Getting Started with Jupyter Notebook for Python

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

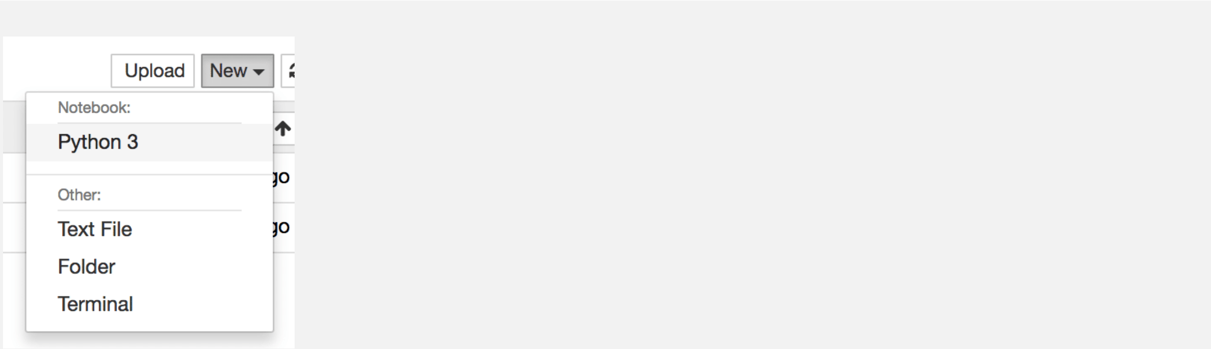
- live code (e.g. Python code)
- visualizations
- explanatory text (written in markdown syntax)

Jupyter Notebook is great for the following use cases:

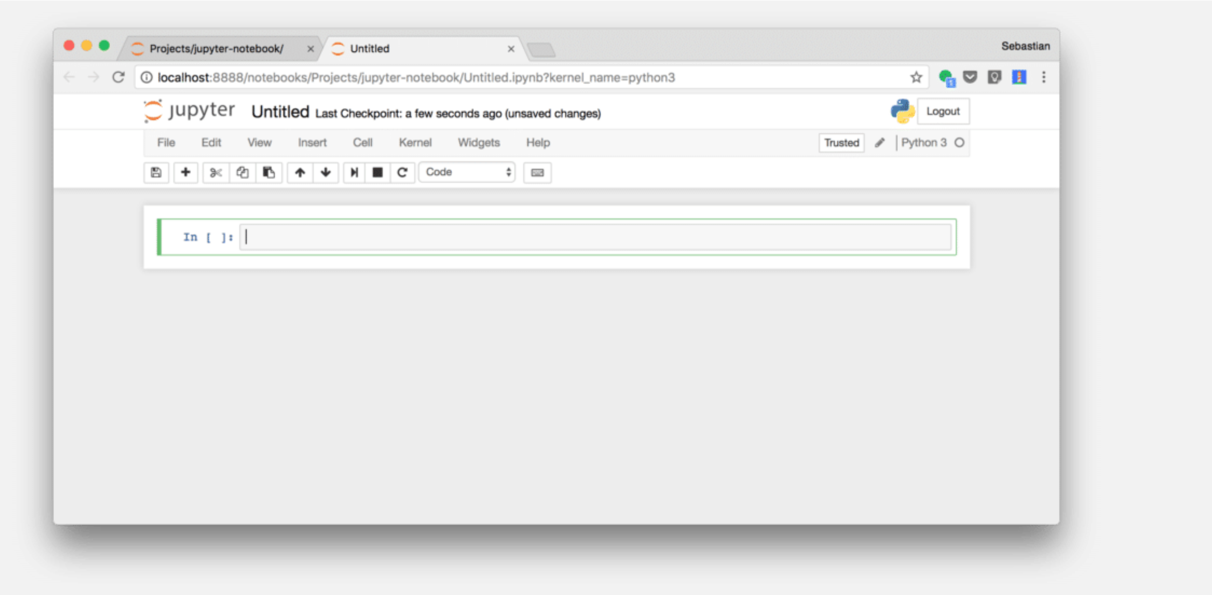
- learn and try out Python
- data processing / transformation
- numeric simulation
- statistical modelling
- machine learning

