	change text of movement command *m*
`cc` *or* `S`	change current line
`C`	change to the end of line
`s`	change one character and insert
`~`	switch case and advance cursor
`g~`*m*	switch case of movement command *m*
`gu`*m* `gU`*m*	lowercase, uppercase text of movement *m*
`<`*m* `>`*m*	shift left, right text of movement *m*
n`<<` *n*`>>`	shift *n* lines left, right

Deletion

`x X`	delete character under, before cursor
`d`*m*	delete text of movement command *m*
n`dd D`	delete *n* lines, to the end of line
n`J` *n*`gJ`	join *n* lines, without space
`:`*r*`d`	delete range *r* lines
`:`*r*`d`*x*	delete range *r* lines into register *x*

Insert mode

`^V`*c* `^V`*n*	insert char *c* literally, decimal value *n*
`^A`	insert previously inserted text
`^@`	same as `^A` and stop insert → command mode
`^R`*x* `^R^R`*x*	insert content of register *x*, literally
`^N ^P`	text completion before, after cursor
`^W`	delete word before cursor
`^U`	delete all inserted character in current line
`^D ^T`	shift left, right one shift width
`^K`*c*1*c*2 *or* *c*1←*c*2	enter digraph $\{c_1, c_2\}$
`^O`*c*	execute *c* in temporary command mode
`^X^E ^X^Y`	scroll up, down
⟨*esc*⟩ *or* `^[`	abandon edition → command mode

Copying

`"`*x*	use register *x* for next delete, yank, put
`y`*m*	yank the text of movement command *m*
n`yy` *or* *n*`Y`	yank *n* lines into register
n`p` *n*`P`	put register after, before cursor position (*n* times)
`]p [p`	like `p`, `P` with indent adjusted
`gp gP`	like `p`, `P` leaving cursor after new text

Advanced insertion

`g?`*m*	perform rot13 encoding on movement *m*
n`^A` *n*`^X`	+*n*, −*n* to number under cursor
`gq`*m*	format lines of movement *m* to fixed width
`:`*r*`ce` *w*	center lines in range *r* to width *w*
`:`*r*`le` *i*	left align lines in range *r* with indent *i*
`:`*r*`ri` *w*	right align lines in range *r* to width *w*
`!`*mc*	filter lines of movement *m* through command *c*
n`!!`*c*	filter *n* lines through command *c*
`:`*r*`!`*c*	filter range *r* lines through command *c*

Complex movement

`- +`	line up, down on first non-blank character
`B W`	space-separated word left, right
`gE E`	end of space-separated word left, right
n`_`	down *n* − 1 line on first non-blank character
`g0`	beginning of *screen* line
`g^ g$`	first, last character of *screen* line
`gk gj`	*screen* line up, down
`f`*c* `F`*c*	next, previous occurence of character *c*
`t`*c* `T`*c*	before next, previous occurence of *c*
`; ,`	repeat last `fFtT`, in opposite direction
`[[ ]]`	start of section backward, forward
`[] ][`	end of section backward, forward
`[( ])`	unclosed (,) backward, forward
`[{ ]}`	unclosed `{`, `}` backward, forward
`[m ]m`	start of backward, forward *Java* method
`[# ]#`	unclosed `#if`, `#else`, `#endif` backward, forward
`[* ]*`	start, end of `/* */` backward, forward

Text Object Selection

`a`*O* `i`*O*	text object, inner object
`w W s p`	word, WORD, sentence, paragraph
`t { [ ( < " '`	XML tags, enclosings

Visual mode

`v V ^V`	start/stop highlighting characters, lines, block
`o`	exchange cursor position with start of highlighting
`gv`	start highlighting on previous visual area
`aw as ap`	select a word, a sentence, a paragraph
`ab aB`	select a block ( ), a block `{ }`

Undoing & repeating

`u U`	undo last command, restore last changed line
`. ^R`	repeat last changes, redo last undo
n`.`	repeat last changes with count replaced by *n*
`:`*r*`g/`*p*`/`*c*	execute *Ex* command *c* on range *r* ⌊ where pattern *p* matches
`:`*r*`g!/`*p*`/`*c*	execute *Ex* command *c* on range *r* ⌊ where pattern *p* does not match

Macros & Registers

`q`*c* `q`*C*	record, append typed characters in register *c*
`q`	stop recording
`@`*c*	execute the content of register *c*
n`@@`	repeat previous `@` command *n* times
`:@`*c*	execute register *c* as an *Ex* command
`:reg` [*c*]	show contents of reg(s)
`"" "%`	special regs: last del or yank, filename
`"/ ":`	special regs: last search, command line
`"* "+`	special regs: clipboard (X11 selection), clipboard
`". "0 "1.."9`	special regs: last insert, yank, del/change

