

Getting Started With Jupyter Notebook for Python

In the following tutorial, you will be guided through installing the Jupyter Notebook. Furthermore, we'll explore the basic functionality of the Jupyter Notebook and you'll be able to try out the first examples.

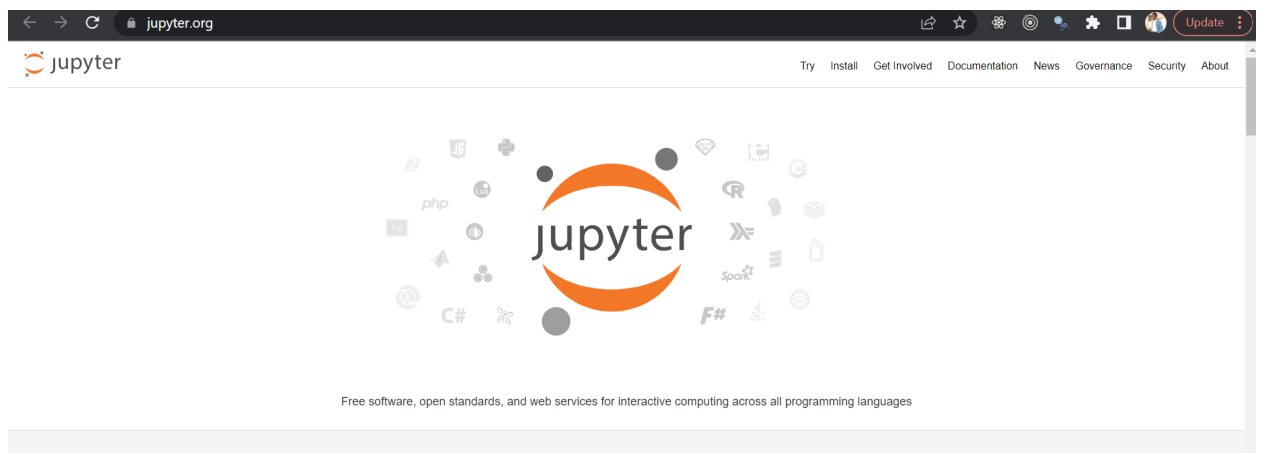
Jupyter Notebook is a web application that allows you to create and share documents that contain:

- live code (e.g. Python code)
- visualizations
- explanatory text (written in markdown syntax)

Jupyter Notebook is great for the following use cases:

- learn and try out Python
- data processing / transformation
- numeric simulation
- statistical modeling
- machine learning

Let's get started and install Jupyter Notebook on your computer ... The first step to get started is to visit the project's website at <https://jupyter.org/>



