# UNIX Workshop 2010

http://uws.assembla.me

Angad Singh  Aldrian Obaja  Dang Cao Khoa  Chong Perry
Saiaswin Saikannan  Jialong Liu  Vu An Hoa  Gary Kwong  Melvin Zhang

August 4, 2010

"It's a UNIX system! I know this."

– Alexis "Lex" Murphy, Jurassic Park (1993)

## 1 Why UNIX?

You may be running Windows on your desktop/laptop, but the next time you visit Google or Facebook, the webpage you see is being served from a computer running UNIX. According to w3techs.com, in August 2010, at least 67% of all web servers are running on UNIX.

UNIX is the name of the operating system developed by a group of AT&T researchers at Bell Labs in 1969. Most modern operating systems, such as Linux, Solaris, Mac OS X, and BSD (see Figure 1), descended from UNIX. All of these operating systems provide a UNIX-like environment to the user. In the rest of these notes, the term UNIX will refer to these modern day descendants of the original AT&T UNIX.

The story of UNIX is also tied in with the creation of the C programming language. Dennis Ritchie invented C in order to rewrite UNIX. Incidentally, starting from the 2010 intake, the School of Computing changed the programming language used in its programming methodology module from Java to C. For those of you doing Scheme, you will learn C when you start on your Operating System module (CS2106).

Finally, in addition to learning about programming methodology in the introductory modules, you will be required to develop your C/Java programs in the UNIX environment.

The basic workflow for writing programs is the edit-compile-run-test cycle (see Figure 2). For interpreted languages, such as Scheme, there is no compile step. In this workshop, we will be focusing on the parts of UNIX related to the edit-compile-run-test cycle.

## 2 Getting started

SoC operates a number of UNIX servers where students can login and interactively develop programs in the UNIX environment. We will be accessing SoC's `sunfire` server from the PCs in the programming lab. Use your NUSNET account to login to the PC in the programming lab.

**Activity: Login to NUSNET**

1. Press `Ctrl-Alt-Delete`.
2. Type in your NUSNET user name, password and select the NUSSTU domain.
3. Click on the `Ok` button.

Before you can login to `sunfire`, you will need to create your UNIX account. In the following activity, you will create your UNIX account.


Figure 1: Modern UNIX-like operating systems (Linux, BSD, Solaris)

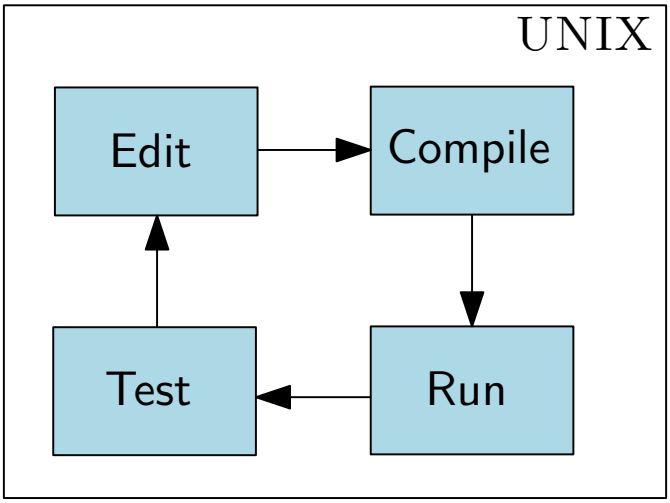


Figure 2: Workflow for program development

Figure 3: `sunfire` server located in the Machine Room with our Central Facilities staff. Clockwise from top-left: Tan Chee Sin, Tan Kwang Pon, Budiman Tsjin and Lai Zit Seng (Systems programmer, ITU).

**Activity: Creating your UNIX account**

1. Login to `https://mysoc.nus.edu.sg/~newacct` using your NUSNET user name and password.
2. Decide your UNIX user name.
3. Type in your new password (twice).
4. Submit your application.

SoC's `sunfire` server runs on the Solaris operating system developed by Sun Microsystem.

# 3 Command line interface

In the next activity, we will connect to the `sunfire` server (see Figure 3) and access the UNIX system via the Secure Shell protocol (ssh).

**Activity: Connecting to `sunfire`**

1. From the desktop, launch the `SSH Secure Shell Client` application.
2. Click on `Quick Connect` Host Name: `sunfire.comp.nus.edu.sg` User Name: your UNIX user name
3. Click on `Connect`.
4. Click on "Yes" at the Host identification dialog.
5. Enter your UNIX password in the password dialog.

Once you have logged in, you will see a command line interface (see Figure 4). In this interface, we interact with the system by typing in a single line of command, followed by the `Enter` key.

A command consists of two parts: the name of the program you wish to execute, followed by a list of arguments or input to that program.

