**Python for Researches**

*A 101 Guide to beginning Python*

Andrew Zheng

02/03/24

Contents

[Introductions 2](#_Toc160247405)

[What is Python exactly? 2](#_Toc160247406)

[Why Python over other languages? 2](#_Toc160247407)

[What is an Editor? 2](#_Toc160247408)

[What are Environments? Or Python Versions? 3](#_Toc160247409)

[L1: Let’s get started with Jupyter! 3](#_Toc160247410)

[Set up Jupyter! What is Jupyter? 3](#_Toc160247411)

[Start with a Jupyter Notebook 4](#_Toc160247412)

[L2: Navigating Jupyter + Python Basics 6](#_Toc160247413)

[Markdowns in vscode + Jupyter basics 7](#_Toc160247414)

[Python Data types 7](#_Toc160247415)

# Introductions

Welcome snake enthusiasts! I heard you were interested in Python! A little about myself, I am a Physics student by training but taught myself how to code and used Python throughout Uni and various projects that I’ve come across. So, by no means am I an expert.

If you have done some Python in the past or heard of its capabilities, this is ideal for you! I have really struggled to find anything useful online specifically for researchers, so here we go!

**Note: This learning series is specific to researches and to enable you to use python quickly and efficiently. This will NOT focus on best practices, optimisation, industry standards etc.**

## What is Python exactly?

Python is a common programming language that allows the programmer to communicate to the CPU to perform desired calculations. Other modern languages you may have heard of are C, C++ (C sharp), Java, JavaScript, R. Out of all these programs, python is in fact the most computationally heavy, meaning it takes the most resources to read each code, however this is negligible for most applications that you would use.

C and C++ are very light weight languages used to perform super heavy matrix computation, they excel in speed and resource efficiency. R is a common programming language used in statistics, many older statisticians still use R but have many challenges like writing axis labels on graphs. Java and Java Script are programming languages that help with visual development, some games, websites, apps are built inside these, also very light weight.

Python has as wide range of packages, from the powerful computation of C (Numpy, Tensorflow, Torch) to GUI development (Tkinter, Streamlit), Web searching (Beautiful Soup) or Data visualisation (matplotlib, pyvis, seaborn).

## Why Python over other languages?

Python is by far the most popular language to use in modern research because it is a very versatile language with lots of capabilities that aren’t often found in other languages. Because many people use python, developers are incentivised to build software shortcuts known as packages or libraries to allow other users to use them, then flex on how popular their library is. This is such a driving force behind python that people have programmed python libraries that secretly wrap on top of other languages just because python runs relatively slowly (Numpy is actually in C, making it fast). Having such a large user base, it makes it a lot easier to seek solutions on StackOverflow (an online forum for programmers).

## What is an Editor?

Just like how you can access everything on the terminal, you can also program python on the terminal. However, it can be very ugly and inconvenient. For larger projects we will use a text editor to help us organise our programs. There are many text editors on the market, most of which are agnostic to any programming language, in this document, I will use **VScode**, no reason in particular, just out of convenience. Many other languages also can be edited in VScode, so, the more people using it, the easier it is to get help. Other popular editors are PyCharm or Atom.

It is not important what editor you use, its mainly an aesthetic choice as it will help with productivity.

## What are Environments? Or Python Versions?

Environments are small units of land in python world, think of them as paddocks in a field. They all run Python (the field) but you might want different buildings (packages) in each field. Some fields might have barns, some might have tractors, and others might have a school, it depends on what you plan on putting in the environment.

*But why do we need environments in general?*

For our case, it isn’t necessary to have separate environments, it is like having a barn, community housing, schools, public transport all in the one field. It can get very busy for your computer to navigate to the right libraries, but we can still manage it since we aren’t running anything too crazy.

Environments are important when running specific codes and programs, some programs might only run on python 3.1 but your python is 3.6. The program might run on a different Numpy version, and getting the latest version might break the older fragile code. In that case you can set an olden day environment to accommodate the specific libraries that was used when the old code was first conceived.

# L1: Let’s get started with Jupyter!

First you will need access to Python 3 (latest version should be okay), that you would need to have installed. If you do not have access, talk to IT. Next, we will need VScode to edit the python files.

For basic installation you will need pip (python installer package), this will allow you to install the funky libraries easily. To install pip go to a command prompt (search cmd on windows search bar) and type in:

**py get-pip.py**

notice the .py, this a python data file and will only activate if you have access to python.