Search & substitution

`/`*s* `?`*s*	search forward, backward for *s*
`/`*s*`/`*o* `?`*s*`?`*o*	search fwd, bwd for *s* with offset *o*
`n` *or* `/`	repeat forward last search
`N` *or* `?`	repeat backward last search
`# *`	search backward, forward for word under cursor
`g# g*`	same, but also find partial matches
`gd gD`	local, global definition of symbol under cursor
`:`*r*`s/`*f*`/`*t*`/`*x*	substitute *f* by *t* in range *r* ⌊ *x* : `g`—all occurrences, `c`—confirm changes
`:`*r*`s` *x*	repeat substitution with new *r* & *x*

Special characters in search patterns

`. ^ $`	any single character, start, end of line
`\< \>`	start, end of word
`[`c_1`-`c_2`]`	a single character in range $c_1..c_2$
`[^`c_1`-`c_2`]`	a single character not in range
`\i \k \I \K`	an identifier, keyword; excl. digits
`\f \p \F \P`	a file name, printable char.; excl. digits
`\s \S`	a white space, a non-white space
`\e \t \r \b`	⟨*esc*⟩, ⟨*tab*⟩, ⟨↵⟩, ⟨←⟩
`\= * \+`	match 0..1, 0..∞, 1..∞ of preceding atoms
`\\|`	separate two branches (≡ *or*)
`\( \)`	group patterns into an atom
`\& \`*n*	the whole matched pattern, n^{th} `()` group
`\u \l`	next character made upper, lowercase
`\c \C`	ignore, match case on next pattern

Offsets in search commands

n *or* `+`*n*	*n* line downward in column 1
`-`*n*	*n* line upward in column 1
`e+`*n* `e-`*n*	*n* characters right, left to end of match
`s+`*n* `s-`*n*	*n* characters right, left to start of match
`;`*sc*	execute search command *sc* next

Buffers

`:ls`	display all buffers
`:bad :bd`	add, delete
`:bn :bN :bf :bl`	next, previous, first, last buffer
`:bufdo` *c*	execute command *c* in each buffer
`:bufdo undo`	undo change in each buffer
`:b`*n*	edit buffer *n*

Arguments

`:args` *files*	print, set argument list
`:all :sall`	open a window for every file in the arg list
`:arga` *files*	add *files* to arg list
`:argd` *files*	delete *files* from arg list
`:wn :wN`	write file and edit next, previous one
`:n :N`	edit next, previous file in list
`:argdo` *c*	execute command *c* for every file in list

Tabs

`:tabs`	list tabs
`:tabe` *file*	new tab
`:tabf` *file*	find *file* and open new tab
`:gt :gT :tabr :tabl`	next, previous, first, last tab
`^Wgf`	file under cursor
`:tabc` *n* `:tabo`	close tab, close others
`:tabdo` *c*	execute command in each tab

Ex commands (↩)

`:e` *f*	edit file *f*, unless changes have been made
`:e!` *f*	edit file *f* always (by default reload current)
`:`*r*`w`	write range *r* to current file
`:`*r*`w` *f*	write range *r* to file *f*
`:`*r*`w>>`*f*	append range *r* to file *f*
`:wall`	write all changed buffers
`:q :q!`	quit and confirm, quit and discard changes
`:wq` *or* `:x` *or* `ZZ`	write to current file and exit
⟨*up*⟩ ⟨*down*⟩	recall commands starting with current
`:r` *f*	insert content of file *f* below cursor
`:r!` *c*	insert output of command *c* below cursor
`:`*r*`co` *a* `:`*r*`m` *a*	copy, move range *r* below line *a*
`:pwd`	print the current directory name
`:cd` *path*	change the current directory to *path*
`:diffs` *f*	diff mode with file *f*
`:norm` *cmds*	execute normal mode commands
`:Explore`	open file browser in current window
`:Sexplore :Vexplore`	file browser in split window, vertically
`:sh :!`*c*	start shell, execute command *c* in shell
`:redir>`*f*	redirect output to file *f*

Ex ranges

`, ;`	separates two lines numbers, set to first line
n	an absolute line number *n*
`. $`	the current line, the last line in file
`% *`	entire file, visual area
`'`*t*	position of mark *t*
`/`*p*`/ ?`*p*`?`	the next, previous line where *p* matches
`+`*n* `-`*n*	$+n$, $-n$ to the preceding line number

Command-line Editing

`^v`*c*	insert char *c* literally
`^V`*n*	enter decimal value of character
`^R`*c*	insert contents of register *c*
`^W`	delete the word in front of the cursor
`^U`	remove all characters
`^B ^E`	cursor to beginning, end of command-line

Multi-windowing

`^Ws` *or* `:split`	split window in two
`^Wv` *or* `:vsplit`	vertically split window in two
`^Wn` *or* `:new`	create new empty window
`^Wo` *or* `:on`	make current window one on screen
`^Wc` *or* `:close`	close current window
`^Wj ^Wk`	move to window below, above
`^Ww ^W^W`	move to window below, above (wrap)
`q: q/`	cmd-line history window, search history
`:windo` *c*	execute command *c* in each window

