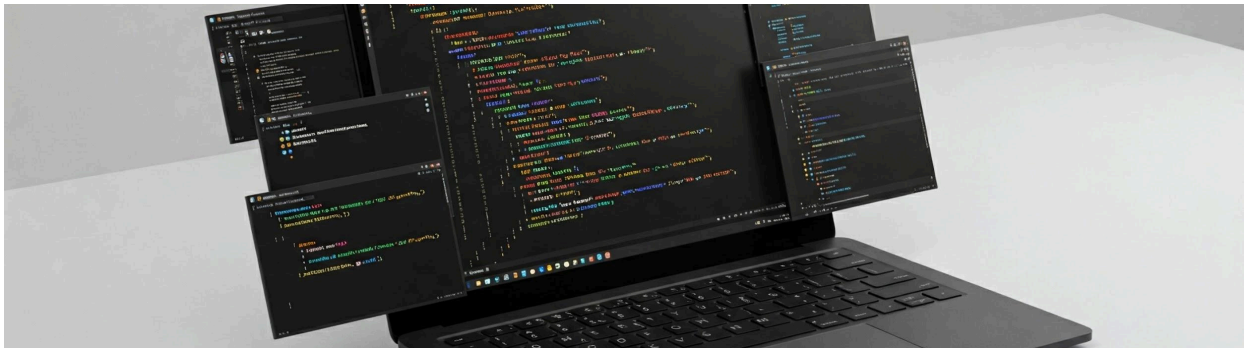


Understanding Python Virtual Environments: What, Why, and How

What is a Python Virtual Environment?

A virtual environment is an **isolated Python environment** that allows you to install packages and dependencies for a specific project without affecting other projects or the system-wide Python installation.

Why Use Virtual Environments?

- **Avoid conflicts**, and **maintain clean**, organized **projects**.
- Virtual environments allow you to **manage** these **dependencies** separately for projects.
- Virtual environments ensure that **everyone working** on a project has the **exact same** set of **packages and versions**

How to Create and Use Virtual Environments

1. Check if Python is Installed

Before creating a virtual environment, ensure Python is installed on your system.

2. Create a Virtual Environment

Python comes with **venv**, a **built-in module** for **creating virtual environments**. Create and Navigate to your project directory in the command prompt as below:

Note: It is **recommended** to create **virtual environments** for each **project** in **same project directory** or create and **maintain** all **virtual environments in single folder**

```
Python
#create new folder name sample project
> mkdir C:\Users\santhosh\sample_project

#move to sample project
> cd C:\Users\santhosh\sample_project

#create virutal environment
> python -m venv sample_venv
```

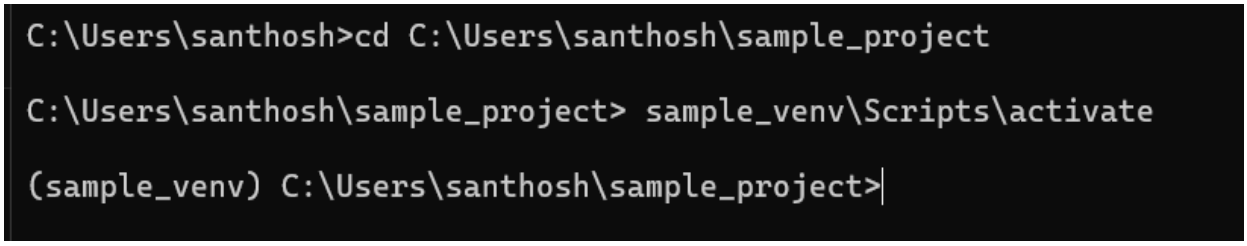
3. Activate the Virtual Environment

To use the virtual environment, you need to **activate** it. The activation command varies depending on your operating system:

- **Windows:**

```
Unset
> sample_venv\Scripts\activate
```

After activation, your terminal prompt will show the name of the virtual environment in parentheses, indicating that it's active.



4. Install Packages

With the virtual environment activated, you can install packages using `pip`:

```
Unset
> pip install package_name
```

For example, to install the `requests` library:

```
Unset
> pip install requests
```

These packages will be installed only in your virtual environment, not system-wide.

5. Deactivate the Virtual Environment

When you're finished working in the virtual environment, you can deactivate it using:

```
Unset
> deactivate
```

Note: **Python Libraries installed** after **deactivating** the **virtual environment**, will get **installed outside** the **virtual environment.**

6. Installing Packages using pip

Open command prompt Activate virtual environment and user **requirements.txt** file **(available in git repo Courses, under week 0 → Day 1 → requirements.txt)**, then enter below command

```
Python

#activate virtual environment
> C:\Users\santhosh\sample_project\sample_venv\Scripts\activate
```

```
#move to local repository
> cd C:\Users\santhosh\local_repo\Courses\Week0\Day1

#install requirements.txt
> pip install -r requirements.txt
```

7. Virtual Environment Commands Summary

Here's a quick recap of the essential commands:

Command	Description
`python -m venv venv_name`	Creates a new virtual environment.
`venv_name\Scripts\activate` (Windows)	Activates the virtual environment on Windows.
`pip install package_name`	Installs a package within the virtual environment.
`deactivate`	Deactivates the virtual environment.