**02 – Unix and Shell Commands**

**Activities**

COMP190 – Tools and Techniques for Software Development

Dickinson College

Fall 2020

Prof. Grant Braught

While Unix/Linux usually comes with a nice graphical user interface (GUI) that looks similar to the Mac or Windows desktops, it is very common to interact with Linux through a *command line interface* (CLI). As introduced in today’s class the CLI in Linux is the Bash *command shell* that runs within the *Terminal application*. The activities for today will familiarize you with the basics of interacting with the operating system through the Bash shell.

In computing you will frequently need to learn new programs, operating systems, programming languages or other technologies outside of a classroom setting. A common way to do so is to find and work though a tutorial on-line. For any given computing topic there are somewhere around a bazillion tutorials ;) out there. Some good, some okay, some downright terrible. So sometimes it takes a little searching to find one that will work for you. For today, I have already found a pretty good tutorial that introduces the basic use of the Unix/Linux Bash Shell CLI. The activities below will walk you through that tutorial using your LinuxLite installation from the last class.

The activities below follow along with the tutorial and are intended to model the types of things you might later do on your own when following other tutorials. This naturally includes trying the specific commands shown in the tutorial. But it is even more important to try to use the commands to do things that are just a little different than what is in the tutorial. At a minimum be sure to test out your answers for every question. Sometimes it may take a few tries to get it right and that’s okay. By doing this practice and experimentation, you reinforce your understanding of the commands and generalize your knowledge so that you’ll be better able to use it for new and different tasks as they come up.

**Before We Start:**

1. Ensure that you have completed the 01-OSandVM Activity Sheet and have a working version of LinuxLite in Virtual Box. The activities below ask you to use LinuxLite as you learn about the Bash shell, so you’ll need to have it running.

2. Bash seems like a strange name for a shell or CLI program. I know sometimes I want to bash my computer, but that’s not why it has this name. What does Bash stand for? Why it is called that? Use google to answer this question.

**The Tutorial:**

For learning about the Unix/Linux CLI and the Bash shell I find that Ryan’s Tutorials are quite good. You can find Ryan’s Linux Tutorial here:

* <https://ryanstutorials.net/linuxtutorial/>

Begin by reading the Introduction. Note that we will not be doing all of the sections that are listed in the outline. The activities below will point you to the relevant sections. Answer the following few questions based on the Introduction.

3. The tutorials use a few conventions to indicate parts of commands that you will need to provide or customize. While these are used in this tutorial they are also fairly general conventions that you will find in other places when reading tutorials or documentation.

a. What does it mean when text appears between < >?

b. Using that convention rewrite the following sentence:

This class is <Course>.

c. What does it mean when text appears between [ ]?

d. Using that convention rewrite the following sentence assuming you can choose up to two options:

I want [Cake] [Pie] [Ice Cream] [Cookies] for dessert.

**The Command Line:**

Open the “The Command Line!” section of the tutorial:

* <https://ryanstutorials.net/linuxtutorial/commandline.php>

As you read through the tutorial try out and experiment with the commands that are presented and then answer each of the questions below. The order of these questions matches the order of the material in the tutorial. So an effective way to work will be to read a question, then read the tutorial, trying commands and experimenting until you think you can answer the question. Then answer the question and move on to the next question. I will include the headings from the tutorial sections before the questions that use the material from that section to help you stay on track.

You may also find it helpful to create a reference sheet for yourself as you go. Add each command and a few notes and examples about each. This will give you a compact reference sheet that you can use to quickly refresh your memory on the commands when you need them later.

It is important to note that there will be some small cosmetic differences between the terminal window shown in the tutorials and the one you will see in LinuxLite. For example, the command prompt shown in the tutorials appears as:

**user@bash:**

While the one in LinuxLite will appear similar to the following where instead of comp256, you will see the username you chose when doing your LinuxLite install:



You will quickly learn to ignore these cosmetic differences and see that all of the commands and the results that they produce are the same in LinuxLite as they are in the tutorial.

*Opening a Terminal*

4. Describe two ways to open a Terminal in LinuxLite.

*The Shell, Bash*

5. Is the LinuxLite terminal running the Bash Shell? What command did you use to check?

**Basic Navigation:**

Open the “Basic Navigation” section of the tutorial:

* <https://ryanstutorials.net/linuxtutorial/navigation.php>

*So where are we?*

6. In Unix/Linux many commands are abbreviations or acronyms for a description of what the command does. What does pwd stand for?

