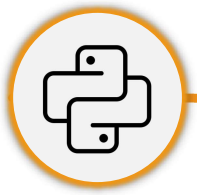
Python & Jupyter Notebook



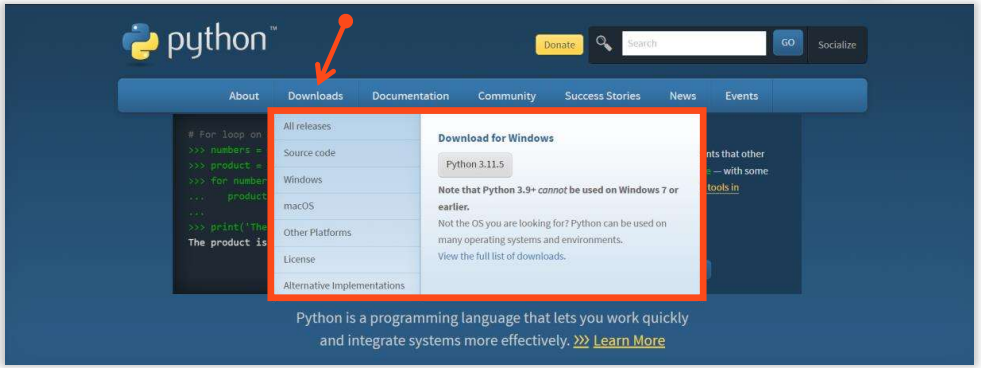
Python & Jupyter Notebook

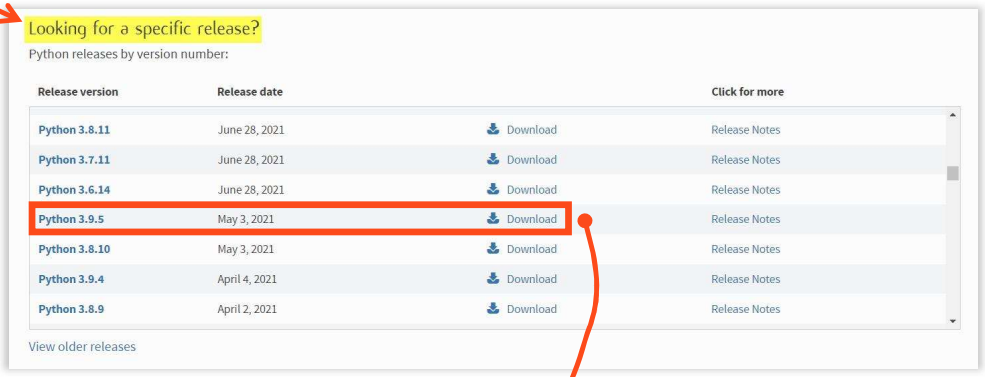
In this section we'll **install** **Python** and after that, **Jupyter Notebook** using **pip & venv** module laid inside of Python. Also we will introduce Jupyter Notebook, a user-friendly coding environment where we'll write our first Python program

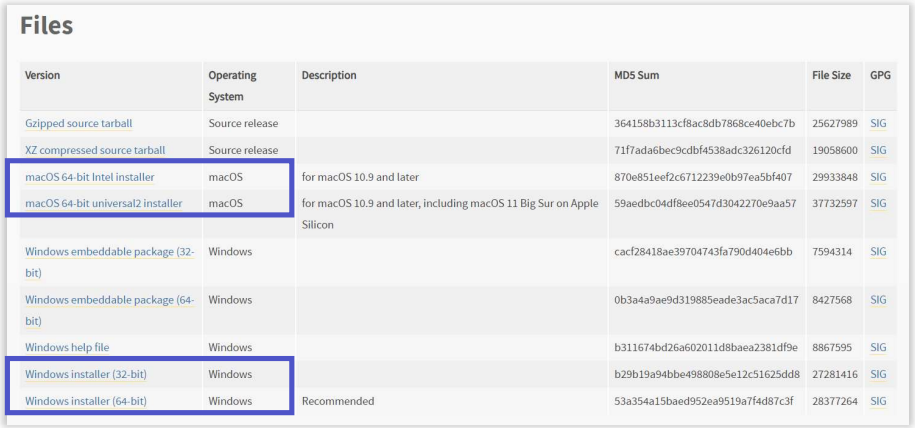
Topics we will cover :	Goal for this section :
Installation & setup	• Installing Python & Jupyter Notebook using CMD (Terminal)
Notebook Interface	• Getting familiar with the Jupyter Notebook environment
Comments and Markdown	• Learning about comments and Markdown in Jupyter environment
The print( ) function	• Writing our first Python program in Jupyter Notebook

Downloading Python IDLE

1. Go to https://python.org
2. Hover over or click on the **Downloads** Section
3. Find **3.9.5** Version (or the version you prefer)
4. Click on the **Download** button







