"Prof" Joe's Tutorial on

The Basics of the vi editor

Joseph Phillips
2013 October 21

What is vi?

- A **primitive** but **powerful** text editor for Unix-style operating systems
  - **Mode based**
    - Move around mode
    - Edit mode
    - Command mode
    - Search mode
  - Pronounced by saying the letters in English: "v-i"
  - "vim" (said like a word): vi-compatible Linux editor
- *Other editors* on Unix/Linux:
  - emacs
  - gedit
  - nano

Why use vi?

- **CON**:
  - Primitive
  - Have to learn funky commands
- **PRO**:
  - Present on practically all Unix systems
  - Powerful
- *"Well, which editor(s) do you use, 'Prof' Joe?"*
  - Extensive editing: ***emacs***
  - Quick-n-dirty edits: ***vi***

Starting vi

- Just type "*vi*"

  - You start out in "moving around" mode (but right now there is no where to go)

Our first text (1)

- Now press lowercase "*i*" to enter "edit mode"

- Now type "*Hello*"

Our first text (2)

- Now press "*Esc*" to get back to "moving around" mode

- Now press "*:*" to enter command mode, then enter "*w hello.txt*" to save your file.

- You are returned to moving-around mode
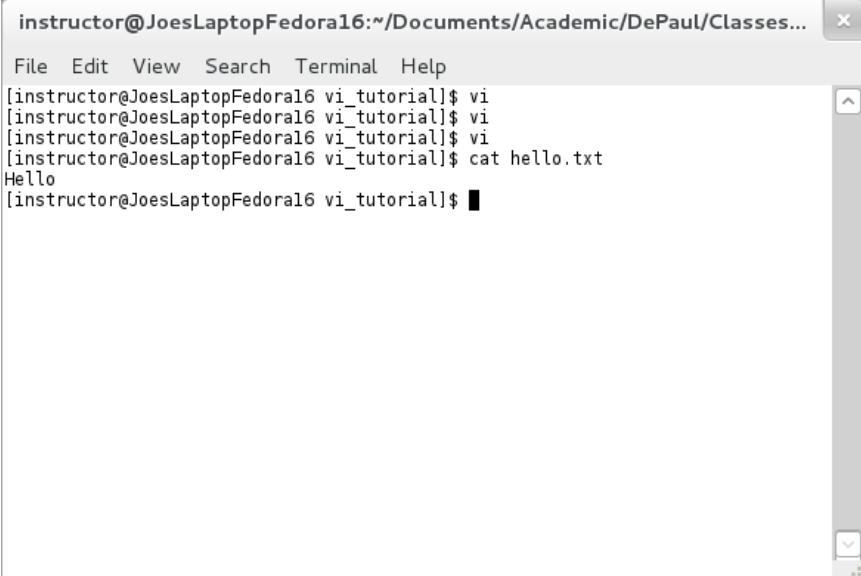
instructor@JoesLaptopFedora16:~/Documents/Academic/DePaul/Classes...

File   Edit   View   Search   Terminal   Help

Hello

instructor@JoesLaptopFedora16:~/Documents/Academic/DePaul/Classes...

File   Edit   View   Search   Terminal   Help

Hello

"hello.txt" [New] 1L, 6C written

Our first text (3)

- To quit press "**:**" to enter command mode, then enter "*q*"

- To see our file from the command line, type "*cat hello.txt*"

General themes of vi commands

- **Abbreviation-based**:
  - `c` = change, `i` = insert, `r` = replace, `a` = append, *etc*.

- **Little letters** do one thing, **big letters** do something similar but different
  - `little-w`: advance one word (punctuation)
  - `Big-W`: advance one word (whitespace)
  - `little-i`: insert where you are
  - `Big-I`: insert at beginning of line

- Misc:
  - `0` (the digit): the beginning (first of the Natural Numbers)
  - `$`: the end (end-of-string marker in some languages like Perl)

How do all these modes work?

Command mode

automatically returned

i,a,o,c,
I,A,O,C,R

:

Moving-around mode

Editing mode

/,?

