CompSoc Presents

INTRO TO TERMINAL

Created by CompSoc Academic Reps, 2016

A quick introduction...

Welcome to CompSoc's first workshop of the year! Over the next two hours (or however long it takes) we'll be learning about some of the introductory commands so that you can start using the Linux Terminal like a pro. All the computers on the first floor of Erskine run the Linux Operating System. (Read more about the specific version here.) Linux is also very common in the 'real developer world', and getting familiar with it now will be beneficial to your future!

This document contains information and a few exercises to help you get acquainted with the terminal. If at any time you get stuck, the instructions don't make sense, or you just want to chat about Linux, use the "red flag" program on your computer and one of our tutors will be happy to chat. However, keep in mind that there's a whole wide world full of commands out there, and this tutorial only starts to scratch at the surface.

What is the command line, anyway?

Linux's command line is a super useful tool that can do anything a GUI can do - and more! In non-jargony terms, the terminal is a way of interacting with your computer using text instead of a graphical interface.

To get detailed (perhaps too detailed) instructions on any terminal command, just type `man` followed by the name of the command. For example, `man ls` . This brings up the 'manual page', that tells you a bit about each command and how it can be used. Type q to exit the manual page.

Give me back my file browser!

Challenge: For the entirety of this section, don't open your file browser.

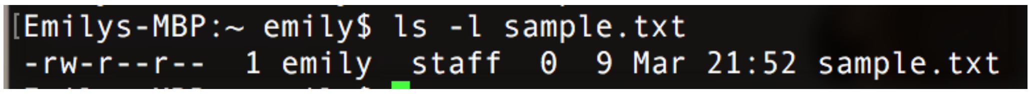
Exercise:

1. With a brand new terminal window open, type in `ls` and press enter. This command shows you the contents of the current directory, including files and subfolders.
2. Next up, we're going to create a new folder and also a file to go inside it. Type the command `cd ~/Documents` so you can create your folder in the right place.
3. Now type, `mkdir myCoolLinuxFolder` This creates a new folder called myCoolLinuxFolder. You can now navigate into it with the `cd` command.
4. Once you've navigated to the folder, you can now create a file using the `touch` command. Create a text file named sample.
5. Now, use `ls` again. You should now see "`sample.txt`" as the output of this command.
6. You can delete your file (if you wish) using the command `rm sample.txt`

Now, enter `man cp` and have a read through how the manual describes the copy function. Try copying "`sample.txt`" into `~/Documents`, the folder one above your current layer. If even after a bit of thought the manual page still looks like hieroglyphics, feel free to ask someone around you or a tutor. And the best bit? Cut is just copy and delete put together, so you just learned how to cut as well!

An extension on file management:
If you're interested in managing who can access your files and how, the two following commands might be of particular interest to you.

chmod allows you to modify file permissions (who can read, write and execute the file)
You can see who currently has permission by typing `ls -l sample.txt` . The output of this command will look a little something like this:



```
Emilys-MBP:~ emily$ ls -l sample.txt
-rw-r--r--  1 emily  staff  0  9 Mar 21:52 sample.txt
```

A quick breakdown:

- The first character specifies file type. '-' means a regular file; 'd' means a directory, l means a symbolic link.
- The next three characters show the owner's permissions. 'r' for read, 'w' for write, and 'x' for execute.
- The next three characters are the permissions for the user group that the file belongs to.
- The final three characters are the permissions for everyone else. In this case, other people cannot write or execute the file, only read it.

Now that we know how to read permissions, we can learn how to set them using `chmod`.
Each permission has a 'point' value: Read is worth 4, Write is worth 2, and Execute is worth 1. These points get combined for each set of people.

For example, if you are the owner you want to be able to read, write and execute. So you would give yourself 4 + 2 + 1 = 7 points. You want your group to be able to read and execute, so you give them 4 + 1 = 5 points. And lastly, you want everyone else to only be able to read the file. So you give them four points.

