University of Science
Programming for Data Science

# Lab 0: Warm-up

Set-up Linux environment and tools

Nguyen Bao Long
baolongnguyen.mac@gmail.com

**Abstract**

The goal of this lab is to teach you how to install and use Linux as well as Python and Jupyter Notebook. There is no submission for this lab, but you have to make sure that you prepare all the needed tools and environment for the rest of subject. This lab should take you a week.

## 1 Set up Linux environment

Your Windows laptop needs to be installed a Linux environment. Then, you will install Git, Jupiter Notebook, Python,... in this environment. There are tons of reasons for you to use Linux instead of Windows. There are three options for you to install Linux environment to your laptop:

- **Option #1**: Linux only
  - Remove Windows OS and install only Linux. This option is the fastest way to help you learn Linux. However, you should mentally prepare for doing this because everything will be tough at the beginning. **Keep in mind that you have to back up your data before doing this**.
  - You can install Linux Mint because it is very easy to use. Be careful with the option **Erase disk** during installation.
- **Option #2**: Parallel installation
  - You can also install Linux in parallel with Windows. Remember to back up your data before installation.
  - It is a little bit harder than option 1 and 3 but you are an IT-er, you can do that stuff naturally.
- **Option 3**: Install Linux in Windows environment
  - **WSL-Windows Subsystem for Linux** is the phrase that you have to keep in mind if you would like to use Linux this way.
  - It is the easiest way for you to use Linux. You can watch this video for more information about installation.

## 2 Install tools

In this section, you will install some tools in the environment that you have just set up in the previous section.

## 2.1 Conda

If you do not use conda, there are some problems:

- In subject *Introduction to Data Science*, you might have studied about *Anaconda*. Anaconda is a tool that helps you on installing packages and package dependencies. The disadvantage of using this tool is the size of Anaconda ($\approx 5GB$).
- In addition, you might want to share your coding environment (which packages are used, which version of this package is used). By sharing these information, everyone can output the same result as yours while using your code.
- You might also have a need to install more than 1 environment when you work on various projects.

Solution for all these above problems:

- Install Miniconda: *Miniconda* is a subset of Anaconda which contains basic packages and **conda** is one of them (the most important package). Conda allows you to install/remove package/environment.
- Use conda to create coding environment, install needed packages or share your coding environment to others.
- After installing conda, there should be a default environment called **base** like the following image appear in your terminal:

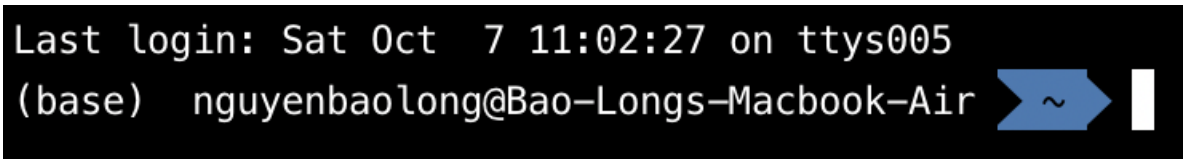


Figure 1: **base** will be your environment by default.

Use conda in terminal: The following commands are helpful and I think that you should keep in mind while using conda.

- Display all environments (activate and deactivate): `conda env list`
- Display the usage of a command (e.g `conda env list`): add the tag `--help`
  → `conda env list --help`
- An environment (like *base*) basically is a folder that contains installed packages. You can use/unuse an environment by turning it on/off.
  - Deactivate (turn off) the current environment: `conda deactivate`
  - Activate (turn on) an environment: `conda activate <env name>`. For example, activate `base`: `conda activate base`
  - Turning an environment on is actually adding the path of the environment's folder to the PATH variable (system variable) and turning an environment off is removing the path of this its folder from the PATH variable.
- Create an environment:

  - Create an environment named `meo` with package `p1, p2`:
    `conda create -n meo p1=<version> p2=<version>`
  - Create an environment from a file that contains information about the environment (you will learn how to create this file later): `conda env create --file <filename>`

- In the current coding environment:
  - Search information about a package: `conda search <package name>`
  - Display list of installed packages: `conda list`
  - Install a package: `conda install <package name>`
  - Install a specific package: `conda install <package name>=<version>`
  - Remove an installed package: `conda remove <package name>`
  - Update the coding environment (using a file that contains information about the environment): `conda env --file <filename> --force`
  - When you run `conda install`, conda will look up the packages in an online channel of Conda (named `default`), download and then install them.

- Remove a coding environment: `conda remove -n <env name> --all`

Note that there is a file mentioned above that contains information about the environment. You can use this file for sharing your environment.

## 2.2 Jupiter notebook, Python and libraries of Python

You are provided a file named `min_ds-env.yml`. This file contains all the needed packages with specific version for this subject. In order for me to grade you well, you will have to use this file to create your coding environment: `conda env create --file min_ds-env.yml`.