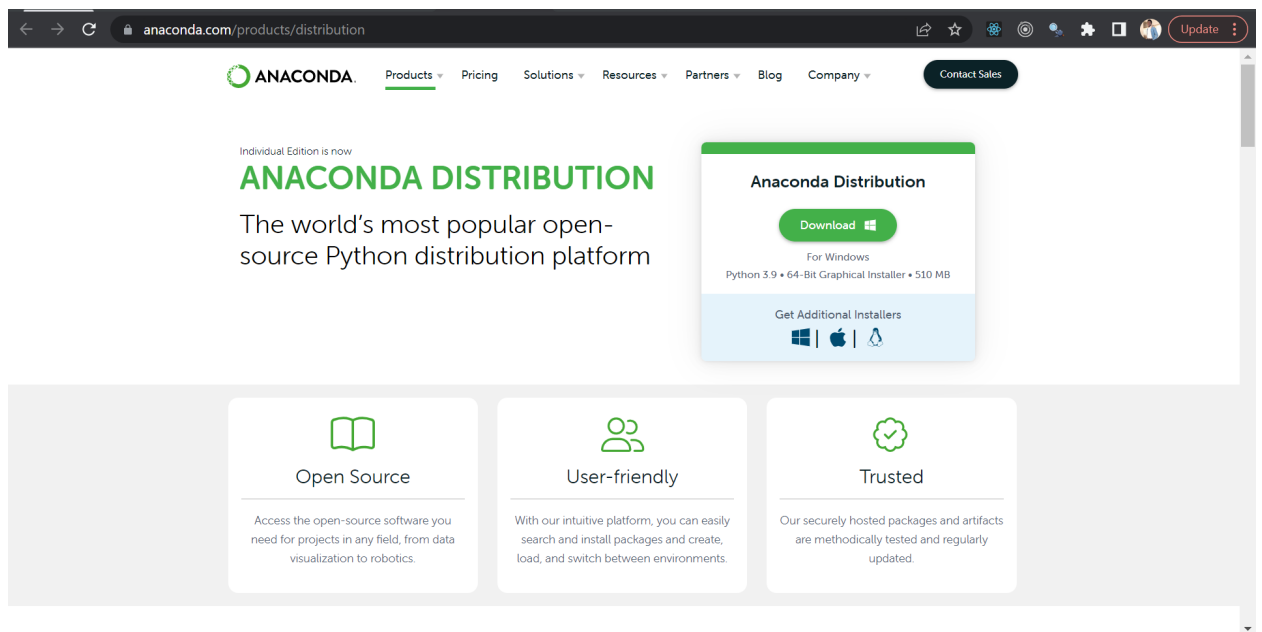
If you scroll down, you'll find two options:

- Try it in your browser
- Install the Notebook

With the first option "Try it in your browser", you can access a hosted version of Jupyter Notebook. This will get you direct access without needing to install it on your computer. The second option "Install the Notebook" will take you to another page that gives you detailed instructions for the installation. There are two different ways:

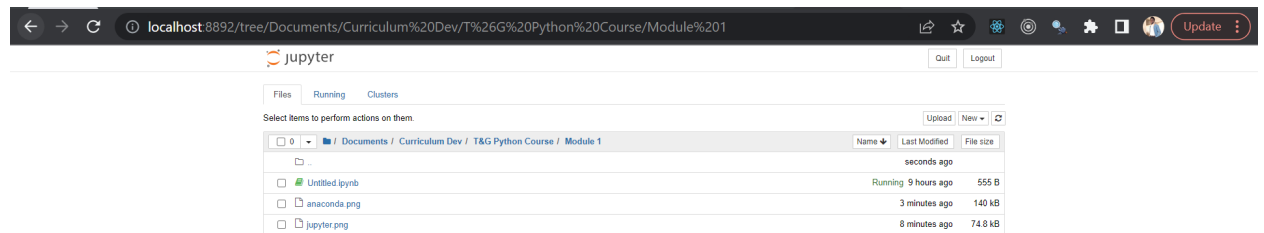
Installing Jupyter Notebook by using Python's package manager pip
Installing Jupyter Notebook by installing the Anaconda distribution

Especially if you're new to Python and would like to set up your development environment from scratch using the Anaconda distribution is a great choice. If you follow the link (<https://www.anaconda.com/products/distribution>) to the Anaconda download page you can choose between installers for Windows, macOS, and Linux:



Download and execute the installer of your choice. Having installed the Anaconda distribution we can now start Jupyter Notebook by clicking the Jupyter notebook icon in the Anaconda GUI, or starting Jupyter from the command prompt.

