Environment and tools setup

This short lesson will teach you the basics of the work environments needed for this course.

Installing Python

What is python?
https://docs.python.org/3/tutorial/index.html

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

What can you do with Python? Just about anything, and most things quite easily. Topics like data analysis, machine learning, web development, desktop applications, robotics, and more are all things that you can immediately begin doing with Python without much effort. Personally, I've used Python to train some AI, to help companies detect diseases, to help detect fraud and abuse against servers, to create games, to trade stocks, and I've built and helped to build multiple businesses with Python.
Python and programming is life-changing, and it's my honor to share it with you!
https://pythonprogramming.net/introduction-learn-python-3-tutorials/

If you want to install python on your machine, you can download from here: https://www.python.org/downloads/
The python version used in this course is the python 3.6 because it's more versatile for all Data Science libraries used in this course.

In some machines (mac, linux) will be installed by default version 2.7 of python, we ask you not to use version 2.x for incompatibility issues.

Install the basic version from the website if you want to start playing with python!

Anaconda

- What is anaconda
- Install

- GUI vs Command Line

Anaconda is a an open source distribution of the Python and R programming languages and it is used in data science, machine learning, deep learning-related applications aiming at simplifying package management and deployment.

Anaconda Distribution is used by over 7 million users, and it includes more than 300 data science packages suitable for Windows, Linux, and MacOS.

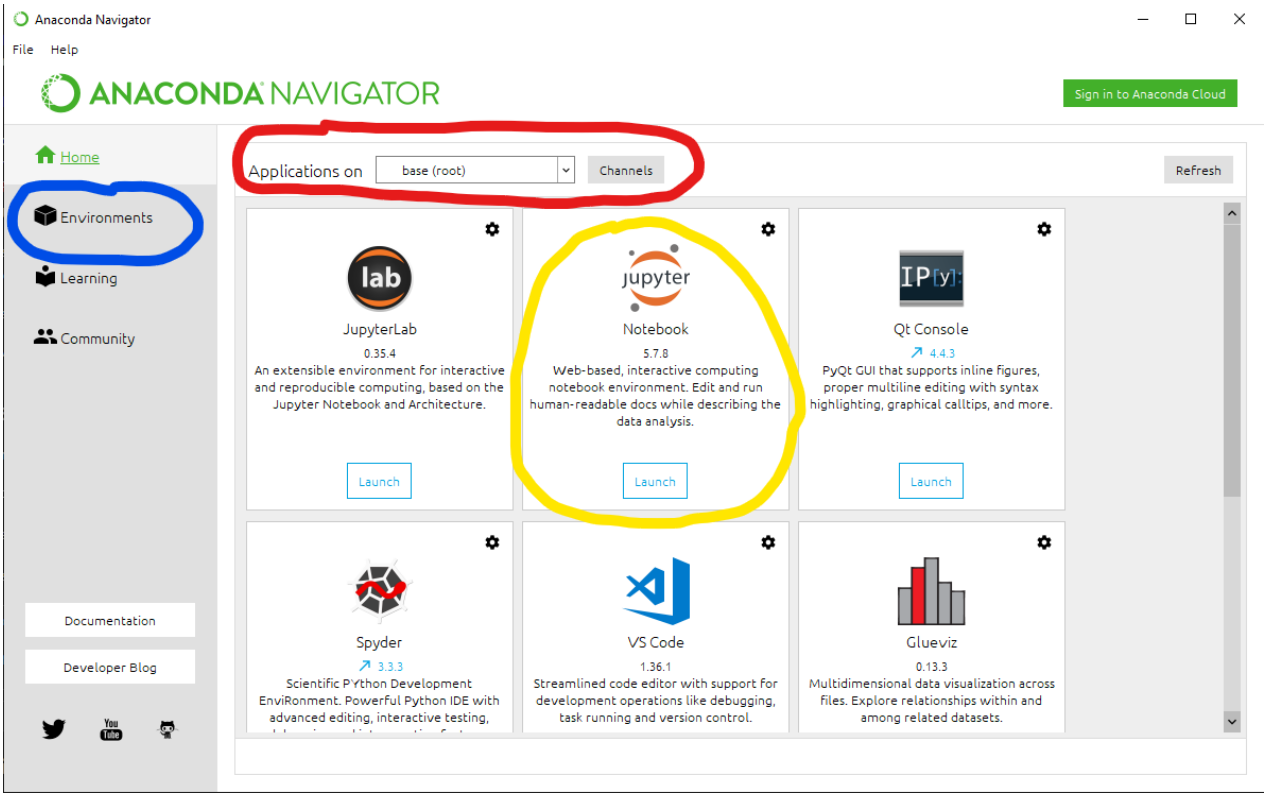
It contains all the packages that you need to start developing with Python and it's the distribution that we recommend because it's very easy to learn and to use.

