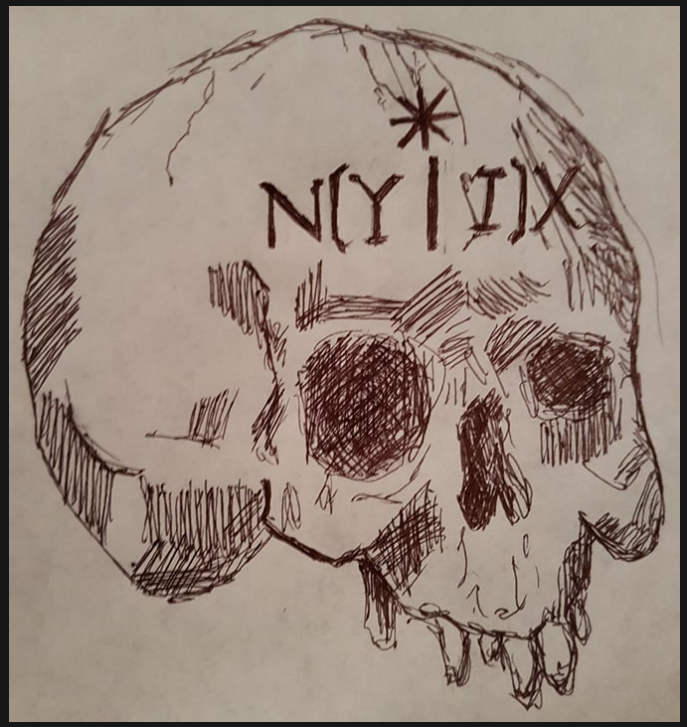
Learn Vim

~~Glorified Vimtutor~~

Ricardo Vela

martes 27 de febrero, 2018

Su. . . *servidor*



- ITC de cuarto semestre
- Pseudo Perl Monger (gotta **grepresent**)
- Comegalletas
- Definitivamente no una autoridad para dar esta plática
- (y hago prácticas en NIC México)

¿Qué saben de Vim?

- ¿Qué han escuchado de Vim?
- ¿Lo han usado?

Temas

Veremos..

- Instalación
- Cómo entrar
- Cómo salir
- Cómo salir
- Lidiar con la frustración de no poder salir
- Cómo salir (reprise)

Temas

- ¿Qué distingue a Vim? ¿Por qué Vim?
- Instalación
- Los modos
  - Normal
  - Insert
  - Command/Ex
  - Visual
- Cómo entrar y salir (en serio)
- Normal Mode
  - Movimientos básicos
  - Text Objects y la filosofía de Vim
- Insert Mode
  - Las reglas de Insert
  - CTRL
- Plugins y vimrc
- Más información
- Cheat Sheet

Propósito

Los propósitos de esta charla son

- Que *intimen* con sus teclados
- Que evalúen las herramientas que utilizan en su día a día.
- Si Vim fuese una espada, que pudieren cortar jamón con ella.

Esto es **independiente** al editor (o cualquier otra herramienta) que decidan utilizar.

¿Qué distingue a Vim? ¿Por qué Vim?

Hay dos cosas que distinguen a Vim entre editores

- Es un editor **modal**
  - Las teclas se comportan de manera diferente según el modo en el que uno esté.
- Sus comandos son **compuestos**
  - Tiene comandos que son **generales** y **contextuales**.
  - Editores como Emacs, Sublime, Atom, etc. tienen comandos que invocan funciones específicas.

Instalación

www.vim.org

- La versión del repositorio de Debian es la más actual. Recomiendo hacer **sudo apt install vim-gtk**.[1]
- Si lo prefieren, pueden instalar gVim, la versión gráfica de Vim.

[1] después de **sudo apt update**.

Antes de empezar..

