B11_jupyter_code

February 22, 2024

0.1 Presentation

A **notebook** is a sort of interactive Python console where the code is contained within **cells**. each celle can be ran alone, but the memory (i.e. variables) is shared by all cells.

Output of each cell is printed below the cell. By default, the last evaluated statement of the cell is an output.

There are several pros of using **notebooks** at a development step:

- split heavy computation into small units that make sense
- access easily to variable values
- describe the scientific flow of your program using *markdown* (discussed hereafter)

0.2 Howto

[5]:
```
area = 100
speed = 30
print(area, speed)
```

```
100 30
```

[6]:
```
flowrate = area * speed
print(flowrate)
```

```
3000
```

Above: `flowrate` is displayed as `print` is used. Below: `flowrate` is returned, and thus displayed, as it is the last statement of the cell

[7]:
```
flowrate
```

[7]:
```
3000
```

One can run nearly every Python code in a **notebook**:

- in script mode: global variables declared directly in any cell
- in function mode: functions are defined and called later on
- in object oriented mode: class and methods are defined

[8]:
```
def compute_flowrate(area, speed):
    return area * speed
```

```
compute_flowrate(1000, 30)
```

[8]: 30000

[9]:
```
class Flowrate():
    def __init__(self, area, speed):
        self._area = area
        self._speed = speed
        self.compute()

    def compute(self):
        self._flowrate =  self._area * self._speed

    def __repr__(self):
        return f"Instance 'Flowrate' with value: {self._flowrate:d}"

Flowrate(100, 30)
```

[9]: Instance 'Flowrate' with value: 3000

Note: during the execution of a cell:

- the cell itself shows an asterisk [*]
- the black circle on the right hand side of the page is colored with black

0.3 Configuration

0.3.1 Files

notebooks files have an extension *.ipynb*: these files cannot be manually read (JSON format) and cannot be ran the same way a Python file (*.py*) is ran. Yet, most IDE have a support plugin for these files.

0.3.2 Kernel

The *Python* version used by the **notebook** depends either on the environment it is running in or on the configuration of the system. If several Python versions are available, one can choose which to use in the notebook by going to Kernel > Change kernel. The *nb_conda* plugins facilitates the discovery of existing *conda* environments.

Other **kernels** can be installed to code in other languages (full list).

0.3.3 Functionalities

Some helper functionalitiesin the development process are not natively available in a **notebook**:

Some of them are available using *plugins*.