

# Text Surgery with vi

Nicholas C. Spinale

DevX  
Carleton College

May 14<sup>th</sup>, 2014



# Introduction

version 1.1  
April 1st, 06

**Esc**  
normal mode

## vi / vim graphical cheat sheet

~ toggle case	! external filter	@ play macro	# prev ident	\$ eol	% goto match	^ "soft" bol	& repeat is	* 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 <sup>1</sup> after	* misc	.	misc
A append at eol	S subst line	D delete to eol	F "back" find ch	G eol/ goto lin	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 cmds <sup>6</sup>	h	j	k	l	. repeat	' goto mk. bol	\ not used!	
Z quit <sup>4</sup>	X back-space	C change to eol	V visual lines	B prev word	N prev (find)	M screen midT	< un-indent	> indent	? find (rev.)			
Z extra <sup>5</sup> cmds	x delete char	c change	V visual mode	b prev word	n next (find)	m set mark	, reverse	. repeat cmd	/ find			

<b>motion</b>	moves the cursor, or defines the range for an operator
<b>command</b>	direct action command, if <b>red</b> , it enters insert mode
<b>operator</b>	requires a motion afterwards, operates between cursor & destination
<b>extra</b>	special functions, requires extra input

q.

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,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](http://www.viemu.com) - home of ViEmu, vi/vim emulation for Microsoft Visual Studio



# My Plan

- 1 Background
  - History
  - The vi Utility
- 2 How to vi
  - General Principles
  - Command Structure
  - Registers
- 3 Modern vi
  - Vim
  - Comparison to Other Editors



# Outline

- 1 Background
  - History
  - The vi Utility
- 2 How to vi
  - General Principles
  - Command Structure
  - Registers
- 3 Modern vi
  - Vim
  - Comparison to Other Editors



# History

## Evolution of line editors

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



# Outline

- 1 Background
  - History
  - The vi Utility
- 2 How to vi
  - General Principles
  - Command Structure
  - Registers
- 3 Modern vi
  - Vim
  - Comparison to Other Editors



# The vi Utility

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



# Outline

## 1 Background

- History
- The vi Utility

## 2 How to vi

- General Principles
- Command Structure
- Registers

## 3 Modern vi

- Vim
- Comparison to Other Editors





# 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

1

## Background

- History
- The vi Utility

2

## How to vi

- General Principles
- **Command Structure**
- Registers

3

## Modern vi

- Vim
- Comparison to Other Editors



# Command Structure

version 1.1  
April 1st, 06

Esc  
normal mode

## vi / vim graphical cheat sheet

~ toggle case	! external filter	@ play macro	# prev ident	\$ eol	% goto match	^ "soft" bol	& repeat is	* 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 <sup>1</sup> after	* misc	.	misc
A append at eol	S subst line	D delete to eol	F "back" find ch	G eol/ 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 cmds <sup>6</sup>	h	j	k	l	* repeat	' goto mk. bol	\ not used!	
Z quit <sup>4</sup>	X back-space	C change to eol	V visual lines	B prev word	N prev (find)	M screen mid	< un-indent	> indent	? find (rev.)			
	x delete char	c change	V visual mode	b prev word	n next (find)	m set mark	reverse	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.

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](http://www.viemu.com) - home of ViEmu, vi/vim emulation for Microsoft Visual Studio



# Outline

## 1 Background

- History
- The vi Utility

## 2 How to vi

- General Principles
- Command Structure
- Registers

## 3 Modern vi

- Vim
- Comparison to Other Editors



# 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

## 1 Background

- History
- The vi Utility

## 2 How to vi

- General Principles
- Command Structure
- Registers

## 3 Modern vi

- Vim
- Comparison to Other Editors



# vi Improved

Ubiquitous

Compatibility mode

Improvements

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



# VimScript

There are TONS of plugins

## Examples

- Commentary
- Tablizarize
- Nerdtree
- Ctrl-P



# Outline

## 1 Background

- History
- The vi Utility

## 2 How to vi

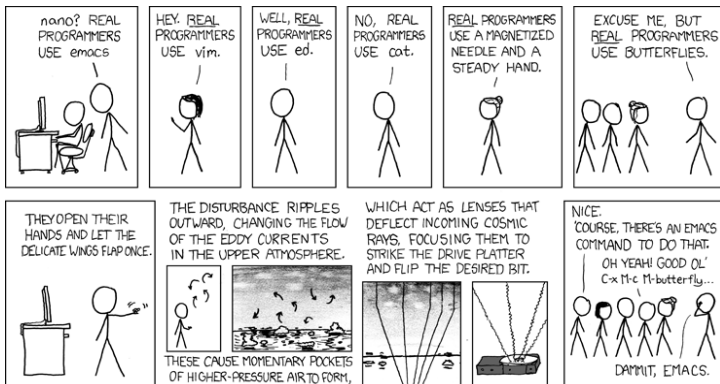
- General Principles
- Command Structure
- Registers

## 3 Modern vi

- Vim
- Comparison to Other Editors



# 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

**Esc**  
normal mode

## vi / vim graphical cheat sheet

~ toggle case	! external filter	@ play macro	# prev ident	\$ eol	% goto match	^ "soft" bol	& repeat is	* 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 <sup>1</sup> after	* misc	.	misc
A append at eol	S subst line	D delete to eol	F "back" find ch	G eol/ 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 cmds <sup>6</sup>	h	j	k	l	* repeat	' goto mk. bol	\ not used!	
Z quit <sup>4</sup>	X back-space	C change to eol	V visual lines	B prev word	N prev (find)	M screen midT	< un-indent	> indent	? find (rev.)			
Z extra <sup>5</sup> cmds	x delete char	c change	v visual mode	b prev word	n next (find)	m set mark	reverse	repeat cmd	/ find			

<b>motion</b>	moves the cursor, or defines the range for an operator
<b>command</b>	direct action command, if <b>red</b> , it enters insert mode
<b>operator</b>	requires a motion afterwards, operates between cursor & destination
<b>extra</b>	special functions, requires extra input

q.

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/s/y/g (replace 's' 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](http://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?

?