Las diapositivas, su código, y algunos ejemplos que trabajaremos están en un repositorio de Github.

git clone https://github.com/aricav96/learnvim.git

Los modos

Hay muchos modos en Vim, pero nos interesan los 4 más básicos. Estos son:

Normal

Es el modo default. Aquí es donde ocurre buena parte de la magia (*la edición*). Se accede a él presionando la tecla ESC.

Insert

Es el modo en el que se escribe el texto. Hay muchas maneras de acceder a él: “A/a”, “I/i”, “c{movement}”, “S/s”, “O/o”...

Los modos

Command/Ex

Es el modo para ejecutar comandos y activar o desactivar variables del editor. Se accede a él tecleando ":" en Normal Mode. En realidad, Command (o Command-line mode) es Ex Mode ejecutado una sola vez. A Ex mode se accede tecleando Q en Normal Mode.

Visual

Es un modo en el que se navega como en Normal Mode, pero hay un indicador de texto elegido. Se accede a él mediante "v", "V" o CTRL-V en Normal Mode.

De estos cuatro, le pondremos particular atención a **Normal** y a **Insert**.

Cómo entrar y salir

Para entrar..

Basta correr **vim** <**filename**> o **vim** (sin archivo). En el segundo caso, se tiene que usar **:e** <**path**> para abrir el archivo deseado.

Para salir..

- En Normal Mode, hay que hacer **:q**
- . . . o **:wq**
  - Esto guarda (write) y cierra (quit)
- Sus equivalentes son (también en Normal Mode)
  - **ZQ**
  - **ZZ**
  - **ZZ** no se queja si no se han hecho cambios. . .

NORMAL MODE

¿Por qué creen que se llama Normal?

Movimientos básicos

- El movimiento en Vim se hace en Normal Mode.
- El movimiento más básico se hace con las teclas **h, j, k y l**
  - h = izquierda
  - j = abajo
  - k = arriba
  - l = derecha
- Los movimientos pueden ser prefijados con un número
  - E.g. 2j mueve el cursor dos líneas hacia abajo.

Text Objects y la filosofía de Vim

Vim reconoce ciertos “objetos”, como palabras, oraciones, párrafos, bloques de código delimitados (por paréntesis, llaves o corchetes) y tags. Estos también tienen sus teclas asignadas en Normal Mode.

- w/W, b/B, e/E para palabras (¿qué constituye una palabra?).
- () para oraciones.
- {} para párrafos.
- % para delimitadores de bloques.

Asimismo, Vim se puede mover a partes del archivo por líneas

- 0/$ para el comienzo/final de la línea.
- ^/_ para el primer caracter de la línea.
- g_ para el último caracter de la línea.
- gg/G para ir a una línea específica (e.g. 2G te deja en la línea 2).

Más movimientos

Algunos movimientos útiles son

- CTRL-B CTRL-U CTRL-Y scroll up
- CTRL-F CTRL-D CTRL-E scroll down
- CTRL-O CTRL-I saltar frenéticamente por el jumplist

Hay otros movimientos que son útiles cuando queremos preservar la posición del cursor

- zz posiciona la línea en el centro de la pantalla
- zt posiciona la línea hasta arriba
- zb posiciona la línea hasta abajo

Búsquedas

En casos más heterodoxos, podemos emplear búsquedas con:

- f/F busca un caracter hacia adelante/hacia atrás
- t/T hace lo mismo, pero mueve el cursor un espacio antes ('til)
- "/" nos permite escribir un regex para buscar

Marcas

En casos aún más heterodoxos, podemos emplear marcas

- m{a-zA-Z} crea una marca en el lugar donde está el cursor
- '{a-zA-Z} navega a dicha marca (principio de línea)
- `{a-zA-Z} navega a dicha (exactamente a donde está la marca).

Verbo + Objeto

Así como podemos prefijar movimientos, podemos prefijar comandos con movimientos.

- cw = change word
- c} = change paragraph
- dw = delete word
- d) = delete sentence
- yw = yank word

Otros comandos ya incluyen una mezcla de movimiento e inserción

- s = substitute
- r = replace
- a = append
- A = append EOL
- S = substitute line ...

