

VI Quick Reference

Entering/leaving vi

<i>vi name</i>	edit <i>name</i> at top
<i>vi +n name</i>	edit <i>name</i> at line <i>n</i>
<i>vi + name</i>	edit <i>name</i> at end
<i>vi -r</i>	list saved files
<i>vi -r name</i>	recover file <i>name</i>
<i>vi name1 name2 ...</i>	edit first file, rest via :n
<i>vi -t tag</i>	start at <i>tag</i>
<i>vi +/pat name</i>	search for <i>pat</i>
<i>view name</i>	read only mode
<i>ZZ</i>	exit from vi, saving changes
<i>~Z</i>	stop vi for later resumption

The display

Last line	Error message, echoing input to : / ? and ! ; feedback about <i>i/o</i> and large changes.
@ lines	On screen only, not in file.
~ lines	Lines past end of file.
~x	Control characters, ~? is delete.
tabs	Expand to spaces; cursor at last.

Vi states (or modes)

Command	Normal and initial state. Others return here. ESC (escape) cancels partial command.
Insert	Entered by a A i I o O c C s S . Arbitrary text then terminates with ESC or abnormally with interrupt.
Last line	Reading input for : / ? or ! ; terminates with ESC or CR to execute, interrupt to cancel.

Counts before vi commands

line/column number	<i>z G</i> —
scroll amount	<i>~D ~U</i>
replicate insert	<i>a i A I</i>
repeat effect	most other commands

Simple commands

<i>dw</i>	delete a word
<i>de</i>	delete a word leaving punctuation
<i>dd</i>	delete a line
<i>3dd</i>	delete 3 lines

Insert/ESC

<i>itertESC</i>	insert <i>text</i>
<i>cwnewESC</i>	Change word to <i>new</i>
<i>eaESC</i>	pluralize word
<i>xp</i>	transpose characters

Interrupting, cancelling

ESC	end insert or incomplete cmd (delete or rubout) interrupts
<i>~?</i>	reprint screen if <i>~?</i> scrambles it
<i>~L</i>	

File manipulation

<i>:w</i>	write back changes
<i>:wq</i>	write and quit
<i>:q</i>	quit
<i>:q!</i>	quit, discard changes
<i>:e name</i>	edit file <i>name</i>
<i>:e!</i>	reedit, discard changes
<i>:e + name</i>	edit, starting at end
<i>:e + n</i>	edit starting at line <i>n</i>
<i>:e #</i>	edit alternate file
<i>~#</i>	synonym for <i>:e #</i>
<i>:w name</i>	write file <i>name</i>
<i>:w! name</i>	overwrite file <i>name</i>
<i>:sh</i>	run shell
<i>:!cmd</i>	run <i>cmd</i> , then return
<i>:n</i>	edit next file in arglist
<i>:n args</i>	specify new arglist
<i>:f</i>	show current file and line
<i>~G</i>	synonym for <i>:f</i>
<i>:ta tag</i>	to tag file entry <i>tag</i>
<i>: </i>	<i>:ta</i> , following word is <i>tag</i>

Positioning within file

<i>~F</i>	forward screenfull
<i>~B</i>	backward screenfull
<i>~D</i>	scroll down half screen
<i>~U</i>	scroll up half screen
<i>G</i>	goto line (end default)
<i>/pat</i>	next line matching <i>pat</i>
<i>?pat</i>	prev line matching <i>pat</i>
<i>n</i>	repeat last / or ?
<i>N</i>	reverse last / or ?
<i>/pat/+n</i>	<i>n</i> 'th line after <i>pat</i>
<i>?pat?-n</i>	<i>n</i> 'th line before <i>pat</i>
<i> </i>	next section/function
<i> </i>	previous section/function
<i>%</i>	find matching () { } [or]

Adjusting the screen

<i>~L</i>	clear and redraw
<i>~R</i>	retype, eliminate @ lines
<i>zCR</i>	redraw, current at window top
<i>z-</i>	... at bottom
<i>z.</i>	... at center
<i>/pat/z-</i>	<i>pat</i> line at bottom
<i>zn.</i>	use <i>n</i> line window
<i>~E</i>	scroll window down 1 line
<i>~Y</i>	scroll window up 1 line