7. What is your working directory? What command did you enter in the Terminal to check?

What’s in Our Current Location?

8. What files are in your current working directory? What command did you enter in the terminal to check?

9. Open “User Files” application on the desktop.

a. What directory does this application indicate it is showing you?

b. Compare the files shown in the “User Files” application to those listed in the terminal. What do you notice?

10. Give a command that displays a long form listing of all of the files in the /usr/bin directory.

11. What are the names of the last three files listed in question #10?

*Paths*

12. In Unix/Linux the *root directory* is specified by what character?

13. What are the two types of paths that can be used in Unix/Linux?

14. Which of those types of paths begins with the / character?

15. Assuming you are in your home directory, give four different ls commands that use different path arguments to list the contents of your Desktop directory. Hint: for the fourth one make use of “..”.

*Let’s Move Around a Bit*

16. Give commands that accomplish each of the following tasks:

a. From your home directory, change the working directory to your Desktop directory.

b. From your Desktop directory, give three different commands that will change the working directory back to your home directory.

c. From your home directory, give two different commands that will change the working directory to the root directory.

d. Which two of the three commands from b will change you back to your home directory from the root directory? Why doesn’t the third one work?

**More About Files:**

Open the “More About Files” section of the tutorial:

* <https://ryanstutorials.net/linuxtutorial/aboutfiles.php>

17. What happens if you run the following command from your home directory? Why?

cd desktop

18. Imagine you have a folder within in your home directory that contains a space. For example: My Stuff. Give two different ways that a cd command could be written to change the working directory from your home directory to the My Stuff directory.

19. What are a few of the hidden files in your home directory? What command did you use to find them?

**Manual Pages:**

Open “Manual Pages!” section of the tutorial:

* <https://ryanstutorials.net/linuxtutorial/manual.php>

20. Once you have a man page open, how do you exit the man page?

21. Use the man command to answer the following questions.

a. What does the ls command do when issued with the flags -alh? (i.e. if you enter the command ls -alh).

b. How does the -t flag modify the behavior of the ls command?

The information displayed by the man command may seem obtuse and confusing, but if you give it time you will find that sometimes it is very useful. Some of the advantages of the man command are that it provides the definitive reference for each command, it is very comprehensive, it is available on all Unix/Linux machines, and it can be used without network access. Sometimes though, if you know exactly what you want to do it will be easier to just use google. For example, try googling: ls sort by modified.

**File Manipulation:**

Open “File Manipulation” section of the tutorial:

* <https://ryanstutorials.net/linuxtutorial/filemanipulation.php>

*Making a Directory:*

22. With your home directory as the working directory, create new directories named Test1, Test2, Test3, Test4 in your home directory. What commands did you use?

As you work through the exercises in this section you should take a moment following each answer to use ls and/or the “User Files” application to confirm that your commands are working as expected. Also, keep in mind that ls and “User Files” are two different ways of looking at the same thing.

Also, note that as you work through the rest of these activities the terms directory and folder mean the same thing and will be used interchangeably. Directory was the original name before graphical user interfaces (GUIs) came along. When GUIs came along directories were visualized on the screen like folders. This led to the common usage of the term folder to mean the same thing as directory.

23. With your home directory as the working directory:

a. Create a folder named Homework inside of the Test1 directory using a relative path. What command did you use?

b. Create directory named Project inside of the Test3 folder using an absolute path. What command did you use?

24. With Test2 as the working directory (use cd to change there), create a directory named Essay in the Test1 directory. What command did you use?

*Removing a Directory*

25. With your home directory as the working directory, complete each of the following tasks:

a. Delete the Test4 directory. What command did you use?

b. Try to delete the Test3 folder. What command did you use? What happens? Why?

c. Delete the Project directory that is in the Test3 folder. What command did you use?

d. Now delete the Test3 folder. What happens? Why?

*Creating a Blank File*

Creating blank files has some uses (e.g. to use the existence of a file as a signal that something has happened). But for our purpose here it just gives us a way to create some files that we can then use for practice in copying, moving, renaming and deleting them. As mentioned in the tutorial, we will get into putting content into files later.

