Lecture 3

- Unix in a Nut-Shell
- Basic Unix Command

Qingyang "Young" Xu

April 7, 2015

Questions on the **Final Project**?

What we've learned so far…

- Basic Python Syntax
  - assigning variables
  - control statements (if, while, for)
  - defining functions

- Advanced Python Programming
  - lists, dictionaries, tuples
  - built-in functions
  - importing modules

Learning goals for today!

- After this lecture, you will…
- Understand how Unix works…
  - kernel and shell
  - file systems (files and directories)
  - basic commands

- Work with Unix…
  - navigating directories
  - create / remove files and directories
  - edit files with text editor

Unix in a Nut-Shell

- Use your computer like a pro!

What is Unix?

- What is an operating system (kernel)?



  - intermediary between human and machine
  - human – software – hardware
  - to find out more, look into CS 107, 110
- User communicates with the kernel through a program known as a shell.

Shells – your new best friend

- The interface (medium of communication) between the user and Unix OS.
- Basically an interpreter of the user's command.
- Shells…

  - interprets the command from the user
  - executes the command based on input
  - displays the command's output (if any)

- It is common to use multiple shells for multitasking in Unix.

File system in Unix

- All data in UNIX is organized into files.
- All files are organized into directories (which is equivalent to folders in Windows)
- All directories are organized into a tree-like structure called the file system.

Commands

- Commands – the order given by the user in the form of strings separated by white space.
- Almost all commands are imperative and are abbreviations of English words.
- Some commands take options, depending on the context, which specifies the details of the action when necessary.
- To look for a complete specification of built-in commands and their options, here is the *meta-command*:

man *command_name*

Navigating the file system

Command	Example	Description
pwd	**pwd**	Print working directory
cd	**cd** *directory*	go to physics91/
	cd ..	go to the ancestor directory
	cd -	go to the most recent directory
	cd ~	go to home directory

Navigating the file system

Command	Example	Description
ls	**ls**	List all files and directories in the current directory
	ls *directory*	List all files and directories in the specified directory This does NOT change the current directory
cat	**cat** *filename*	Display file content

File management

Command	Example	Description
touch	**touch** *filename* touch ~/filex	Create file without editing the content
cp	**cp** *file1 file2*	Copy the first file into the other
mv	**mv** *file1 file2*	Move the first file into the other; Commonly used for renaming.
rm	**rm** *filename*	Removes the file

Directory management

Command	Example	Description
mkdir	**mkdir** *dirname*	Create new directory
	mkdir *dirname1, dirname2, ...*	Create multiple new directories
rmdir	**rmdir** *dirname*	Remove the directory only if it is empty
	rmdir *dirname1, dirname2, ...*	Remove multiple directories

Foreground vs. Background

Command	Example	Description
&	command &	run the command in background
bg	ctrl + Z bg	suspend the current program and then run it in background
xterm	xterm &	start an extra terminal and run it in the background

- To run a program which prints many things, xterm is usually preferred over bg (e.g. uranium238.vi)

Additional Unix tricks…

- Enter the beginning of the file/directory name and click <TAB> for Unix to match the right filename.
- Hit the <up> button to refer to the previous command (useful when entering complicated commands or repeating the same one)
- Use * to sort and list filenames which share the same part of their name / type. (e.g. list *.txt)
- Use Ctrl + C to interrupt a running program.
- Use jobs to see all suspended jobs and use kill %n (with option -9) to kill the n-th suspended job.

How to edit files in Unix?

- There are many text editors: emacs, vim, etc.
- Nothing special... just plain old notebook (i.e. collection of strings).
- So what?

  - emacs offers a bewildering number of keyboard short cuts (which unfortunately you have to get used to)
  - this keeps your hands on the center of the keyboard (more efficient compared to mouse control)
  - emacs is also highly configurable and easy to customize

Basic Emacs Commands

Command	Description
emacs *filename*	1. Edit an existing file 2. Create new file and edit
Ctrl + X Ctrl + S	Save changes to the file.
Ctrl + X 2	Split the current window into two horizontal ones.
Ctrl + X Ctrl + C	Exit the current file.

Basic Emacs Commands

Command	Description
Ctrl + X 0	Close the active window.
Ctrl + X F	Opens the new file, whose name is yet to be specified.
Ctrl + K	"Kill" – cuts the rest of the line following the curser.
Ctrl + Y	"Yank" – pastes the text immediately after the curser.

So what?

- The "Unix Philosophy"
  - a set of simple tools
  - each performs limited, well-defined function
  - unified file system and shell scripting for complex work flow

So the Unix OS is really just…

What are .cshrc files?

- When you first login to a UNIX system, the shell commands contained in the **.cshrc** file are executed, followed by the commands in the **.login** file.

- What goes into **.cshrc** file?
  - all **set** comands, including path setting
  - all **setenv** commands
  - **alias** commands

Demystifying git

- We have used Git to clone and submit lab 2 last week, but what REALLY did we do?

git init milestone1

cp *final_first_crack.ipynb* milestone1/

cd milestone1/

git add *final_first_crack.ipynb*

git commit –m "Hooray! First assignment!"

git remote **add** origin

https://github.com/physics91si/username-milestone1.git

git push origin master

Wrap-up and Review

- We have learned today the following…
  - what is an operating system?
  - what is a shell?
  - how to use common Unix commands to…
    - navigate directories
    - create & remove files
    - execute programs
  - how to edit files with text editor?

Looking forward…

- Next lecture will build on what we learned today and cover…

  - advanced Unix commands (grep & pipe)
  - file I/O and workflow
  - basic shell scripts (ensemble of commands)
  - (much) more to say about git…

grep and pipe

Command	Example	Description
grep	**grep** *pattern*	search a file or files for lines that have a certain pattern
pipe	**command2 | command1**	connects two commands so that the output from one becomes the input of the next

- Can think of the vertical bar as meaning "given that" as in set theory or probability.