Creating A New Notebook

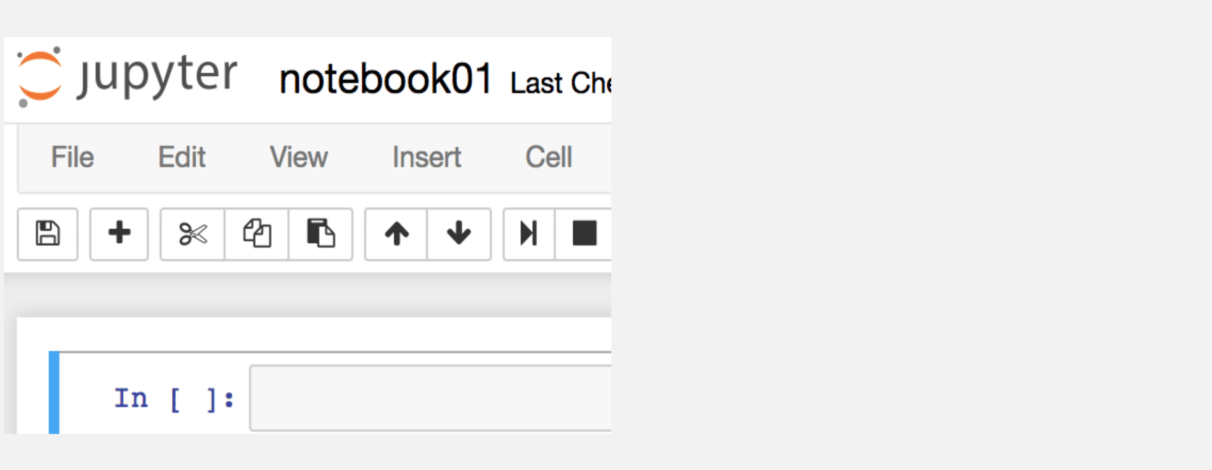
Creating a new Jupyter Notebook is easy. Just use the *New* dropdown menu and you'll see the following options:



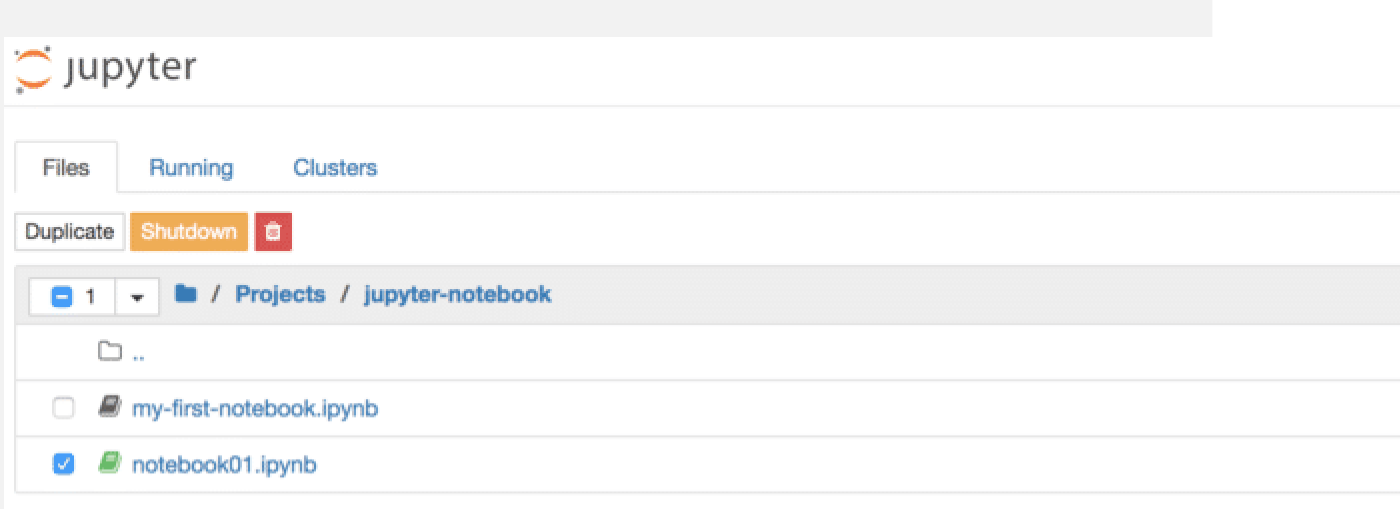
Select option *Python 3* to open a new Jupyter Notebook for Python. The notebook is created and you should be able to see something similar to:


Projects/jupyter-notebook/
Untitled
Sebastian
localhost:8888/notebooks/Projects/jupyter-notebook/Untitled.ipynb?kernel_name=python3
jupyter
Untitled Last Checkpoint: a few seconds ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Code
In [ ]:

The notebook is created but still untitled. By clicking into 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*:

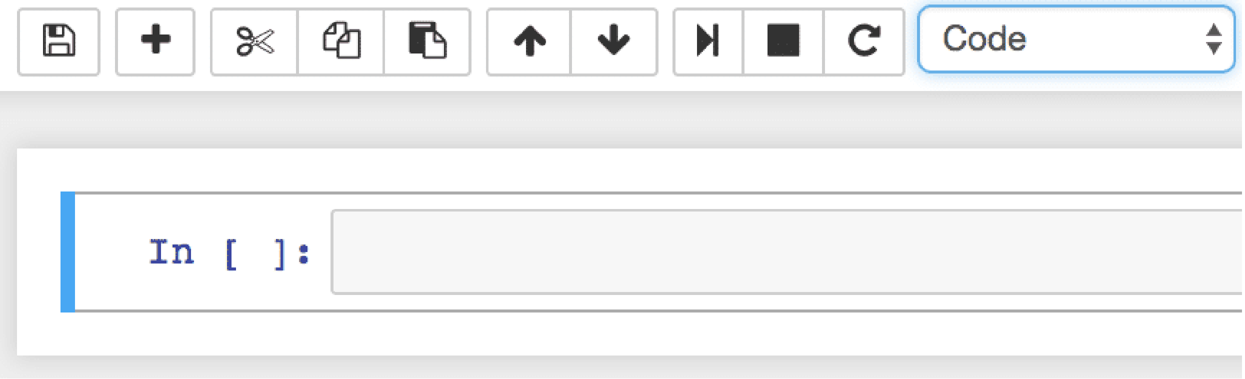


Because this notebook file is opened right now the file is marked with status *Running*. From here you can decide to shut down this notebook by clicking on button *Shutdown*.

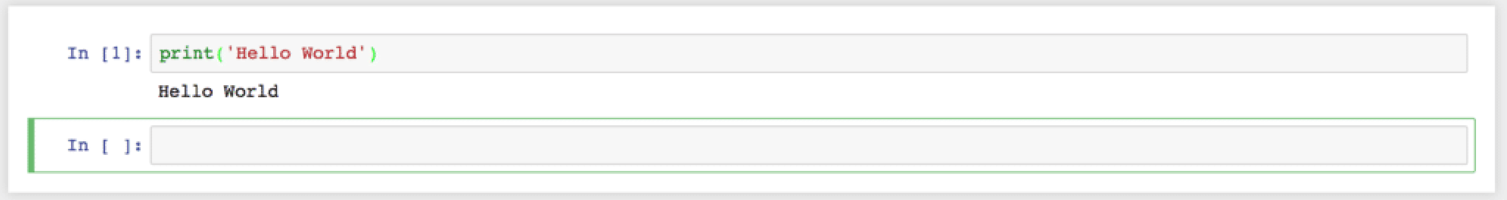
However before shutting down the notebook let's switch back to the notebook view and try out a few things to get familiar with the notebook concept.

