Tutorial 6

Python & Bash

What's our goal?

- We'll create a bash script that will leverage the following tools
  - wget - used to download resources online.
  - gunzip - used to unpack a zipped file.
  - Python - once the file we're after has been saved, we'll process the data. In this case, since we're more interested in the use of Bash itself, we'll do something simple like reading the file.

First things first, instantiate your bash script.

- To start a bash script, you'll need to go to the command line and navigate to the directory you want to have your script in.
  - To create a bash script simply use touch NAME_OF_FILE or you can do vim NAME_OF_FILE
    - I'll typically just run vim to start editing.

Setting up your Bash script.

- At the top of your Bash script, you'll need to enter what's called a shebang. This is important to tell the system what the script it is about to run is.
  - #!/bin/bash

This will always be at the top of your Bash scripts.

Now to do something with our script.

- We'll use wget to download some resources online.
  - http://mtarchive.geol.iastate.edu/2018/04/01/mrms/ncep/PrecipRate/
  - The above link is where our data is going to be.
  - The file we'll grab is PrecipRate_00.00_20180401-000000.grib2.gz
- Our wget call will look like the following
  - wget http://mtarchive.geol.iastate.edu/2018/04/01/mrms/ncep/PrecipRate/PrecipRate_00.00_20180401-000000.grib2.gz --directory-prefix /scratch/scholar/YOUR_USERNAME/
  - This will save your file to the scratch directory

Step two: Unzip your file

- The file we downloaded was gzipped file, we'll need to run gunzip to fix this.
  - Your next line should be the following...
  - gunzip -f /scratch/scholar/YOUR_USERNAME/PrecipRate_00.00_20180401-000000.grib2.gz

- Your file is now unzipped and ready to be operated on!

Now to bring in Python

- We haven’t created a python script just yet to do anything with the data but we can go ahead and set it up in Bash.
- You may notice, we aren’t doing much with the file here, I’ll leave that part to you. The point here is to demonstrate the workflow.

Running your Python script in Bash

- Your next line will be just like running a Python script from the terminal.
  - python myScript.py PARAMETER_TO_PASS

- The parameter you choose to pass can be whatever you would like, for our purposes here we'll pass in the filename.

Exit your bash script and make it executable.

- If you're using vim, hit escape and enter ':wq ` to write your file and quit vim.
- Now, back in your terminal you can run the following.
  - Chmod 700 YOUR_BASH_SCRIPT

Two things to note.

- A lot of times in tutorials you'll see chmod +x myScript.sh, +x just refers to various chmod codes (see resources).
- Second, your script can be myScript or myScript.sh. It doesn't matter, I like the .sh ending as an explicit reminder that it's a shell script. It works either way.

Now to make our python script.

- Creating the python script is pretty straightforward. We won't be doing much here.
  - vim myScript.py

- In this file you'll want to add this to the top.
  - import sys
- To get access to your python script we'll be using the sys.argv command.
  - sys.argv[0] will access the first parameter passed to Python.
  - Let's just print it for simplicity sake.

Your turn!

- See if you can automate the download of a week of data.
- We haven’t covered loops for bash but a quick search should show you how it is done. I’ll be around to help.

Hints.

- The file name has a structure, you can take advantage of that.