

part__0

June 4, 2023

1 Part 0

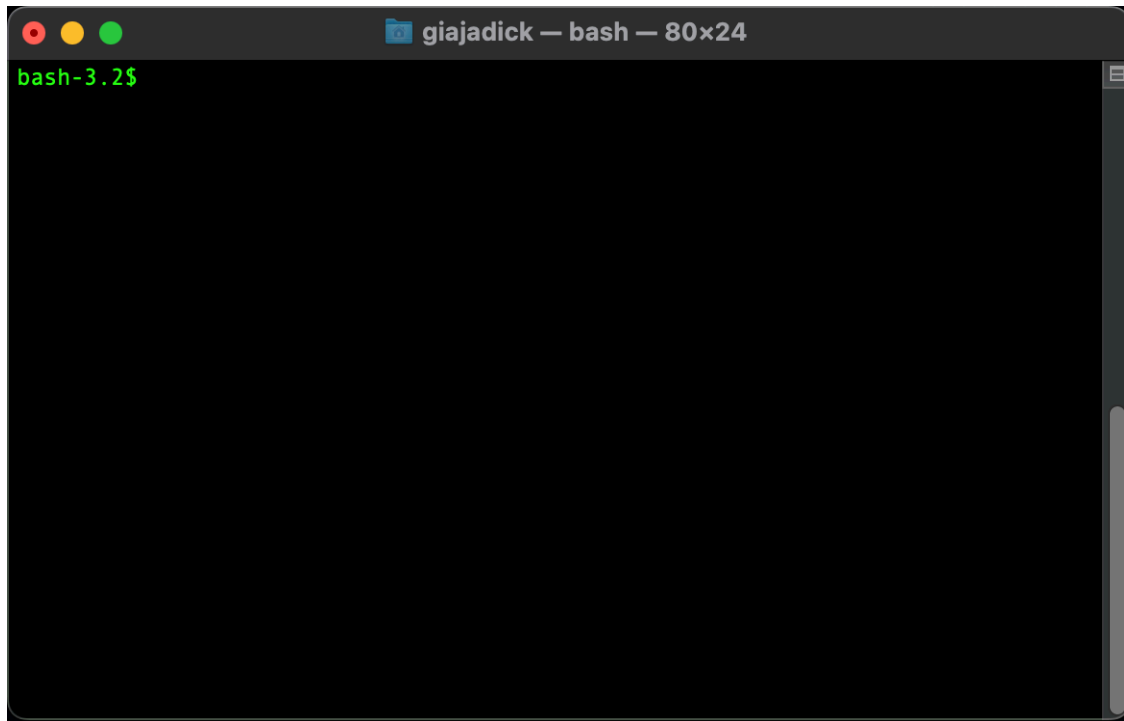
This coding tutorial will be interactive and rely on you creating your own code snippets to get the hang of things as we go along, so before anything, we need some tools.

1.1 Opening terminal

First, we want to have access to a bash (or zsh) shell. This is a direct way of issuing commands to your computer by typing, rather than being forced to work with applications (“graphical user interfaces” or GUIs) that typically have who-knows-what buttons in who-knows-where locations for different people. Getting access to the command line via a shell will help us all get on the same page for our coding lessons.

A terminal is a way of accessing this sort of shell. On Mac, it comes pre-installed, and you can open it as an app by just searching “Terminal” in finder. On my computer, it is in the Applications/Utilities folder. On Windows, search for “cmd”. Then open the “Command Prompt” app. Or maybe this is called “Power Shell”. Unfortunately I am not well-versed in Windows. If you’re on Linux, you probably know how to open a terminal.