<Esc>

automatically returned

Search mode
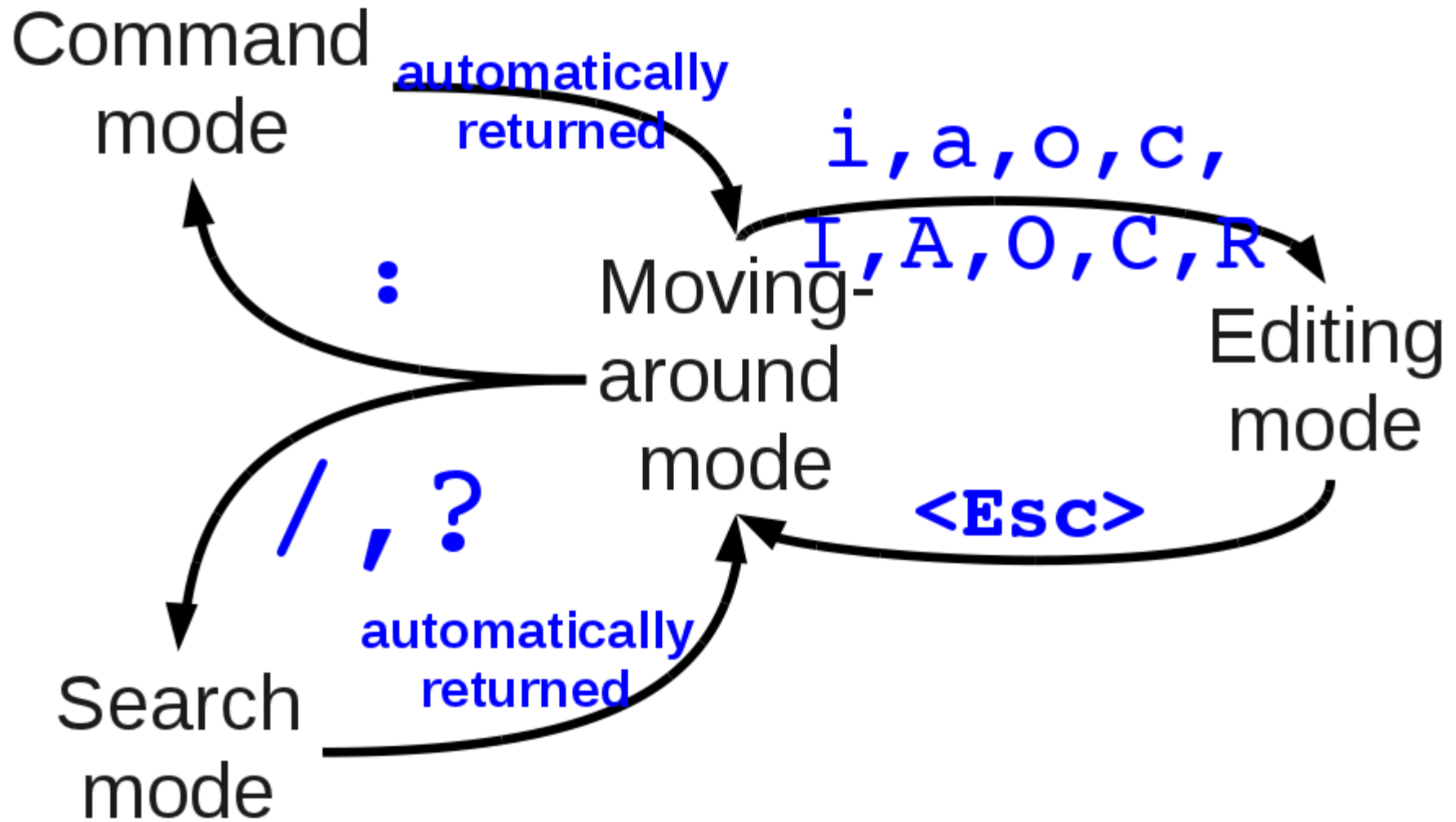
Moving-around mode: adjacent chars

- Of course, the keys let you move **left**, **down**, **up**, **right**.