(**ANSTO ALREADY HAS PIP INSTALLED, DON’T BE ALARMED IF THIS DIDN’T WORK)**

## Set up Jupyter! What is Jupyter?

Programming in Jupyter is a way of programming python, just like a text editor. It is a modern form of programming that allows for easier editing. We can discuss why Jupyter is so good a little later, but rest assured that Jupyter will save you lots of time as a beginner.

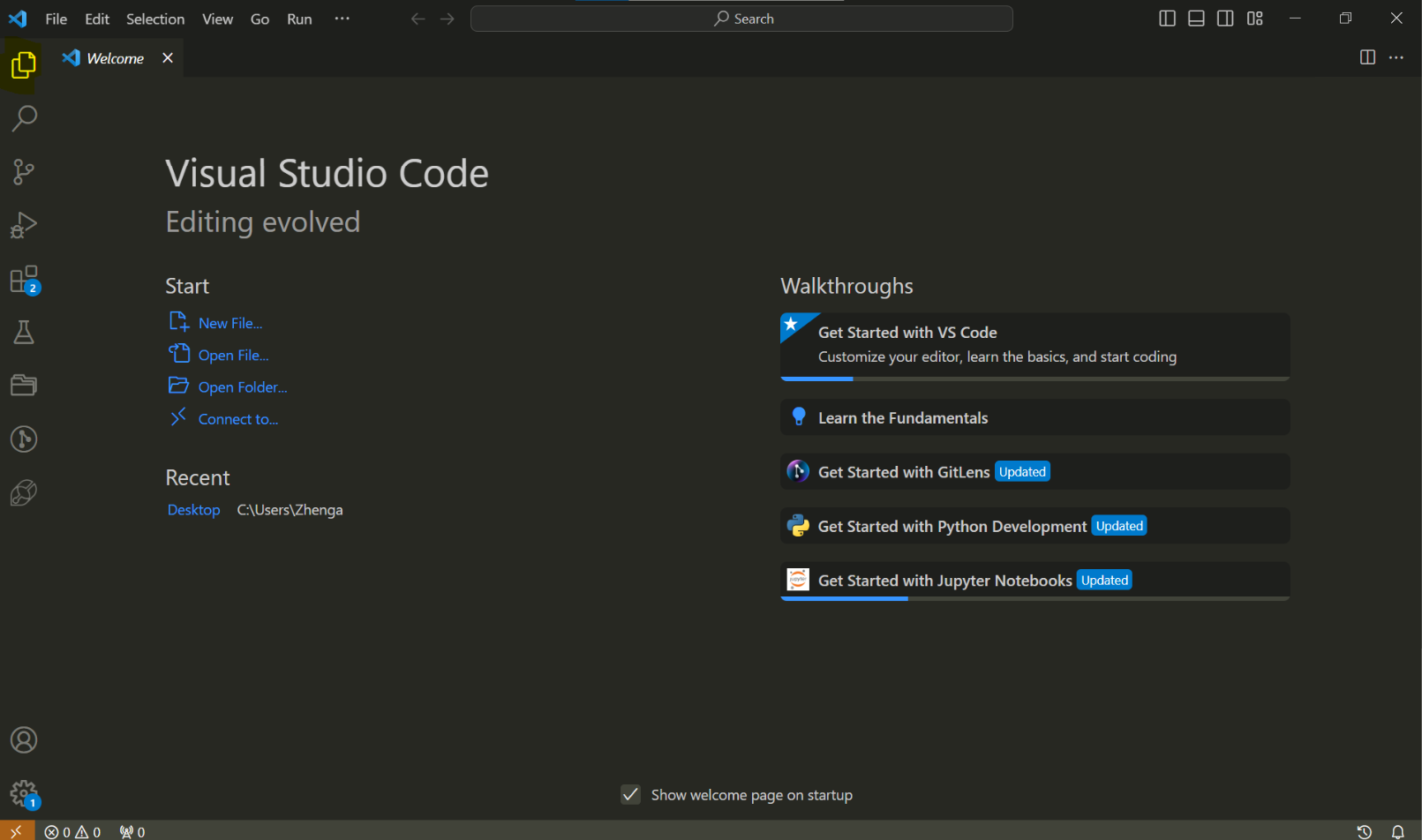
To install Jupyter, we need to open a command prompt (search cmd on windows search bar). Since we have pip, we can copy/call/type this function into the command prompt:

**pip install jupyter**

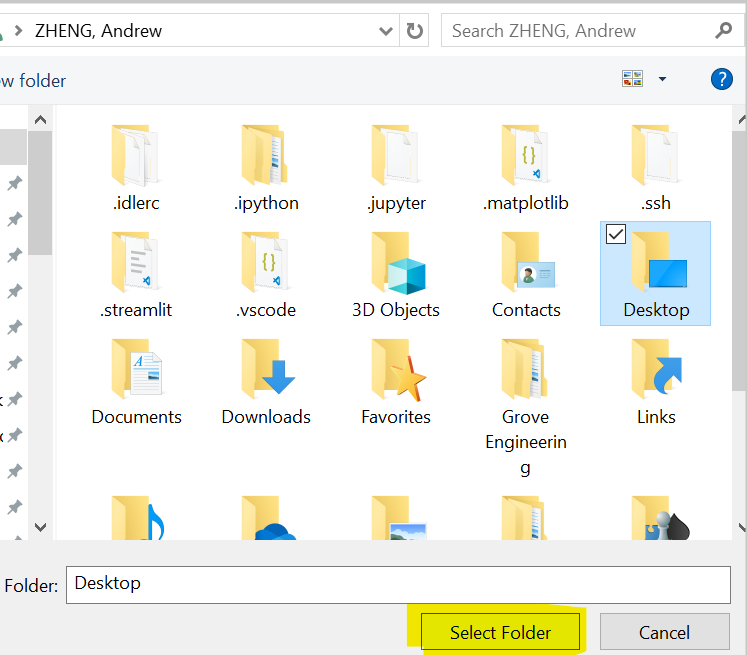
Now we are ready to set up a jupyter notebook!

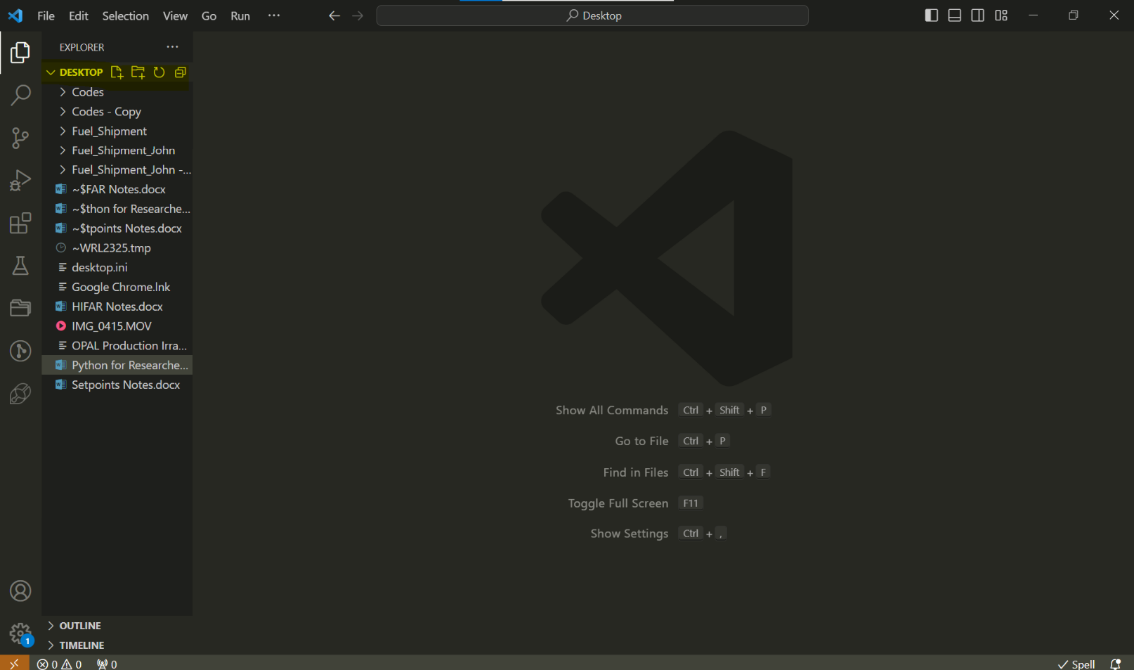
## Start with a Jupyter Notebook

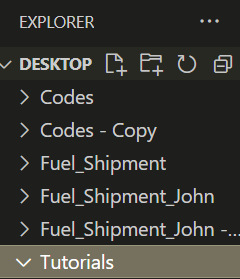
Open up the vscode application, save it to your task bar to make life easy! This is what it should look like (minus a few notifications and stuff)



Click the highlighted box on the top left, this button is setting up a project, you can choose a folder to begin your project in. If you are using a VPN its recommended to not use a folder from the shared drives, because when you unplug your computer you might face saving/connection problems.

