Introduction to Python

Elements of Applied Data Security

Alex Marchioni – alex.marchioni@unibo.it

Outline

- Lab’s objectives and modalities
- Python: what and why.
- Development Environment: Jupyter
- Let’s get started

Objectives

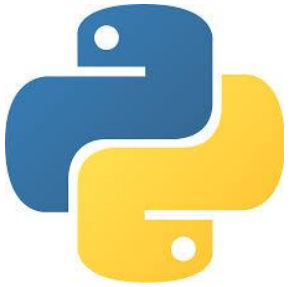
1. Hands on methods and algorithms explained in the theory lessons
   - Focus on concepts and working principles
   - Minor concerns on implementation details

2. Get experienced with Python programming language
   - Start with the basics (assuming you have C/C++ background)

How it works

1. Python Tutorial
   - Soft start with Python
2. Tasks
   - Once a week, you will be assigned a task to work on
   - We will go through solutions all together
   - No marks
3. Final Project
   - Implementation + brief explanation (as a report or presentation)
   - You can work in groups (max 4 students) but the assessment is individual.
   - Mark is Pass/Fail. Pass is required to access to the theory exam.

Python



Python

Python is an Interpreted and Object-Oriented Programming Language.

WHY Python?

- Simple syntax
- Very flexible
- Highly extensible
- Cross-platform
- Open-source with a huge community

Google says: *Python where we can, C++ where we must*

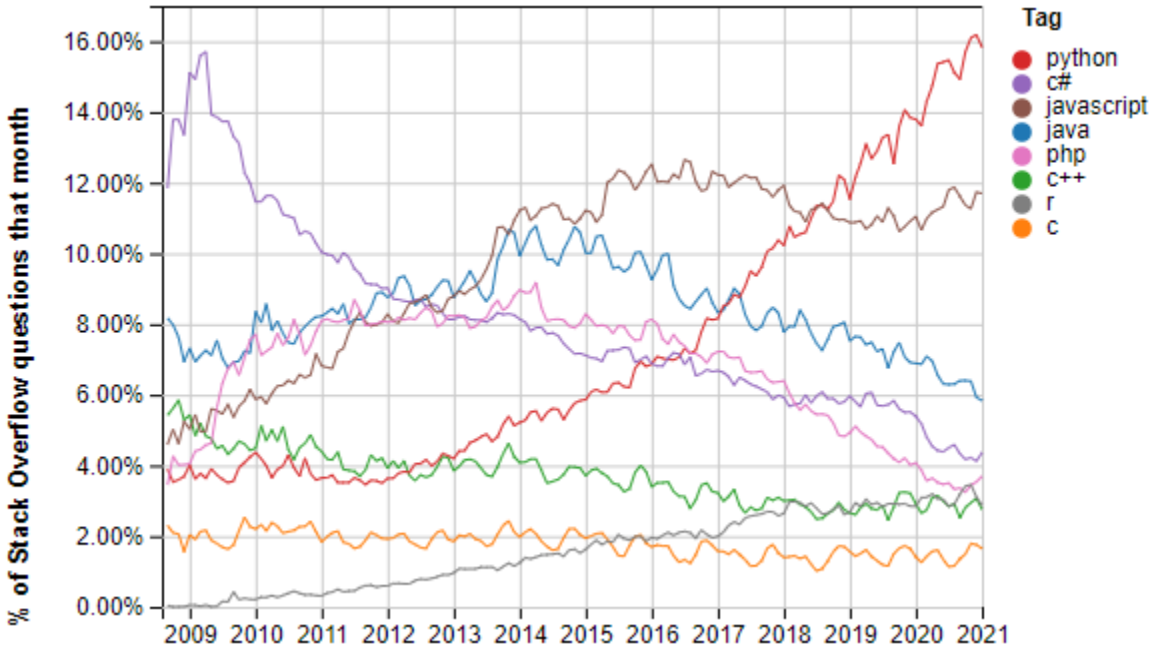
A bit of History

- It is not so recent, since it was conceived in the late 1980s and implemented in 90s by Guido van Rossum.
- Name is a tribute to Monty Python (a British surreal comedy group)
  indeed, as metasyntactic variables spam and eggs preferred to the traditional foo and bar
- **Python 2.0** released on Oct 2000, not supported since 01/2020
- **Python 3.0** released on Dec 2008
- **Last release** is **3.9.1** on Dec 2020 (https://www.python.org/downloads/)

