**Virtual Environments in Python**

**What is Virtual Environment?**

Virtual environments in Python allow you to have different versions of packages and libraries for different projects you work on regardless of your global setup. For example, you might have package v2 installed on your machine but one of your projects depends on v1 of the same package. Regardless of what your global setup looks like, you can create a virtual environment for your project and use package v1. Note that, v2 will still be your default version of that package but you’ll be able to use v1 in your virtual environment. You can have as many different virtual environments on your machine as you want but it is important to keep track of where they are and what you name them. In Python, we use virtualenv to create a virtual environment.

**Setting up a Virtual Environment**

Let’s first install virtualenv.

**pip3 install virtualenv**

Now let’s check if the setup worked.

**virtualenv --version**

Now that you’ve ensured that virtualenv is installed on your system, we can move on to creating our first virtual environment. But before that, let’s create a folder to put the environment details inside.

**mkdir my\_project**  
**cd my\_project**

Okay, we’ve created a new folder and we are inside it. This folder will ideally host all you project files. Now, let’s create a new virtual environment (venv) for Python3.

**virtualenv --python=python3 funProjectDependencies**

This will create a new folder in the current directory with Python3 executable files and a copy of the **pip3** library so that you can install new packages in the environment.

If you don’t want to create a new folder inside your project to store the virtual environment, then you can instead run:

**virtualenv --python=python3 .**

This will create all the executables in the current directory but it is always recommended to create a separate folder as it keeps everything organised.

**Note:** Leaving out the flag --python=python3 in the above commands will setup the venv for the version of Python that is called by python, if it exists. On many systems like the CSE machines, python calls to Python2, which is not the version we want for this course.

**Using a Virtual Environment**

To run the virtual environment, use

**source funProjectDependencies/bin/activate**

This will start the virtual environment. You should be able to see the name of the virtual environment on the left of the prompt from your terminal.

If you do a **pip list** or **pip3 list**, you’ll be able to see all the packages that are installed in the current directory (read the section *“Pip or Pip3?”* below for why we can use either commands).

Since the virtual environment is activated, any packages you install now will be added to the virtual environment. To see this, try

**pip3 install flask**

and then do a **pip3 list**. You should be able to see the newly installed package there. Flask also comes with a few extensions that will be installed when flask is installed. You should be able to see them there as well.

It is always recommended that you keep a documented list of all the packages used in a certain project so that anyone collaborating can install the same versions of the same packages and run your project seamlessly. To do that, run

**pip3 freeze > requirements.txt**

This will store all the details of the packages that are being used in this virtual environment. Collaborators can simply run the following command install all the packages from the requirements.txt in their virtual environment.

**pip3 install -r requirements.txt**

If you want to deactivate the current virtual environment, just type

**deactivate**

and you should see the name of the virtual environment no longer shows up in the prompt from the terminal.

You can create virtual environments for each project you work on.

**Pip or Pip3?**

pip is a package manager for Python, and it is the recommended way to install your Python packages. pip3 is a version of pip for Python3.

In the case of CSE machines, by default, using pip will install packages for Python2 (and pip3 for Python3).

However, if you setup a virtual environment with the --python=python3 flag on, using pip within the venv will now have the same affect as using pip3. This is because pip in the venv will simply be pointing to (symlinked to) the pip3 installed in the venv; in otherwords, executing pip will actually execute pip3 behind the scenes.