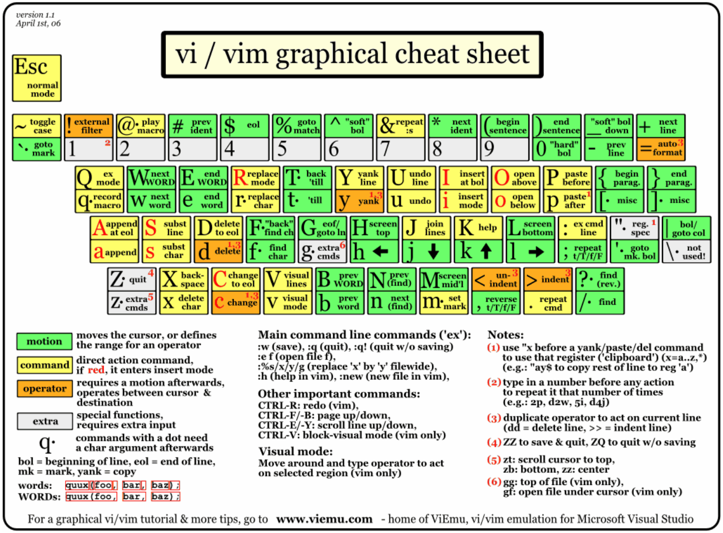
Text Surgery with vi

Nicholas C. Spinale

DevX
Carleton College

May 14th, 2014

Introduction



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)

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

My Plan

Outline

History

Evolution of line editors

- 1971: ed at AT&T
- 1976: ex 0.1 by Bill Joy
- 1979: ex 2.0/vi

Outline

The vi Utility

```
vi [-rR] [-c command] [-t tagstring] [-w size]
[file...]
```

Outline

wut is editor

wut is editor

At the very least, it allows a user to

wut is editor

At the very least, it allows a user to

- Load state of system into internal state

wut is editor

At the very least, it allows a user to

- Load state of system into internal state
- Modify internal state

wut is editor

At the very least, it allows a user to

- Load state of system into internal state
- Modify internal state
    - Buffers

wut is editor

At the very least, it allows a user to

- Load state of system into internal state
- Modify internal state
    - Buffers
        - Contents
        - Position of cursor
        - Positions of other markers

wut is editor

At the very least, it allows a user to

- Load state of system into internal state
- Modify internal state
  - Buffers
    - Contents
    - Position of cursor
    - Positions of other markers
  - Registers

wut is editor

At the very least, it allows a user to

- Load state of system into internal state
- Modify internal state
    - Buffers
        - Contents
        - Position of cursor
        - Positions of other markers
    - Registers
    - Layout of buffers

wut is editor

At the very least, it allows a user to

- Load state of system into internal state
- Modify internal state
    - Buffers
        - Contents
        - Position of cursor
        - Positions of other markers
    - Registers
    - Layout of buffers
- Apply changes in internal state to system

A Clever Approach

Most editors: Key combinations

A Clever Approach

Most editors: Key combinations

vi: Modal editing/key sequences

Outline

Ex Commands

```
:______<cr>
```

- `w[rite]`
- `q[uit]`
- `wq`
- `e[dit]`
- ...

("`!`" means what you might expect)

Nouns

By themselves, they are considered 'motions'

Left/down/up/right	`h/j/k/l`
Beginning of next word/Word	`w/W`
Beginning of last word/Word	`b/B`
Next/last occurence of `x`	`f/Fx`
Beginning/end of line	`^/$`
Next/last occurence of current word	`#/*`

Numbers are adjectives

Intransitive Verbs

Standalone actions, some of which ask for strings (which are ended with `<esc>`)

ight	`h/j/k/l`
Beginning of next word/Word	`w/W`
Beginning of last word/Word	`b/B`
Next/last occurence of `x`	`f/Fx`
Beginning/end of line	`^/$`
Next/last occurence of current word	`#/*`

Numbers are adverbs too...

Visual Mode

The Rest

Scrolling

`zz/zq`

`.`

`u`

Outline

Registers

Named variables that store strings

Cut/copy to them
Record them

Paste from them
Play them

The Art of Macros

Stay abstract.

The Art of Macros

Stay abstract. Practice.