Marking and returning

<i>‘</i>	previous context
<i>‘</i>	... at first non-white in line
<i>‘,</i>	mark position with letter <i>x</i>
<i>mx</i>	to mark <i>x</i>
<i>‘x</i>	... at first non-white in line
<i>‘x</i>	

Line positioning

<i>H</i>	home window line
<i>L</i>	last window line
<i>M</i>	middle window line
<i>+</i>	next line, at first non-white
<i>-</i>	previous line, at first non-white
<i>CR</i>	return, same as +
<i>j</i> or <i>↓</i>	next line, same column
<i>k</i> or <i>↑</i>	previous line, same column

Character positioning

<i>^</i>	first non-white
<i>0</i>	beginning of line
<i>\$</i>	end of line
<i>l</i> or <i>→</i>	right
<i>h</i> or <i>←</i>	left
<i>~ l</i>	same as <i>h</i>
<i>space</i>	same as <i>l</i>
<i>fx</i>	find <i>x</i> forward
<i>Fx</i>	find <i>x</i> backward
<i>tx</i>	find just before <i>x</i> forward
<i>Tx</i>	find just before <i>x</i> backward
<i>;</i>	repeat last F F t or T
<i>,</i>	inverse of ;
<i>%</i>	to column <i>n</i>
<i>%</i>	find matching () { } [or]

Words, sentences, paragraphs

w next word forward
b back one word
e end of word
) to next sentence
} to next paragraph
(to next sentence
{ to previous paragraph
W blank delimited word
B back W
E to end of W

Commands for LISP

) Forward s-expression
} ... but don't stop at atoms
(Back s-expression
} ... but don't stop at atoms

Corrections during insert

~H erase last character
~W erase last word
erase your erase, same as ~H
kill your kill, erase this line
\ ends insert, back to command
ESC ends insertion, back to command
~? interrupt, terminates insert
~D backtab over *autoindent*
↑~D kill *autoindent*, save for next
0~D ... but at margin next also
~V quote non-printing character

Insert and replace

a append after cursor
i insert before
A append at end of line
I insert before first non-blank
o open line below
O open line above
rx replace single character with *x*
R replaces characters

Operators (double to affect lines)

d delete
c change
< left shift

> right shift
! filter through unix command
= indent for LISP
y yank lines to buffer

Miscellaneous operations

C change rest of line
D delete rest of line
s substitute chars
S substitute lines
J join lines
x delete characters
X ... before cursor
Y yank lines

Yank and put

P put back lines
P put before
"xp put from buffer *x*
"xy yank to buffer *x*
"xd delete into buffer *x*

Undo, redo, retrieve

u undo last change
U restore current line
· repeat last change
"dp retrieve *dth* last delete

Scanning pattern formation

· beginning of line
\$ end of line
· any character
< beginning of word
> end of word
[*abc*] any character in *abc*
[~*abc*] any character not in *abc*
[*x - y*] any character between *x* and *y*
* any number of preceding

Useful options (:se option, :se noop- tion)

autoindent ai apply indent
autowrite aw write before changing files
ignorecase ic in scanning
lisp () { } are s-expressions

list print ~I for tabs, \$ at end
magic · { * special in patterns
number show line numbers
paragraphs para macro names that start paragraph
redraw simulate smart terminal
scroll command mode lines
sections sect macro names for sections
shiftwidth shw for < >, and input ~D
showmatch sm to) and] as typed
slowopen slow choke updates during insert
window visual mode lines
wrapscan ws around end of buffer?
wrapmargin wm automatic line splitting

Aliasing and Mapping

:ab <enter-str> <real-str>
:ab <enter-str> <real-str>
When <enter-str> is typed,
replace by <real-str>
:ab move more corrects spelling mistake
:ab tse *Transactions on Software Engineering*
expands tse
:map char commands (re)defines char
:map ; : ; is now equivalent to :
:map g :l,\$ s/ Frequent command
:map F ifor (i = 1; i <= N; i++)~M~MESCKtEN
a C for loop
:map B i{ESCea}ESC LaTeX boldface a word
:map F i}fnt -70~M Format a paragraph

Jeff Offutt, 1995