Working with The Notebook

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



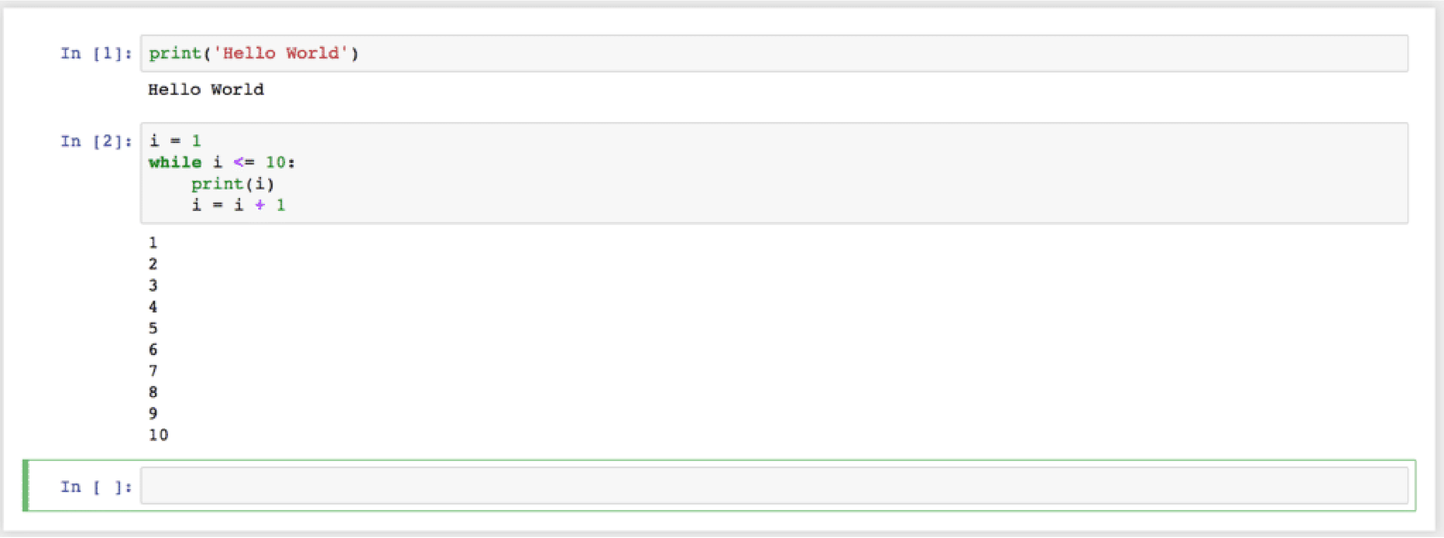
This cell is of type "Code" and you can start typing in Python code directly. Executing code in this cell can be done by either clicking on the *run cell* button or hitting Shift + Return keys:



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

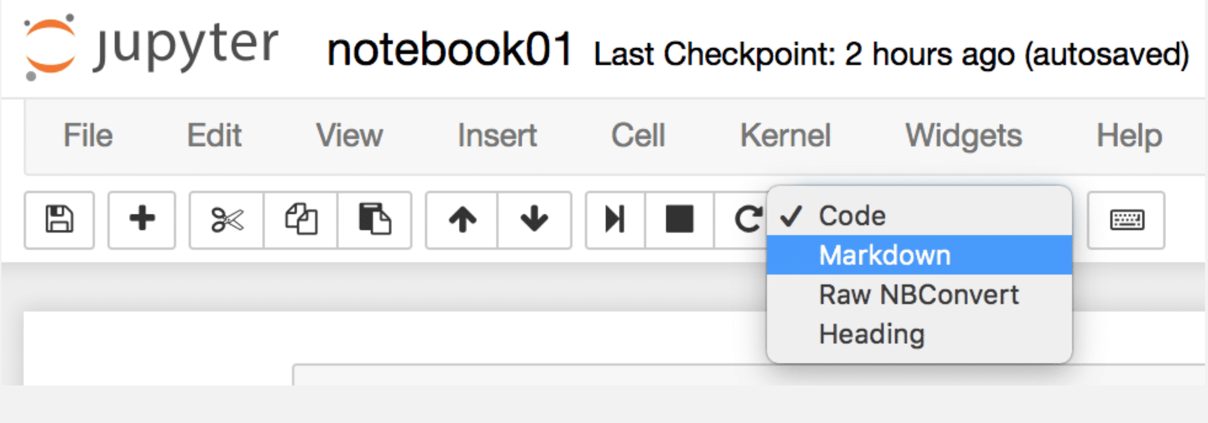
The resulting output becomes visible right underneath the cell.