Now, you put those three values together in that order (754) and use this value in your command. It would look something like this:

```
chmod 754 sample.txt
```

Or, if you wanted to have a secret file that can only be accessed by you, the owner, you would type:

```
chmod 700 sample.txt
```

chown allows you to modify the owner/s of the file. For example, to change the owner of the file `dna_evidence.txt` to the user `TomMagnum`, you would enter the command `chown TomMagnum ./dna_evidence.txt`. If you wanted to change the owner of the file `oahu_prison_records` and all of its children to `TomMagnum`, you would enter the command `chown -R TomMagnum oahu_prison_records`

Some Helpful Tips on File Navigation

1. The ~, aka tilde, is the symbol you use to tell the terminal "go to my home folder", or the topmost level of your file organisation.
2. To go 'back up' a level, use the command `cd ../` The number of times you type this applies to the number of levels up you'll travel. For example, `cd ../../../` will take you back up three levels of directories.
3. **What follows is a command you should never, ever use unless you want to go upstairs and explain that you've deleted everything: rm -r /***

Pipes

What is a pipe? It's a means for taking the output of one program and using it as the input for another program.
The terminal you are working with is a program. It takes the input from you keyboard **stdin** and pipes it into other programs such as **ls** which then prints back out to the terminal as **stdout**.

The Pipe Operator '|'

First let's create some sample data
copy this block of commands into your terminal and press enter.

```
mkdir ~/learningAboutPipes/
mkdir ~/learningAboutPipes/nextLevelPipes/
touch ~/learningAboutPipes/pipingHotFile.txt
touch ~/learningAboutPipes/thisIsNotWhatIWanted.txt
touch ~/learningAboutPipes/nextLevelPipes/pipingToTheNextLevel.txt
touch ~/learningAboutPipes/nextLevelPipes/something.txt
```

Example Situation: I want to know if there are files about piping in the learningAboutPipes directory (assuming the file names have the word piping in them)

So let's look at this command: `ls -R ~/learningAboutPipes | grep piping`
You should see output like the following:

```
pipingHotFile.txt
pipingToTheNextLevel.txt
```

A little about what's going on here: first we have used the `ls` command to list the contents of the `~/learningAboutPipes` directory. We have used the `-R` command with `ls` which tells it to also list files in subdirectories.
If we ran the command by itself, `ls -R ~/learningAboutPipes` we would get the following output:

```
/home/cosc/student/dmw99/learningAboutPipes/:
nextLevelPipes  pipingHotFile.txt  thisIsNotWhatIWanted.txt
```

```
/home/cosc/student/dmw99/learningAboutPipes/nextLevelPipes:
pipingToTheNextLevel.txt  something.txt
```

Above is the **stdout** for the `ls` command. We can then pipe this into the `grep` command like so :

```
 ls -R ~/learningAboutPipes | grep
```

Now, the grep command won't do much on its own. We have to tell it that we are interested in finding files with the word 'piping' in the file name:

```
ls -R ~/learningAboutPipes | grep piping
```

Extension:

Can you guess what the following commands will output?

```
1. ls -R ~/learningAboutPipes | grep Level
2. ls ~/learningAboutPipes | grep piping
3. ls ~/learningAboutPipes | grep Next
```

Summary

Piping is awesome, it can be used to chain multiple programs together, this makes the terminal extremely powerful.
There are many examples and tutorials on the internet which go more indepth than what we can cover today. This section is only meant to give you a taste of what you can do with pipes.

Python is bae

We can use the terminal to run Python programs!

The Python Shell

You might have seen your lecturers working with the python shell in the terminal. Here's how to get it yourself:

1. Type `python3`
2. That's it! You're in!

Now you can type python commands and watch them run right in front of you. Some things to try:

1. Simple math (e.g. 2 + 4)
2. Variable setting and getting
3. Using built in Python functions - check the cheat sheet for some ideas

Remember to type exit() once you're done in the Python shell to return to the regular terminal.

