

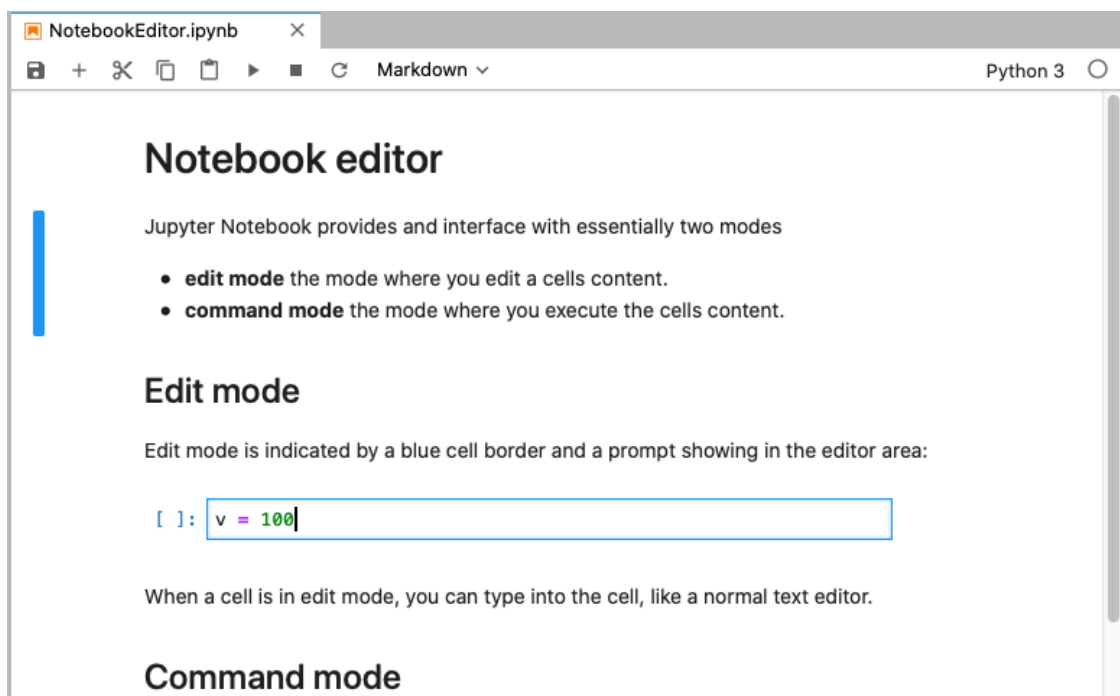
## 2\_NotebookEditor

October 17, 2023

### 1 Notebook editor

```
[3]: from IPython.display import Image, display;
```

```
[6]: display(Image(filename='img/notebook.png'))
```



A Jupyter Notebook provides an interface with essentially two modes

- **edit mode** the mode where you edit a cells content.
- **command mode** the mode where you execute the cells content.

In the more advanced version of JupyterLab, which we are using on myBinder, this will look like that

#### 1.1 Edit mode

Edit mode is indicated by a blue cell border and a prompt showing in the editor area:

```
[11]: display(Image(filename='img/edit_mode.png'))
```

```
[ ]: v = 100|
```

When a cell is in edit mode, you can type into the cell, like a normal text editor.

## 1.2 Command mode

Command mode is indicated by a grey cell border:

```
[ ]: display(Image(filename='img/command_mode.png'))
```


## 1.3 Keyboard navigation

If you have a hardware keyboard connected to your iOS device, you can use Jupyter keyboard shortcuts. The modal user interface of the Jupyter Notebook has been optimized for efficient keyboard usage. This is made possible by having two different sets of keyboard shortcuts: one set that is active in edit mode and another in command mode.

In edit mode, most of the keyboard is dedicated to typing into the cell's editor. Thus, in edit mode there are relatively few shortcuts. In command mode, the entire keyboard is available for shortcuts, so there are many more. Most important ones are:

1. Switch command and edit mods: **Enter** for edit mode, and **Esc** or **Control** for command mode.
2. Basic navigation: **↑/k**, **↓/j**
3. Run or render currently selected cell: **Shift+Enter** or **Control+Enter**
4. Saving the notebook: **s**
5. Change Cell types: **y** to make it a **code** cell, **m** for **markdown** and **r** for **raw**
6. Inserting new cells: **a** to **insert above**, **b** to **insert below**
7. Manipulating cells using pasteboard: **x** for **cut**, **c** for **copy**, **v** for **paste**, **d** for **delete** and **z** for **undo delete**
8. Kernel operations: **i** to **interrupt** and **0** to **restart**

## 1.4 Running code in your notebook

Code cells allow you to enter and run code. Run a code cell by pressing the  button in the bottom-right panel, or **Control+Enter** on your hardware keyboard.

```
[1]: v = 10
```

```
[2]: print(v)
```

10

There are a couple of keyboard shortcuts for running code:

- **Control+Enter** run the current cell and enters command mode.
- **Shift+Enter** runs the current cell and moves selection to the one below.
- **Option+Enter** runs the current cell and inserts a new one below.

## 1.5 Managing the kernel

Code is run in a separate process called the **kernel**, which can be interrupted or restarted. You can see kernel indicator in the top-right corner reporting current kernel state: `Ready` means kernel is **ready** to execute code, and `Busy` means kernel is currently **busy**. Tapping kernel indicator will open **kernel menu**, where you can reconnect, interrupt or restart kernel.

Try running the following cell — kernel indicator will switch from `Ready` to `Busy`, i.e. reporting kernel as “busy”. This means that you won’t be able to run any new cells until current execution finishes, or until kernel is interrupted. You can then go to kernel menu by tapping the kernel indicator and select “Interrupt”.