The next empty code cell is created automatically and you can continue to add further code to that cell. Just another example:

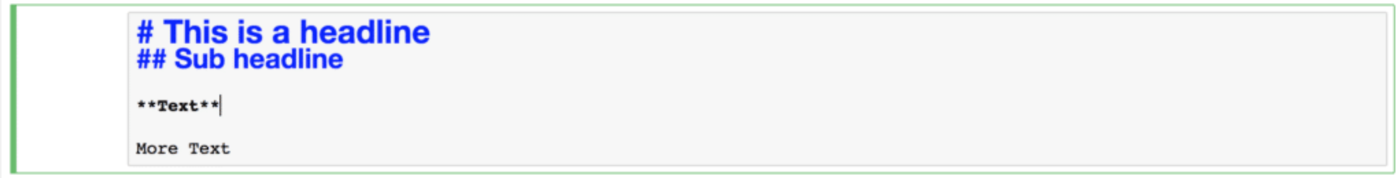


```
In [1]: print('Hello World')

        Hello World

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

        1
        2
        3
        4
        5
        6
        7
        8
        9
        10

In [ ]:
```

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:

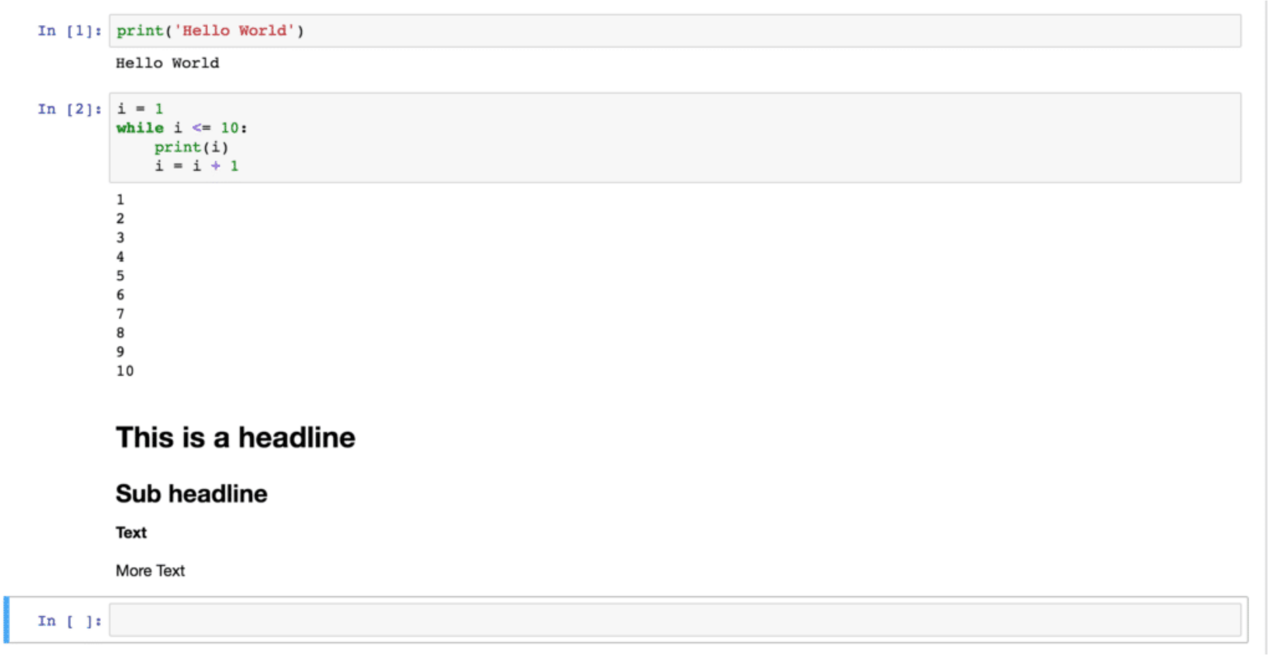


Once switched the type to *Markdown* you can start typing in markdown code:



```
# 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 [ ]:

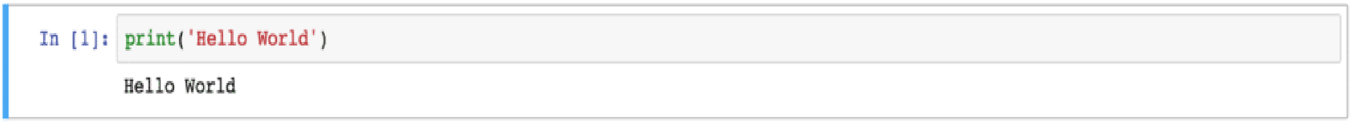
