Vim Skill Module

Parth Upadhyay

1 How to get credit for this Skill Module

There are 4 steps to completing this Skill Module.

1. Read this document
2. Run `vimtutor` and complete it
3. Use vim for a while on your own
4. Complete the questions and return them to your UGTA

Your finished product should have the following things:

- A brief description of your experience with vim.
- The questions answered to your best ability

2 Introduction

Vim is a text editor written in 1988 by Bram Moolenar, now an employee at Google. Vim stands for *Vi IMproved* as it is based on the older text editor Vi, but is thought to have a feature set that surpasses that of Vi. It now comes default on basically any linux distribution you could possibly imagine using, and has become extremely popular in the Unix community since its onset.

2.1 Why Learn Vim?

With powerful IDEs like Eclipse and Netbeans available, you might ask why its even worth learning vim. While to some extent vim does not contain all the features that these IDEs contain, vim is still an extremely powerful text editor that will make you much more efficient. It has an extremely rich set of functionality that will allow you to edit text in ways that conventional text editors do not allow. Also, vim is extremely lightweight and there will be many times when you do not need a powerful IDE, or cannot even use one. It's one of those things that once you start using, you will begin to appreciate.

And bottom line, if you are a computer science major then you will **need** to learn how to use a unix text editor. You'll be forced more and more to work in terminal environments where getting a graphical interface is not really possible, and your efficiency in coding will be highly dependent upon your ability to use a text editor effectively.

3 Getting Vim

Vim is available for all operating systems and architectures, and I will outline how you can get a hold of it on different systems. However, **I highly recommend you work on a linux system. Most of the skill module assumes you are in a Linux environment, and some parts even require it.**

If you do not have linux installed on your computer, or you find it troubling to installing it, you can always use the CS machines in ENS basement or in PAI.

3.1 Linux

If you want to install in a linux environment, obtain vim from your package manager. For instance, on Ubuntu you would do:

```
sudo apt-get install vim
```

You can also obtain the graphical version of vim by doing (this package contains both the normal version and the graphical version):

```
sudo apt-get install vim-gnome
```

3.2 Windows

Follow this link to obtain `Gvim`. `Gvim` is a graphical version of vim: `http://www.vim.org/download.php#pc`

3.3 Mac

Consult the following link. `http://www.vim.org/download.php#mac`.

4 Understanding Vim Configuration Files

Note: this only applies if you develop on a *nix environment.

As a consequence of vim's rich functionality, vim is extremely configurable. You can change how it behaves with regards to colorschemes, syntax coloring, tab spacing, highlighting, etc. To deal with all of this, Vim uses a *configuration file* named `.vimrc`. On linux systems, files whose names are prefixed by dots are hidden. This means that when running

`ls` or just looking through your file explorer, they will not be shown by default. (As an aside, to view hidden files when in terminal run `ls -a`, the `-a` standing for 'all'.)

Vim knows where to look for this configuration file. The location it looks first is in your home directory, i.e. `~/.vimrc`. I won't go into detail here about actually writing your own configuration files; for now, copy mine and as you use vim more you can change it suit your own purposes. I've included the file on the website. Download the file and place it in your home directory after you install vim. *Ensure that the filename is .vimrc, not anything else like .vimrc.txt.*

5 Editing in Vim

It's finally time to begin learning to use vim!

If you have already tried opening vim and editing a file, you may have noticed that it's not exactly intuitive. In fact, without any previous knowledge of vim it is probably impossible to edit a file. However, there aren't a lot of things you need to learn to become relatively efficient in vim.

There are two modes in vim, **Command Mode** and **Insert Mode**. **Insert Mode** is what you'd expect; you can type normally just like any other text editor. **Command Mode**, or the mode in which Vim normally begins, is the mode where you can run different commands to edit the text.

5.1 Insert Mode

Insert Mode is just like being in notepad. You can type text, delete text, etc. There's not much to it.

5.2 Command Mode

Command Mode is where vim's true usefulness shows. From command mode, you can run different commands, or basically keyboard shortcuts, to manipulate and traverse text. For instance when in Command Mode, pressing 'w' will move your cursor to the next word, pressing 'gg' will take your cursor to the top of the file, pressing 'dd' will delete the current line, pressing 'p' will paste the last thing in the buffer, etc. etc. This list goes on and on, and in the next section we'll tackle learning a useful subset of this list.

5.3 Switching Between Modes

This all begs the question, how does one get to either mode? By default, Vim starts you out in Command Mode. To get to insert mode, press 'i'. To get out of insert mode and back to command mode, press '⟨ESC⟩'. And it's that easy.

6 `vimtutor`

So there already exist many tools for learning to use Vim. One among them is called `vimtutor`. This program can be obtained from your package manager in Linux, and is also on the CS machines.

To run this program, open up a terminal and run the command `vimtutor`. After that, just follow along. It's honestly not that long, and is very helpful in learning how to use Vim properly. Note that vimtutor is really nothing more than editing a text file in vim.

