

Vim Visual Cheat Sheet

Serge Y. Stroobandt

Copyright 2016, licensed under [Creative Commons BY-NC-SA](#)

This document is still under construction.

Introduction

Vim (and vi for that matter) is renowned for its notoriously steep learning curve.

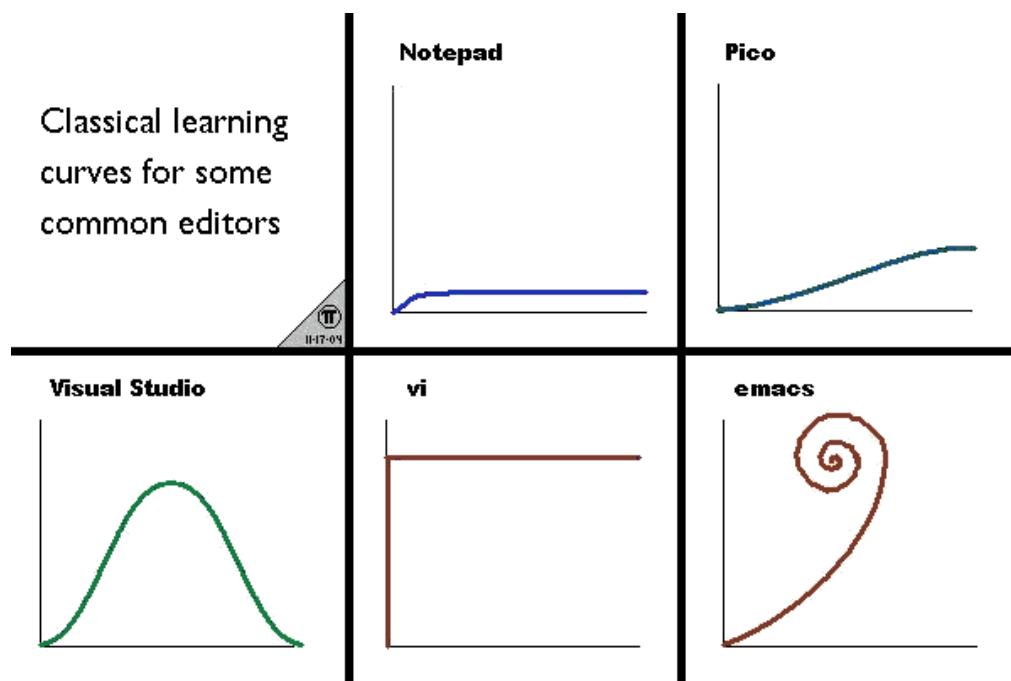


Figure 1: Comparison of editor learning curves *Source:*

This should not scare you off. If you have not done so yet, first read *“Why I Use Vim”* If you cannot bother reading it, here is my personal break down.

No, this is not an extreme case of fanboyhood. Vim is a way of life, or rather, a means towards extreme levels of productivity with the [Vim-Mark-down-Pandoc-make](#) input combo with any of the [LaTeX/Con-TeXt/XHTML/Prince/DZSlides](#) back-ends.

Among other things, Vim will teach you to use a keyboard professionally; keeping your index fingers stuck to the home row. Vim allows you to edit texts lightning fast without resorting to any computer mouse or other pointing device. Imagine the space you will gain on the folding table on your next transatlantic business flight! You will no longer have to use the `Home`, `End` or arrow keys, if you chose so.



vi gang sign

Vim's predecessor vi has been around since 1976 and has been continuously improved ever since. If computer application has been around for such an extended period in time, for sure, there must be a good reason to it! Are you not curious towards experiencing for yourself why?

My interest in Vim grew out of my annoyance with other text editors not recognising words, parenthesis, brackets, etc. I got fed up with manually tapping the cursor to the desired editing position. Vim is one of the few text editors which is a word processor in the true sense of the word. It is a word processor which effectively recognises word, bracket and parenthesis boundaries, counts words, etc.

Vim continuously records a small macro of your last command combination, which is easily accessible through the dot `.` key. Don't be deceived; this functionality is far more powerful than the usual repeat function!

