Vim: A great tool for your toolbox!

When things are dim
Give 'em VIM

CLEANSER & POLISHER
LEVIR BRITHEIERS LIMITED, PORT SUNLIGHT.
History

- Originally created for the Unix operating system by Bill Joy (one of the founders of Sun)
- Created a visual interface for the original ex line editor on UNIX
- The name vi is derived from the shortest unambiguous abbreviation for the command visual in ex
- The Single UNIX Specification specifies vi, so every conforming system must have it.
- vi is not open source but there are several clones with the most popular being Vim
Vim

- Vim is a clone of vi created by Bram Moolenaar and was first released in 1991
- Vim stands for Vi IMproved
- Vim is not 100% compatible with vi as defined in the Single Unix Specification
- Vim has many improvements over standard vi
- Vim has a built-in tutorial for beginners (accessible through the `vimtutor` command)
- Vim has 6 modes, 4 of which you will spend 99% of your time in.
Vim Modes

- **Normal mode** - For navigation and manipulation of text. This is the mode that vim will usually start in, which you can usually get back to with ESC.

- **Insert mode** - For inserting new text. The main difference from vi is that many important "normal" commands are also available in insert mode - provided you have a keyboard with enough meta keys (such as Ctrl, Alt, Windows-key, etc.).

- **Visual mode** - For navigation and manipulation of text selections, this mode allows you to perform most normal commands, and a few extra commands, on selected text.
Vim Modes - continued

- **Select mode** - Similar to visual but with a more MS-Window like behavior.
- **Command-line mode** - For entering editor commands -
  - Try :help
  - Try :Ni!
- **Ex-mode** - Similar to the command-line mode but optimized for batch processing.
Using Vim

- When using Vim the goal is to keep your hands on the home row
- Primarily you will use normal mode, and insert mode when editing text
- Visual mode allows you to cut and paste large swaths of text all at once
- Command mode allows you to send commands to the editor such as save (:w) or quitting (:q)
Learning Vim - basic commands

- Saving a file - ESC then :w
- Opening a file - ESC then :e <filename>
- Quitting - ESC then :q
- Save then quit - ESC then :wq
- To switch to insert mode hit i. To get back to normal mode hit ESC
- Here is good guide to get you started with the basics.
  http://cs.boisestate.edu/~amit/teaching/handouts/vi-two-page-ref.html
- vimtutor will walk you through all the basic commands.
Vim has a powerful regular expression engine built in to help you find and edit text.

While in normal mode type `/<regex>` and Vim will find all the text that matches.

You can also do a complex find and replace with :%s/<regex>/<new text>/cg which can be read as globally (g) find all the text that matches the <regex> and replace it with <new text> and confirm with me (c) before you actually make the change.
Customizing Vim

- When Vim starts up it looks for a file named `.vimrc` in your home directory.
- You can use your `.vimrc` with any version of Vim.
- You can customize your `.vimrc` to add in any plugins or settings that make Vim easier to use. For example, changing the font to something easier to read.
- Take a look at Shane’s vimrc as a starting point: [https://github.com/shanep/vim](https://github.com/shanep/vim)
- Vim takes some time to learn but once you have mastered it the payoff is faster editing and coding.
The font is bad with the default install you can fix it with the following code in your .vimrc

```vi
if has("win32")
    set guifont=Consolas 18
endif
if has("unix")
    if system('uname')=~'Darwin'
        set guifont=Menlo\ Regular:h18
    else
        set guifont=Inconsolata\ Medium\ 18
    endif
endif
```
Vim has an extensive set of plug-ins that you can leverage to add additional functionality. Below is a few plug-ins that you may want to try out.

- **NERDTree** - Gives you a buffer view of all the files and directories in your current working directory. (Similar to the project window in eclipse)

- **TagBar** - Gives you a layout off all the functions, methods, classes, and structs of the currently loaded buffer. (Similar to the class view window in eclipse)

- **SuperTab** - Ties the built in omnicomplete to the tab button. (Not as great as code complete in eclipse but better than nothing at all!)
There is a GUI version of Vim that is easier to use and more friendly in a graphical environment. You get a toolbar that provides many of the common commands so you don’t have to memorize everything!

You can install this on Fedora Linux with the command:

```
yum install vim-X11
```

MacVim is a version of vim that has been customized for Mac

GVim is also available on windows.

You should generally use GVim or MacVim for daily usage. The terminal version is handy if you are working over a slow ssh connection.
References

- wikibooks entry on Vim: http://en.wikibooks.org/wiki/Learning_the_vi_Editor/Vim/Modes
- Vim homepage: http://www.vim.org