The Art of Macros

Stay abstract. Practice. That is all.

vi Golf

```
i1<esc>qqyyp<c-a>q98@qqqcc
Buzz<esc>5-q19@q2-qqciwFizz<esc>3+0q32@q
```

Outline

vi Improved

Ubiquitous

Compatability mode

Improvements

- Aesthetics
- Much more custmizability (including aesthetics...)
- GUI mode

VimScript

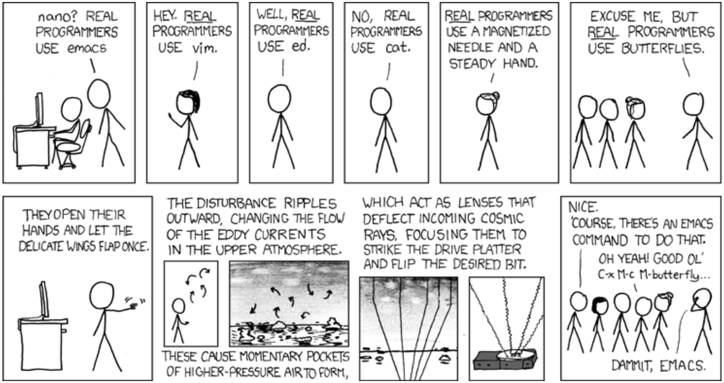
There are TONS of plugins

Examples

- Commentary
- Tablizarize
- Nerdtree
- Ctrl-P

Outline

Emacs


nano? REAL PROGRAMMERS USE emacs
HEY. REAL PROGRAMMERS USE vim.
WELL, REAL PROGRAMMERS USE ed.
NO, REAL PROGRAMMERS USE cat.
REAL PROGRAMMERS USE A MAGNETIZED NEEDLE AND A STEADY HAND.
EXCUSE ME, BUT REAL PROGRAMMERS USE BUTTERFLIES.
THEY OPEN THEIR HANDS AND LET THE DELICATE WINGS FLAP ONCE.
THE DISTURBANCE RIPPLES OUTWARD, CHANGING THE FLOW OF THE EDDY CURRENTS IN THE UPPER ATMOSPHERE.
THESE CAUSE MOMENTARY POCKETS OF HIGHER-PRESSURE AIR TO FORM,
WHICH ACT AS LENSES THAT DEFLECT INCOMING COSMIC RAYS, FOCUSING THEM TO STRIKE THE DRIVE PLATTER AND FLIP THE DESIRED BIT.
NICE.
'COURSE, THERE'S AN EMACS COMMAND TO DO THAT.
OH YEAH! GOOD OL' C-x M-c M-butterfly...
DAMMIT, EMACS.

IDE's

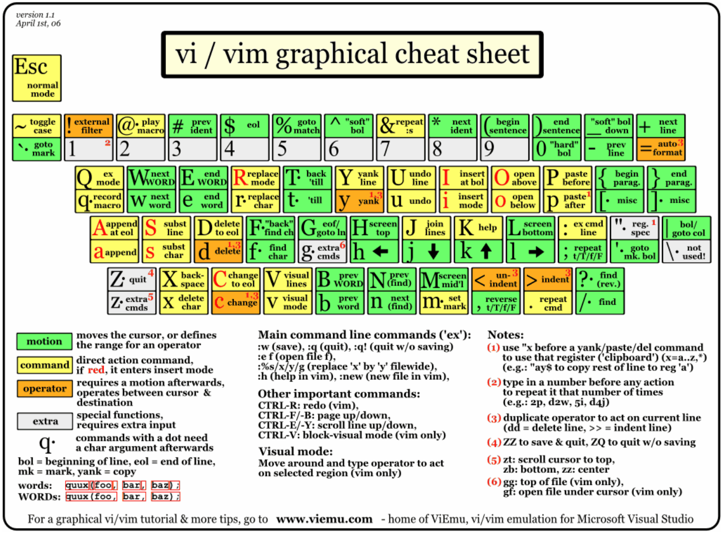
Pro: stay in terminal (seamlessly integrates with tmux, screen, etc)

IDE's

Pro: stay in terminal (seamlessly integrates with tmux, screen, etc)

Con: learning curve

Conclusion



version 1.1
April 1st, 06

vi / vim graphical cheat sheet

Esc normal mode

~ toggle case · ` goto mark
! external filter · 1 [2]
@ 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 [3]

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 [1,3]
U undo line · u undo
I insert at bol · i insert mode
O open above · o open below
P paste before · p paste after [1]
{ begin parag. · [· misc
} end parag. ·]· misc

A append at eol · a append
S subst line · s subst char
D delete to eol · d delete [1,3]
F· "back" find ch · f· find char
G eof/ goto ln · g· extra cmds [6]
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 [1] · '· goto mk. bol
| bol/ goto col · \· not used!

Z· quit [4] · z· extra cmds [5]
X back-space · x delete char
C change to eol · c change [1,3]
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 [3] · , reverse t/T/f/F
> indent [3] · . 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 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)

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

Further Learning

How to continue learning

- Cheatsheets (a few commands at a time)
- VimDoc
- VimCasts

Further Learning

How to continue learning

- Cheatsheets (a few commands at a time)
- VimDoc
- VimCasts
- Me

Further Learning

How to continue learning

- Cheatsheets (a few commands at a time)
- VimDoc
- VimCasts
- Me (I love talking about vim)

Questions?

?