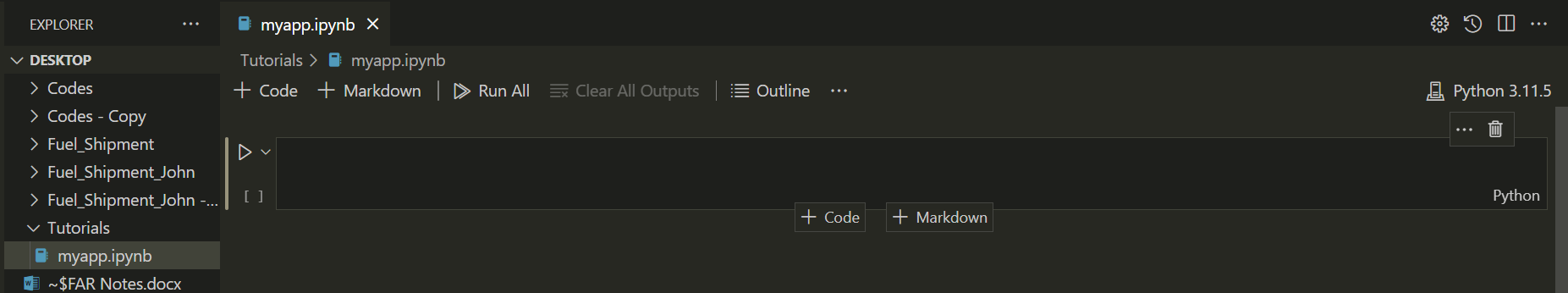
As an example, I will just click **Desktop.**



You have just linked a path to vscode, with these paths, you cannot go to the folder before it, so keep it general! This column behaves exactly like the file explorer function on your computer, you can tunnel into folders, create new folders, and create files. Let’s create a new folder called “**Tutorials**”. Click the second icon in the highlighted yellow.

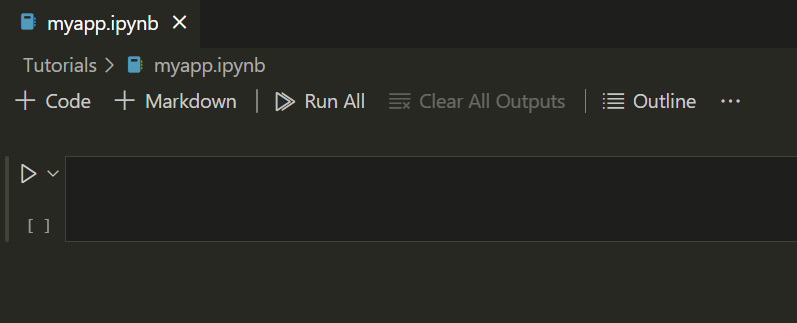


We have just created a folder in our Desktop called “Tutorial” through vscode. Now to create a jupyter notebook inside the tutorials folder, click the tutorials folder first, then click the new file icon. Call this file “**myapp.ipynb**”. For naming, it doesn’t really matter what you name it, so long as it ends in “.ipynb”.

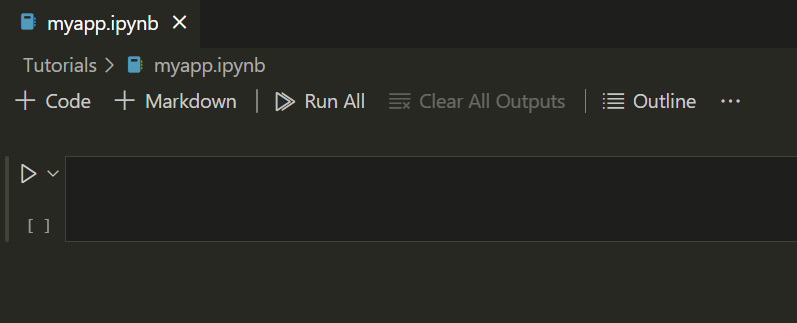
****

This is what it should look like, if you have gotten here, you have successfully set up your jupyter notebook in vscode ready to run python!