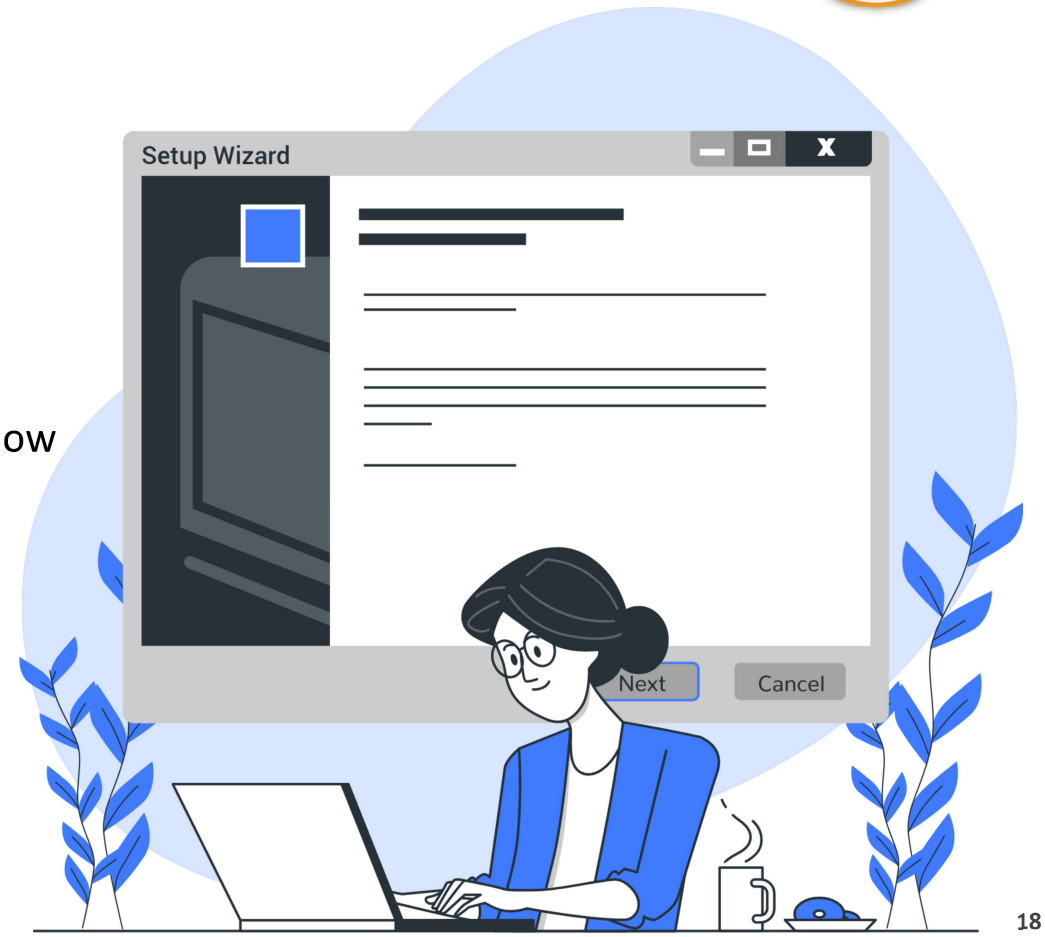
Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		364158b3113cf8ac8db7868ce40ebc7b	25627989	SIG
XZ compressed source tarball	Source release		71f7ada6bec9cdbf4538adc326120cfd	19058600	SIG
macOS 64-bit Intel installer	macOS	for macOS 10.9 and later	870e851eef2c6712239e0b97ea5bf407	29933848	SIG
macOS 64-bit universal2 installer	macOS	for macOS 10.9 and later, including macOS 11 Big Sur on Apple Silicon	59aedbc04df8ee0547d3042270e9aa57	37732597	SIG
Windows embeddable package (32-bit)	Windows		cacf28418ae39704743fa790d404e6bb	7594314	SIG
Windows embeddable package (64-bit)	Windows		0b3a4a9ae9d319885eade3ac5aca7d17	8427568	SIG
Windows help file	Windows		b311674bd26a602011d8baea2381df9e	8867595	SIG
Windows installer (32-bit)	Windows		b29b19a94bbe498808e5e12c51625dd8	27281416	SIG
Windows installer (64-bit)	Windows	Recommended	53a354a15baed952ea9519a7f4d87c3f	28377264	SIG

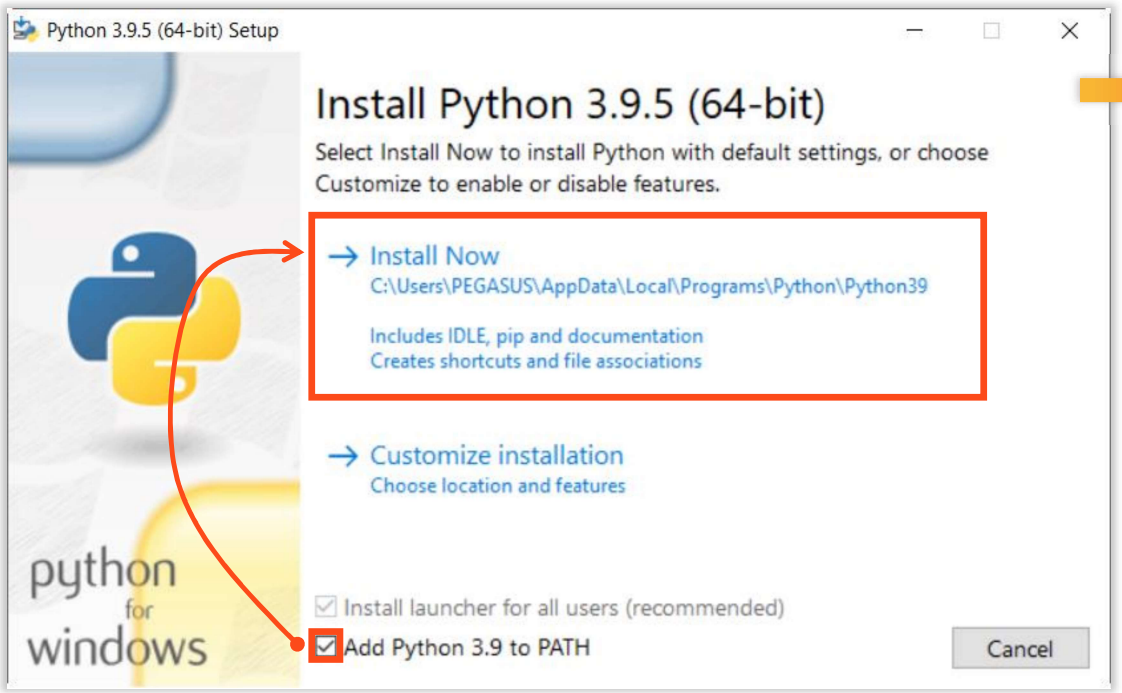
Installing Python

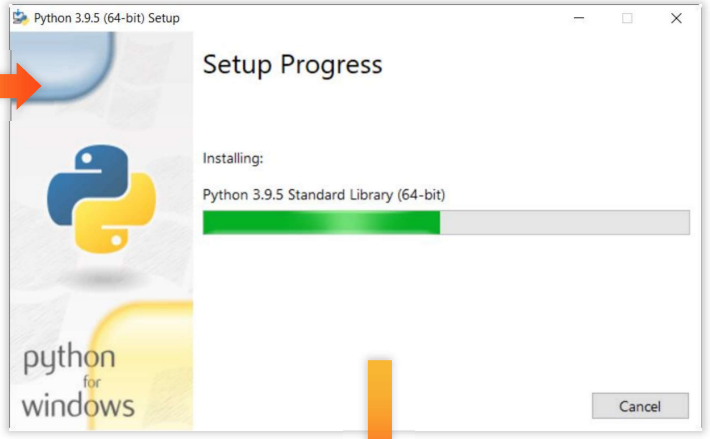
1. Run **python.exe** file
2. Select **Add Python 3.9 to PATH**
3. Click on **Install Now**
4. Wait until you see the **Setup Was Successful** window

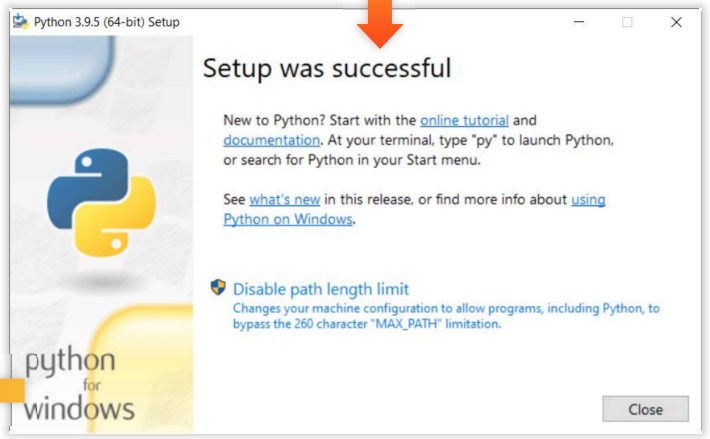
This PC \Downloads \Programs**python-3.9.5-amd64.exe**