- So do the keys **h** (left), **j** (down), **k** (up), **l** (right).

```
A            B           C

         up-arrow
             ↑
D ←  left    E  right → F
             ↓
        down-arrow

G            H           I


A            B           C

             k
             ↑
D  ←  h      E     l  → F
             ↓
             j

G            H           I
```
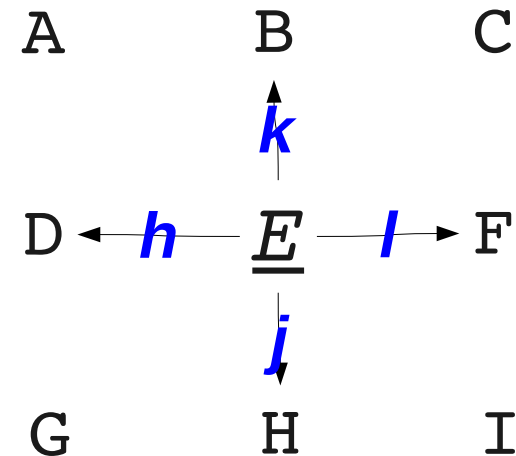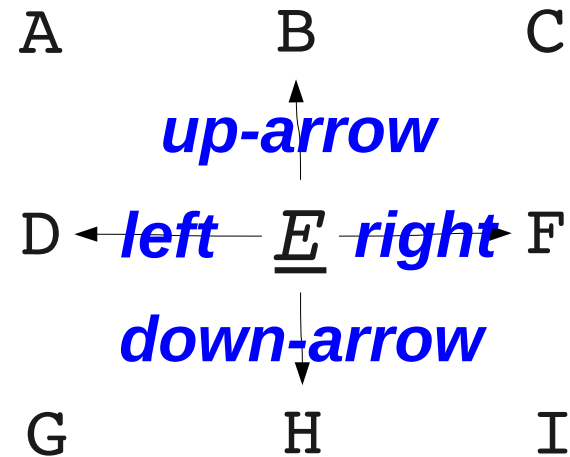
Moving-around mode: Page up, page down

- ***Page–Up*** works

- So does ***Ctrl–b*** ("back")

- ***Page–Dn*** works

- So does ***Ctrl–f*** ("forward")

Previous screen

page up
Ctrl-b

Current screen

page down
Ctrl-f

Next screen
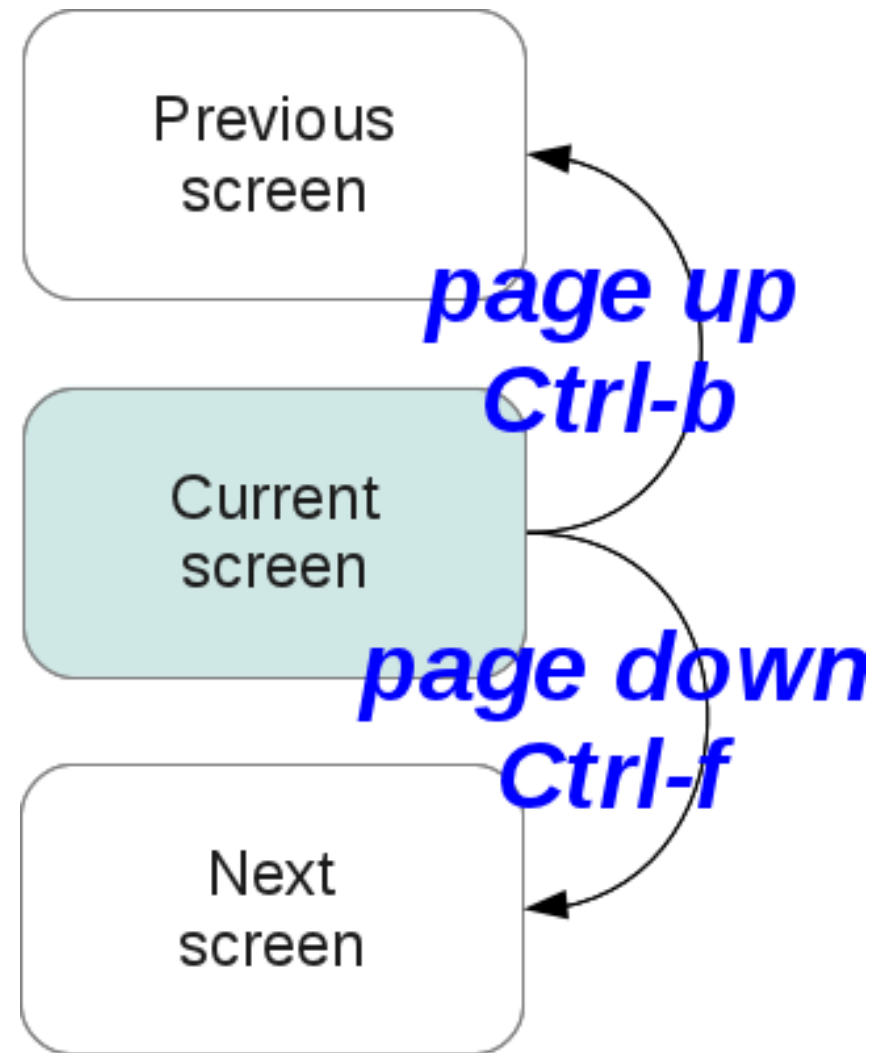
Moving-around mode: Next word, previous word