Scrolling

`^E ^Y`	scroll line up, down
`^D ^U`	scroll half a page up, down
`^F ^B`	scroll page up, down
`zt` *or* `z`	set current line at top of window
`zz` *or* `z.`	set current line at center of window
`zb` *or* `z-`	set current line at bottom of window
`zh zl`	scroll one character to the right, left
`zH zL`	scroll half a screen to the right, left

Folding

`zf`*m*	create fold of movement *m*
`:`*r*`fo`	create fold for range *r*
`zd zE`	delete fold at cursor, all in window
`zo zc zO zC`	open, close one fold; recursively
`[z ]z`	move to start, end of current open fold
`zj zk`	move down, up to start, end of next fold

Marks and motions

`m`*c*	mark current position with mark $c \in [a..Z]$
`` ` ``*c* `` ` ``*C*	go to mark *c* in current, *C* in any file
`` ` ``0..9	go to last exit position
``` `` `" ```	go to position before jump, at last edit
``` `[ `] ```	go to start, end of previously operated text
`:marks`	print the active marks list
`:jumps`	print the jump list
n`^O`	go to n^{th} older position in jump list
n`^I`	go to n^{th} newer position in jump list

Tags

`:ta` *t*	jump to tag *t*
`:`*n*`ta`	jump to n^{th} newer tag in list
`^] ^T`	jump to the tag under cursor, return from tag
`:ts` *t*	list matching tags and select one for jump
`:tj` *t*	jump to tag or select one if multiple matches
`:tags`	print tag list
`:`*n*`po :`*n*`^T`	jump back from, to n^{th} older tag
`:tl`	jump to last matching tag
`^W} :pt` *t*	preview tag under cursor, tag *t*
`^W]`	split window and show tag under cursor
`^Wz` *or* `:pc`	close tag preview window

Key mapping & abbreviations

`:map` *c e*	map $c \mapsto e$ in normal & visual mode
`:map!` *c e*	map $c \mapsto e$ in insert & cmd-line mode
`:unmap` *c* `:unmap!` *c*	remove mapping *c*
`:mk` *f*	write current mappings, settings... to file *f*
`:ab` *c e*	add abbreviation for $c \mapsto e$
`:ab` *c*	show abbreviations starting with *c*
`:una` *c*	remove abbreviation *c*

Miscellaneous

`K`	lookup keyword under cursor with `man`
`:make`	start `make`, read errors and jump to first
`:cn :cp`	display the next, previous error
`:cl :cf`	list all errors, read errors from file
`g^G`	show cursor column, line, and character position
`ga`	show ASCII value of character under cursor

Best of Vim Tips

`ci"`	change text inside "..."
`daw`	delete a word
`ct`*c*	replace text until next occurrence of *c*
`xp`	exchange 2 characters
`/.*fred\&.*joe`	search for FRED AND JOE in any ORDER
`3/joe/e+1`	find 3rd joe cursor set to End of match plus 1
`/begin\_.*end`	search over possible multiple lines
`/\<fred\>/`	search for fred but not alfred or frederick
`/^\n\{3}`	find 3 empty lines
`/\(^str.*\n\)\{2}`	find 2 successive lines starting with str
`/bugs\(\_.\)*bunny`	bugs followed by bunny anywhere in file
`/^\([^,]*,\)\{8}`	repeating the Regexp
`^\(.*nos.*\)\@!.*str.*$`	find str in lines not containing nos
`:%s/old/new/gi`	replace case insensitive
`:2,35s/old/new/g`	replace between lines 2 and 35
`:5,$s/old/new/g`	replace all occurrences from line 5 to EOF
`:%s/^/hello/g`	replace the begining of each line by hello
`:%s/$/Harry/g`	replace the end of each line by Harry
`:%s/ *$//g`	delete all white spaces at end of lines
`:%s/^\(.*\)\n\1$/\1/`	delete duplicate lines
`:%s,\(all/.*\)\@<=/,_,g`	replace all / with _ AFTER "all/"
`:g/string/d`	delete all lines containing string
`:v/string/d`	delete all lines not containing string
`:g/^\s*$/d`	delete all blank lines
`:g/{/ ,/}/- s/\n\+/\r/g`	del empty lines only between {...}
`:g/fred/,/joe/j`	join lines
`:g/fred/y A`	append all lines containing fred to register a
`:g/^/t.`	duplicate every line
`:v/./,/./-j`	compress empty lines
`:%norm jdd`	delete every other line
`:1,10 w >> outfile`	append lines 1 to 10 to outfile
`:bufdo /searchstr/`	multiple file search
`:bufdo %s/old/new/g`	multiple file search & replace
`:earlier 15m`	set doc back to how it was 15 minutes ago
`:%!xxd :%!xxd -r`	enter hex edit mode, revert
`:%!sort -u`	use external program to filter content
`:.g/^/ exe ".!sed 's/N/X/'" | s/I/Q/`	chain external cmd
`gvim -O file1 file2`	open with vertical split (-o horizontal)
`:w !sudo tee %`	save current file as root