Some information about channel named `conda-forge` in `min_ds-env.yml`:

- This is an online repository that contains packages made by user community. The official repository of Conda is named 'default'.
- `conda-forge` contains more packages than `default` and is updated more frequently.
- In a specific coding environment, you can install packages from various channels. But you should install packages from only one channel and `conda-forge` is a good choice.

Test the new environment created from `min_ds-env.yml`:

- Type `conda activate min_ds-env` to activate the environment.
- Then, it should run the python version specified in the file. To check that:
  - Option 1: Type `which python` in your terminal
  - Option 2: Open command line mode of Python by typing `python`. Then, run:

```
import sys
sys.executable
```

- The result will be something like this:

```
(base)  nguyenbaolong@Bao-Longs-Macbook-Air ~ conda activate min_ds-env
(min_ds-env)  nguyenbaolong@Bao-Longs-Macbook-Air ~ python
Python 3.11.0 | packaged by conda-forge | (main, Oct 25 2022, 06:21:25) [Clang 14.0.4 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>> sys.executable
'/Users/nguyenbaolong/miniconda3/envs/min_ds-env/bin/python'
>>> quit()
(min_ds-env)  nguyenbaolong@Bao-Longs-Macbook-Air ~ which python
/Users/nguyenbaolong/miniconda3/envs/min_ds-env/bin/python
(min_ds-env)  nguyenbaolong@Bao-Longs-Macbook-Air ~
```

Figure 2: Test the new environment

# 3 Use Jupiter notebook at basic level

- First, download this file and save it as a `*.ipynb` file.
- In order to open this file, you change the direction (`cd`) to the path of the file and type `jupiter notebook` in your terminal. You will see a local server created and a home page will appear on your web browser (if it doesn't, read the text in your terminal to know what to do). In the home page, click on the `*.ipynb` file that you have just saved and a page will be opened in new tab.
- A notebook is made of cells. You can use up/down arrow keys to move between cells. There are 2 types of cell: **markdown cells** and **code cells**. You can write and edit text using markdown syntax in markdown cells and write Python/Julia/Command line code in code cells. There will always be `In [...]` at the beginning of a code cell.
- If you can move between cells, that means you are in **command mode**. To edit the cell, you have to switch to **Edit mode** by hitting `Enter`. After that, you can press `Esc` to get back to command mode.
- In order to run a code cell, you can try `Ctrl + Enter` or `Shift + Enter`.
- For a more detail on doing this stuff, I recommend that you should take a look at `Help/User Interface Tour`. There are also a lot of keyboard shortcuts which help you write and run code more efficiently. You can find them in `Help/Keyboard Shortcuts` (see the following picture).

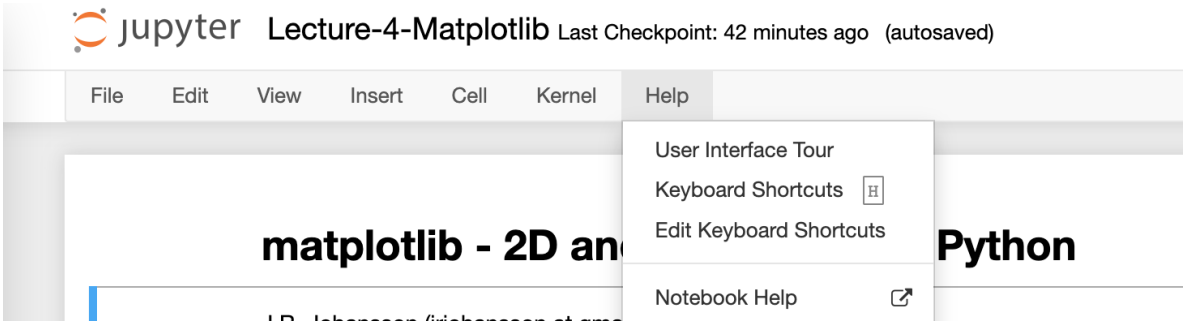


Figure 3: Guideline for newbie

- Create a new notebook: Go to home page and click `New/Python 3`.

# 4 Use Python at basic level

- Take this course on Coursera. This course is on the list of the best online courses of all-time.
- The total time of all videos is about 4 hours. You should watch all the videos (in English, turn on subtitle if needed), do the reading and quiz (optional).
- You should use Jupiter notebook (instead of IDLE) to get used to this tool. I suggest you follow these steps:
  - First, create a markdown cell for heading. For example: "Week 1 - Python, Variables and Functions".
  - Then, you create a subheading for each of section in week 1 and do the code in code cells.
- If you have already achieved a certain level on Python/Jupiter notebook, just skip the tutorial.
- **Set in Python**: In the above course, the author did not mention 'set' - a really helpful datatype. This datatype is usually used in Data Science. You should follow this tutorial (about 15 mins) to get more knowledge about set.

# References

I re-write this assignment from the Vietnamese version of Thầy Trần Trung Kiên.