Visual Cheat Sheet

For English keyboards

vim graphical cheat sheet
(english keyboard layout)

Esc normal mode																						
~ toggle case	! external filter	@ play macro	# search backwards	\$ end of line	% match (bracket)	^ "soft" begin of line	& repeat %	* search forward	(begin sequence) end sequence	_ "hard" begin of line	+ next line										
.` jump to mark	1	2	3	4	5	6	7	8	9	0	_ prev. line	= auto format										
		Q ex mode	W next WORD	E end WORD	R replace mode	T back tab	Y yank line	U undo line	I insert at bol	O open below	P paste before	{ begin para	} end para									
		q record macros	w next word	e end word	r replace char	t tab	y yank	u undo	i insert mode	o open before	p paste after	. misc	 misc									
		A append at eol	S subst. char	D delete to eol	F "back" find char	G end/ goto line	H screen top	J join lines	K look-up keyword	L screen bottom	: cmd. line	" register/ spec	 begin of line/ goto column									
		a append	s subst. char	d delete!	f find char	g extra/ cmd	h ←	j ↓	k ↑	l →	: repeat (u/r/W)	' goto mark	\ unused									
> indent	< un-indent	Z quit!	X back-space	C change to eol	V visual mode	B prev. word	N find prev	M screen middle	< un-indent	> indent	? find (rew)											
		z extra cmds	x delete char	c change	v visual mode	b prev. word	n find next	m set mark	/ never use (u/r/W)	* repeat command	/ find											

motion moves the cursor or defines the range for an operator

command direct action cmd, if red it enters insert mode

operator requires a motion afterwards, operates between cursor and destination

extra special functions, requires extra input

Q commands with a dot need a char argument afterwards

Main command line commands ("ex"):

- W [file] [save]
- Q [quit]
- Q! [quit without saving]
- W [save & quit]
- W [foo] [open file foo]
- n [new file]
- SP [split window horizontally]
- svp [split window vertically]
- reg [display content of named registers]
- Explore [dir] [open file-explorer]
- h [help]
- h help-gmail [list all commands]

Other important commands:

- CTRL + R [redo]
- CTRL + P / N [complete the current word]
- CTRL + W [move cursor to next window]
- [n] CTRL + S [toggle [n]h alternate file]
- CTRL + F / B [page up / down]
- CTRL + E / Y [scroll lin up / down]
- CTRL + V [block-visual mode]

Find and replace:

- :%s/<RegExp>/<String>/g [replace <RegExp> by <String> filewide]
- s/<RegExp>/<String>/ [search current line and replace first match]

Vim 7.x only commands:

- CTRL + X - CTRL + D [omni completion in insert mode]
- tabe [file] [edit file] in a new tab
- tabe [n] [close tab [n]]
- tabonly [close all other tab pages]
- tabmove [n] [move tab to position [n]]
- tab [cmd] [execute [cmd] and when it opens a new window open a new tab page instead, e.g. :tab split opens current buffer in a new tabm; :tab help gt opens tab page with help for gt]
- taba [list all tab pages]
- [n] gt [goto next or tab [n]]
- gt [goto previous tab]
- undolist [list leafs in tree of change]
- earlier [n] [s/m/h] [goto older text state [n] times / sec / min / hours]
- g: [goto older text state]
- later [n] [s/m/h] [goto newer txt state [n] times / sec / min / hours]
- g: [goto newer text state]

Notes:

- [1] use 'x before a yank/paste/del command to use that register ('dipboard') (x=a,x,*) (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 d4)
- [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] z: scroll cursor to top, zb bottom, zz: center
- [6] gg: top of file (vim only), gf: open file under cursor (vim only)

Visual mode:
Move around and type operator to act on the selected region.

Vim-Help navigation:
CTRL + ALT GR +] or :a [tag] (jump to subject using tags, CTRL + O to jump back)

Figure 2: Vim visual cheat sheet for English keyboards

For German keyboards

vim graphical cheat sheet (german keyboard layout)

Esc normal mode

o begin of line

! external filter

'' register spec

\$ end of line

% inside (bracket)

& repeat

/ find

(begin sequence

) end sequence

= auto format

? find (rev)