- ***little-w***: move to next word (punctuation)

- ***Big-W***: move to next word (whitespace)

- ***little-b***: move ***back*** to prev word (punctuation)

- ***Big-B***: move ***back*** to prev word (whitespace)
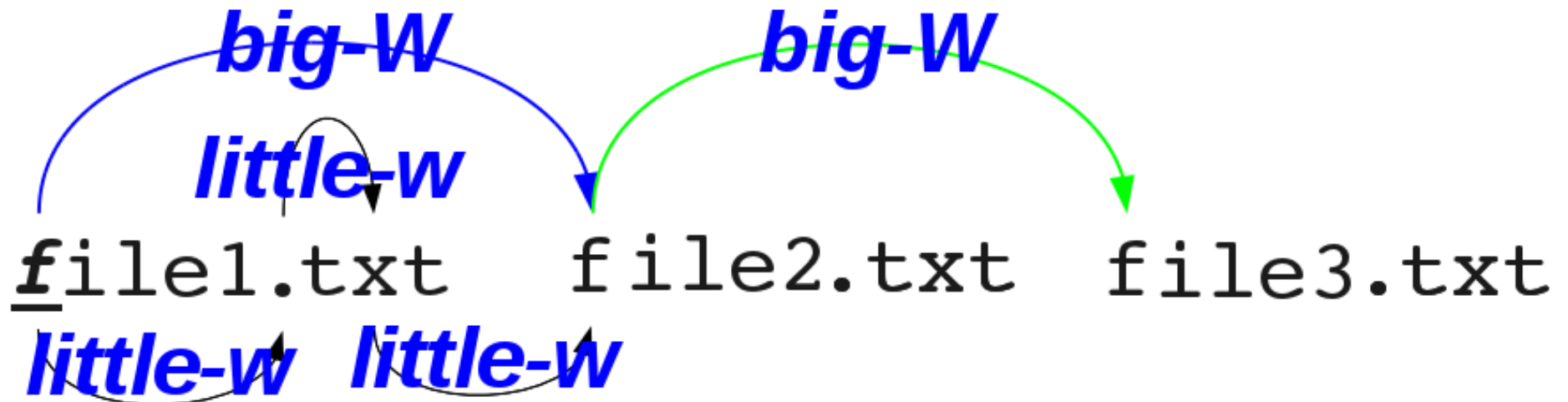
Moving-around mode:
First char of line, last char of line

- *0*: (the digit) takes you to the beginning of the line
  - "0" as in "the first of the Natural numbers"