If you want to install Anaconda download the 3.X version from here: https://www.anaconda.com/distribution/

Anaconda has two type of interaction:

- Graphical approach
- Terminal based approach

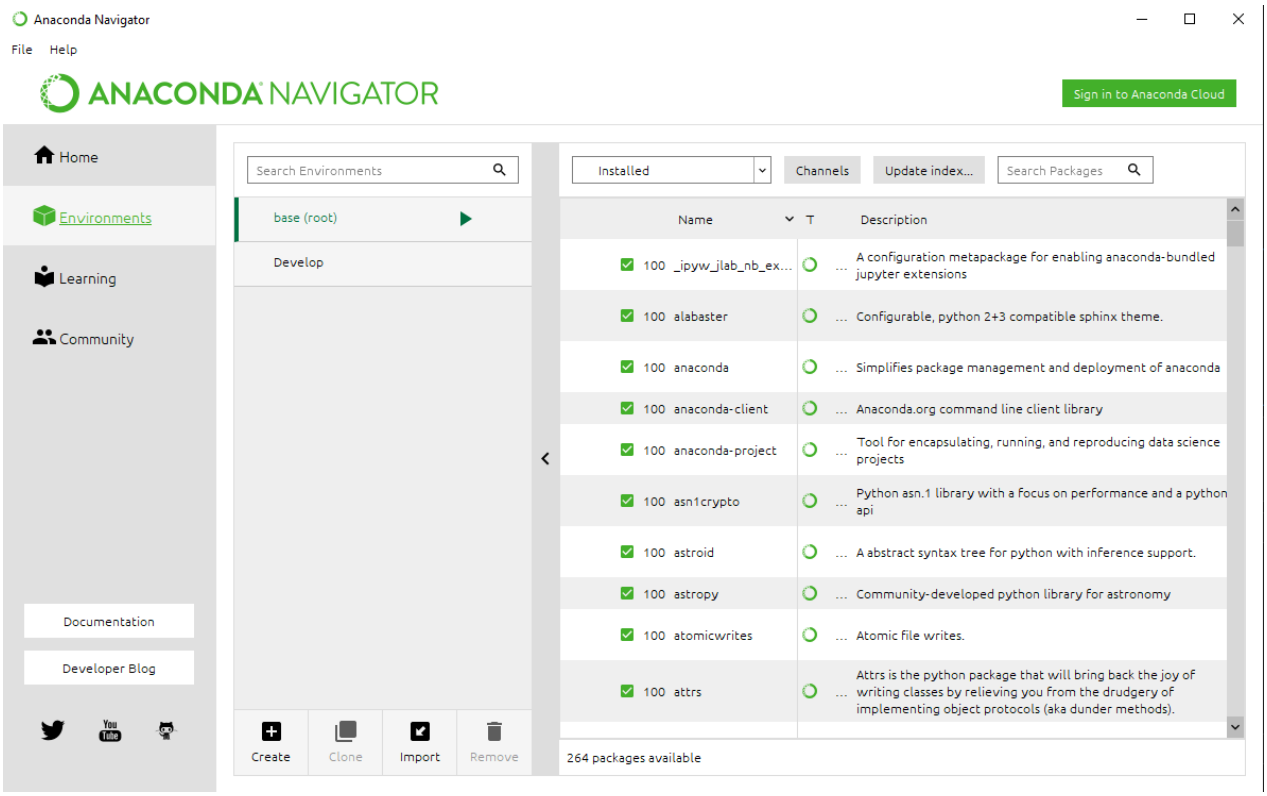
The graphical approach is with the Anaconda Navigator a GUI that can you help to use the tools