Here a great Hackaday's post: Stop using Python2: What you need to know about Python3

Popularity

- According to StackOverflow's survey and trends Python among all programming languages is the:
  - **1st** most questioned
  - **4th** most used
    behind JavaScript, HTML/CSS, and SQL
  - **2nd** most loved
    behind Rust
  - **1st** most wanted
    developers who do not yet use it say they want to learn it



Applications

- Web and Internet Development
- Scientific and Numeric
- Education
- Desktop GUIs
- Software Development
- Business Applications

Basically anything, like English for spoken languages

Development Environment

Python as is

```
alex@sheldon:~$ python3
Python 3.7.3 (default, Jul 25 2020, 13:03:44)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello world')
hello world
>>>
```

Python, such as it is, is quite useless. There is the need for:

- a **code editor** or an integrated development environment (**IDE**)
- **packages** for specialized libraries

Scientific packages

Scipy.org = Python for math/science/engineering

- **Numpy**: Numerical Python package (inspired by Matlab)

  N-dimensional array capabilities and some linear algebra, Fourier analysis, random number capabilities, etc.

- **Scipy**: Scientific Python

  For Matlab users, it's very much like many of the core toolboxes.

- **Matplotlib:** most popular data visualization package for Python

  Inspired by Matlab plots, but then it has evolved into something more.

- **Pandas:** Data Science Python

  high-performance, easy-to-use data structures and data analysis tools.