The web server is started and the Jupyter Notebook application is opened in your default browser automatically. You should be able to see a browser output, which is similar to the following screens



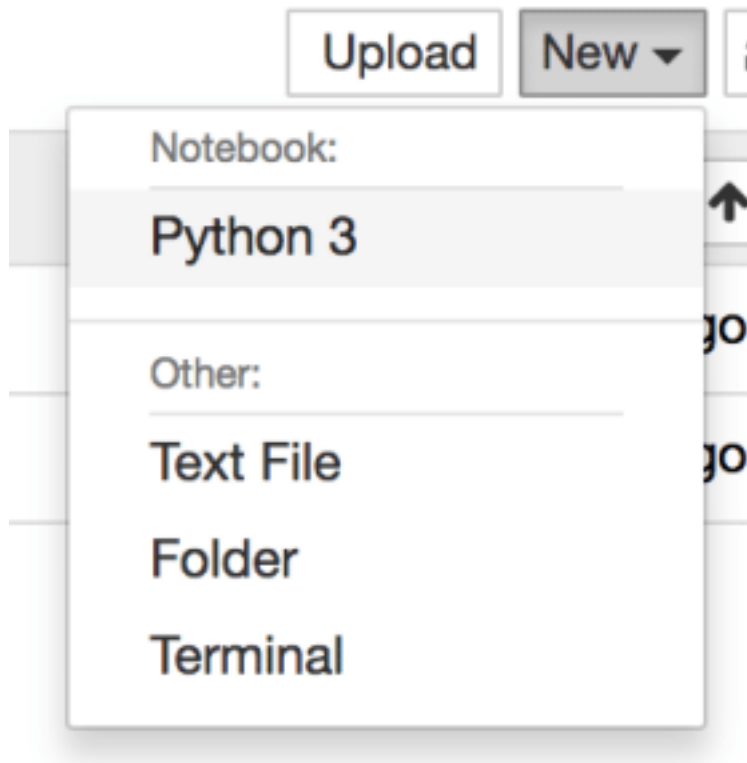
As you can see the user interface of the Jupyter Notebook is split up into three sections (tabs):

- Files
- Running

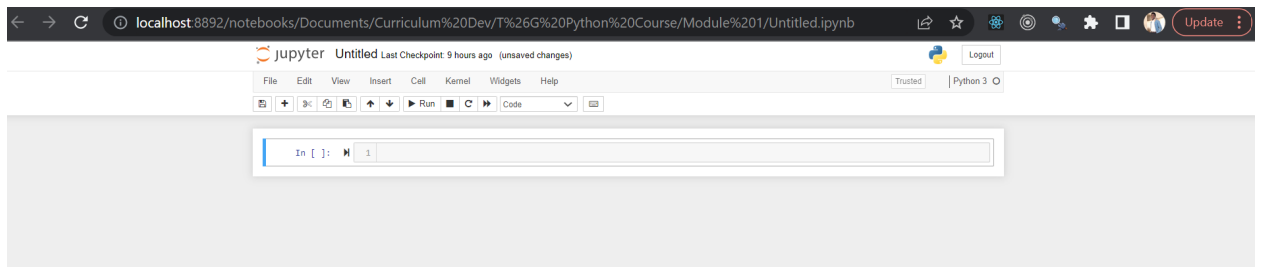
The default view is the Files tab from which you can open or create notebooks.

