

vim Quick Reference		gk, gj	Up or down one <i>screen</i> line	?<string><cr>	To previous occurrence of <string>
Many keys allow a <count> before them, to execute the command multiple times		0	To first character on line	n	Repeat last / or ?
Many commands work line-wise by doubling them up (e.g. d = delete, dd = delete line)		^	To first NW character on line	N	Reverse of last / or ?
Command line commands (starting with :) require <cr> to run		\$	To last character on line	/<string>/e<cr>	To end of next occurrence
NW = non-whitespace, (I) = in insert mode		w	To next word	/<string>/e+1<cr>	To after end of next occurrence
		b	To previous word	/<string>/+<num><cr>	Line <num> after <string>
vi Command Line Flags		e	To space before next word	m<char>	Mark position with letter <char>
vi <file>	Edit <file>			'<char>	Jump to mark <char>
vi +<num> <file>	Edit <file> at line <num>	W	As w (space delimited only)	` `	To position before last mark jump / search
vi + <file>	Edit <file> at last line	B	As b (space delimited only)	' '	To line before last jump / search
Mode Change Commands		E	As e (space delimited only)		
a	Append text after highlighted character	+, <CR>	To next line (first NW)	<num>G	To line <num>
A	Append text after last character on line	-	To previous line (first NW)	G	To last line
i	Insert text before highlighted character	)	To next sentence	<num>	To column <num>
I	Insert text before first NW character on line	(	To previous sentence	%	To matching bracket
C<motion>	Delete up to <motion>, then insert	}	To next paragraph (i.e. blank line)	Window Scrolling	
C	Delete rest of line, then insert	}	To next paragraph (i.e. blank line)	C - f	Scroll down (forwards) a page
s	Delete current character, then insert	{	To previous paragraph	C - d	Scroll down half a page
S	Delete current line, then insert			C - e	Scroll down one line
o	Add new line after current, then insert	H	To first line of window	C - b	Scroll up (backwards) a page
O	Add new line before current, then insert	M	To middle line of window	C - u	Scroll up half a page
R	Enter replace (overtyping) mode	L	To last line of window	C - Y	Scroll up one line
Esc	Return to normal mode			Z<cr>	Scroll so current line is at top
v	Enter / exit visual mode	f<char>	To next <char> on line (inclusive)	Z .	Scroll so current line is at middle
V	Enter / exit linewise visual mode	t<char>	To next <char> on line(exclusive)	Z -	Scroll so current line is at bottom
gv	Enter visual mode, remember previous area	F<char>	To previous <char> on line(inc)	File commands (require <cr> to end)	
Movement Commands		T<char>	To previous <char> on line (exc)	:w	Write current file
k		;	Repeat last f, F, t or T	:wa	Write all files
h l	Cursor keys	,	Reverse of last f, F, t or T	:w <file>	Write to <file>
j		/<string><cr>	To next occurrence of <string>	:w! <file>	Write to <file>, force overwrite

:q	Quit current file (! forces)	dd	Cut line	Other Commands	
:qa	Quit all files (! forces)	x	Cut character under cursor	><motion>	Indent up to <motion>
:wq	Writes file, then quits (! forces)	X	Cut character before cursor	<<motion>	Unindent up to <motion>
:e	Re-edit (load external changes, ! forces)	y<motion>	Yank (copy) <motion>	J	Join line to next
:e <file>	Edit (open) <file>	yy, Y	Yank line	C - a	Increase number / char under cursor (I)
:e <file>	Edit <file>	p	Put (paste) after cursor	C - x	Decrease number / char under cursor (I)
:e +<num> <file>	Edit at line <num> (default last)	P	Put before cursor	*	Find word under cursor downwards
:enew	Edit a new file	gp, gP	Put, leaving cursor after text	#	Find word under cursor upwards
		]p, ]P	Put, adjusting indent to current line	g*, g#	... including part words
:r <file>	Insert <file> at cursor	"<char><clipbrd action>	Use register <char>	C - n	Word completion (I)
:! <cmd>	Run <cmd> in shell	"_	Do not store in any register	C - x C - l	Line completion (I)
!! <cmd>	Insert output of <cmd> at cursor	:reg, :di	Display contents of all registers	C - r=7*6	Insert 42 into document (I)
				:<range>S / <string1> / <string2>	
Buffers & Splits		Macros		Find <string1> in <range>, replace with <string2>	
:ls	List all open buffers	q<char>	Start recording, storing in register <char>	C - g	Display current file information
:ls!	List all (including unlisted)	q	Stop recording		
:b <num>	Go to buffer <num>	@<char>	Run macro contained in register <char>	If this key doesn' work:	Try:
:sb <num>	Split window & edit buffer <num>	@@	Repeat previous @ command	Backspace	C - h
:bn	Go to next buffer in list			Tab	C - i
:bp	Go to previous buffer in list	Undoing		Esc	C - [
:bf	Go to first buffer in list	u	Undo (multiple times)		
:bl	Go to last buffer in list	U	Undo all changes to current line	Ranges (two positions, with comma between)	
:bm	Go to next modified buffer in list	C - r or :redo	Redo undone changes	<num>	Absolute line number
:ba	Show all listed buffers on screen in splits			.	Current line
		Changing Case		\$	Last line in buffer
:new	Create new file in new split	~	Swap case of current character	%	Entire buffer
:sp <file>	Open <file> in new split	g~<motion>	Swap case up to <motion>	' <char>	Position of mark <char>
:vsp <file>	Open <file> in new vertical split	g~~	Swap case of current line	/ <string>	Next line where <string> occurs
C - w <cursor key>	Change active split	gu	Make <motion> text lowercase	? <string>	Previous line where <string> occurs
:clo	Close split, hide buffer	guu	Make current line lowercase	\ /	Next line where previously used search matches
		gU	Make <motion> text uppercase	\ ?	As above, but previous line
Register (clipboard) commands		gUU	Make current line uppercase	\ &	Next line where last substitute pattern matches
d<motion>	Cut (and delete) <motion>				