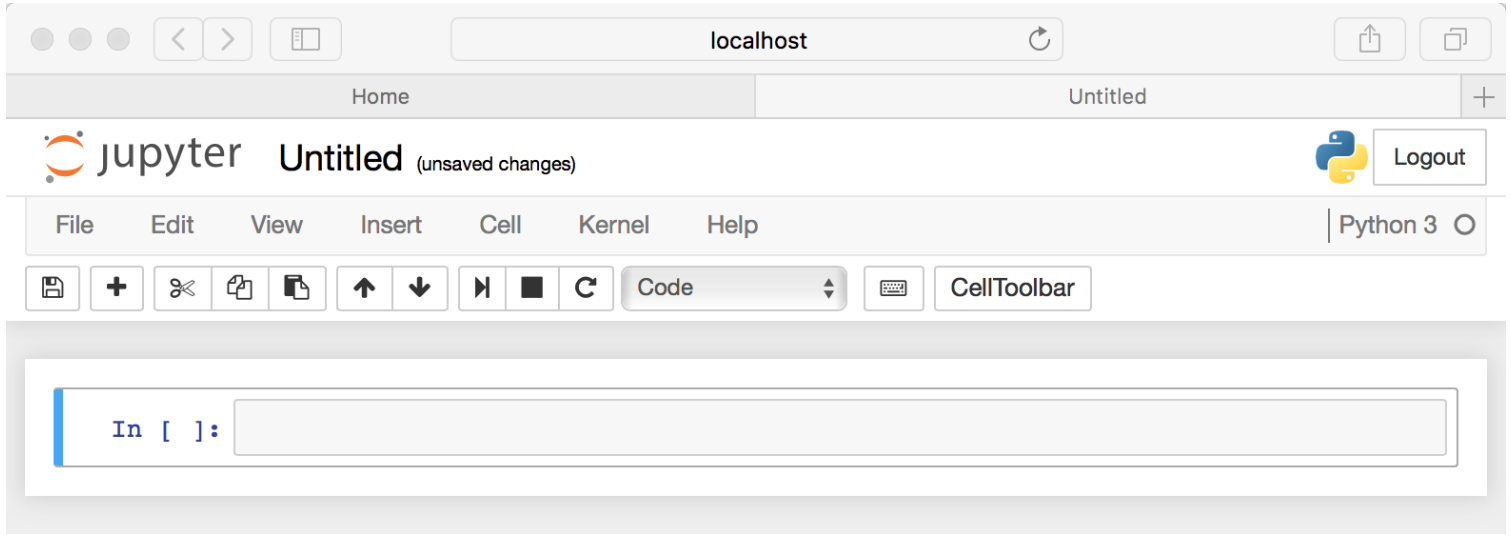
Code editors/IDEs and notebooks

- **Code Editor**
  - tool that is used to write and edit code with features such as text highlighting.
  - Examples: Vim, Atom, Sublime Text
- **IDE** (integrated development environment):
  - integrate text editor with other tools such as compiler, build or make integration, debugging and so on.
  - Examples: IDLE, VS Code, PyCharm, Spyder
- **Notebook**: Jupyter
  - web-based interactive development environment that allows you to create documents that contain live code, equations, visualizations and narrative text.

Jupyter Notebook

web application (it runs on a browser) made of 2 main components:

- **Kernel**: program that runs and introspects the user's code.
- **Dashboard**: shows you the notebook documents and manage the kernels (see which are running or shut them down)