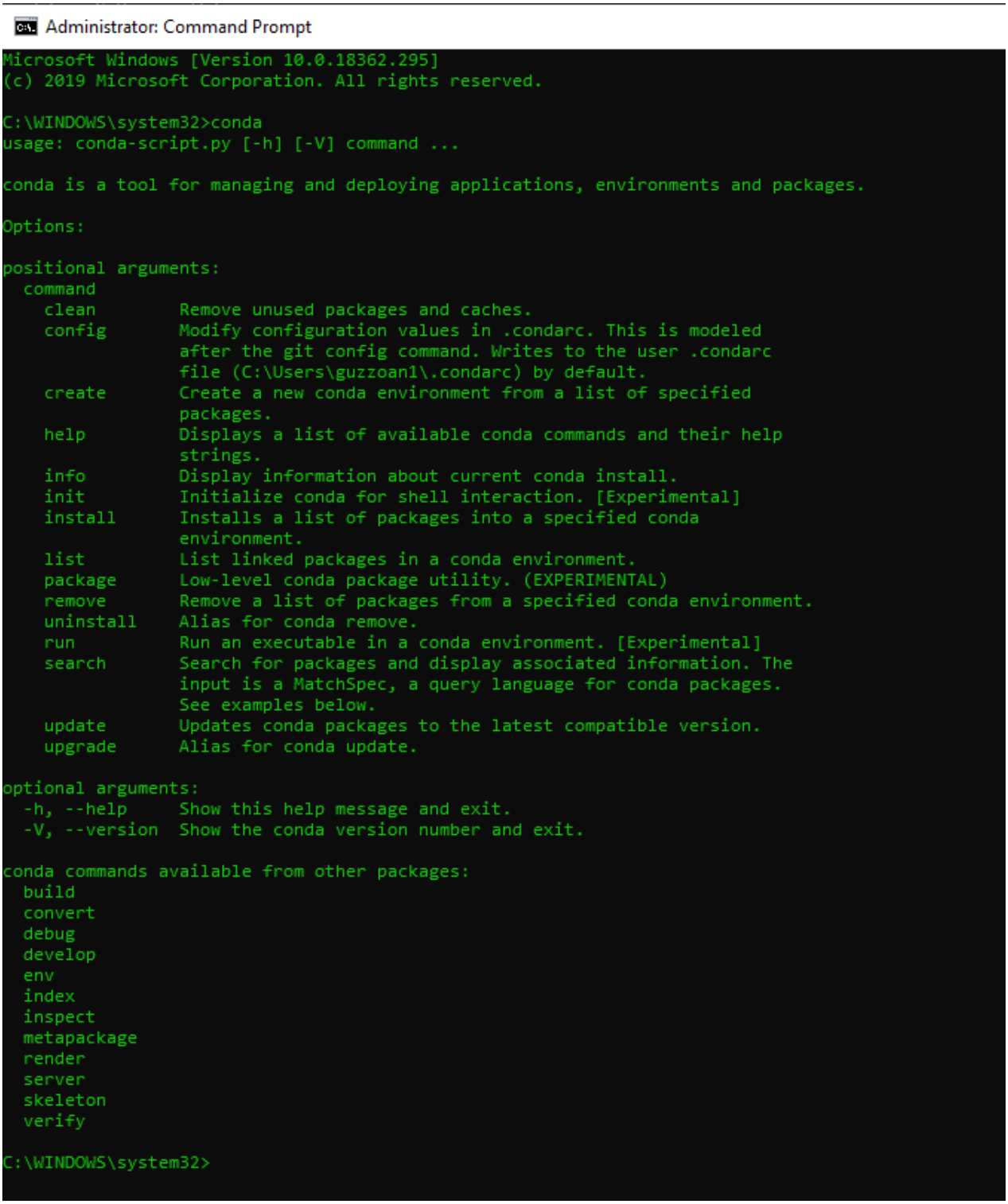
In the immage above you can see some different parts:

- The blue part: is where you can manage some different python-conda environments (we talk about this in the next cpt.)
- The red part: is where you can change the installed environments and related apps
- The yellow part: are the apps installed in a specific environment that you can use