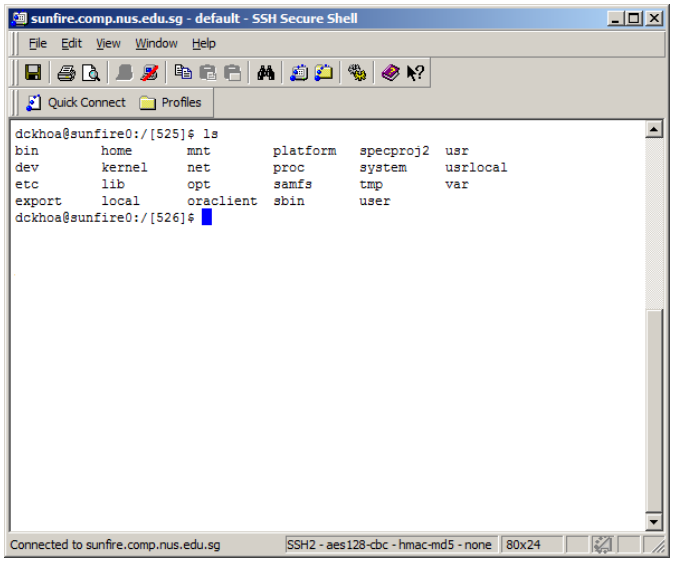


Figure 4: Command line interface on `sunfire`

```
$ program_name argument1 argument2 ...
```

Such an interface might seem rather primitive compared to the rich graphical user interfaces (GUI) we are used to. However, there are a number of advantages to using the CLI. It is a reasonable interface for programs that do not need to interact with the user and it makes it easy to invoke any program even if there are thousands of programs installed. Once you are comfortable with this interface, it can be much faster than using the GUI.

Our next activity involves working with files and directories. On UNIX, files are arranged in a hierarchy (see Figure 5) of directories. The top of the hierarchy is traditionally called root (written as a slash /).

In the following activity, we will use the command line to create a directory to store files related to this workshop. There is a concept of a *working directory*. Commands that operate on files or directories assumes that you are modifying your working directory.

**Activity: Working with files and directories**

1. After login, you are place in your home directory, e.g. `/home/m/melvin`
2. You can check your working directory using the `pwd` command

   ```
   pwd
   ```

3. The `ls` command shows you the files in your working directory

   ```
   ls
   ```

4. Now create a new directory called `UNIXWorkshopFiles`

   ```
   mkdir UNIXWorkshopFiles
   ```

5. Switch to the new folder using the `cd` command

   ```
   cd UNIXWorkshopFiles
   ```

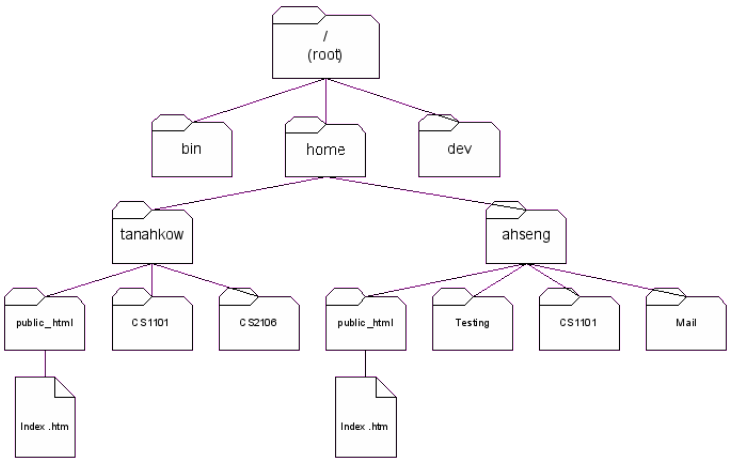


Figure 5: A subset of the UNIX directory tree showing home directories

6. Use the `pwd` again command to check your working directory

   ```
   pwd
   ```

# 4 Text editing

There are many types of files in UNIX, the include:

- a document (report, essay etc.)
- the text of a program written in some high-level programming language
- instructions comprehensible directly to the machine and incomprehensible to a casual user, for example, a collection of binary digits (an executable or binary file)
- a directory, containing information about its contents, which may be a mixture of other directories (subdirectories) and ordinary files.

By far the most common are text files, which are used extensively on a UNIX system for storing system data, program configuration files, scripts and source code. They are preferable to binary files because they can be easily read/modified. The most important program for working with text files is a *text editor*, a program to interactively create/edit text files. Examples of editors on UNIX are nano, Vim, and Emacs.

In the following activity, we will use nano, http://www.nano-editor.org/, to create a simple text file. Nano is a basic text editor that is easy to learn, if you are interested in a more sophisticated text editor, you should try learning Vim or Emacs.

