**Virtual Environments**

When developing software with Python, the basic approach is to install Python on our machine, install all our required libraries, write all our code in .py files or notebook and run your Python program/script.

Consider this scenario: we are working on project A, using your system installed Python and we **pip install package==1.0** to your global Python library. Then we switch to project B and you install the same packageX but version 2.0, which has some breaking changes between version 1.0 and 2.0.

When we go back to run our project A, we might encounter errors and our application does not run. This is a common scenario we can run into when building software with Python. To get around this, we use **virtual environments**.

A virtual environment is a Python runtime environment in which we have Python interpreter, core libraries and dependencies installed into it, they are isolated from those installed in other virtual environments and any libraries installed in a “system” Python, i.e., one which is installed after downloading (or which was bundled with OS).

With this new environment, our application becomes self-contained and we get some benefits such as:

* Our development environment is contained within our project, is isolated and does not interfere with our system installed Python or other virtual environments
* We can create a new virtual environment for multiple Python versions
* We are able to download packages into our project without admin privileges
* We can easily package our application and share with other developers to replicate
* We can easily create a list of dependencies in a file for our project (requirements.txt in case of pip, for example **pip install -r requirements.txt**), which makes it easy for other developers to replicate and install all the dependencies used within our environment.

**How to create a Virtual Environment?**

In the earlier days, Virtualenv tool was used to set up Python environments. Since Python 3.3, a subset of it has been integrated into the standard library under the venv module.

Run the following in the terminal to create a new virtual environment:

**python -m venv path/env\_name**

In successful, a new folder with name env\_name will be created n the mentioned path. env\_name is the name of our virtual environment (it can be named anything we want).

In the env folder, on a Mac we will see a bin folder. We will also see scripts that are typically used to control our virtual environment, such as activate and pip to install libraries and the Python interpreter for the Python version we had used to create the environment (**This folder will be called Scripts on Windows**).

The lib folder will contain a list of libraries that we have/will install.

**How to Activate the Virtual Environment?**

On a Mac:

**source envA/bin/activate**

This will activate our virtual environment. We can observe that our terminal path includes env name at the beginning, signifying an activated virtual environment.

On Widows :

**envA/Scripts/activate.bat (Command Prompt)**

**envA/Scripts/Activate.ps1 (PowerShell)**

**How to Deactivate a Virtual Environment**

To deactivate the virtual environment, run the following code in the terminal:

On a Mac (the following in the terminal):

**~ deactivate**

On Windows:

**env/Scripts/deactivate.bat (Command Prompt)**

**deactivate (PowerShell)**

**Verify Virtual Environment usage**

On Mac/Linux:

**which python**

**which python3**

On Windows:

**where python**

**Python interpreter must be picked up from the virtual environment folder.**

**---------------------------------------------------**

**Manage Virtual environments using conda (install miniconda and then use the following commands)**

**1) conda env list**

**2) conda deactivate**

**3) conda activate (base env get activated)**

**4) conda activate env-name (activates the specified env)**

**5) conda create --name envA python=3.10**

**6) conda env remove --name env-name**

**7) conda create --clone envA --name envB**

**8) to update the python version in virtual environment**

**conda activate env-name**

**and**

**conda install python=3.10**

**9) conda search python (get list of supported versions)**

**10) conda install notebook**