If you open the blue part (enviroments) you can find all the environments, all the packages and you can create new environments, packages and libraries or uninstall and manage the others already in



The best way to use anaconda is with the terminal, after the installation open CMD (or your terminal app) and you can interactive with Anaconda using the command: conda



Here some useful command:

- Conda installation info
- To see your environments
- List of packages in your environments
- Update anaconda

Virtual Environments

- What is an virtual environment

- Create a new virtual environment
- Install python packages into libreries and packages (conda vs pip)
- Change environment and use different environments

The main purpose of Python virtual environments (also called venv) is to create an isolated environment for Python projects.
This means that each project can have its own dependencies, regardless of what dependencies every other project has.

In our little example above, we'd just need to create a separate virtual environment for both ProjectA and ProjectB, and we'd be good to go.
Each environment, in turn, would be able to depend on whatever version of ProjectC they choose, independent of the other.

The great thing about this is that there are no limits to the number of environments you can have since they're just directories containing a few scripts.
Plus, they're easily created using the virtualenv or pyenv command line tools.

It's possibile to create a virtual environment with default python, but we use environments with Anaconda.
For standard python info about the virtual environments, see this link below:
https://realpython.com/python-virtual-environments-a-primer/

Here some useful command to use with Anaconda to create, check, validate and update a Conda Venv

WARNING: if you are on windows, use CMD (as an admin if possibile) and try to avoid Powershell until you are confident with this technology

To visualize Conda information about your installation

```
conda -v
```

Check conda is up to date

```
conda update conda
```

Create a new virtual environment (venv) with a specific python version
Remember to replace x.x with your python version (we use principally the 3.6 version) and "yourenvname" with your environment name

```
conda create -n yourenvname python=x.x anaconda
```

If you want to create an empty environment without the default conda libraries you can do:

```
conda create -n yourenvname python=x.x
```

without the anaconda label

Now you have to activate your conda environment

```
conda activate yourenvname
```

To install a new package in your new environment you can...

```
conda install -n yourenvname [package]
```

but if you are already in your conda environment you can do simply:

```
conda install [package]
```

*always without the [ ] parenthesis

To exit from your virtual environment

```
conda deactivate
```

If you want to delete your anaconda virtual environment

```
conda remove -n yourenvname -all
```

If you want to see your installed anaconda virtual environments

```
conda env list
```

If you want remove your conda environment

```
conda remove --name yourenvname --all
```

If you want to have some info about your conda venv or if an environment is deleted launch:

```
conda info --envs
```

There are 2 types of scenario that you can use to install new python packages or libraries in Conda:

- Using pip
- Using conda

both are two library managers, the first one is the default python manager and the second is the anaconda default manager.
The available libraries that have both can be the same or different, so we suggest to use both manager but prioritising the use of Conda.

WARNING: If you are using pip, you must have your environment activated and be inside it.

If you want some other informations see this article (expecially if you want to use a custom requirements.yml file for your python libraries)
https://towardsdatascience.com/getting-started-with-python-environments-using-conda-32e9f2779307

Jupyter

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.
Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

Is the default tool for this lab and it's one of the common tools for Data Science used worldwide.

Jupyter is installed by default inside the base conda environment, but if you want to use inside your new conda virtual environment you have to install on it.

To install jupyter inside conda you have to:

1. activate your conda venv
2. launch `conda install jupyter`
3. run jupyter typing `jupyter notebook`

Everytime you want to launch Jupyter notebook with your custom conda virtual environment you have to:

1. activate your conda env
2. run: `jupyter notebook` inside the terminal

then a new browser window will appear and you can use Jupyter from there with your venv.

If you want to close jupyter

1. save your work
2. close the browser tabs
3. press: CRTL + C inside the console windows to kill all the kernels and jupyter server

Set Jupyter default project folder

You can set the default Jupyter home main folder with this simple guide

Use the jupyter notebook config file:
After you have installed Anaconda..