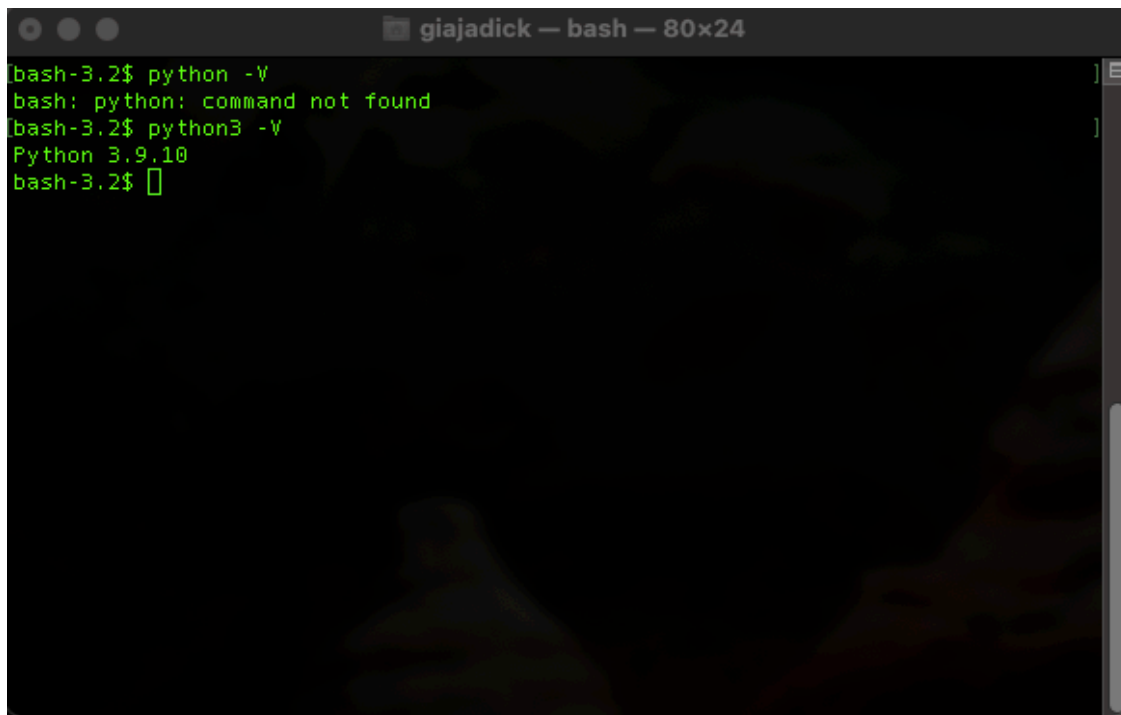
Once you open it, you should see something like this:



1.2 Installing Python3

Typically, I believe Macs come with Python pre-installed but Windows does not. If Python is pre-installed, the version will depend on how old your computer is. We want to have Python 3, specifically.

You can check whether you have Python already installed by typing into your terminal the command `python3 -V`, then hit the “enter” key to execute it. If you get a “command not found” error, try removing the 3: `python -V`. The “V” indicates that you are asking what, if any, version of Python you have already installed. Capitalization, spacing, and spelling must be exact when issuing any terminal command. Here is what that looks like for me:

A terminal window titled 'giajadick — bash — 80x24' with a dark background and green text. The terminal shows the following commands and output:

```
bash-3.2$ python -V
bash: python: command not found
bash-3.2$ python3 -V
Python 3.9.10
bash-3.2$
```

Hopefully one of these print something like “Python 3.x.x”. If you get either an error message or “Python 2.x.x” output, then you can install Python from the website: <https://www.python.org/downloads/>. There should be a big button below the text “Download the latest version”. Follow the prompts there. When you open the installer, you will see three options “Modify”, “Repair”, and “Uninstall”. Click “Modify” to install it and add features. If you have to choose options, go with the defaults.

(sidenote – whenever I say Python, I specifically mean Python 3. This version made some big changes to Python 2, so it is important we are on the same page here. Since some computers have different versions installed, you may need to use the “python3” or “python” commands in the terminal. I recommend starting with “python3”, then trying “python”, and observing the outputs to choose which to use. In my case, shown above, I had to use “python3”).

After installing, you can confirm that the installation worked by running the same commands above. You might need to restart your terminal or reboot your computer.

1.3 Installing packages

We will walk through package details in our first coding lesson, but the gist is that packages are useful code snippets that have been written by other people that will make your life easier.

Pip is the Python package manager, as in you can use it to install extra things to work with in Python. It stands for “Pip Installs Packages”, and it comes with the default Python installation. Depending on your installation setup, you might have to use the command `pip3` or `pip`. This likely (but not always) matches with whether you had to remove or add the “3” when checking your Python version above.