**Activity: Text editing with nano**

1. From the Secure Shell Client window start nano and create a new file using the command

   ```
   nano lorem_ipsum.txt
   ```

2. Type the following paragraph as carefully as possible.

```
Lorem ipsum dolor sit amet,
consectetur adipisicing elit,
sed do eiusmod tempor incididunt
ut labore et dolore magna aliqua.
Ut enim ad minim veniam, quis
nostrud exercitation ullamco
laboris nisi ut aliquip ex ea
commodo consequat.
```

3. Save the file and exit nano by pressing

```
Ctrl-x
```

# 5 Working with text files

**Activity: playing with `diff` and `grep` I**

1. What is `diff`? – compare differences between files
2. Text editing usually leaves a lot of backup files ending with `~`. One day you want to figure out the differences between a file `text` and its backup `text~`... Open in two editors and then eye-ball?

```
diff firstFile secondFile
```

3. A quick how-to

**Activity: playing with `diff` and `grep` II**

Let's see how `text` and `text~` look like first

Output of `cat text`

```
same text
same text
Hello World!
still the same
still the same
```

Output of `cat text~`

```
same text
same text
Hello World~
still the same
still the same
```

**Activity: playing with `diff` and `grep` III**

Output of `diff text text~` :

```
3c3
< Hello World!
---
> Hello World~
```

**Activity: playing with `diff` and `grep` IV**

1. What is `grep`? *– look for a pattern in file(s)

   ```
   grep pattern file
   ```

2. Sometimes it is useful to find the occurrences of some word in a (list of) file. Say you suspect a typo in you source code, open a text editor and 'Find'?

3. But what if you made the same typo in a lot of files? 'grep' makes your life easier Let's find out how to 'grep'

**Activity: playing with `diff` and `grep` V**

Sample output of `grep h1 a.html`:

```
grep h1 a.html

  <h1>Hello World!</h1>
```

Contents of a.html

```
<html>
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```

A bit too easy, isn't it? Ready to get nasty?

1. Output of `grep h1 a.html*`

   ```
   a.html:<h1>Hello World!</h1>
   a.html~:<h1>Hello World~</h1>
   ```

2. Output of `grep -n h1 a.html*`

   ```
   a.html:3:<h1>Hello World!</h1>
   a.html~:3:<h1>Hello World~</h1>
   ```

3. Output of `grep -n -i 'heLlO wORlD' a.html*`

   ```
   a.html:3:<h1>Hello World!</h1>
   a.html~:3:<h1>Hello World~</h1>
   ```

4. Find out more in `'man grep'` !

5. grep on Linux is more fun! :p

# 6 Combing multiple tools

UNIX is more than just a family operating systems. It also an approach towards software that emphasizes creating small sharp tools that work well together. This is embodied in the UNIX philosophy.

**The UNIX Philosophy**

> Write programs that do one thing and do it well.
>
> Write programs to work together.
>
> Write programs to handle text streams, because that is a universal interface.
>
> – Douglas McIlroy(inventor of UNIX pipes)

The next activity demonstrates the UNIX philosophy, by making use of three simple UNIX utility programs to analyse SMS messages.

**Activity: SMS Word Count**

Your friend from FASS is studying SMS language as part of a course project. She collected a number of SMS messages and would like to find out the frequency of each word.

For example, given the following text file:

```
U wan 2 haf lunch i'm in da
canteen now.
Haf u found him? I feel so
stupid da v cam was working.
Where r we meeting?
I went to ur hon lab but no
one is there.
```

The desired output is:

```
.
.
.
1 we
1 went
1 Where
1 working.
2 da
2 I
```

**Activity: `sort` and `uniq`**

Two UNIX utility programs are related to our task.

`sort`

| Input: | | Output: |
|---|---|---|
| dog | | bat |
| bat | $\longrightarrow$ | cat |
| log | | dog |
| cat | | log |

`uniq`

| Input: | | Output: |
|---|---|---|
| dog | | dog |
| dog | $\longrightarrow$ | cat |
| cat | | dog |
| cat | | cat |
| dog | | |
| cat | | |
| cat | | |

**Activity: SMS Word Count**

1. Download the file containing sms messages from `http://uws.assembla.me/SMSwords.txt` using wget

   ```
   wget http://uws.assembla.me/SMSwords.txt
   ```

2. Sort the file.

   ```
   sort SMSwords.txt
   ```

3. Sort and remove duplicates.

   ```
   sort SMSwords.txt | uniq
   ```

4. We need to use a particular option of `uniq` which counts the number of duplicates, read the manual page for `uniq`. Press `q` to leave the manual page.

   ```
   man uniq
   ```

5. Sort and count words,

   ```
   sort SMSwords.txt | uniq -???
   ```

6. Sort by the frequency, so that more frequent words appear later,

   ```
   sort SMSwords.txt | uniq -???  | sort -n
   ```

# 7 Summary

Finally, after we are done with what we need to do in the UNIX environment, we have to logout of the system.

**Activity: Logging out of `sunfire`**

To log out of `sunfire`, use the `logout` command,

```
logout
```

**Useful programs/websites**

- KiTTY, SSH client for Windows `http://www.9bis.net/kitty/`
- Cygwin, UNIX-like environment for Windows `http://www.cygwin.com/`
- Description of computing facilities in SoC `http://docs.comp.nus.edu.sg/cf`
- mySoC, web service portal `https://mysoc.nus.edu.sg`