Jupyter Notebook

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

```
Hello World
```

Getting started with Python

We have done the following

- installed Python
- started iPython Notebook

Create variables in Python

In [3]:
```
i = 4 # int
```

In [4]:
```
type(i)
```

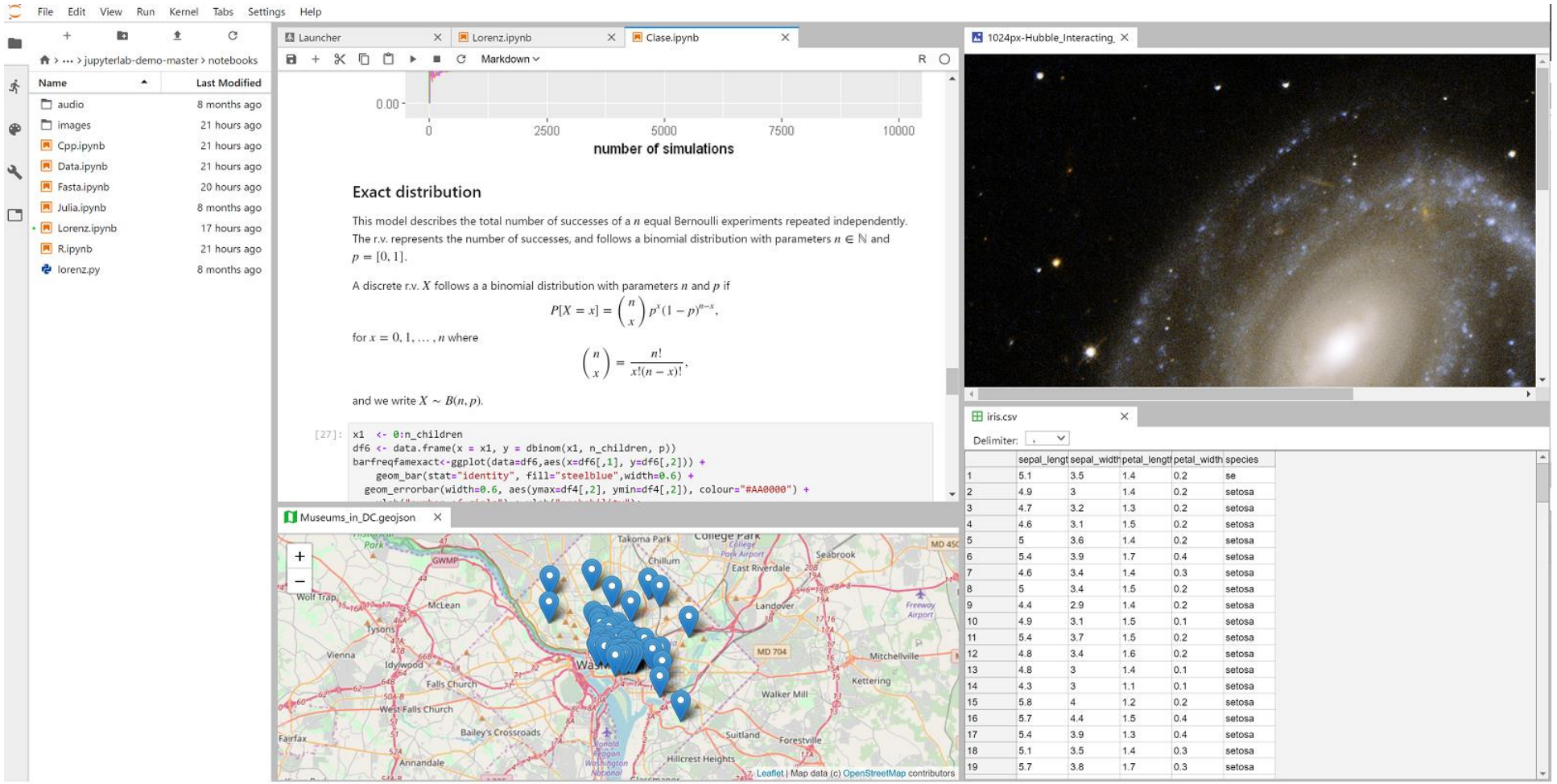
Out[4]:
```
int
```

In [5]:
```
f = 3.14
```

In [6]:
```
type(f)
```

Out[6]:
```
float
```

Jupyter Lab


File Edit View Run Kernel Tabs Settings Help
jupyterlab-demo-master > notebooks
Name
Last Modified
audio 8 months ago
images 21 hours ago
Cpp.ipynb 21 hours ago
Data.ipynb 21 hours ago
Fasta.ipynb 20 hours ago
Julia.ipynb 8 months ago
Lorenz.ipynb 17 hours ago
R.ipynb 21 hours ago
lorenz.py 8 months ago
Launcher
Lorenz.ipynb
Clase.ipynb
Markdown
number of simulations
Exact distribution
This model describes the total number of successes of a n equal Bernoulli experiments repeated independently. The r.v. represents the number of successes, and follows a binomial distribution with parameters $n \in \mathbb{N}$ and $p = [0, 1]$.
A discrete r.v. X follows a a binomial distribution with parameters n and p if
$$P[X = x] = \binom{n}{x} p^x (1-p)^{n-x},$$
for $x = 0, 1, \ldots, n$ where
$$\binom{n}{x} = \frac{n!}{x!(n-x)!},$$
and we write $X \sim B(n, p)$.
[27]:
x1 <- 0:n_children
df6 <- data.frame(x = x1, y = dbinom(x1, n_children, p))
barfreqfamexact<-ggplot(data=df6,aes(x=df6[,1], y=df6[,2])) +
geom_bar(stat="identity", fill="steelblue",width=0.6) +
geom_errorbar(width=0.6, aes(ymax=df4[,2], ymin=df4[,2]), colour="#AA0000") +
Museums_in_DC.geojson
Leaflet | Map data (c) OpenStreetMap contributors
1024px-Hubble_Interacting.
iris.csv
Delimiter:
sepal_lengt sepal_width petal_lengt petal_width species
1 5.1 3.5 1.4 0.2 se
2 4.9 3 1.4 0.2 setosa
3 4.7 3.2 1.3 0.2 setosa
4 4.6 3.1 1.5 0.2 setosa
5 5 3.6 1.4 0.2 setosa
6 5.4 3.9 1.7 0.4 setosa
7 4.6 3.4 1.4 0.3 setosa
8 5 3.4 1.5 0.2 setosa
9 4.4 2.9 1.4 0.2 setosa
10 4.9 3.1 1.5 0.1 setosa
11 5.4 3.7 1.5 0.2 setosa
12 4.8 3.4 1.6 0.2 setosa
13 4.8 3 1.4 0.1 setosa
14 4.3 3 1.1 0.1 setosa
15 5.8 4 1.2 0.2 setosa
16 5.7 4.4 1.5 0.4 setosa
17 5.4 3.9 1.3 0.4 setosa
18 5.1 3.5 1.4 0.3 setosa
19 5.7 3.8 1.7 0.3 setosa