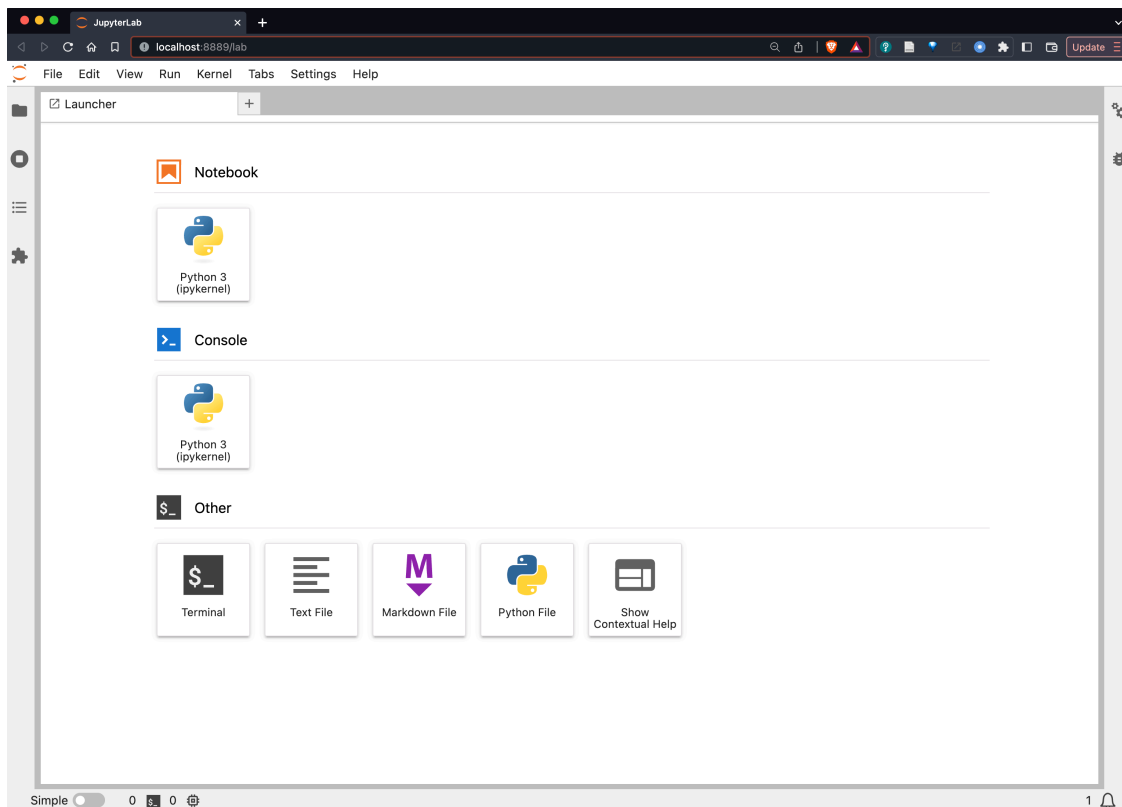
You can check your version in the terminal just like we did with Python: `pip -V`. To make sure it is up-to-date with the latest version, run: `pip install --upgrade pip`

Then, run the command `pip install jupyterlab numpy scipy matplotlib` to install the four packages:

- **JupyterLab**: a nice user interface for writing and executing Python commands.
- **numpy**: useful for efficiently dealing with numbers and datasets.
- **scipy**: useful for scientific analysis and more advanced math functions, for example curve fitting.
- **matplotlib**: useful for data visualization.

1.4 Running Jupyter Lab

Now that you have installed everything, let's confirm that you can open Jupyter Lab. In terminal, run `jupyter lab`. This will open a window in your web browser. Safari, Firefox, and Chrome should definitely work. Hopefully you see something like this:



If it has you select a kernel, choose Python 3.

You'll see some options for different files to create. Before we do that, let's create a folder where you can organize this coding course content. Click on the folder icon in the upper left to open Jupyter Lab's built-in file browser. Navigate by clicking through files, then once you are in the right place, create a new folder by clicking the icon of a folder with a "+" sign in it. Then, navigate into this folder. It should be empty.

To create your first file, you can choose an icon on the right. Let's make a Jupyter Notebook. So,

click the box under the Notebook title. This will open a blank “.ipynb” (or python notebook) file. You can rename it in the file manager on the left, if you want.

And that’s it! We’ll get to typing stuff in these notebooks in the first lesson.

1.5 Troubleshooting

If you get a “WARNING” message, as menacing as it looks, you are probably fine. If you get an “ERROR” message, this probably means something is broken. My first line of defense is to copy-paste the error message into a search engine and see whether I can find a solution. Try this, and if you are still stuck after 15 minutes, please email me with an explanation of what you have already tried, and we will figure it out together (giavanna at uchicago dot edu).

```
[1]: import os
      os.getcwd()
```

```
[1]: '/Users/giajadick/Library/CloudStorage/Box-Box/BoxFiles/UChicago/D&O/EYES/glj-
      intro-to-coding'
```

```
[ ]: '/Users/giajadick/Library/CloudStorage/Box-Box/BoxFiles/UChicago/D&O/EYES/
      ↪glj-intro-to-coding'
```