

vi / ex Quick Reference

Interrupting, canceling

<ESC> end insert or complete command
^? (delete or rubout) interrupts
^L reprint screen if ^? scrambles it

File manipulation

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

Positioning within file

^F forward screenful
^B backward screenful
^D scroll down half screen
^U scroll up half screen
G goto line (end default)
/pat next line matching *pat*
?pat prev line matching *pat*
n repeat last / or ?
N reverse last / or ?
/pat/+n n'th line after *pat*
?pat?-n n'th line before *pat*
|| next section/function
|| previous section/function
% find matching () { or }

Adjusting the screen

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

Marking and returning

`` previous context
' ' ... at first non-white in line
nx mark position with letter *x*
`x to mark *x*
'x ... at first non-white in line

Line Positioning

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

Character positioning

- first non-white
O beginning of line
\$ end of line
h or _ forward
l or _ backwards
^H same as _
<Space> same as _
fx find *x* forward
Ff backward
tx upto *x* forward
Tx back upto *x*
; repeat last *f F t* or *T*
, reverse of ;
/ to specified column
% find matching ({ } or }

Words, sentences, paragraphs

w	word forward
b	back word
e	end of word
)	to next sentence
}	to next paragraph
(back sentence
{	back 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

Correction during Insert

^H	erase last character
^W	erases last word
erase	your erase, same as ^H
kill	your kill, erase input this line
\	escapes ^H your erase and kill
<ESC>	ends insertion, back to command
^?	interrupt, terminates insert
^D	backtap 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 above
rx	replace single char with x
R	replace characters

Operators (double to affect lines)

d	delete
c	change
<	left shift
>	right shift
!	filter through 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 d th last delete

Ex Quick Reference

Entering/leaving ex

% ex name	edit name , start at end
% ex +n name	... at line n
% ex -t tag	start at tag
% ex -r	list saved files
% ex -r name	recover file name
% ex name ...	edit first, rest via :n
% ex -R name	read only mode
:x	exit, saving changes
:q!	exit, discarding changes

Ex states

Command	Normal and initial state. Input prompted by : . Your kill character cancels partial command.
Insert	Entered by a i and c . Arbitrary text then terminates with line having only . character on it or abnormally with interrupt.
Open/visual	Entered by open or vi , terminates with Q or ^\ .

Ex commands

abbrev	ab	recover	rec
append	a	rewind	rew
args	ar	set	se
change	c	shell	sh
copy	co	source	so
delete	d	stop	st
edit	e	substitute	s
file	f	unabbrev	una
global	g	undo	u
insert	i	unmap	unm
join	j	version	ve
list	l	visual	vi
map		write	w
mark	ma	xit	x
move	m	yank	ya
next	n	window	w
number	nu	escape	!
open	o	lshift	<
preserve	pre	print next	<CR>
print	p	resubst	&
put	pu	rshift	>
quit	q	scroll	^D
read	re		

Ex command address

n	line n	/pat	next with pat
.	current	?pat	previous with
pat			
\$	last	x-n	n before x
+	next	x,y	x through y
-	previous	'x	marked with x
+n	n forward	''	previous context
%	1,\$		

Specifying terminal type

% setenv TERM type **cs**h and all version 6
\$ TERM-type; export TERM **sh** in Version 7

Some terminal types

2621	2645	300s	33
37	4014	43	733
745	act4	act5	adm3
adm31	adm3a	c100	dm1520
dm2500	dm2510	dm3025	dw1
dw2	gt40	gt42	h1500
h1510	h19	i100	mime
owl	sun	t1061	vt52
vt100			

Initializing options

EXINIT	place set's here in environment var.
set x	enable option
set nox	disable option
set x=val	give value val
set	show changed options
set x?	show value of option x

Useful options

autoindent ai	supply indent
autowrite aw	write before changing files
ignorecase ic	in scanning
lisp	() {} are s-exp's
list	print ^I for tab, \$ at end
magic . [*	special in patterns
number nu	number lines
paragraphs para	macro names which start ...
redraw	simulate smart terminal
scroll	command mode lines
sections sect	macro names ...
shiftdwidth sw	for < >, and input ^D
showmatch sm	for) and } as typed
slowopen slow	choke updates during insert
window	visual mode lines
wrapsan ws	around end of buffer?
wrapmargin wm	automatic line splitting

Scanning pattern formation

-	beginning of line
\$	end of line
.	any character
\<	beginning of word
\>	end of word
[str]	any char in str
[_str]	... not in str
[x-y]	... between x and y
*	any number of preceding

Vi Quick Reference

Entering/leaving vi

% **vi** *name* edit *name* at top
% **vi** **+n** *name* ... at line *n*
% **vi** **+** *name* ... at end
% **vi** **-r** list saved files
% **vi** **-r** *name* recover file *name*
% **vi** *name* ... edit first; rest via **:n**
% **vi** **-t** *tag* start at *tag*
% **vi** **+/pat** *name* search for *pat*
% **view** *name* read only mode
ZZ exit from vi, saving change
^Z stop vi for later resumption

The display

Last line Error messages, echoing input to **:** **/?** and **!**, feedback about i/o 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

Command Normal and initial state.
Others return here. **<ESC>**

Insert

(escape) cancels partial command.

Entered by **a i A I o O c C s S R**. Arbitrary text then terminates with **<ESC>** character, or abnormally with interrupt.

Last line

Reading input for **;** **/** **?** or **!**; terminate with **<ESC>** or **<CR>** to execute, interrupt to cancel.

Counts before vi commands

line/column number **z G /**
scroll amount **^D ^U**
replicate insert **a i A I**
repeat effect most rest

Simple commands

dw delete a word
de ... leaving punctuation
dd delete a line
3dd ... 3 lines
i text<ESC> insert text *text*
cnwew<ESC> change word to *new*
eas<ESC> pluralize word
xp transpose characters

DOCUMENT CODE: UNIX-30203B

DATE REVISED: July 2, 1997