Installing Python







Virtual Environment in Python

A virtual environment is a tool that helps to keep dependencies required by different projects separated by creating isolated python virtual environment for them | From geeksforgeeks.com

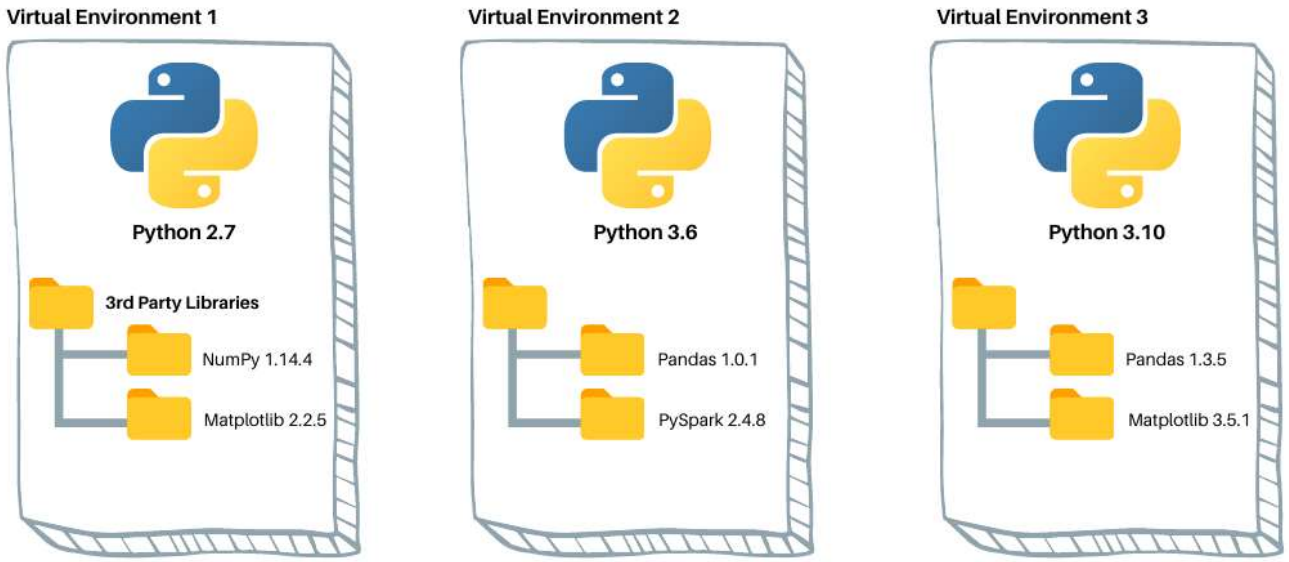
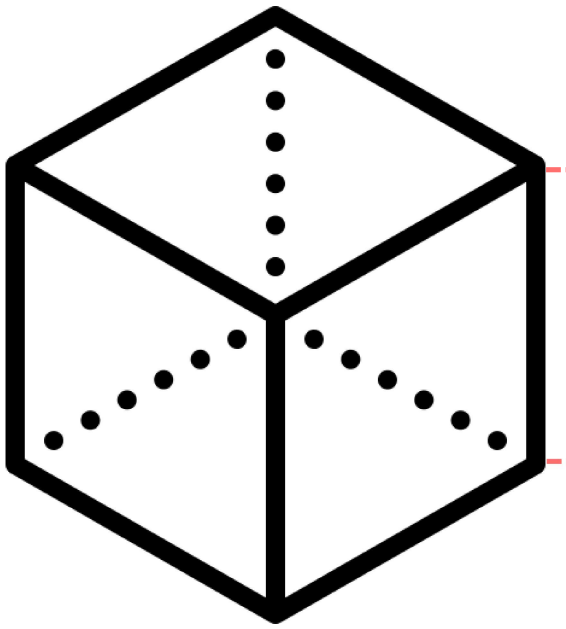


Image copyright to: dataquest.io

Virtual Environment in Python



myenv

Installing libraries

Installing Jupyter

Scripts

activate

myenv\Scripts\activate

ON

Creating **virtual environments** is a way of managing your projects in a proper way.

Installing Jupyter Notebook (Windows)

Command Prompt

```
C:\Users\Ahmad>cd Desktop\my_jupyters
C:\Users\Ahmad\my_jupyters>python -m venv myenv
C:\Users\Ahmad\my_jupyters>myenv\Scripts\activate
(myenv)C:\Users\Ahmad\my_jupyters\pip install jupyter notebook==6.5.6
```

```
(myenv)C:\Users\Ahmad\Desktop\my_jupyters>jupyter notebook
```

Installing Jupyter Notebook (Mac)

```
Last login: Tue Sep 19 18:16:02 on console
ahmad@ahmad-MacBook-Pro ~ % cd Desktop/my_jupyters
ahmad@ahmad-MacBook-Pro my_jupyters % python3 -m venv myenv
ahmad@ahmad-MacBook-Pro my_jupyters % source myenv/bin/activate
(myenv) ahmad@ahmad-MacBook-Pro my_jupyters % pip3 install jupyter notebook==6.5.6
```

```
(myenv) ahmad@ahmad-MacBook-Pro my_jupyters % jupyter notebook
```

Let’s Test What We’ve learned!

- Working with CMD
- Creating Virtual Environment
- Installing Jupyter Notebook

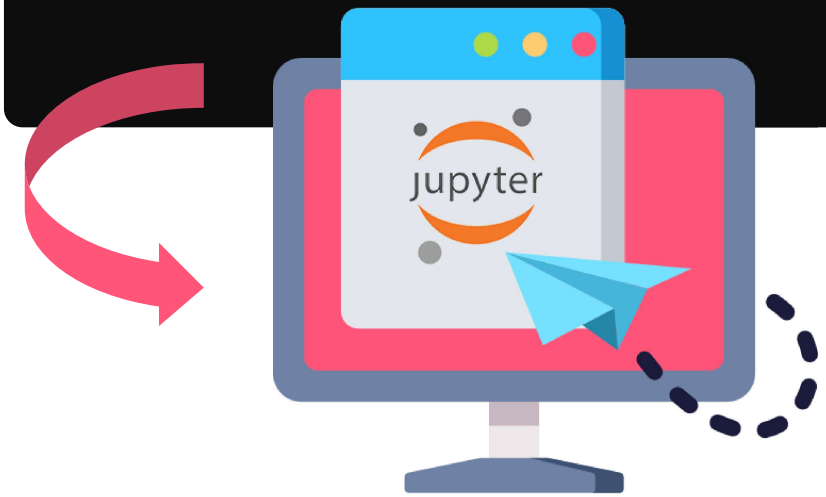
Launching Jupyter

Windows Users

```
C:\Users\Ahmad>cd Desktop\my_jupyters
C:\Users\Ahmad\Desktop\my_jupyters>myenv\Scripts\activate
(myenv)C:\Users\Ahmad\Desktop\my_jupyters>jupyter notebook
```