Me tomaré la libertad de poner un **gran recordatorio** aquí para parar y demostrar las demás.

Modificadores especiales

dw vs daw vs diw

- dw borra la palabra **desde el cursor hasta el inicio de la siguiente palabra**
- daw borra la palabra **sin importar la posición del cursor**
- diw borra la palabra **sin importar la posición del cursor hasta antes de la siguiente palabra**

Vim Golf

¿dbx, bdw o daw?

Si las tres combinaciones hacen lo mismo y requieren teclear tres veces, ¿cuál es la mejor?

- **daw** porque es **un cambio**

INSERT MODE

Las reglas de Insert

En **Insert Mode** las teclas actúan como siempre. Es el modo donde se puede escribir como lo sabemos (y amamos) hacer.

Esto tiene una implicación importante respecto a teclas modificadoras: Las combinaciones de CTRL también son únicas dependiendo del modo en el que nos encontremos.

CTRL

- **CTRL-a** inserta texto recientemente añadido. Es el dot command de Insert Mode.
- **CTRL-r** + **registro** nos permite poner texto en el **registro** referido.
- **CTRL-n y CTRL-p** nos permiten autocompletar según lo que hay en el documento.
- **CTRL-x** nos pone en un modo especial para autocompletar

Vim tiene definido en dónde va a buscar (set complete?). Para indicarle que busque en archivos de un mismo proyecto, es preciso utilizar **ctags**.

Plugins y vimrc

- Vanilla(???) Vim es bonito, pero hay algunos plugins que valen la pena.
- Tim Pope es nuestro pastor, nada nos faltará.

Plugins y vimrc

Algunos de los plugins más célebres de Tpope son

- vim-surround
- vim-commentary
- vim-sensible

¿Dónde van?

Sin un administrador, hay que descargarlos, ponerlos en .vim/plugin. Es importante que tengan permiso de ejecución.

Más información

Hay muchas fuentes para consultar sobre Vim:

Documentación

- vimtutor
- :help
- man pages

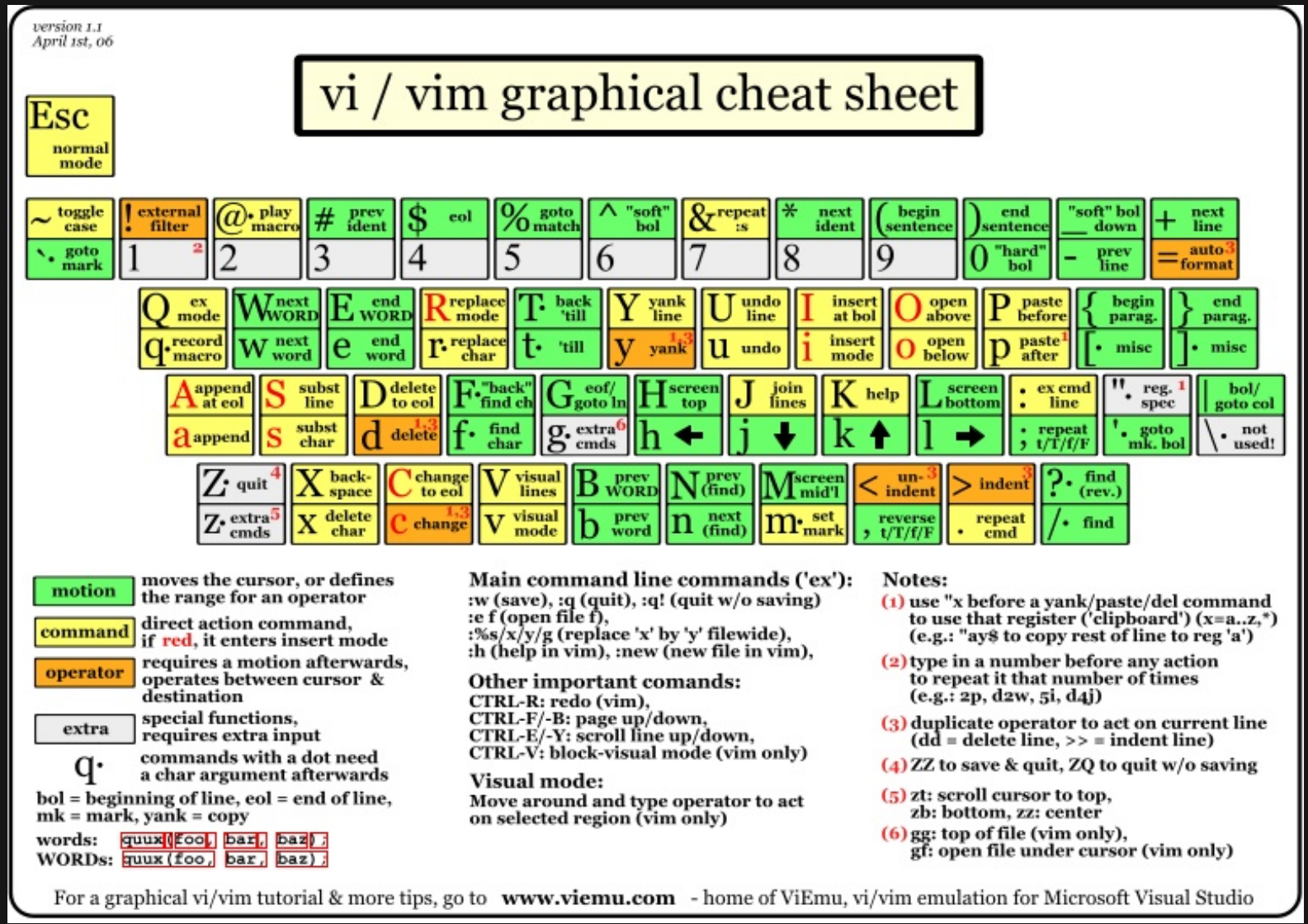
Libros

- “Practical Vim” de Drew Neil
- “The Vim Tutorial and Reference” Steve Oualline
- “Seven habits of effective text editing” de Bram Moolenaar

¿Dudas?

(si no, party tricks maybe?)

Cheat Sheet


version 1.1
April 1st, 06
vi / vim graphical cheat sheet
Esc
normal mode
~ toggle case
` goto mark
! external filter
1
@ play macro
2
prev ident
3
$ eol
4
% goto match
5
^ "soft" bol
6
& repeat :s
7
* next ident
8
(begin sentence
9
) end sentence
0 "hard" bol
"soft" bol
_ down
- prev line
+ next line
= auto format
Q ex mode
q record macro
W next WORD
w next word
E end WORD
e end word
R replace mode
r replace char
T back 'till
t 'till
Y yank line
y yank
U undo line
u undo
I insert at bol
i insert mode
O open above
o open below
P paste before
p paste after
{ begin parag.
[misc
} end parag.
] misc
A append at eol
a append
S subst line
s subst char
D delete to eol
d delete
F "back" find ch
f find char
G eof/ goto ln
g extra cmds
H screen top
h
J join lines
j
K help
k
L screen bottom
l
: ex cmd line
; repeat t/T/f/F
" reg. spec
' goto mk. bol
\ not used!
bol/ goto col
Z quit
z extra cmds
X back-space
x delete char
C change to eol
c change
V visual lines
v visual mode
B prev WORD
b prev word
N prev (find)
n next (find)
M screen mid'l
m set mark
< un-indent
, reverse t/T/f/F
> indent
. repeat cmd
? find (rev.)
/ 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 comands:
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)
For a graphical vi/vim tutorial & more tips, go to www.viemu.com - home of ViEmu, vi/vim emulation for Microsoft Visual Studio