1. Open cmd (or Anaconda Prompt) and run jupyter notebook --generate-config.
2. This writes a file to C:\Users\username.jupyter\jupyter_notebook_config.py.
3. Browse to the file location and open it in an Editor
4. Search for the following line in the file: #c.NotebookApp.notebook_dir = ''
5. Replace by c.NotebookApp.notebook_dir = '/the/path/to/home/folder/'
6. Make sure you use forward slashes in your path and use /home/user/ instead of ~/ for your home directory, backslashes could be used if placed in double quotes even if folder name contains spaces as such : "D:\yourUserName\Any Folder\More Folders"

7. Remove the # at the beginning of the line to allow the line to execute

If you want to extend and upgrade Jupyter with new features, you can follow this guide:
https://ndres.me/post/best-jupyter-notebook-extensions/

Jupyter Lab

JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data.
JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning.
JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.

Compared to jupyter notebooks, jupyter lab is a single web page with much more functionalities and extended interface, it's almost an IDE more complex.

To install jupyter lab inside conda you have to:

1. activate your conda venv
2. launch `conda install jupyterlab`
3. run jupyter typing `jupyter lab`

Everytime you want to launch Jupyter notebook with your custom conda virtual environment you have to:

1. activate your conda env
2. run: `jupyter lab` inside the terminal

Problemi noti

Utilizzando jupyter su un environment creato a mano è possibile che non vi trovi i pacchetti installati, questo perchè si sta utilizzando jupyter installato nell'environment di default oppure in un altro ambiente, ma non nell'ambiente di riferimento della libreria installata.
Quando questo si verifica ricordate di installare jupyter all'interno del vostro nuovo ambiente (environment) di lavoro.

Visual Studio Code

https://code.visualstudio.com/

Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring.

It's a useful IDE to develop powerful and complex applications with python and it's suggested when you want to create, design, engineer and build large application or production code.

Visual Studio Code is compatible with Python and you can follow this guide to use with:
https://code.visualstudio.com/docs/python/python-tutorial

With visual studio code you can also use code cells like jupyter notebook.
They are not the same, but the use it's quite similar thanks to IPython that is the base package on which Jupyter was built.
To use notebooks follow this guide:
https://code.visualstudio.com/docs/python/jupyter-support

Here a list of useful extension for visual studio code that we use:

- Anaconda extension pack
- Code Runner
- Git History
- Git Lens
- Live share
- Powershell
- Python
- Project manager
- Shell launcher
- vscode-icons

Git

- What's Git?
- Why Git?
- How to use it
- Suggested course for GIT
- Using Github

Git is a distributed version control software
Created by Linus Torvalds in 2005 to manage Linux code
Can be used from the command line
Also available on Windows
It may have a more important "central" repository than the others

It's the basic foundamental tool to cooperate in a team, sharing code and "programming thing" with each others.

The purpose of Git is to manage a project, or a set of files, as they change over time.
Git stores this information in a data structure called a repository. A git repository contains, among other things, the following: A set of commit objects.

There are also companies that extends and use git for many purpose, two examples are: Github and GitLab.

GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface.
It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project.

Version Control repository management services like Github and GitLab are a key component in the software development workflow. In the last few years, GitHub and GitLab positioned themselves as handy assistants for developers, particularly when working in large teams.

Both, GitLab and GitHub are web-based Git repositories.

The aim of Git is to manage software development projects and its files, as they are changing over time. Git stores this information in a data structure called a repository.

Such a git repository contains a set of commit objects and a set of references to commit objects.

A git repository is a central place where developers store, share, test and collaborate on web projects.

There are some differences beetween Gitlab and Github, but the key points are the same.

Install Git

Download git from here: https://git-scm.com/downloads using POSIX emulation in windows.

Or for geeks, you can follow this guide for windows with Linux Subsytem: https://docs.microsoft.com/en-us/windows/wsl/about

Git the simple guide

http://rogerdudler.github.io/git-guide/

Git Interactive Tutorial

https://learngitbranching.js.org/

Using Github

Using GitHub is absolutely suggested and recommended to familiarize yourself with these tools for this course.

We recommend that you create an account on GitHub and use it for the projects and code that you will build in this lab and path.

Use this tutorial to understand how to use GitHub
https://product.hubspot.com/blog/git-and-github-tutorial-for-beginners