lynda.com

Up and Running with vi

with David D. Levine

Entering and leaving vi

- $ **vi** ***file*** edits existing or new file
- *[Shift]* + **ZZ** writes file and quits (or **:wq**[Return])
- **:q!**[Return] quits without saving changes

File management

:w	Write without quitting	**:q**	Quit without writing
:q!	Abandon changes	**:vi**	Edit another file
:n	Go to next file	**:N**	Go to previous file
:rew	Rewind to first file	**:r**	Read file into this one

- Use *[Control]* + **g** to display line number and file status

Basic cursor movement

h	Left one character
j	Down one line
k	Up one line
l	Right one character (lowercase L)

More ways of moving the cursor

- Arrow keys
- *[Space]* and *[Backspace]* for forward and backward
- *[Return]* to move to beginning of next line
- **–** (minus) to move to beginning of previous line
- Can use a number before any movement command

Moving by textual units

w	Forward a word	**b**	Backward a word
e	Forward to end of current or next word		
)	Forward a sentence	**(**	Backward a sentence
}	Forward a paragraph	**{**	Backward a paragraph

- Can use a number before any movement command

Moving by lines

^	Beginning of line	**$**	End of line
1G	First line of file	**G**	Last line of file
*n***G**	Line *n* of file	**%**	Matching paren/brace

- Use *[Control]* + **g** to display line number and file status
- Can use a number before any movement command

Summary of movement commands

h,l	Characters	**j,k**	Lines
w, b, e	Words	**^, $**	Begin/end of line
(,)	Sentences	**{, }**	Paragraphs

- Can use a number before any movement command
- Can be used as modifiers for some other commands

Scrolling

- *[Control]* + **e** Scroll down one line ("expose")
- *[Control]* + **y** Scroll up one line
- *[Control]* + **d** Scroll down half a screen
- *[Control]* + **u** Scroll up half a screen
- *[Control]* + **f** Scroll down one screen ("forward")
- *[Control]* + **b** Scroll up one screen ("back")

Inserting text

i	Insert before cursor	**I**	Insert at beginning of line
a	Append after cursor	**A**	Append at end of line
o	Open a new line below	**O**	Open a new line above

- All of these commands enter insert mode
- Use *[Esc]* to return to command mode

Deleting text

x	Current character	**dd**	Current line
dw	Current word	**de**	To end of word
d^	To beginning of line	**d$**	To end of line (also **D**)

- Can use **d***x*, where *x* is any movement command
- Can use a number before any deletion command

Changing text

r	Current character *	**s**	Current character
cc	Current line	**cw**	Current word (also **ce**)
c^	To beginning of line	**c$**	To end of line (also **C**)

- * **r** leaves you in command mode. Others: insert mode
- As with delete, **c** can use number or any movement cmd

Miscellaneous editing commands

- **R** enters overwrite mode (*[Esc]* to exit)
- **~** changes the case of the character at the cursor
- **J** joins the next line to the current line
- Can use a number before any of these

Undo, redo, and repeat

- **u** undoes last change (repeat to undo more)
- *[Control]* + **r** redoes last change (undoes undo)
- **U** undoes all changes to current line
- **.** (period) repeats last change
- Multiple undo and redo are **Vim** features

Delete, yank, and put

dd	Delete (cut) line	**yy**	Yank (copy) line
d*x*	Delete an amount	**y***x*	Yank an amount
p	Put after cursor	**P**	Put before cursor

- As with delete, **y** can use number or any movement cmd
- Deletes and yanks go into vi's "buffer," not OS clipboard

Searching

- **/***text* searches forward for *text*
- **?***text* searches backward for *text*
- **n** repeats previous search
- **N** repeats previous search in opposite direction

Regular expressions

- **.** matches any single character
- \ removes special meaning
- **^** and **$** match line starts and ends
- **[ ]** matches any character in set
- **\\(\\)** repeats multiple items

Search and replace

- **:s/***old/new***/** replaces first *old* with *new* on current line
- **:s/**old/new**/g** replaces every old with new on current line
- **:%s/***old/new***/** replaces first old with new on every line
- **:%s/**old/new**/g** replaces every old with new on every line

More about search and replace

- / can actually be any character
- *old* is a regular expression
- *new* is generally a string, except…
- **&** is replaced with the found text
- \n is replaced with the text found for the *n*th \\(…\\)

Indenting, auto-indent, and word wrap

- **>>** indents current line
- **<<** outdents current line
- As with delete, can use number or movement command
- **:se ai** enables auto-indent **:se noai** disables it
- **:se wm=8** enables wrap margin **:se wm=0** disables it

Filtering through shell commands

!!	filters current line through shell command		
*n***!!**	filters *n* lines	**!%**	filters to matching paren
!}	filters next paragraph	**!{**	filters previous paragraph

- Useful commands include **fmt**, **tr**, **grep**, **sed**, and **awk**

Using line ranges with colon commands

- : *line* [, *line*] *cmd* performs *cmd* on *line*(s) specified as:

n	Line number *n*	**.**	Current line
$	Last line of file	**'x**	Mark x (set with **m**x)
/re/	Regular expression	**%**	All lines (same as **1,$**)

- *cmd* includes **s**, **d**, **y**, **c**, **!**, **>**, and others