Mac Users

```
source myenv/bin/activate
```

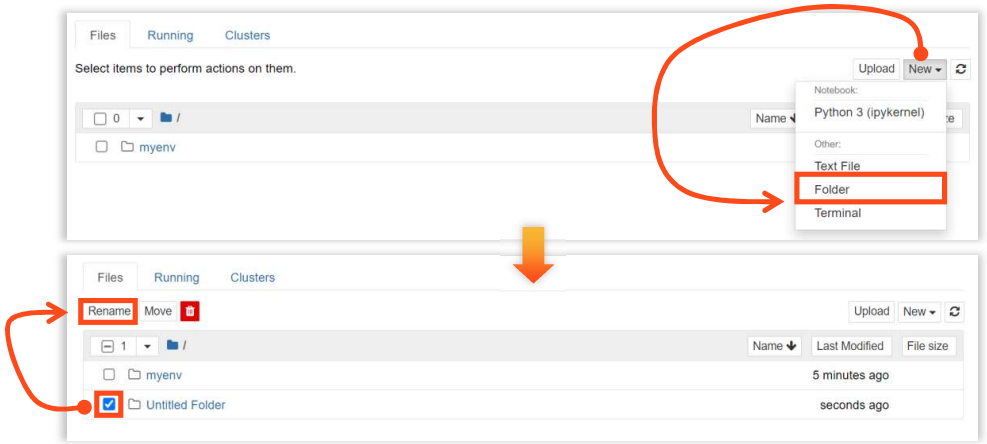


1. **Move to the folder** containing virtual environment
2. **Activate** the environment
3. **Type** jupyter notebook command and press **Enter**

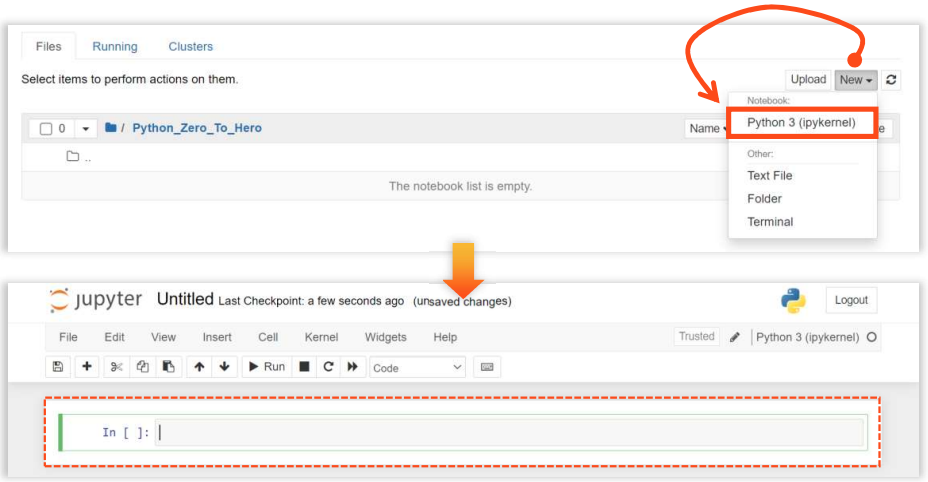
Your First Jupyter Notebook

1. Once the Jupyter Notebook has launched on your browser, **create a folder** to store your notebook

We can **RENAME** the folder by clicking "**Rename**" button in the top left corner



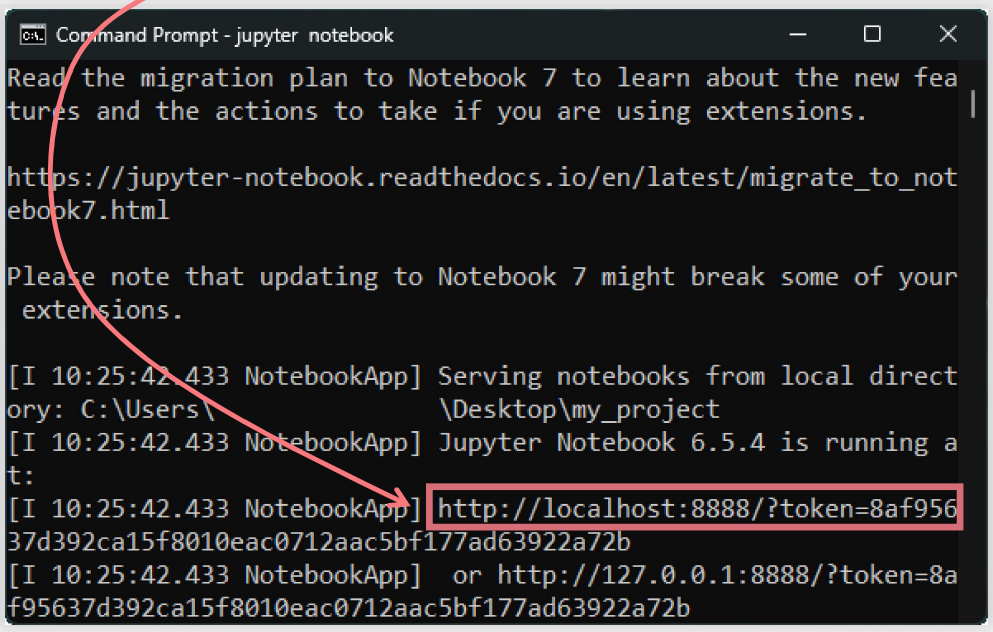
2. Open that folder and create your first Jupyter file using **New** button in the right corner.



Jupyter's Server

IMPORTAN NOTE: When you run Jupyter Notebook, a terminal (weird black box) would pop up as well, this is called **Jupyter Notebook's server**, and it makes your Jupyter's file perform correctly.

Jupyter Notebook's Server



If you close the server window;

Your Notebook will Not Be Working!

Let’s Test What We’ve learned!

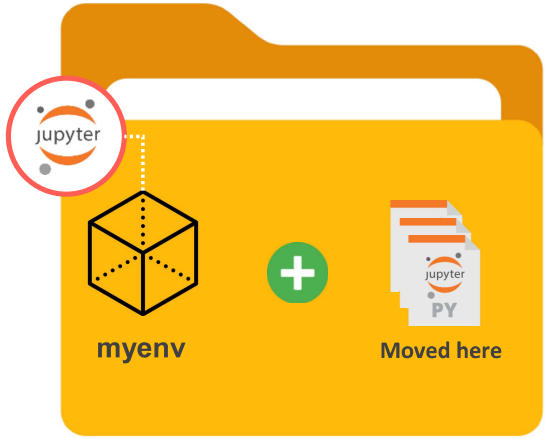
- Launching Jupyter Notebook
- Creating a Folder Inside It
- Creating a Jupyter File

Jupyter_Practice_01.ipynb

How to Open Other Jupyter Files?

To access and open your downloaded jupyter files, simply move them into your folder containing the created virtual environment.

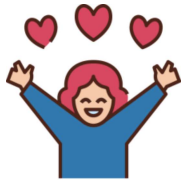
Your Downloaded Jupyter Files



my_jupyters

The Commands

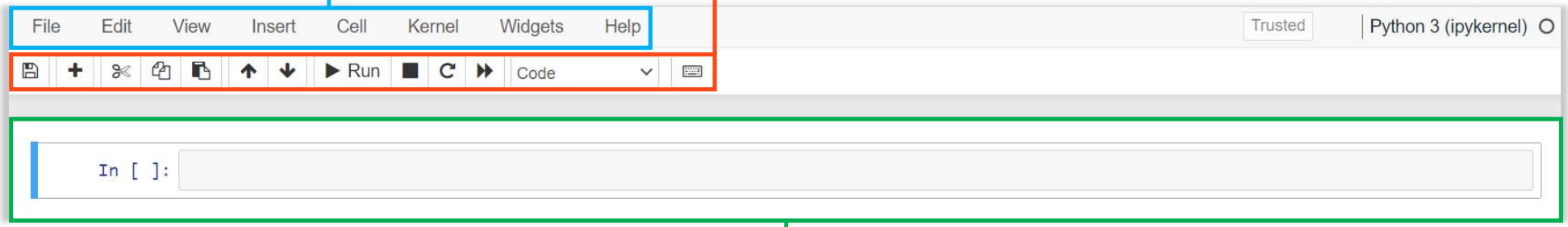
1. cd desktop\my_jupyters
2. myenv\Scripts\activate
3. jupyter notebook



The Notebook Interface

Menu Bar:
You have some options to manipulate the notebook.

Tool Bar:
Buttons for the most-used actions within the notebook!

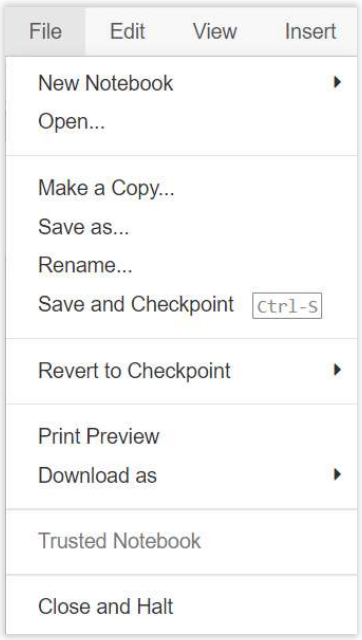


Code Cell:
Input field where you can write your code and run it!

Menu Options

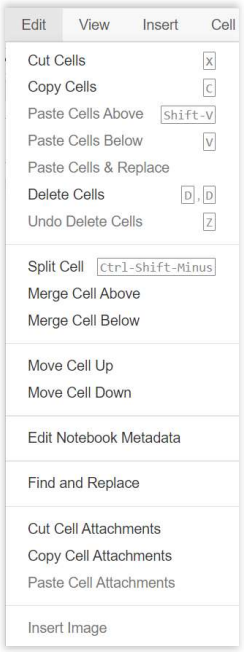
File

Save or **open**, **make a copy**, a **new notebook**, **download** and etc.



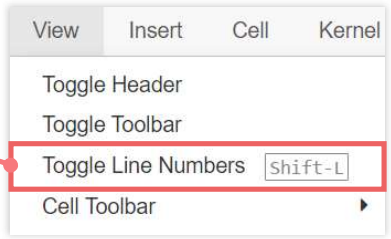
Edit

Edit cells within your notebook



View

Edit the appearance of the notebook



This is a useful one in this menu!

```
print('Hello World!')
print('Welcome To The Course')
```

```
1 print('Hello World!')
2 print('Welcome To The Course')
```

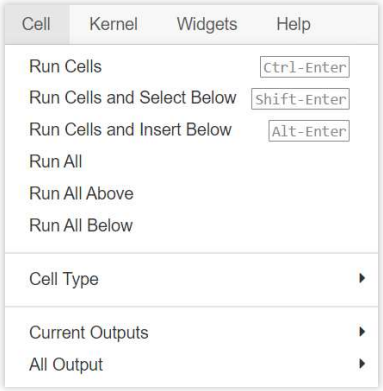
Insert

Insert new cells into your Jupyter notebook

Menu Options

Cell

Access options for running the cells in your notebook

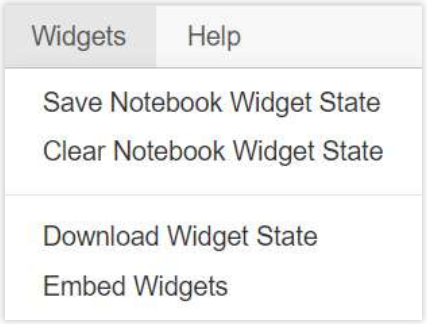


Kernel

Interact with the Kernel (Jupyter core) that runs your code

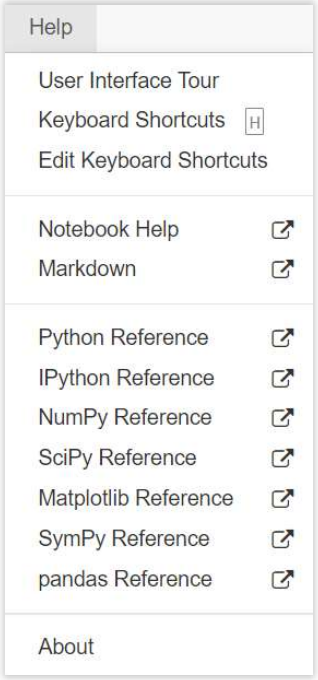
Widgets

Manage interactive elements or "widgets" in your notebook



Help

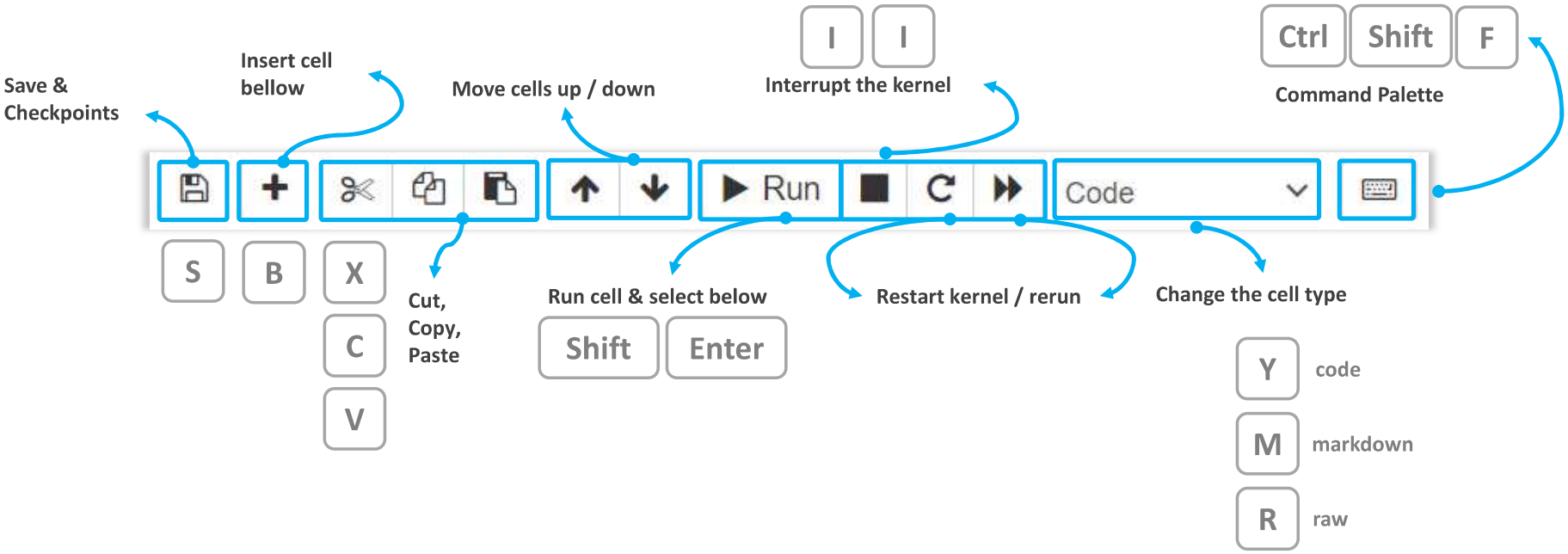
View or edit keyboard shortcuts and access Python reference pages



Toolbar

The **toolbar** provides easy access to the most-used notebook actions

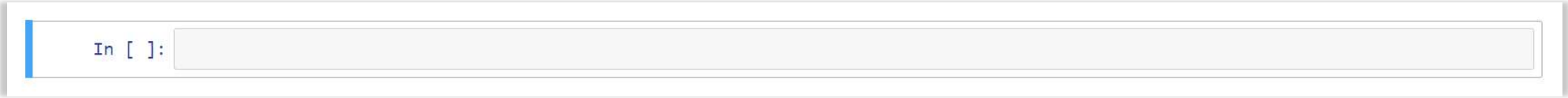
- **These actions can also be performed using hotkeys. (keyboard shortcuts)**



Edit and Command Mode

EDIT MODE is for **editing content within cells** (usually for writing code), and is determined by **green** highlights

COMMAND MODE is **for editing the notebook**, and is determined by **blue** highlights (**no pencil icon**)

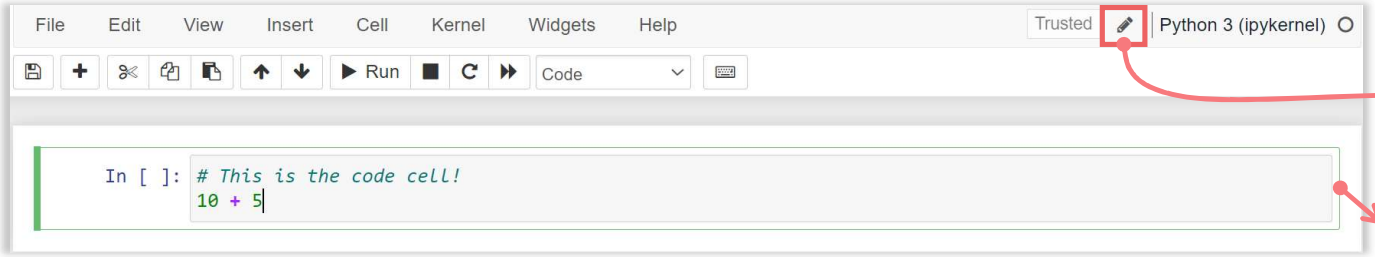


Most of the keywords work when we're in the **COMMAND MODE** (like x, v, m, ...)

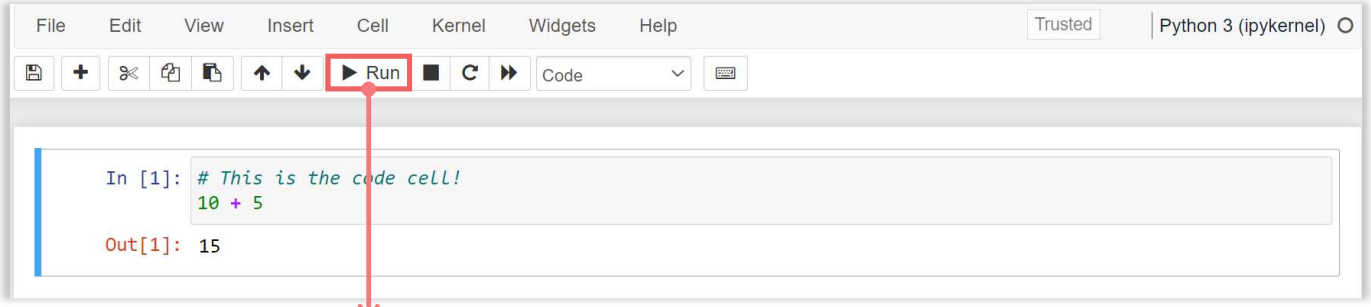
Use "**Left Click**" and "**Esc**" key to switch between **edit** mode & **command** mode!

Code Cell

The **code cell** is where you'll **write** and **execute** your Python code.



In edit mode, the cell color is in **GREEN**, and **pencil icon** is appeared



Write some code, then click **Run** (**Shift + Enter**)

- `In[ ]:` Our code (input)
- `Out[ ]:` Result of our code (output)

- ***Note:*** *Your code* ***may not have output***

Code Cell

The **code cell** is where you'll **write** and **execute** your Python code

```
In[1]:  10 + 5

Out[1]  15
```

Click **SHIFT + ENTER** to **rerun** this cell again!

```
In[2]:  10 + 5

Out[2]  15
```

Note that our output hasn't changed, but the number in the brackets increased from **1** to **2**.

This is a **cell execution counter**, and indicates **how many cells** you've run in the current notebook's session

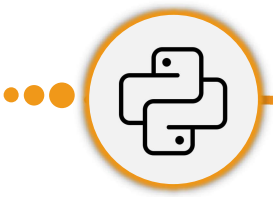
- If the cell is processing, you'll see `In[*]`

Let’s Test What We’ve learned!

- Launching Jupyter Notebook
- Jupyter’s Shortcuts
- Edit & Command Mode
- Code Cell in Action

Jupyter_Practice_02.ipynb

Commenting!



Comments are lines of code that starts with **"#"** (Hash symbol) and **do not run**

- They are great for **other developers who want to read your code**
- They are great for **temporarily stop piece of your code from executing**

1

```
# Let's add 5 to 10 and see the result!
10 + 5
```

15

```
# Let's subtract 5 from 10!
10 - 5
```

5

To **explain** portions of your code

2

```
# Comment out the line below
# 10 - 5
```

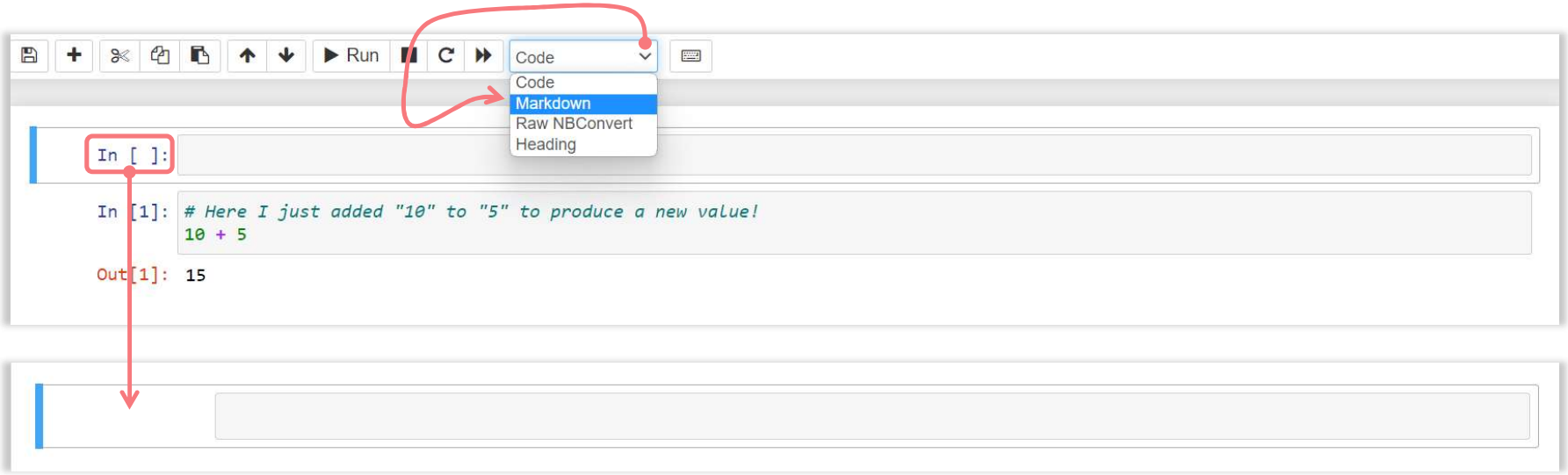
To temporarily **deactivate** portions of your code

Markdown Cells

Markdown cells let you write **nice-formatted text** to explain your code workflow in your Jupyter notebook!

To create a markdown cell:

1. Create a new cell above the top cell (press ***A*** with the cell selected)
2. Select "**Markdown**" from the cell type of menu bar (or press ***M***)



Note that `In [ ]:` has been disappeared.

Markdown Syntax

Markdown cells use a special text formatting syntax.

```
## ✔ Commenting & Markdown

**Comments** are lines of code that starts with **#** (Hash symbol) and *do not run*.

**Markdown** cells let you write your **text passages** to explaing your workflow in your jupyter notebook! (amazing!)
```

```
In [1]: # Here I just added 10 to 5 to produce a new value!
        10 + 5

Out[1]: 15
```

Shift Enter

✔ Commenting & Markdown

Comments are lines of code that starts with **#** (Hash symbol) and *do not run*.

Markdown cells let you write your **text passages** to explaing your workflow in your jupyter notebook! (amazing!)

```
In [1]: # Here I just added 10 to 5 to produce a new value!
        10 + 5

Out[1]: 15
```

Markdown Syntax

All You need to know …

```
# Jupyter Notebook Markdown

## Topic : Markdown Basics

You can write text to demonstrate the workflow of your code in Jupyter environment.

* Extra explanation
* Steps of your work
* etc

To create bulleted lists simply start your line with *.

To create numbered lists, start your line with 1., 2., and so on.
1. Item (1)
2. Item (2)
3. Item (3)

Markdown has a **lot** of capabilities. The essentials to get started with are:

1. Create headers with # ( one is the biggest and six is the smallest header! )
2. **Bold**, *Italic*, ***Bold & Italic***
3. Creating bulleted or numbered lists usinig ( * or 1. at the beginnig of your lines)
4. Code highlighting for example : `my_name = "Ahmad"` (use backtick not ' or ")

To explore more, I highly recommend [this article](https://www.markdownguide.org/basic-syntax/)
```

Creating a link

Markdown Syntax

All You need to know …

Jupyter Notebook Markdown

Topic : Markdown Basics

You can write text to demonstrate the workflow of your code in Jupyter environment.

- Extra explanation
- Steps of your work
- etc

To create bulleted lists simply start your line with *.

To create numbered lists, start your line with 1., 2., and so on.

1. Item (1)
2. Item (2)
3. Item (3)

Markdown has a **lot** of capabilities. The essentials to get started with are:

1. Create headers with # (one is the biggest and six is the smallest header!)
2. **Bold**, *Italic*, ***Bold & Italic***
3. Creating bulleted or numbered lists usinig (* or 1. at the beginnig of your lines)
4. Code highlighting for example : `my_name = "Ahmad"` (use backtick not ' or ")

To explore more, I highly recommend this article

Let’s Test What We’ve learned!

- Launching Jupyter Notebook
- Commenting in Python
- Markdown Cell in Jupyter

Markdown_&_Comment.ipynb

Markdown_Guides.ipynb

The print ( ) function

The **print( )** function will **display** a specified value to the **screen**

```
In[1]: print('Hello World!')
       Hello World!
```

Simply specify the value you want to print out to the screen, **inside the parenthesis**

```
In[2]: print(5, 10 + 5)
       5 15
```

You can print multiple values by separating them with **commas**

Note that, there is no `out[2]:` Why? **print**( ) is one of the few functions doesn't return an output

Congrats! You just wrote your **first Python code** in Jupyter Notebook!

