Linux & Vim Guide

1 The Linux environment

- You will be using the terminal for almost everything at first. This is vital to learn, as there will be times when a Linux terminal is the only access you have to a system.
- Type commands at the prompt and press 'enter' ↩ to process them. (This may be called 'return' if you are using a Mac.)
- Most commands are of the form: command [option(s)] [filename(s)]
- Don't forget: Commands and filenames are case sensitive!
- The mouse does not work within the terminal – use the arrow keys to move around.
- This may all seem very new and strange, but *practice makes perfect*! And there are always people you can ask for help if you get stuck.

1.1 Starting Linux

If you are in the Linux labs in building 314, you can simply double click on the Terminal icon on the left hand side to begin work. Your work will be stored, but getting access to it is a little tricky outside the building – see Section 3 for a brief guide.

On Windows computers at Curtin, or from home, the standard method is to open a virtual machine:

---

1. Open mydesktop.curtin.edu.au in a web browser.
2. Select "HTML Access", and log in with your student ID and password. (Make sure to change STAFF to STUDENT).
3. Select "Computer Science Linux" (usually on the left).
4. Wait about 30-60 seconds, until all the network drives have been loaded.
5. Double click on the "I: Drive" icon. Press "Cancel" if asked for a password. This stage is very important, as the virtual machine's own storage is destroyed when you log out – so having a permanent place to store your programs and files is essential!
6. Once the "I: Drive" window opens, navigate to the directory you wish to work in. Then, right-click on any empty area in the window and select "Open in Terminal".
7. Congratulations! You made it!

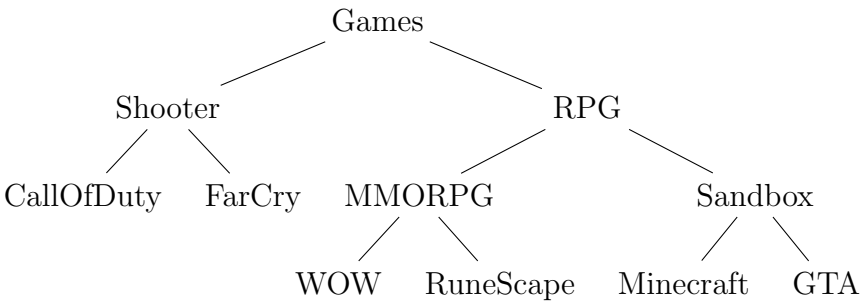
A third way is to create your own virtual machine. This has many advantages – it doesn't destroy itself on logout, you have full access to it as it is yours, and of course, it is much faster as it doesn't have to compete with all the other VMs on Curtin's networks. ComSSA runs workshops on setting up virtual machines twice a year – keep an eye out for announcements!

1.2 A bit about directories

In your Computing units, you will end up creating directories to house your files and data. For example, each week's practical might be stored in a different directory. If you're used to Windows or Mac folders, this is essentially the same thing, and while command-line access may be daunting at first, you'll soon see how powerful it is (and how it will ultimately save you from RSI from constantly clicking the mouse).

Linux directories start at the *root* (/) which is at the top of a very big tree with lots of branches. One such branch is `/home`, which you might recognise from `/home/12345678` in the labs.

Let's use this tree to explore some ideas:



You can go up or down the tree easily:

- To go down: specify the directory, e.g. `cd Shooter`
- To go up: `cd ..` (`..` means "parent")

You can move multiple steps at the same time by using / between steps.

- To go down three steps: `cd RPG/Sandbox/GTA`
- To go up three steps: `cd ../../..`
- To go from WOW to Sandbox: `cd ../../Sandbox` (trace it with your finger to check!)

You can always find out where you are by typing `pwd`. Do this if you're lost – it never hurts.

Now we have covered the *do's*, here are some *don'ts*.

- Let's say you're in Minecraft and want to get to Sandbox. The correct syntax is `cd ..`, not "cd ..Sandbox" or anything similar.
- Never type "cd" all by itself, especially in the virtual machines. If you accidentally do this, just close the terminal window and start a new one by right-clicking in the correct folder and selecting "Open in Terminal" again.
- cd only ever takes one argument. "cd Minecraft GTA" will produce a syntax error.
- Avoid using spaces when naming directories – use underscores (_) instead. This avoids having to use `cd Badly\ Named` or `cd 'Badly Named'` to change to it.

1.3 Useful commands

- **ls** ("list") – This tells you what files are in a directory. `ls -la` shows a long listing of all files in the current directory (including hidden files starting with ".").
- **cd** ("change directory") – Go to a particular directory. For example:

From	To	Command
FOP	FOP/Prac02	`cd Prac02`
FOP	FOP/Prac04/test	`cd Prac04/test`
PDI/Prac02	PDI	`cd ..`
PDI/Prac02	PDI/Prac01	`cd ../Prac01`
UCP/Prac04/test	UCP	`cd ../..`