' jump to mark

^ begin of line

1 1

2 2

3 3

4 4

5 5

6 6

7 7

8 8

9 9

0 0

**** \

Q ex mode

W next WORD

E end WORD

R replace mode

T back tab

Z quit

U undo line

I insert at bol

O open before

P paste before

U undo

***** search Forward

q play record macros

w next word

e end word

r replace char

t extra crnl

Z extra crnl

u undo

i insert mode

o open below

p paste after

ü undo

+ next line

A append at end

S subst line

D delete to end

F find char

G end of file

H screen top

J join lines

K back up keyword

L screen bottom

Ö ö

Ä ä

. goto mark

a append

s subst char

d delete

f find char

g extra crnl

h ←

j ↓

k ↑

l →

ö ö

ä ä

search backwards

> indent

Y yank line

X back space

C change to end

V visual mode

B prev WORD

N find prev

M screen middle

, repeat 8/7/9/9

. cmd_line

- begin of line

< un-indent

y yank

x delete char

c change

v visual mode

b prev word

n find next

m set mark

. never edit (8/7/9/9)

***** repeat command

_ prev line

Q commands with a dot need a char argument afterwards

Motion moves the cursor or defines the range for an operator

command direct action cmd, if red it enters insert mode

operator requires a motion afterwards, operates between cursor and destination

extra special functions, requires extra input

Main command line commands ("ex"):

- :w [file] [save]
- :q [quit]
- :q! [quit without saving]
- :wq [save & quit]
- :x [open file Foo]
- :n [new file]
- :sp [split window horizontally]
- :vsp [split window vertically]
- :reg [display content of named registers]
- :Explore [dir] [open file-explorer]
- :h [help]
- :h holy-grail [list all commands]

Other important commands:

- CTRL + R [redo]
- CTRL + P / N [complete the current word]
- CTRL + W [move cursor to next window]
- [n] CTRL + 6 [toggle [n]th alternate file]
- CTRL + F / B [page up / down]
- CTRL + E / Y [scroll lin exp / down]
- CTRL + V [block-visual mode]

Find and replace:

- %s<RegExp><String>/g [replace <RegExp> by <String> filewide]
- %s<RegExp><String>/ [search current line and replace first match]

Vim 7.x only commands:

- CTRL + X - CTRL + O [omni completion in insert mode]
- :tabe [file] [edit [file] in a new tab]
- :tabc [n] [close tab [n]]
- :tabonly [close all other tab pages]
- :tabmove [n] [move tab to position [n]]
- :tab [cmd] [execute [cmd] and when it opens a new window open a new tab page instead, e.g. :tab split opens current buffer in a new tab, :tab help :gt opens tab page with help for gt]
- :tabs [list all tab pages]
- [n] :gt [goto next or tab [n]]
- gt [goto previous tab]
- :undolist: [list leaves in tree of change]
- :earlier [n] [y/m/h] [goto older text state [n] times / sec / min / hours]
- g_ [goto older text state]
- :later [n] [y/m/h] [goto newer text state [n] times / sec / min / hours]
- g^ [goto newer text state]

Notes:

- [1] use 'x' before a yank/paste/del command to use that register ('dipboard') (x=a,x*) (e.g.: 'ygt' 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, 5t, d4)
- [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)

Visual mode: Move around and type operator to act on the selected region.

Vim-Help navigation: CTRL + ALT GR + or ta [tag] (jump to subject using tags, CTRL + O to jump back)

Figure 3: Vim visual cheat sheet for German keyboards

Tutorial

vi/vim graphical cheat sheet tutorial

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	= auto 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	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 cof 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/f/F	* 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/f/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
 commands with a dot need a char argument afterwards
 bol = beginning of line, eol = end of line, mk = mark, yank = copy
 words: `quux(foo, baz, baz)`
 WORDS: `quux (foo, baz, 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, 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)

For a graphical vi/vim tutorial & more tips, go to www.viemu.com - home of ViEmu, vi/vim emulation for Microsoft Visual Studio

Figure 4: Vi/vim graphical cheat sheet tutorial

Vim game

VIM Adventures

From here on

By now, you might have Vim is actually a word processor which you can easily modify and/or extend yourself.

In other applications



This work is licensed under a [Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-nc-sa/4.0/).

Other licensing available on request.

Unattended CSS typesetting with [Print with CSS Prince](https://princeps.org/).

This work is published at <https://hamwaves.com/vim.tutorial/en/>.

Last update: Monday, March 1, 2021.