26. With your home directory as the working directory, complete each of the following tasks:

a. Create empty files names File1 and File2 in your home directory. What commands did you use?

b. Create an empty file named File3 in the Test1 directory. What command did you use?

c. Create an empty file named File4 in the Homework directory that is inside the Test1 directory. What command did you use?

*Copying a File or a Directory*

27. With your home directory as the working directory, complete each of the following tasks:

a. Copy File1 into a new file in your home directory named File5. What command did you use?

b. Copy File2 from your home directory into a file named File6 in the Essay directory that is within the Test2 directory. What command did you use?

c. Copy the entire Test1 directory to a directory named Test5. What command did you use?

28. With Test1 as the working directory (use cd to change there), make a copy of File6 from the Essay directory into a file named File7 in the Test2 directory.

*Moving a File or a Directory*

29. With your home directory as the working directory complete each of the following tasks:

a. Move File5 from your home directory into the Test1 directory. What command did you use?

b. Move the Test5 directory to a directory into the Test2 directory. What command did you use?

c. Move File1 in your home directory to NewFile1 in your home directory. What command did you use?

It is worth mention that there is no rename command in Unix/Linux. But if you think about it, what you just did in part c above was to rename File1 to be NewFile1.

*Removing a File (and non empty Directories)*

30. With your home directory as the working directory complete each of the following tasks:

a. Delete NewFile1. What command did you use?

b. Delete the Test5 directory and all of its contents with a single command. What command did you use?

**Running & Managing Programs:**

At this point we will move away from Ryan’s tutorial and explore how programs are run and managed from the command line in a Unix/Linux environment. The materials for this part of the activities are self-contained here in this document.

*Running a Program*

To run a program from the command line in Unix/Linux you simply type its name. In fact many of the commands we have been using (e.g. ls, cp, mv, rm) are actually small programs that are run when you enter their name at the command prompt.

31. You can see the full path to any program you are running at the command line by using the which command. What is the absolute path to the ls program? (Hint: You can use man to learn about the which command, then use it to find the path to ls).

32. Explain in a sentence what the command which which tells us?

Notice that each time you have run a program (e.g. ls, pwd) the results appear in the Terminal and the command prompt returns so that you can enter additional commands. In each case the command prompt returns because the program has finished running. That is, the prompt comes back when for example, ls has finished printing the directory contents and exited. Not all programs behave this way, for example some programs appear in a window and run for a long time.

33. Run the xclock program by entering its name on the command line.

a. What happens?

b. Does the command prompt return right away?

c. Does the command prompt return when you close the clock window?

*Terminating Programs:*

We were able to terminate the xclock program using the controls on its window. However, not all programs run in a window. Further, some programs might enter an infinite loop or otherwise get stuck running forever. There are several ways we can terminate these types of programs.

34. Run the xeyes program from the command line. Notice that the command prompt does not return and xeyes is not running within a window, so there are no window controls to use to terminate it. Press the key combination ctrl-c (or control-c). That is, hold down the ctrl (or control) key and press the c key. What happens?

35. Run the xeyes program again. This time instead of using ctrl-c to terminate the program open another terminal window and complete the following steps in that window:

a. Run the command: ps -u

This command lists all of the processes (i.e. running programs) that were started by you. You should see four processes listed, the two bash shells (one or each Terminal), one for the ps -u command that you just ran and one for the xeyes program. Each of these processes has a numeric *PID (Process IDentifier)*.

What is the PID of the xeyes process?

b. The Unix/Linux command kill can be used to terminate a process using its PID. The format of this command is: kill <PID>

Terminate the xeyes process using kill. What is the full command you used?

It is worth note that sometimes programs are particularly hung up in nasty ways and do not respond to the simple kill command. In these cases, you can send an additional signal using kill asking the OS to use extreme measures to terminate the process. The format of this command is kill -s SIGKILL <PID>. You should always try a simple kill command before resorting to the SIGKILL form.

*Background Processes:*

When launching a program that will run for a long time you may actually want to have the command prompt return so that you can do other tasks or so that you can launch other long running programs. To do so, you need to launch the program as a *background process*. This is done by appending an & to the end of the command.

36. Run both the xclock and xeyes programs as background processes. What commands did you use?

You can easily terminate the xclock process because it has window controls. For background processes that do not have window controls you can use the kill command in another Terminal as above. Or without using another Terminal you can bring the process back to the foreground using the fg command and then use ctrl-c to terminate it. I encourage you to try both techniques with the xeyes process.