- *$*: takes you to the end
  - "$" sometimes marks the end of strings (*e.g.* Perl)

0

a   b   c   **A**   B   C

$
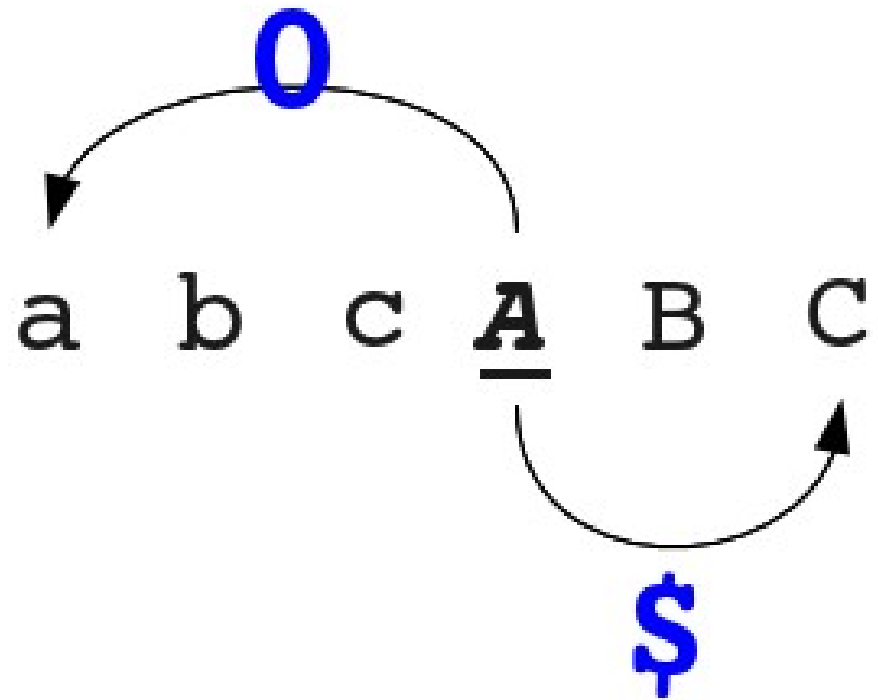
Moving around:
General

- *:<number><enter>*: Go to line numbered *<number>*

- *:0<enter>*: Go to first line

- *:$<enter>*: Go to last line

- *[[*: Go back to previous top-level function.

- *]]*: Go forward to next top-level function.

Moving-around mode: Deleting

- **`dl`** ("delete-letter"): deletes current letter
  - **`x`**: also deletes current letter
- **`dw`**, **`dW`**: delete rest of word from cursor
- **`dd`**: delete current line
  - **`d0`**: on current line delete from beginning to cursor
  - **`d$`**: on current line delete from cursor to end
- **`u`** ("undo"): when back in move-around mode, undo previous delete

Moving-around mode:
Replacing and Joining:

- `r`: **Replace** the current char (no need for `Esc`)

- `R`: **Replace** chars starting from the current until press `Esc`.

- `J`: **Join** this line with the next by placing the next line at the end of the current one.

# Moving-around mode: Changing

- `cl`: change (i.e. delete) current letter, stay in insert mode until press *Esc*.

- `cc`: change (i.e. delete and retype) current line until press *Esc*.

- `c0`: change (i.e. delete) current line from beginning to cursor, stay in insert mode until press *Esc*.

- `c$`: change (i.e. delete) current line from beginning to cursor, stay in insert mode until press *Esc*.

