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 savdes
% **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 for 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 ˆ\\.

Ex commands

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

Ex command addresses

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*	*csh* and all version 6
$ TERM=*type***; export TERM**	*sh* in Version 7
See also *tset*(1)	

Some terminal types

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

Initializing options

EXINIT	place **set**'s here in environment var.
set *x*	enable option
set no_x_	disable option
set *x***=***val*	give value *val*
set	show changed options
set all	show all 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 ...
shiftwidth	sw	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

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 changes
ˆZ	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.
ˆ*x*	Control characters, ˆ? is delete.
tabs	Expand to spaces, cursor at last.

Vi states

Command	Normal and initial state. Others return here. ESC (escape) cancels partial command.
Insert	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 2000 cancel.

Counts before vi commands

line/column number	**z G |**
scroll amount	**ˆD ˆU**
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 *abc*
cw*new*ESC	change word to *new*
ea*s*ESC	pluralize word
xp	transpose characters

Interrupting, cancelling

ESC	end insert or incomplete cmd
ˆ?	(delete or rubout) interrupts
ˆL	reprint screen if **ˆ?** 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
ˆ↑	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
ˆG	synonym for **:f**
:ta *tag*	to tag file entry *tag*
ˆ]	**:ta**, following word is *tag*

Positioning within file

ˆF	forward screenfull
ˆB	backward screenfull
ˆD	scroll down half screen
ˆU	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

ˆL	clear and redraw
ˆR	retype, eliminate @ lines
zCR	redraw, current at window top
z−	... at bottom
z .	... at center
/*pat***/z−**	*pat* line at bottom
z*n* **.**	use *n* line window
ˆE	scroll window down 1 line
ˆY	scroll window up 1 line

Marking and returning

``	previous context
´´	... at first non-white in line
m_x_	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
0	beginning of line
$	end of line
h or →	forward
l or ←	backwards
ˆH	same as ←
space	same as →
f_x_	find _x_ forward
F_x_	**f** backward
t_x_	upto _x_ forward
T_x_	back upto _x_
;	repeat last **f F t** or **T**
,	inverse 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

Corrections during insert

ˆH	
ˆW	erases last word
erase	your erase, same as **ˆH**
kill	your kill, erase input this line
****	escapes **ˆH**, your erase and kill
ESC	ends insertion, back to command
ˆ?	interrupt, terminates insert
ˆD	backtab over *autoindent*
↑ˆD	kill *autoindent*, save for next
0ˆD	... but at margin next also
ˆV	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
r*x*	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
"*x***p**	put from buffer *x*
"*x***y**	yank to buffer *x*
"*x***d**	delete into buffer *x*

Undo, redo, retrieve

u	undo last change
U	restore current line
.	repeat last change
"*d* **p**	retrieve *d*'th last delete