Unix Tutorial 3: Vim for text editing

Today, we are going to use the vim text editor. It is pre-installed on most Linux systems. However if you are running your own Linux distro, you may need to install vim. Issue a command like:

```
$ sudo apt install vim
```

on Ubuntu distros.

Plenty of other text editors are available, including Emacs (Jeremy's favourite!), gedit (more like Notepad++) and nano (very basic). We are focusing on vim since it is a nice, small application that is installed by default on most Linux servers.

Writing a Java Program

Remember Java from last year? Maybe you used eclipse or VS Code or IntelliJ? Now let's use vim instead!

Let's write a one-class Java program.

```
$ cd ~
$ vim Hello.java
```

This should drop you into the vim screen, where you can start to edit the file. Press [i] to enter *insert mode* then type the Java program, something like...

```
public class Hello {
  public static void main(String [] args) {
    System.out.println("Hello world");
  }
}
```

Now we need to save the Java program. Press [Esc] (escape) to exit the insert mode. Then press the following two keys one after another (i.e. not at the same time): [:] [w] [Enter] This will save your Java source code file.

Use the cursor keys (or [h] [j] [k] [l] keys) to navigate to the second line, containing the `main` method signature. Press [o] to *open* a line underneath and transition to insert mode. Enter a comment, something like...

```
public class Hello {
  public static void main(String [] args) {
    // simple console output
    System.out.println("Hello world");
  }
}
```

Press Escape to exit the insert mode. Now navigate so your cursor is over the w character of the word `world` in the output string. Press the following sequence of keys (one after another) to delete this word: [d] [w]

Let's do some cutting and pasting now (in vim terminology, this is called yanking). Move the cursor to the start of the System.out.println statement. Press [v] to enter visual mode. Move the cursor to the semicolon at the end of the statement. Press [y] to yank (i.e. copy) this text. Then move the cursor down a line and press [p] to paste. You might need to rearrange your curly braces and space-tabbing for the formatting. Hopefully you will end up with a program like:

```
public class Hello {
  public static void main(String [] args) {
    // simple console output
    System.out.println("Hello ");
    System.out.println("Hello ");
  }
}
```

We can delete a whole line at once. Let's get rid of the comment line. Move the cursor to this line, check you are not in insert mode, then press the key sequence [d] [d] For single-character edits, use [x] to delete the single character under the cursor.

Let's save our program again. Press [Esc] [:] [w] Enter to write the file.

Is your source code shown using syntax highlighting? If not, try entering the command `:syntax on`.

Vim is very powerful—we have only looked at the basic editing facilities. If you like it, you are encouraged to find out more online. In particular, you should note that we started in *normal* mode, then switched to *insert* mode and finally did some work in *visual* mode. Key presses mean different things in different modes.

To quit the vim program, type [Esc] [:] [q] [Enter] in sequence. If you have unsaved files, you are prompted to save these.

Compiling and Running Java Programs

Now we have created a Java program, let's compile it. We are going to use the command-line Java compiler, which appears to be less user-friendly than Eclipse—but hopefully you will get used to it.

First check whether the Java compiler is installed on your system:

```
$ which javac
```

If this command returns a line showing the location of the javac file, then it is installed. If this command does not print out any text, then javac is not installed. To install it on Ubuntu-style systems, do this:

```
$ sudo apt install default-jdk
```

Or if you do not have permissions to do this, try this sequence of commands instead:

```
$ curl -s "https://get.sdkman.io" | bash
$ source "$HOME/.sdkman/bin/sdkman-init.sh"
$ sdk list
$ sdk install java VERSION
```

where VERSION is the appropriate version for you to install.

Presuming you are still in the directory where you saved Hello.java, execute these commands:

```
$ javac Hello.java
$ ls *.class
```

If the compilation completes successfully, you should see a Hello.class file. If the compilation fails, read the error message and work out where the error is in your file. Use vim again to correct the file, then recompile it.

You can run the Java class file as follows:

```
$ java Hello
```

Notice that the Java output is sent to the standard console output, just like for other Linux utilities. This means you can put Java apps into pipelines with grep, etc. For example, let's count the number of times that the Java program prints Hello:

```
$ java Hello | grep -ci hello
```

which should print out the result 2.

Further Reading

There are lots of vim tutorials online, including plenty of interactive screencasts. Search youtube for `vim power user` videos and see what you find. Let me know if you watch any really good videos!

Here is a nicely formatted vim cheatsheet for commands and shortcuts: `http://tnerual.eriogerg.free.fr/vimqrc.pdf`

You will have noticed that we transitioned between various user *modes* while editing our documents in vim. There is a good explanation of the different modes at `https://www.warp.dev/terminus/vim-modes`.

Sadly, the vim developer passed away in August 2023. He was an influential open source coder. You can read his obituary at `https://www.theregister.com/2023/08/07/bram_moolenaar_obituary/`.