- **mkdir** ("make directory") – Creates a folder or directory. For example, `mkdir test` would create a directory called `test` under the current directory.
- **pwd** ("print working directory") – Tells you where you are. Helpful when you can't find a file or directory that should be there.
- **cp *from to*** ("copy") – Copies files from one place to another. For example, if you are in Prac04 and want to copy Prac03's readme, type `cp ../Prac03/README .` (the dot at the end means "where I am right now".)
  You can also use this command to duplicate a file - e.g. `cp strings1.py strings2.py`
- **mv *from to*** ("move") – Moves files from one place to another. Much like `cp`, but it does not leave the original behind.
  Can also be used to rename files, e.g. `mv stirngs2.py strings2.py`
- **rm** ("remove") – Deletes files. Use with care!
- **cat** – Displays a file - e.g. `cat strings1.py`. If the file is likely to be long, consider " `cat main.py |more` " which views the file one page at a time, or for a *very* long data file, use `head` (displays first 10 lines) or `tail` (displays last 10 lines).
- **man** ("manual") – Get help on a command. e.g. `man cd` or `man ls`
- **exit** – Closes the terminal gracefully.

There are many others, but these are the ones you're most likely to find useful.

Making archives

zip *nameofzip contents* creates a compressed archive.
`zip Prac1_12345678 *` is quite common in the labs, this means "Put everything in the current directory into this zip file".
If you have subdirectories you want to include, use the -r (recursive) option:
`zip -r PracTest1_12345678 PracTest1` means "Put PracTest1 (a directory under the current directory), and everything under it, into this zip file".

2 VIM tips

In Fundamentals of Programming, you have to use the vim text editor. The three things you need *every* session are:

- **"I"** to start editing
- [***Esc***] to stop editing
- `:wq` [***enter***] ↩ to save and exit.

If you are in edit mode, the bottom left of the screen will read `--INSERT--`. If it is blank (command mode), type **"I"** to edit. If it says `--REPLACE--`, type [***Esc***] then **"I"**. Other useful commands (make sure you are not in edit mode!):

- **"dd"** to delete the current line
- **"G"** to move to the last line of the document
- **"20G"** to move to line 20 of the document
- **":num"** to move to the line number *num*
- **"/*abcd*"** highlights everything matching "abcd". Use N to move to the next item, Ctrl-N to move to the previous.
- **"u"** to undo the last action. Type "u" multiple times to undo more actions.
- **":w"** saves the file without exiting. Type [***Esc***] `:w` then **I** every so often while working on a big program or document.
- **":w *filename*"** saves the file as another filename. This may be helpful if you have totally forgotten which version you're in, and gives you a chance to save it *somewhere* and then figure out later which version should be kept.
- **":q!"** quits immediately *without saving*. This is useful when you didn't make any edits, or made a horrible (or unknown) one by accident.

If you accidentally hit Ctrl-Z out of habit and end up at a Linux prompt, type `fg` to get back into vim.

Advanced tip: cut/copy and paste

- Enter **"V"** to enter visual mode
- Use the arrow keys to select the text you want to cut or copy
- Once you're happy, enter **"d"** for cut or **"y"** for copy
- Then go to where you want to put it and enter **"p"** (for paste)
- If this process has added or removed a space character, go into insert mode (**"I"**) and adjust accordingly. This is especially important in Python due to its insistence on consistent indenting.

Advanced tip: fixing the swap file issue

If vim crashes or closes ungracefully while closing your file, it will create a swap file, and will hassle you when you next go to edit it. If this happens to you:

- When prompted, type **"R"** to recover the file
- If vim gives you more than one recovered file, select the one with the most likely last modified time, and enter its number (e.g. "1").
- It will usually tell you it has recovered the file, press 'enter' ↩
- After saving the file and exiting, type `rm .yourfile` then [***Tab***], where "yourfile" is the name of your file, to remove the swap file.

3 Accessing your files from outside Curtin

On occasion, you may wish to copy your files from Curtin machines to your home machine to work on – for example, while doing an assignment. There are several ways of doing this:

Removable media: The 314 Linux machines do not support flash drives or external HDDs, but if you are working mainly from the I: drive, you can copy your files onto such media from any Windows machine on campus. The Abacus labs in buildings 303, 408 and 501 are accessible at all times.

OneDrive: Curtin has created a OneDrive for each student. While working from a Curtin lab machine or MyDesktop, you can back up your work to OneDrive by using drag and drop, and then retrieve it later.

VPN: You can install VPN software to access Curtin's internal systems, and then use secure methods to copy the files. Instructions for this are provided by DTS for Windows and Mac computers. Once inside the network, you can use WinSCP or Cyberduck to copy files to or from Curtin.

References

1. Barrett, D. J. (2016). *Linux Pocket Guide*. O'Reilly Media. ISBN 978-1-49192-757-1.
2. Linuxize. (2020, October). How to Copy, Cut and Paste in Vim / Vi. `https://linuxize.com/post/how-to-copy-cut-paste-in-vim/`
3. McMeekin, D. (2023). Practical 1 [PDF document]. In *Programming Design and Implementation*. Curtin University.
4. Maxville, V. (2023). Lecture 1 [PDF slides] and Practical 3 worksheet. In *Fundamentals of Programming*. Curtin University.
5. Naushad, A. (2018, March 21). Basic Linux commands for beginners. *Maker Pro*. `https://maker.pro/linux/tutorial/basic-linux-commands-for-beginners`
6. Shotts, W. (2019). *The Linux command line: a complete introduction* (2nd ed.). No Starch Press. ISBN 978-1-59327-952-3.