Running Files

Running Python files is also very simple. Simply navigate to a directory where you have an existing python file and type `python3 filename.py`
If you don't have any Python files of your own, copy and paste the following into a new file, and save it as HelloWorld.py:

```
print("Hello, World!")
```

You can copy any files into your current cool linux directory from any other folder by navigating to the file you want and using the command
`cp filename.py ~/Documents/myCoolLinuxFolder`

This can be considered the copy/paste command: it uses the syntax copy [file] [destination]

Help! I left my work at Uni...

Scenario: You're at home, there's a snowstorm outside, and your assignment is due in five hours. It's okay, we've all been there.
But what happens when you the work you've done on the lab computers, and you can't get into uni because of the aforementioned snowstorm?!

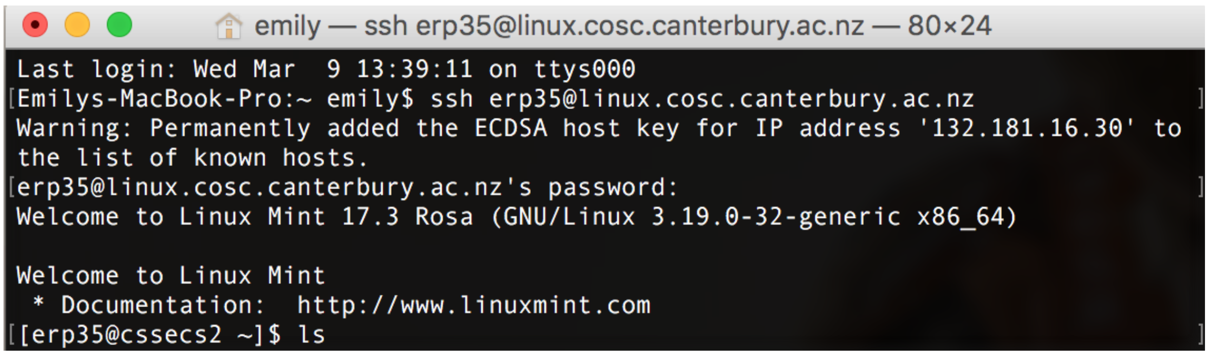
Solution! Use SSH (Secure SHell) to access your uni account!
If you have your own laptop handy, you can try this in the lab. Otherwise, you could try working with a friend to access their uni account from yours. (Don't share passwords, obviously.) You can even SSH into your uni account, from your uni account!

Linux/Mac Users

If you run Linux or Mac OSX at home (Mac's terminal is surprisingly similar!), you can open up a terminal and type in the following:
`ssh abc123@linux.cosc.canterbury.ac.nz`

Where 'abc123' is replaced with your user code.

You'll then be prompted to enter your password - this is the same as you would use to log onto the uni computers in person. It should look a little something like this:



```
Last login: Wed Mar  9 13:39:11 on ttys000
[Emilys-MacBook-Pro:~ emily$ ssh erp35@linux.cosc.canterbury.ac.nz
Warning: Permanently added the ECDSA host key for IP address '132.181.16.30' to
the list of known hosts.
[erp35@linux.cosc.canterbury.ac.nz's password:
Welcome to Linux Mint 17.3 Rosa (GNU/Linux 3.19.0-32-generic x86_64)

Welcome to Linux Mint
 * Documentation:  http://www.linuxmint.com
[[erp35@cssecs2 ~]$ ls
```

You may also need to type "yes" during the connection process to give your permission. To conclude your SSH section, just type exit.

Windows Users

Using SSH on Windows is a bit more complicated, as it doesn't have a terminal similar to Linux. Instead, you'll need to install a program to help you out. One that we recommend is PuTTY. For any given SSH client for windows, you'll be able to find an in-depth tutorial somewhere online. But the basic info you will need is as follows:

- Host name: linux.cosc.canterbury.ac.nz
- Port number : 22
- Your login credentials: usercode and password.

You can also use the built in Windows Remote Desktop client, with the address linux.cosc@canterbury.ac.nz. This will give you the graphical representation of your uni desktop on your home PC. But be warned, it's pretty slow.