If you want to change the markdown code again you can simply click into the compiled result and the editor mode opens again.

Edit and Command Mode

If a cell is active two modes distinguished:

- edit mode
- command mode

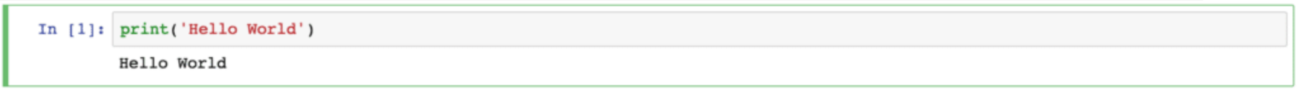
If you just click in one cell the cell is opened in command mode which is indicated by a blue border on the left:



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

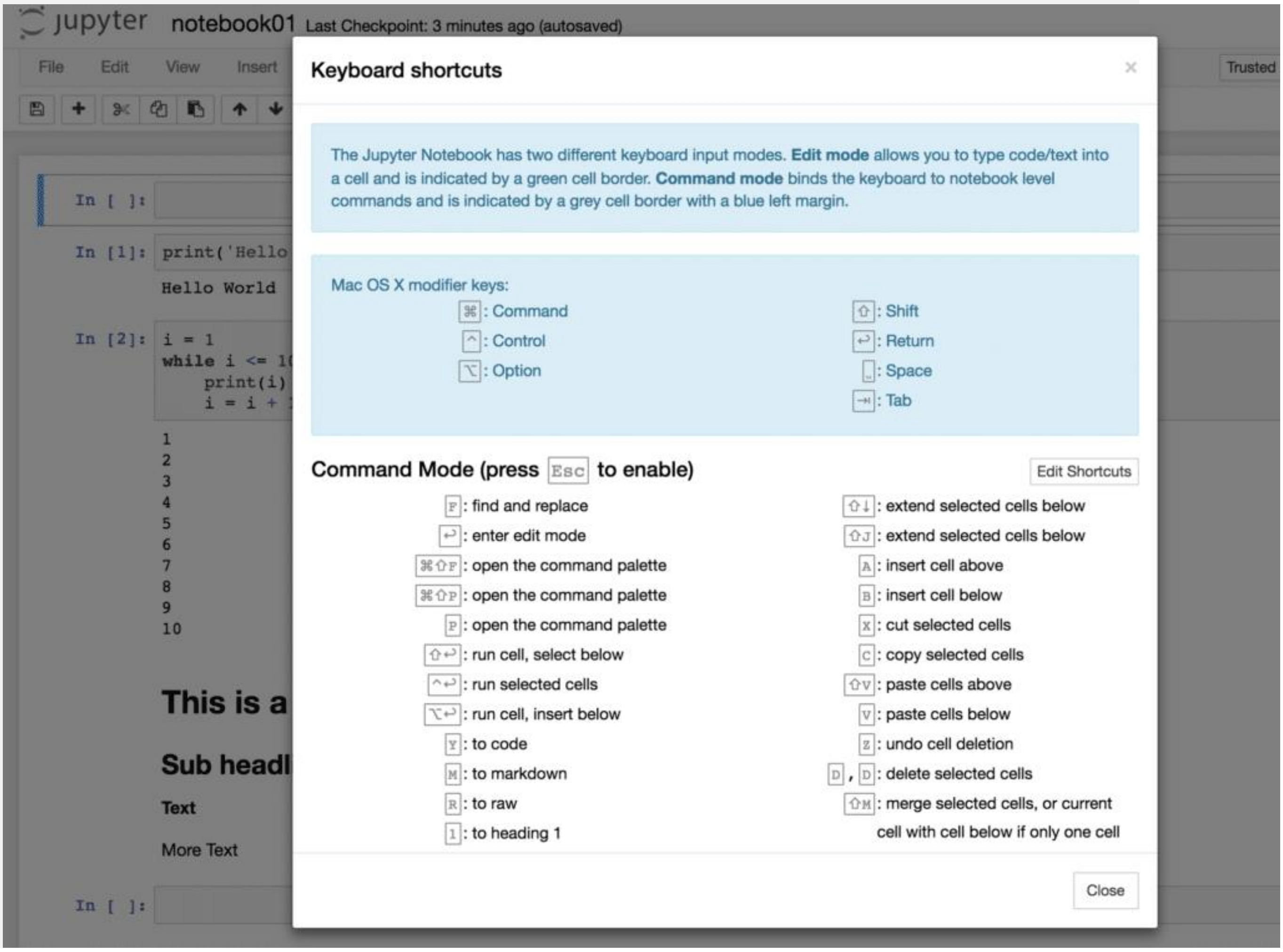
```
Hello World
```

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:



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

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 Shortcuts*:



```
In [ ]:

In [1]: print('Hello
        Hello World

In [2]: i = 1
        while i <= 1(
            print(i)
            i = i + 1
        1
        2
        3
        4
        5
        6
        7
        8
        9
        10
```

This is a

Sub headl

Text

More Text

In [ ]:

Keyboard shortcuts ×

The Jupyter Notebook has two different keyboard input modes. **Edit mode** allows you to type code/text into a cell and is indicated by a green cell border. **Command mode** binds the keyboard to notebook level commands and is indicated by a grey cell border with a blue left margin.

Mac OS X modifier keys:

⌘ : Command	⇧ : Shift
^ : Control	↩ : Return
⌥ : Option	␣ : Space
	⇥ : Tab

Command Mode (press Esc to enable) — Edit Shortcuts

F : find and replace	⇧↓ : extend selected cells below
↩ : enter edit mode	⇧J : extend selected cells below
⌘⇧F : open the command palette	A : insert cell above
⌘⇧P : open the command palette	B : insert cell below
P : open the command palette	X : cut selected cells
⇧↩ : run cell, select below	C : copy selected cells
^↩ : run selected cells	⇧V : paste cells above
⌥↩ : run cell, insert below	V : paste cells below
Y : to code	Z : undo cell deletion
M : to markdown	D , D : delete selected cells
R : to raw	⇧M : merge selected cells, or current cell with cell below if only one cell
1 : to heading 1	

Close

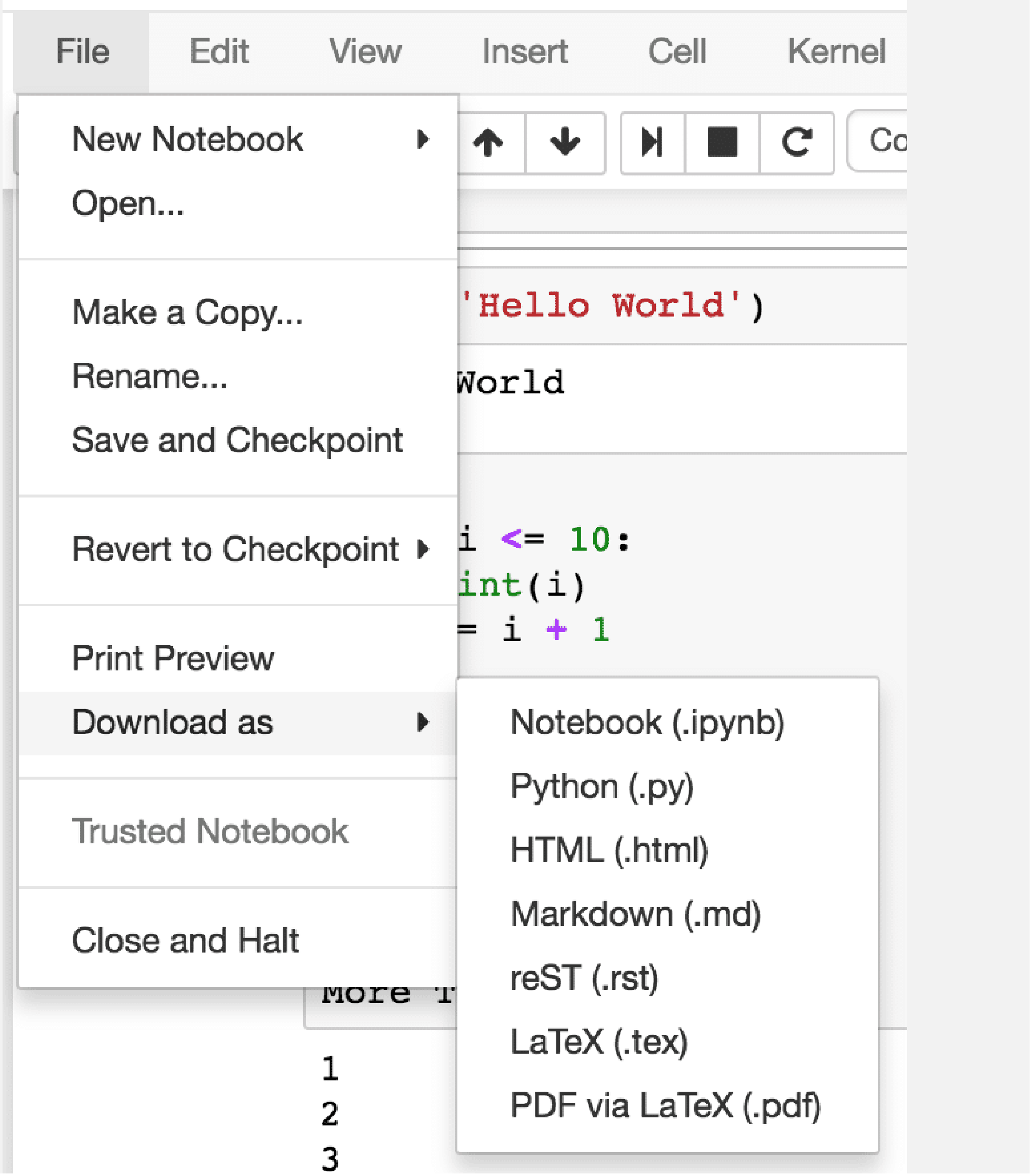
Checkpoints

Another 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*:



```
'Hello World')
World
i <= 10:
int(i)
= i + 1
More T
1
2
3
```