In addition to **print**( ), Python has many **built-in functions** and also it gives you the ability to create your own function (we will see that ☺)

DevTip: Add a **"?"** after a function to access the documentation

```
In [3]: print?
```

```
Signature: print(*args, sep=' ', end='\n'
Docstring:
Prints the values to a stream, or to sys.

sep
  string inserted between values, default
end
  string appended after the last value, d
file
  a file-like object (stream); defaults t
flush
  whether to forcibly flush the stream.
Type:      builtin_function_or_method
```

Let’s Test What We’ve learned!

- Working with **print( )**
- Checking its Docstring

Print_Function_01.ipynb

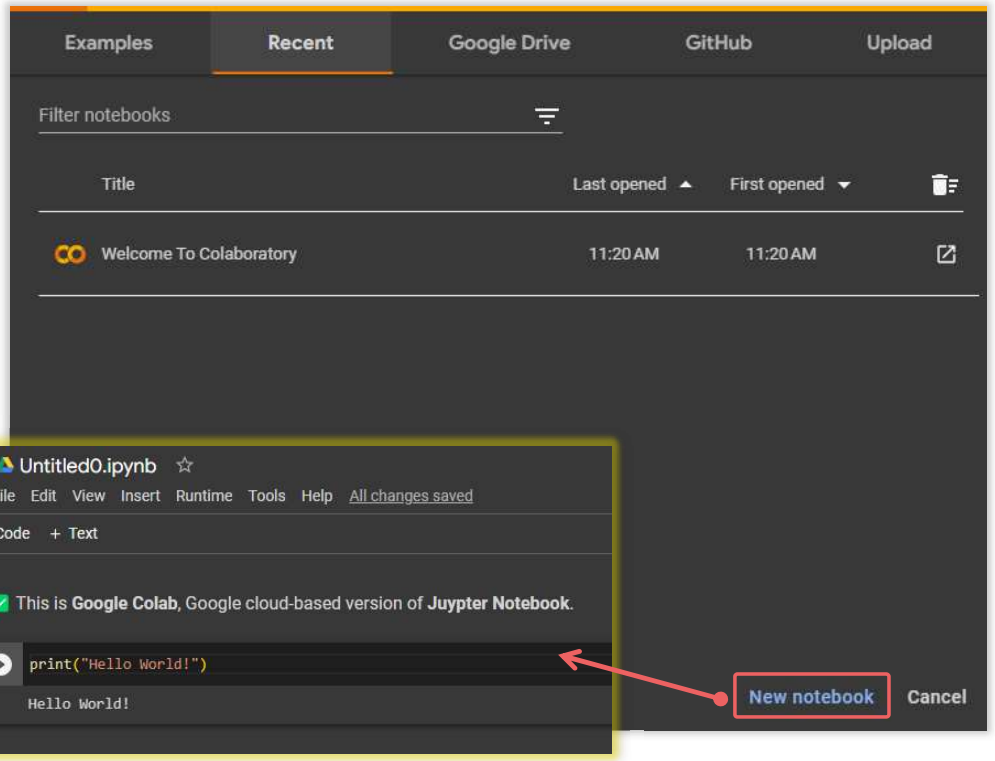
Google Colab (a Powerful Tool)

For the users with less than **4.0 GB RAM**, There is an alternative way to access Jupyter notebook

- **Google Colab** is Google's cloud-based version of Jupyter notebook

To create a Colab notebook:

1. Login to a **Gmail account**
2. Go to **colab.research.google.com**
3. Click "**new notebook**"



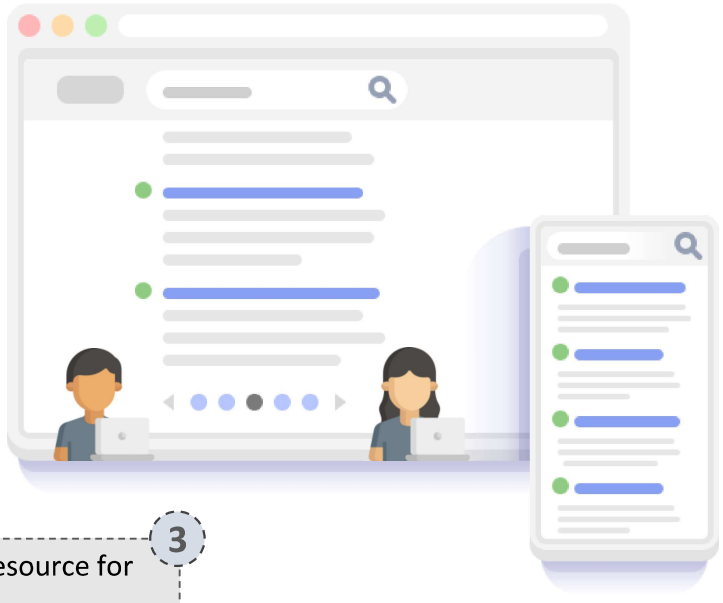
Colab is very similar to **Jupyter Notebooks** (Colab extension is similar to Jupyter Notebook **.ipynb**)

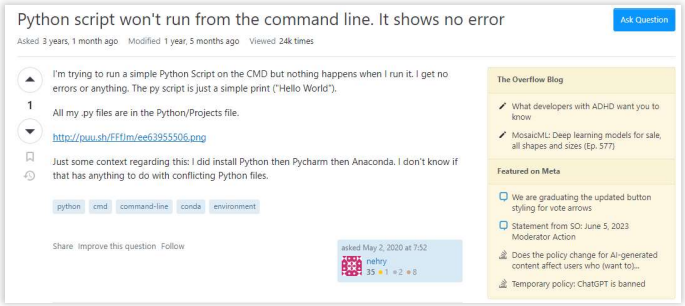
The main difference is that because Colab **is cloud-based** you need to **connect to the internet** (and to your Google Drive) and your files will be stored in **Google Drive**

Helpful Resources

range function in Python

1. **Google all your questions!** (include Python in your search)





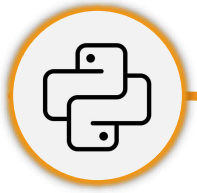
2. **Stack Overflow** is a public coding platform in which developers ask their questions

https://stackoverflow.com/

3. The **Official Python Documentation** is great resource for library and language references.

https://docs.python.org/3/

Section Wrap-Up



✔ **Jupyter Notebooks** are user-friendly coding environments

- Jupyter notebooks are popular among **data analysts** and **data scientists**

✔ **Code cells** are where you write & execute your Python code

- We learn how to **run**, **insert**, **move**, and **remove** cells.

✔ Use **comments** and **markdown** cells for your documentation

- Comments used to **explain/deactivate** specific portions of your code, and markdown should be used to **document** your workflow

✔ **Google Colab** is popular cloud-based alternative to Jupyter Notebooks

- Colab & Jupyter are very similar and have the same file extension (.ipynb) instead Colab store your files in **Google Drive** instead of your machine (computer)

Data Types in Python