


vi(m) tutorial

Steven Ahrendt
PFB2014

command mode vs insert mode

- vim starts in command mode
- insert mode lets you type
 - command → insert: i (or a)
 - insert → command: 

Useful commands

Many commands start with a colon (:)

<code>:q</code>	quit
<code>:q!</code>	force quit (quit!)
<code>:w</code>	save
<code>:wq</code>	save and quit
<code>:w filename</code>	save with new filename
<code>:set nu</code>	show line numbers

Moving around

h j k l

← ↓ ↑ →

w b

forward / back

e

end of word

0 \$

beginning / end of line

Arrow keys are valid.

Letters are considered more efficient since you don't remove your hands from the home row

Modifying text

<code>dd</code>	delete whole line
<code>cc</code>	cut whole line
<code>yy</code>	copy whole line
<code>p</code>	paste line (after cursor)
<code>P</code>	paste line (before cursor)
<code>x</code>	delete character under cursor

These can be
modified with
numbers

note the lack of
a colon (:)

Searching text

<code>/string</code>	search forward for <i>string</i>
<code>?string</code>	search backward for <i>string</i>
<code>n</code>	repeat search, same direction
<code>N</code>	repeat search, opposite direction
<code>:%s/old/new/g</code>	replace 'old' with 'new'
<code>:nohl</code>	clear search highlighting

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	2	3	4	5	6	7	8	9	0 "hard" bol	- prev line	= auto ³ format
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 ^{1,3}	u undo	i insert mode	o open below	p paste ¹ 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. ¹ spec	bol/ goto col	
a append	s subst char	d delete ^{1,3}	f find char	g extra ⁶ cmds	h ←	j ↓	k ↑	l →	· repeat ; t/T/t/F	' goto mk. bol	\ not used!	
Z quit ⁴	X back-space	C change to eol	V visual lines	B prev WORD	N prev (find)	M screen mid'l	< un- ³ indent	> indent ³	? find (rev.)			
z extra ⁵ cmds	x delete char	c change ^{1,3}	v visual mode	b prev word	n next (find)	m set mark	reverse ; t/T/t/F	· 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/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, 5i, 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)

VIM Adventures

<http://vim-adventures.com/>

- online puzzle game for practicing/memorizing VIM's keyboard shortcuts
- “It's the ‘Zelda meets text editing’ game.”
- easy way to learn VIM with a more shallow learning curve