7 Conclusion

As I hope you will come to see, Vim is an extremely powerful text editor that despite its age has still maintained vast popularity in the development community. **If you read nothing else in this entire Skill Module, please at least read this. The only way to really learn Vim is to go out and use it. It will be annoying to use initially, and you will feel less efficient. However this 'breaking-in' period will be short, and as you become more proficient you will come to appreciate Vim's many features in editing text.**

The only way to get better at Vim is to constantly question how you do things. Do not get set in habits, but rather catch yourself and always question if there is a faster way to do what you are doing.

8 Advanced Vim

There are more advanced features of vim that `vimtutor` did not cover, and I will not cover here. However, I encourage you to go out on your own time and learn about these things as you become more familiar with vim. Here's a list of things that I think you may find interesting to learn more about:

- Vim Plugins and Colorschemes
- Markers
- Registers
- Macros
- Window Splits
- Vimdiff
- Fixing Code Indentation
- Code Complete (Yes, Eclipse-style)

And there's much more that I don't even know about!

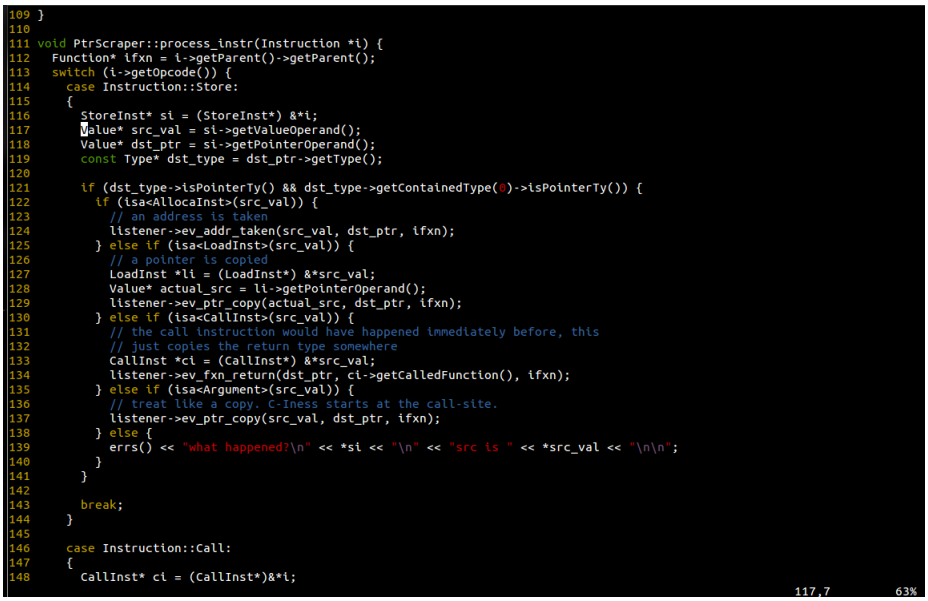

Figure 1: Code Snippet

9 Questions

Consider Figure 1. Each question will ask you to take some action upon the text, and answer as if your cursor is in the current position.

In **bold** next to every question, i've indicated how many characters it took me to do it. This is not suggesting that you should get the same number, or that I've achieved the best way to do it. It is just a guide to help you think in the right direction, and you won't be penalized for not getting it in the exact same way. Just try your best.

1. This is a gimme. Describe how to move left, right, up and down from the keyboard.
2. How do you undo a change? Redo? (**1**)
3. How can you delete the whole line? (**2**)
4. How can you delete the current line and the next 5? (**3**)
5. How can you duplicate the current line in place? (**3**)
6. How can you bring your cursor to sit upon the equal sign? (**3**)
7. For this question assume your cursor is at the equal sign. Now delete the rest of the line. (**2**)
8. Same context as previous question. Delete until beginning of the line (**2**)

9. How can you delete only the word 'Value' upon which the cursor sits? (**2**)
10. Page Up? Page Down? (Don't hit the page-up key). (**2 for each**)
11. How can you jump backward by words? (**1**)
12. How can you get to the beginning/end of the line? Of the file? (**2, 1**)
13. Delete the contents of the entire file. (**4**)
14. Find the matching end-brace of the brace on line 115. **3**
15. Delete everything contained from the brace on line 115 to its match. (**4**)
16. Search for all instances of `src_val`. (**Theres really only one way to do the search and replace ones, and the length isn't important**)
17. Replace `src_val` with `sv`.
18. Now modify that to work for any number of instances on the line.
19. Now modify that to work for the entire program.
20. Now modify that to prompt you before making changes.

10 Contact

If at any point you have questions about this Skill Module, Vim, or anything in general, do not in the slightest hesitate to email me at `parth.upadhyay@gmail.com`. I would be more than happy to help you out. Also feel free to forward me suggestions/thoughts you have about this skill module.

You can freely get a copy of the source of the skill module here: `https://github.com/parthupadhyay/Vim-Skill-Module`.