The vi editor ("vee eye")

NOTE: You will not be examined on the detailed usage of vi, but you should know the basics

Adapted by Dr. Andrew Vardy from

www.wildbill.org/rose/Fall09/ch03.ppt

supercomputingchallenge.org/98-99/stts-99/vi.ppt

What is vi ?

- The *visual editor* initially developed on Unix.
- Before vi the primary editor used on Unix was the line editor
  - User was able to see/edit only one line of the text at a time
- The vi editor is a **text editor**, not a text formatter (like MS Word)
  - You cannot set margins…
  - Center headings…
  - Set text as bold…

Vi History

- Originally written by **Bill Joy** in 1976.
- Who is Bill Joy**?**
  - He co-founded Sun Microsystems in 1982 and served as chief scientist until 2003.
- Joy's prowess as a computer programmer is legendary, with an oft-told anecdote that he wrote the vi editor in a weekend. Joy denies this assertion.

Characteristics of vi

- The vi editor is:
  - Very powerful
  - …But cryptic
- The best way to learn vi commands is to use them
- So practice…

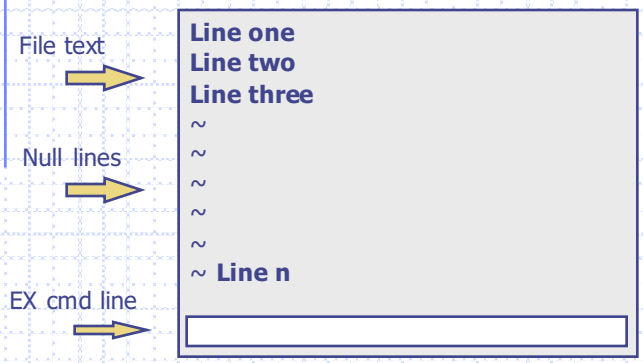
Vim equals Vi

- Most installations of vi actually use a different program called **vim**
  - **Vi** I**m**proved
  - http://www.vim.org
  - Charityware – donations accepted to help children in Uganda through the ICCF
  - Main author is Bram Moolenaar

Starting vi

- First, see what version of vi is installed on your system through "man vi"
- Type **vi** <filename> at the shell prompt
- After pressing enter the command prompt disappears and you see tilde(~) characters on all the lines
- These tilde characters indicate that the line is blank

vi Window Display



Vi is a Modal Editor

- There are (at least) three modes in vi
  - Command mode (a.k.a. normal mode)
  - Input mode (a.k.a. insert, replace mode)
  - Command-line mode (a.k.a. ex mode)
- When you start vi by default it is in command mode
- You enter the input mode through various commands
- You exit the input mode by pressing the Esc key to get back to the command mode
- You can go to command-line mode from command mode with the ":"
- (vim actually has another mode called "visual mode")

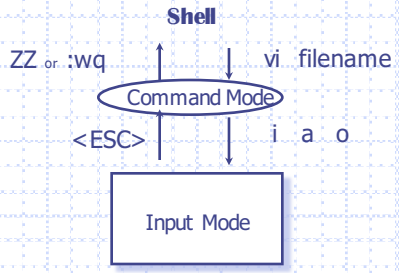
How to exit from vi

- First go to command mode
  - press **Esc** There is no harm in pressing **Esc** even if you are in command mode. Your terminal will just beep and/or or flash if you press **Esc** in command mode
- There are different ways to exit when you are in the command mode

How to exit from vi (comand mode)

- **:q** <enter> is to exit, if you have not made any changes to the file
- **:q!** <enter> is the forced quit, it will discard the changes and quit
- **:wq** <enter> is for save and Exit
- **ZZ** is for save and Exit (Note this command is uppercase)
- The **!** Character forces over writes, etc. **:wq!**

Vi Modes



Moving Around

- You can move around only when you are in command mode
- Arrow keys may work, but are not the standard way of moving the cursor in vi
- The standard keys for moving are:
  - **h** - for left
  - **l** - for right
  - **j** - for down
  - **k** - for up

Moving Around

- **w** - to move one word forward
- **b** - to move one word backward
- **$** - takes you to the end of line
- **^** - takes you to the first non-blank character of the line
- **0** - takes you to the begginning of line

Moving Around

- **)** - moves cursor to the next sentence
- **}** - move the cursor to the beginning of next paragraph
- **(** - moves the cursor backward to the beginning of the current sentence
- **{** - moves the cursor backward to the beginning of the current paragraph
- **%** - moves the cursor to the matching parentheses

Moving Around

- **Control-d** scrolls the screen down (half screen)
- **Control-u** scrolls the screen up (half screen)
- **Control-f** scrolls the screen forward (full screen)
- **Control-b** scrolls the screen backward (full screen).

Entering text

- To enter text in vi you should first switch to input mode
  - To switch to input mode there are several different commands
  - **a** - Append mode places the insertion point after the current character
  - **i** - Insert mode places the insertion point before the current character

Entering text

- **o** - opens a new line **after** the current one and goes to insert mode
- **O** - opens a new line **before** the current one and goes to insert mode

Editing text

- **x** - deletes the current character
- **d** - is the delete command but pressing only d will not delete anything; you need to press a second key
  - **dw** - deletes to end of word
  - **dd** - deletes the current line
  - **d0** - deletes to beginning of line
- There are many more keys to be used with delete command

The change command

- The change (**c**) commands delete the text specified then change to input mode.
- **cw** - Change to end of word
- **cc** - Change the current line
- There are many more options

Structure of vi commands

- vi commands can be prefixed by a number indicating how many times to execute the command
  n<command key(s)>
  - **10j** goes down by 10 lines
  - For example **dd** deletes a line, **5dd** will delete five lines.
- This applies to almost all vi commands

Undo and repeat command

- **u** - undo the changes made by editing commands
- **.** (dot or period) repeats the last edit command

Copy, cut and paste in vi

- **yy** - (yank) copy current line to buffer
- **nyy** - Where **n** is number of lines
- **p** - Paste the yanked lines from buffer to the line below
- **P** - Paste the yanked lines from buffer to the line above

(the paste commands will also work after the **dd** or **ndd** command)

Indenting

- Indenting code is crucial for good style!
- Indent current line: >>
- Indent 4 lines: 4>>
- Unindent: <<
- Unindent 10 lines: 10<<

Ex Commands

- We have already seen the following:
  - **:q** <enter> exit without saving
  - **:q!** <enter> exit without saving or prompting
  - **:wq** <enter> is for write (save) and exit
- There are also text processing commands with the following syntax:
  - : [range] command [args ...]

: [range] command [args ...]

- The range can be:
  - A number for that line (e.g. 7)
  - A pair of numbers for a range (e.g. 7,10)
  - $: The last line
  - %: All lines
- The command can be:
  - s: Search and replace
  - g: Perform a global action
  - Others…

Global Search and Replace

- **:%s/oldstring/newstring/g**
- This will change oldstring into newstring wherever it occurs throughout the entire text:
  - **%** - Means to try and apply this to all lines
  - **s** – Stands for "substitution"
  - **g** – Means to replace as many times as possible within the line (otherwise there is only replacement per line)

General Global Actions

- The g command uses some existing vi command, but applies it globally (to the specified range of lines)
- It has the following form:
  - **:[range]g/pattern/command**
- e.g. Delete all lines that start with #
  - :g/^#/d
- e.g. Delete all empty lines
  - :g/^$/d

Vi References

- **The Vi Lovers Home Page**
  http://thomer.com/vi/vi.html
- **The Editor War**
  http://en.wikipedia.org/wiki/Editor_war

Use a vi Cheat Sheet