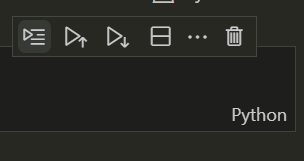
# Navigating Jupyter + Python Basics

Now that you have a jupyter notebook up, lets go through some basics of how to use it.



Highlighted in yellow, you have the tabs column, you can have multiple files open at once. Multiple python scripts to edit, copy code from, sanity check your outputs etc.



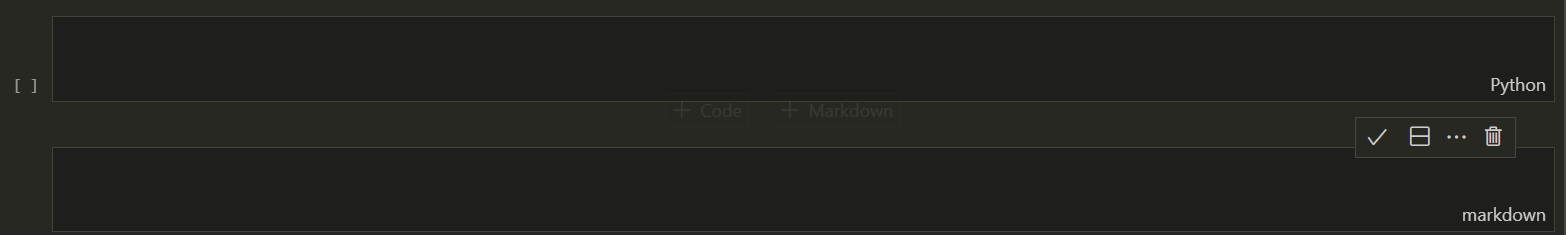
The other 2 buttons we are interested in are the + Code and + Markdown buttons. This creates a code and markdown cell respectively. What is a cell? A cell is a box of text you tell vscode to read, and it will run. A code cell will run python, and a markdown cell are narrations, often when writing, documentation or tutorials, you will use Markdowns.



On the right you will use the trash can icon the most, this will delete the whole cell. The other buttons can help with productivity and are a handy feature of Vscode. Experiment at your own time, but they aren’t important now.

## Markdowns in vscode + Jupyter basics

Just because it’s short, we are going to learn how to edit markdowns.



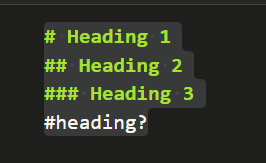


You can tell the difference between a markdown and code cell by the label on the right and the added square box on the code cell. Type the following code into the **markdown cell**.

**# Heading 1**

**## Heading 2**

**### Heading 3**

**#heading?**

Now run the cell, there are 2 ways to run the cell:

* shift + Enter, which runs the cell and goes to the next cell
* ctl + Enter, which runs the cell and stays on the current cell

I mainly use shift + Enter though. To edit the markdown, double click it.

## Python Data types

This may be frustrating, but before beginning python we need to understand some level of basics, or you will get very frustrated whilst you code. I want to start by introducing some data types:

* strings (str): letters/characters, represented with quotation marks. Strings are a common term in programming. There is no bias to the characters, the first letter is the same value as the next unlike numbers. You can use numbers in strings and they will hold the same meaning as another letter.
* Integers (int): a whole number, you can add these together, subtract them, do anything you want with them. They are commonly used when looking up a certain value in table like objects like an excel grid or matrices.
* Float (float): floating point number, or any decimal. Uses more computation, but generally used for calculations, whereas integers are used for indexing.

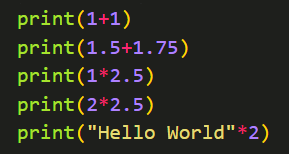
ONE MORE THING before we begin coding, is **how to be use** these data types. A way to show something is the **print** function. It tells the code to print something then and there. Let’s print “Hello World” with this syntax. Remember to use shift + Enter for convenience.

**print("Hello World")**

Strings don’t care about whether you use single or double quotation marks, so long as you use them in pairs.

Let’s go through some examples:

* Integer: **print(1+1)**
* Float: **print(1.5+1.75)**
* Integer/float??: **print(1\*2.5)**
* Integer/float??: **print(2\*2.5)**
* What?: **print("Hello World"\*2)**

You can put them all on the same cell by pressing Enter (Enter doesn’t run the cell, shift + enter does 😊)

This concludes this section of understanding python. Next lesson will be directly on a Jupyter Notebook so you don’t need to copy and paste code on to your python. I will be communicating via markdowns and cell codes.