Creating A New Notebook

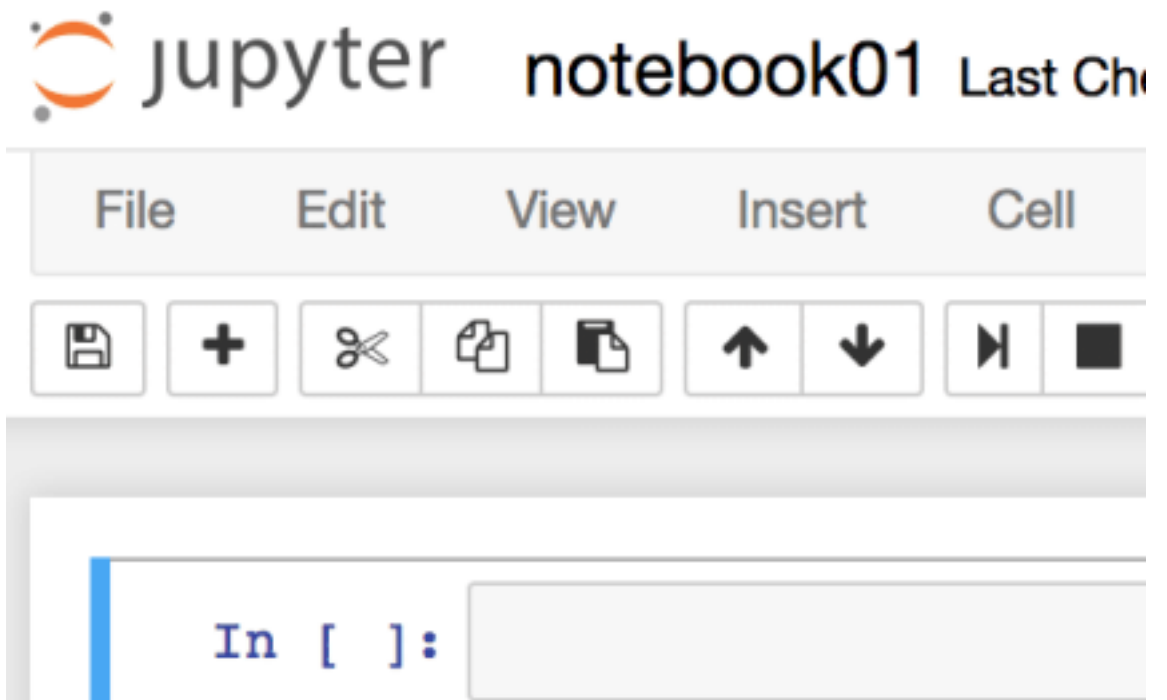
Creating a new Jupyter Notebook is easy. Just use the New dropdown menu and you'll see the following options:



Select option Python 3 to open a new Jupyter Notebook for Python. The notebook is created and you should be able to see something similar to:



The notebook is created but still untitled. By clicking on the text “Untitled” on the top, you can give it a name. By giving it a name the notebook will also be saved as a file of the same name with extension .ipynb. E.g. name the notebook notebook01:



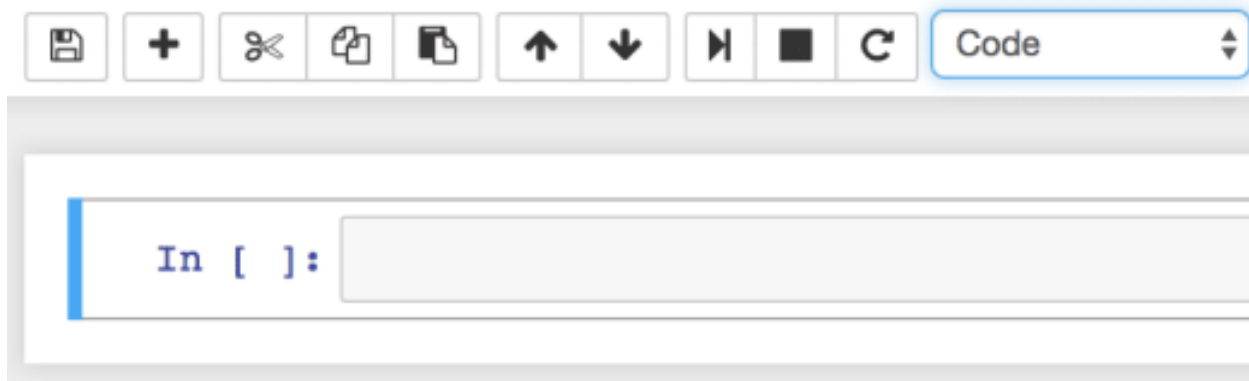
Switching back to the Files tab you'll be able to see a new file notebook01.ipynb:



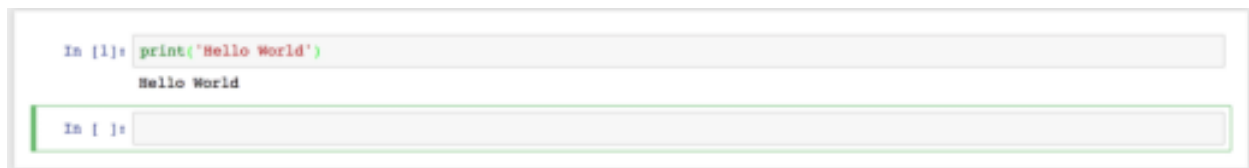
Because this notebook file is opened right now the file is marked with the status Running. From here you can decide to shut down this notebook by clicking on the button Shutdown. However, before shutting down the notebook let's switch back to the notebook view and try out a few things to get familiar with the notebook concept.

Working with the Notebook

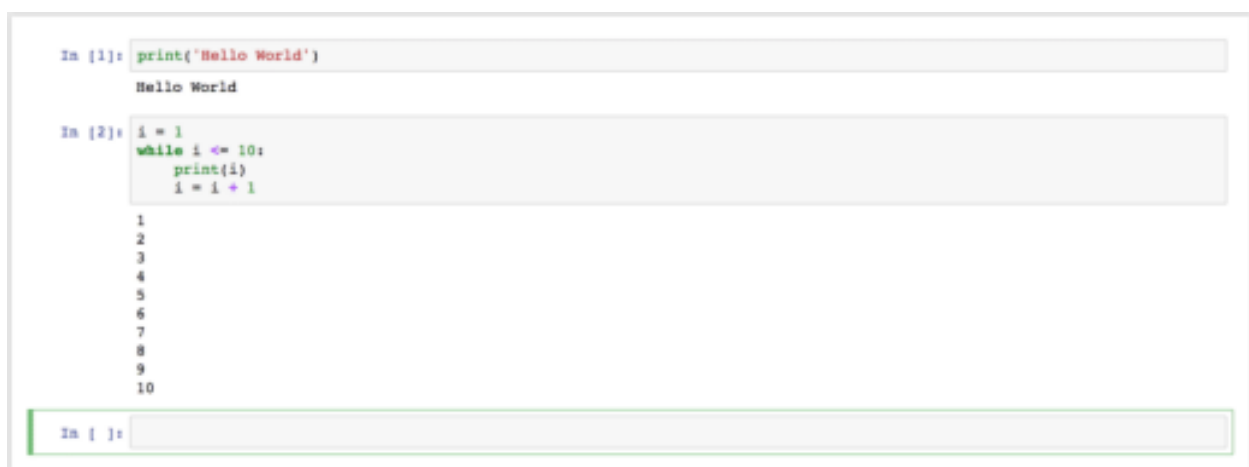
The notebook itself consists of cells. A first empty cell is already available after having created the new notebook:



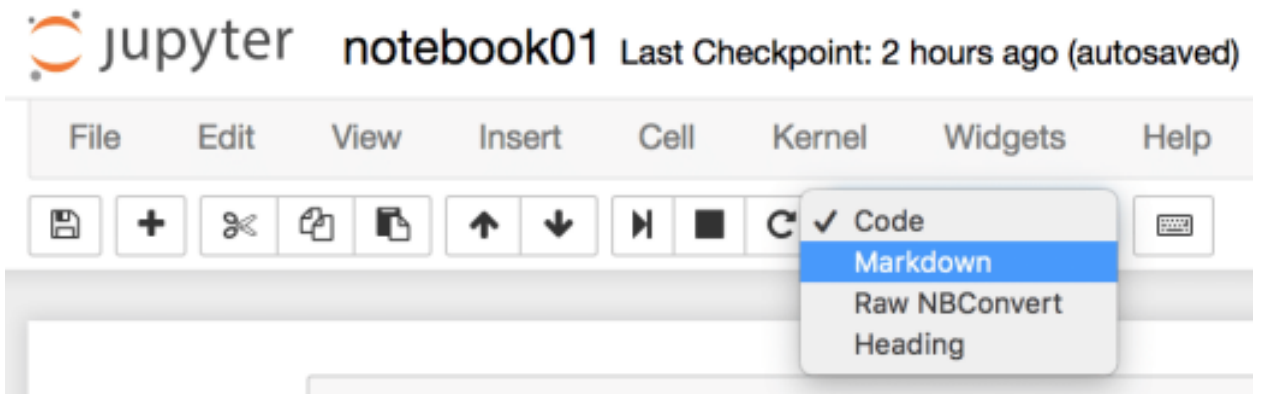
This cell is of type “Code” and you can start typing in Python code directly. Executing code in this cell can be done by either clicking on the run cell button or hitting Shift + Return keys:



The resulting output becomes visible right underneath the cell. The next empty code cell is created automatically and you can continue to add further code to that cell. Just another example:



You can change the cell type from Code to Markdown to include explanatory text in your notebook. To change the type you can use the dropdown input control:



Once switched the type to Markdown you can start typing in markdown code:

```
# This is a headline
## Sub headline

**Text**
More Text
```

After having entered the markdown code you can compile the cell by hitting Shift + Return once again. The markdown editor cell is then replaced with the output:

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

In [2]: i = 1
while i <= 10:
    print(i)
    i = i + 1

1
2
3
4
5
6
7
8
9
10

This is a headline

Sub headline

Text

More Text

In [ ]:
```

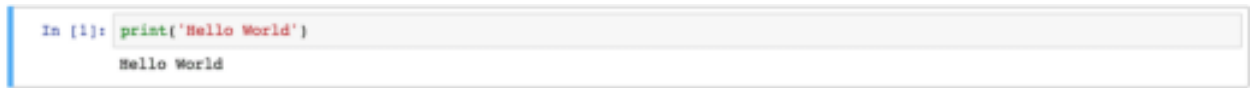
If you want to change the markdown code again you can simply click into the compiled result and the editor mode opens again.

Edit And Command Mode

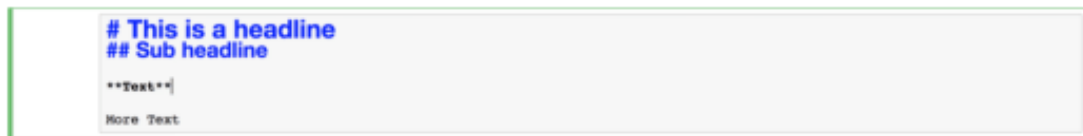
If a cell is active, two modes distinguished:

- edit mode
- command mode

If you just click in one cell the cell is opened in command mode which is indicated by a blue border on the left:

A screenshot of a Jupyter Notebook cell in command mode. The cell has a blue border on the left side. The code inside is `In [1]: print('Hello World')` and the output below it is `Hello World`.

The edit mode is entered if you click into the code area of that cell. This mode is indicated by a green border on the left side of the cell:

A screenshot of a Jupyter Notebook cell in edit mode. The cell has a green border on the left side. The code inside is `# This is a headline`, `## Sub headline`, `**Text**`, and `More Text`.

If you'd like to leave edit mode and return to command mode again you just need to hit ESC. To get an overview of functions which are available in command and in edit mode you can open up the overview of key shortcuts by using menu entry Help → Keyboard Shortcut

Checkpoints

Another cool function of Jupyter Notebook is the ability to create checkpoint. By creating a checkpoint you're storing the current state of the notebook so that you can later on go back to this checkpoint and revert changes which have been made to the notebook in the meantime. To create a new checkpoint for your notebook select menu item Save and Checkpoint from the File menu. The checkpoint is created and the notebook file is saved. If you want to go back to that checkpoint at a later point in time you need to select the corresponding checkpoint entry from menu File → Revert to Checkpoint.

Exporting The Notebook

Jupyter Notebook gives you several options to export your notebook. Those options can be found in menu File → Download as:

