

Ex vs. Teco

Brandon Moore and John Markiewicz

Neo-Vim vs Spacemacs

Brandon Moore and John Markiewicz

Vim vs. Emacs

Brandon Moore and John Markiewicz

Emacs is a great operating system, lacking only a decent editor

–Anonymous

Using a free version of vi is not a sin but a penance.

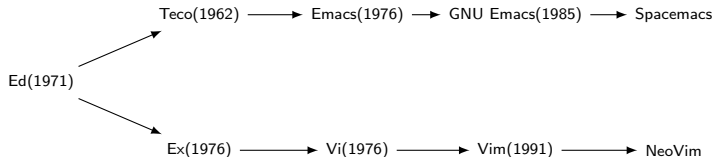
–Richard Stallman, head of the Church of Emacs

Using a free version of vi is not a sin but a penance.

- ~~–Richard Stallman, head of the Church of Emacs~~
- ~~–Saint IGNUcious, head of the Church of Emacs~~

A historic holy war

Emacs vs. Vi(m)





Let's look at Vim

version 1.1
April 1st, 06

vi / vim graphical cheat sheet

Esc
normal mode

~ toggle case	! external filter	@, play macro	# prev ident	\$ eol	% goto match	^ "soft" bol	& repeat :s	* next ident	(begin sentence) end sentence	_ "soft" bol down	+ next line
. goto mark	1	2	3	4	5	6	7	8	9	0 "hard" bol	- prev line	= autoformat
Q ex mode	W next word	E end word	R replace mode	T back 'till	Y yank line	U undo line	I insert at bol	O open above	P paste before	{ begin parag.	}	end parag.
q record macro	w next word	e end word	r replace char	t 'till	y yank	u undo	i insert mode	o open below	p paste after	[misc]	misc
A append at eol	S subst line	D delete to eol	F, "back" find ch	G eof/ goto ln	H screen top	J join lines	K help	L screen bottom	. ex cmd line	" reg. spec	bol/ goto col	
a append	s subst char	d delete	f find char	g extra cmds	h ←	j ↓	k ↑	l →	. repeat t/T/t/T	' goto mk. bol	\ not used!	
Z quit	X back-space	C change to eol	V visual lines	B prev word	N prev (find)	M screen mid'l	< un-indent	> indent	? find (rev.)			
Z extra cmds	X delete char	c change	V visual mode	b prev word	n next (find)	m set mark	, reverse t/T/t/T	. repeat cmd	/ find			

motion	moves the cursor, or defines the range for an operator
command	direct action command, if red, it enters insert mode
operator	requires a motion afterwards, operates between cursor & destination
extra	special functions, requires extra input
q.	commands with a dot need a char argument afterwards

bol = beginning of line, eol = end of line, mk = mark, yank = copy

words: quux(foo, bar, baz);
WORDS: quux(foo, bar, baz);

Main command line commands ('ex'):

:w (save), :q (quit), :q! (quit w/o saving)
:e f (open file f),
:s/s/x/y/g (replace 'x' by 'y' filewide),
:h (help in vim), :new (new file in vim),

Other important commands:

CTRL-R: redo (vim),
CTRL-F/-B: page up/down,
CTRL-E/-Y: scroll line up/down,
CTRL-V: block-visual mode (vim only)

Visual mode:

Move around and type operator to act on selected region (vim only)

Notes:

- (1) use "x before a yank/paste/del command to use that register ('clipboard') (x=a..z,*) (e.g.: "ay\$ to copy rest of line to reg 'a')
- (2) type in a number before any action to repeat it that number of times (e.g.: 2p, d2w, 5l, d4j)
- (3) duplicate operator to act on current line (dd = delete line, >> = indent line)
- (4) ZZ to save & quit, ZQ to quit w/o saving
- (5) zt: scroll cursor to top, zb: bottom, zz: center
- (6) gg: top of file (vim only), gf: open file under cursor (vim only)

Let's look at Emacs

GNU Emacs Reference Card

(for version 20)

Starting Emacs

To enter GNU Emacs 20, just type its name: **emacs**

To read in a file to edit, see Files, below.

Leaving Emacs

suspend Emacs (or iconify it under X)	C-x
exit Emacs permanently	C-x C-c

Files

read a file into Emacs	C-x C-f
save a file back to disk	C-x C-s
save all files	C-x s
insert contents of another file into this buffer	C-x i
replace this file with the file you really want	C-x C-v
write buffer to a specified file	C-x C-w
version control checkin/checkout	C-x C-q

Getting Help

The help system is simple. Type C-h (or F1) and follow the directions. If you are a first-time user, type C-h t for a tutorial.

remove help window	C-x 1
scroll help window	C-M-v
apropos: show commands matching a string	C-h a
show the function a key runs	C-h c
describe a function	C-h f
get mode-specific information	C-h m

Error Recovery

abort partially typed or executing command	C-g
recover a file lost by a system crash	M-x recover-file
undo an unwanted change	C-x u or C-r
restore a buffer to its original contents	M-x revert-buffer
redraw garbled screen	C-l

Incremental Search

search forward	C-s
search backward	C-r
regular expression search	C-M-f
reverse regular expression search	C-M-r
select previous search string	M-p
select next later search string	M-n
exit incremental search	RET
undo effect of last character	DEL
abort current search	C-g

Use C-s or C-r again to repeat the search in either direction. If Emacs is still searching, C-g cancels out the part not done.

© 1997 Free Software Foundation, Inc. Permissions on back. v2.2

Motion

entity to move over	backward	forward
character	C-b	C-f
word	M-b	M-f
line	C-p	C-n
go to line beginning (or end)	C-a	C-e
sentence	M-a	M-e
paragraph	M-{	M-}
page	C-x [C-x]
sexp	C-M-b	C-M-f
function	C-M-A	C-M-E
go to buffer beginning (or end)	M-<	M->
scroll to next screen	C-v	
scroll to previous screen	M-v	
scroll left	C-x <	
scroll right	C-x >	
scroll current line to center of screen	C-u	C-l

Killing and Deleting

entity to kill	backward	forward
character (delete, not kill)	DEL	C-d
word	M-DEL	M-d
line (to end of)	C-O C-k	C-k
sentence	C-x DEL	M-k
sexp	M-- C-M-k	C-M-k
kill region	C-w	
copy region to kill ring	M-w	
kill through next occurrence of char	M-x char	
yank back last thing killed	C-y	
replace last yank with previous kill	M-y	

Marking

set mark here	C-@ or C-SPC
exchange point and mark	C-x C-x
set mark any words away	M-@
mark paragraph	M-h
mark page	C-x C-p
mark sexp	C-M-@
mark function	C-M-h
mark entire buffer	C-x h

Query Replace

interactively replace a text string	M-x
using regular expressions	M-x query-replace-regexp
Valid responses in query-replace mode are	
replace this one, go on to next	SPC
replace this one, don't move	DEL
skip to next without replacing	!
replace all remaining matches	*
back up to the previous match	RET
exit query-replace	C-r
enter recursive edit (C-M-c to exit)	

Multiple Windows

When two commands are shown, the second is for "other frame."

delete all other windows	C-x 1
split window, above and below	C-x 2
delete this window	C-x 0
split window, side by side	C-x 3
scroll other window	C-M-v
switch cursor to another window	C-x o
select buffer in other window	C-x 4 b
display buffer in other window	C-x 4 C-o
find file in other window	C-x 4 f
find file read-only in other window	C-x 4 r
run Dired in other window	C-x 4 d
find tag in other window	C-x 4 .
grow window taller	C-x ^
shrink window narrower	C-x {
grow window wider	C-x }

Formatting

indent current line (mode-dependent)	TAB
indent region (mode-dependent)	C-M-\
indent sexp (mode-dependent)	C-M-q
indent region rigidly any columns	C-x TAB
insert newline after point	C-o
move rest of line vertically down	C-M-o
delete blank lines around point	C-x C-o
join line with previous (with arg, next)	M-~
delete all white space around point	M-\
put exactly one space at point	M-SPC
fill paragraph	M-q
set fill column	C-f
set prefix each line starts with	C-x .
set face	M-g

Case Change

uppercase word	M-u
lowercase word	M-l
capitalize word	M-c
uppercase region	C-x C-u
lowercase region	C-x C-l

The Minibuffer

The following keys are defined in the minibuffer.

complete as much as possible	TAB
complete up to one word	SPC
complete and execute	RET
show possible completions	?
fetch previous minibuffer input	M-p
fetch later minibuffer input or default	M-n
regexp search backward through history	M-r
regexp search forward through history	M-s
abort command	C-g

Type C-x ESC ESC to edit and repeat the last command that used the minibuffer. Type F10 to activate the menu bar using the minibuffer.

Why use Vim?

Pros of Vim

- ▶ Customizable
- ▶ Rich library of extensions
- ▶ Constantly improving
- ▶ Portable
- ▶ ...in multiple ways

Cons of Emacs

- ▶ Reliant on customization
- ▶ Comes with bloat
- ▶ Hello Carpal Tunnel
- ▶ Tries to do too much

Why use Emacs?

Pros of Emacs

- ▶ Flexible, powerful, pleasant to use language for adding features (elisp).
- ▶ Features.
- ▶ Compatibility with the past few decades' worth of software
- ▶ Org-mode
- ▶ Evil-mode for those used to vim keybindings

Cons of Vim

- ▶ Community fragmentation (Neovim vs Vim 8.0) and incompatibility between versions.
- ▶ Fewer features.
- ▶ Vimscript.
- ▶ Inconsistent async methods between Vim 8 and Neovim.
- ▶ Performance issues with syntax highlighting/line numbers/large files.

stdlinux (and problems with old systems)

- ▶ stdlinux is old and out of date. vim and emacs don't work well
- ▶ a basic .vimrc is needed to use vim on stdlinux
- ▶ It should be updated soon!

Vim

- ▶ vim help pages (via `:help <topic>`)
- ▶ `$vimtutor`
- ▶ <http://www.openvim.com/>

Emacs

- ▶ Emacs help features (internal documentation accessible via `C-h C-a`)
- ▶ Emacs wiki
- ▶ Worg (for org-mode)

<https://moore3071.github.io/vim-vs-emacs>