Editing mode, entering

- ***little-i***: insert where you are
- ***little-a*** (append): insert after cursor
- ***little-o***: create new line after and insert there

- ***Big-I***: insert at beginning of line
- ***Big-A***: append at end of line
- ***Big-O***: create new line before and insert there

O →

I   **A**

a b c **A** B C

o →

↑   ↑

i a
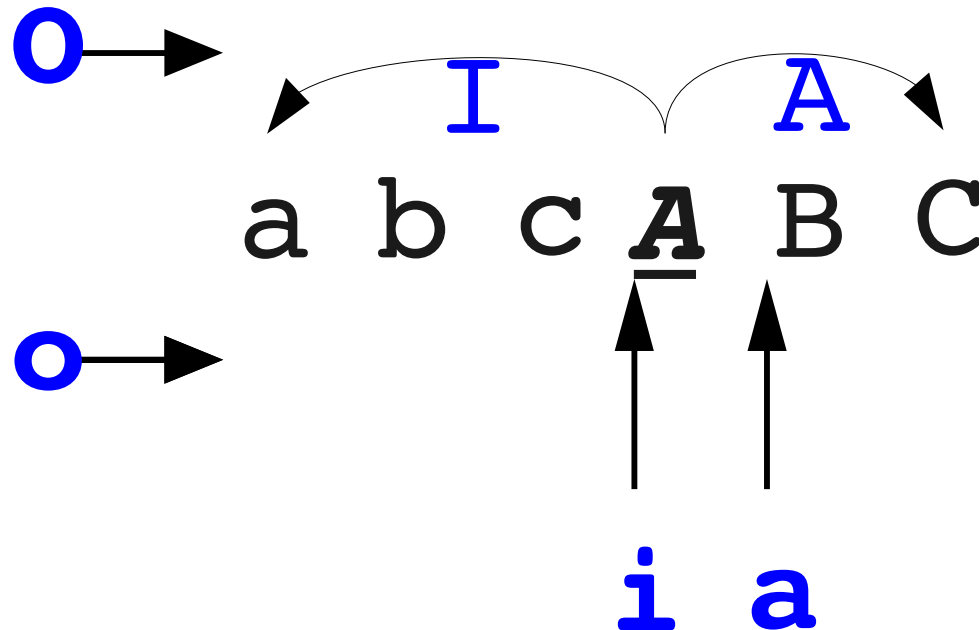
Editing mode, inside

- Whatever you type is entered into document
- *Backspace* and *Delete* keys remove text
- (Cursor was at **A**, typed "*i*" then "*123*")

a b c *1* *2* *3* <u>A</u> B C

Editing mode, leaving

- **_Esc_**: leaves editing mode
  - Goes back to moving-around mode
- **_u_** ("undo"): Once back in moving-around mode, undo previous editing

Cutting and pasting

- *yy*: "yanks" current line into buffer
- *p* (little): "pastes" it at line below
- *P* (big): "pastes" it at line above

Cool stuff

- *u*: Undo (you know this already)

- *.* (period): Do the most recent insertion or deletion again.

- *<number><command>*: Do *<command>* *<number>* times:
  - *4w*: "Go forward 4 words"
  - *4yy*: "Yank 4 lines into buffer"
  - *6x*: "Delete the next 6 characters"

- *Ctrl-G*: Tells current filename, line number and total number of lines

Search:

- Basic search:
  - */*: Brings you to bottom line (command line) and lets you enter text for which to search (forward)
  - *?*: Same thing, but searches backwards
  - After the search you're back in move-around mode
- Do it again!
  - *n*: in move-around mode, search for the next match
  - *N*: in move-around mode, go back to the previous match
- Misc.
  - Pressed */* or *?* accidentally? Just press backspace

Commands

- Except for "**zz**", all commands begin with "**:**"

Command	Description
:e \<filename\>	Stop editing current file and start editing file \<filename\>
:n	Get the next file to edit on the command line
:r \<filename\>	Insert file \<filename\> into current file
:w	Save ("write") current file
:w \<filename\>	Save ("write") current file as \<filename\>
:w!	"***I don't care that it is write-protected, write it!***"
:q	Exit ("quit") the editor
:q!	"***I don't care that I didn't save, quit!***"
:wq or **zz**	Save and then immediately quit

"*But wait, there's more!*"

- More sophisticated commands to:
  - Access multiple yank-buffers
  - Write specific lines only
- Try the following tutorial: "`vimtutor`"
- Type "`vi commands`" into your favorite web browser!