Conda

- is an open source **package** and **environment management system**
  - installs, runs and updates packages and their dependencies
  - creates, saves, loads and switches between environments on your local computer
- runs on Windows, macOS and Linux.
- can handle any language:
  - Python, R, Ruby, Lua, Scala, Java, JavaScript, C/ C++, FORTRAN, and more

Here a towardsdatascience's post: Getting Started with Conda

Conda

```
alex@sheldon:~$ conda env list
# conda environments:
#
base                  *  /home/alex/miniconda3
aml                      /home/alex/miniconda3/envs/aml
eads                     /home/alex/miniconda3/envs/eads
gp                       /home/alex/miniconda3/envs/gp
tf                       /home/alex/miniconda3/envs/tf
```

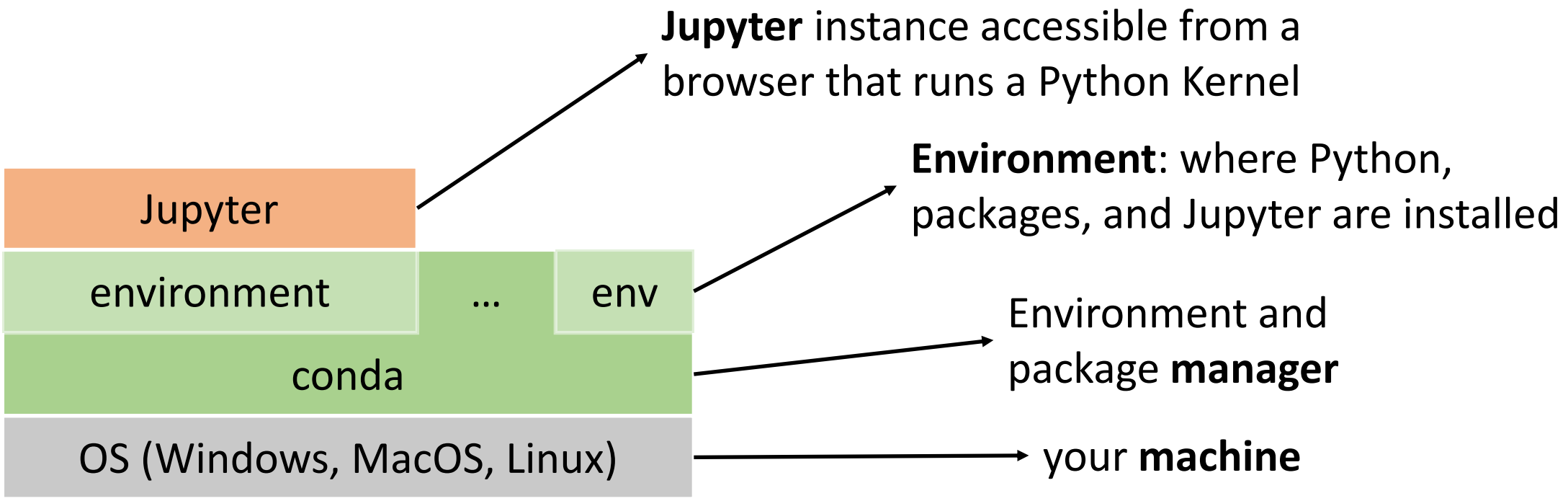
List of environment available on the machine and their location.

```
alex@sheldon:~$ conda activate base
(base) alex@sheldon:~$ python
Python 3.8.5 (default, Sep  4 2020, 07:30:14)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello world')
hello world
>>>
```

Environment that is currently active

Version of the Python installed in the active environment

Summary


Jupyter
environment
…
env
conda
OS (Windows, MacOS, Linux)
Jupyter instance accessible from a browser that runs a Python Kernel
Environment: where Python, packages, and Jupyter are installed
Environment and package **manager**
your **machine**

Let’s get started

Let’s start

We will get start with a short tutorial on Python.

- Clone/download this Github repository (https://github.com/marchioa/data-security) or download the material from virtuale.
- There, the file enviroment_setup.md lists all the instructions to set up the environment we will work in.
- Once the environment is ready and Jupyter has started you can start opening the notebooks (.ipynb files) composing the tutorial. With great imagination, 1.LETS_START.ipynb is the first one.