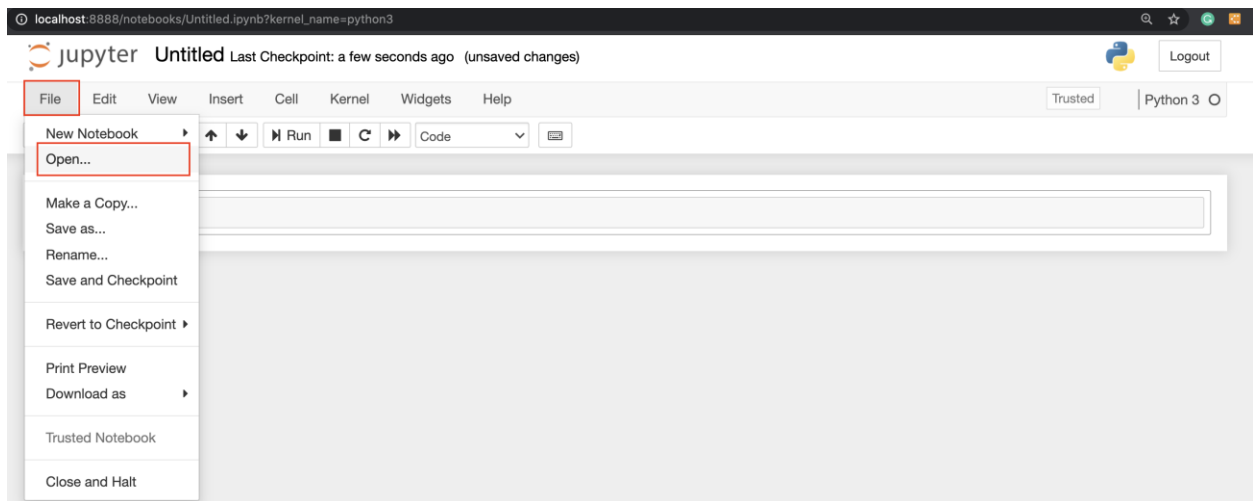


Code Cells

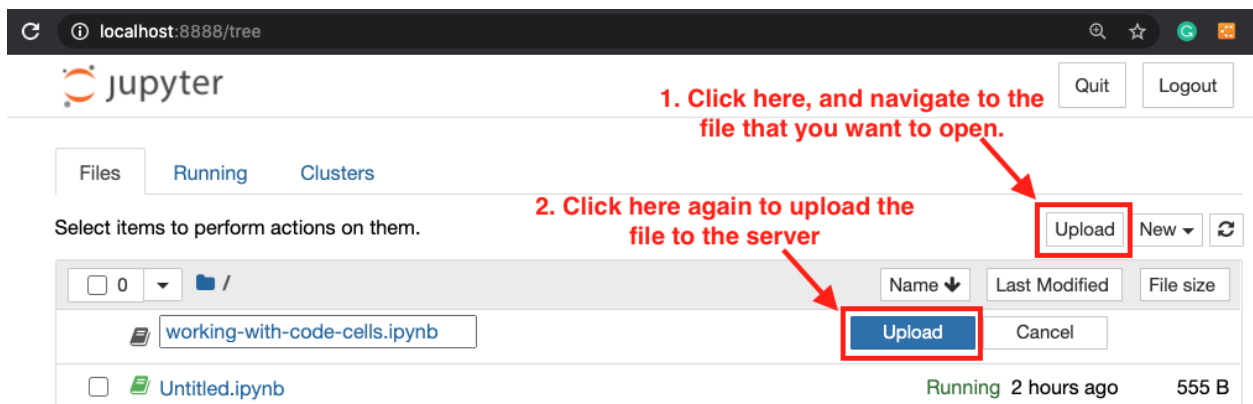
Most of your work in notebooks will be done in code cells. This is where you write your code and it gets executed. In code cells, you can write any code, assigning variables, defining functions and classes, importing packages, and more. Any code executed in one cell is available in all other cells.

To give you some practice, I created a notebook you can work through. Download the notebook **Working With Code Cells** below then run it from your own notebook server. Your browser might try to open the notebook file without downloading it. If that happens, right-click on the link then choose "Save Link As...". There are two ways to open the downloaded notebook locally:

1. In your terminal, change to the directory with the notebook file, then enter `jupyter notebook`.
2. If you have a Notebook server already up and running, you can either put the newly-downloaded file into the current working directory or use the "Upload" option in the Notebook server.



Open a file using the "File" menu.



Steps to upload any file to the current running Notebook server.

localhost:8888/notebooks/working-with-code-cells.ipynb

jupyter working-with-code-cells (unsaved changes) Python 3

File Edit View Insert Cell Kernel Widgets Help Trusted

Run

Working with code cells

In this notebook you'll get some experience working with code cells.

First, run the cell below. As I mentioned before, you can run the cell by selecting it then clicking the "run cell" button above. However, it's easier to run it by pressing **Shift + Enter** so you don't have to take your hands away from the keyboard.

```
In [ ]: # Select the cell, then press Shift + Enter
3**2
```

Shift + Enter runs the cell then selects the next cell or creates a new one if necessary. You can run a cell without changing the selected cell by pressing **Control + Enter**.

The output shows up below the cell. It's printing out the result just like in a normal Python shell. Only the very last result in a cell will be printed though. Otherwise, you'll need to use `print()` to print out any variables.

Exercise: Run the next two cells to test this out. Think about what you expect to happen, then try it.

```
In [ ]: 3**2
4**2
```

```
In [ ]: print(3**2)
4**2
```

Now try assigning a value to a variable.

A snapshot of the working-with-code-cells.ipynb file

Supporting Materials

- [Working With Code Cells](#)