TecAdmin

Home » Programming » Python » How to Setup Python Virtual Environment on Ubuntu & Debian

How to Setup Python Virtual Environment on Ubuntu & Debian

By Rahul — March 17, 2020 4 Mins Read — Updated: October 2, 2021

Python3 venv module is used for creating virtual environments. It helps us to create and manage isolated environments for Python applications with different versions of Python. These are the lightweight environment that has its own independent set of Python packages in its site directories.

For example, you need to deploy multiple Python applications on a single server, which required different-2 module versions. In that situation, you may face a conflict between module versions. The Python virtual environment plays its role to isolate the application environment within the application.

This tutorial will help you to create a Python virtual environment on Ubuntu and Debian Linux systems.

Prerequsities

This article assumes, that you already have installed Python 3 or Python 2.7 on your system.

Step 1 – Install venv

First of all, you need to install the Python module for the virtual environment on your system. Python3 users can directly install the package for the env. The Python 2.7 users need to install **virtualenv** Python module. This will also install other required modules on your system.

- **For Python3:**

```
$ sudo apt install python3-venv
```

- **For Python 2.7:**

```
$ sudo pip2 install virtualenv
```

Step 2 – Create Virtual Environment

Once the installation is finished. Let's create an isolated Python environment for your application.

01. Locate the python binary files location on your system. I have just installed Python 3.9 and Python 2.7 on our Debian system. In my case both binary files are located at `/usr/bin/python3.9` and `/usr/bin/python2.7` .

```
$ which python2.7
/usr/bin/python2.7

$ which python3.9
/usr/bin/python3.9
```

You can choose any other Python version as per the application requirements.

02. Now, Create a separate environment for your Python Application. We are usir **venv** as the environment directory name, but you can use any other name of your choice. You can also define the environment directory in a different location.

First, navigate the your Python project directory.

```
$ cd myPythonApp
```

03. Then create Python isolated environment based on the Python version requirements.

- **For Python 3.9:**

  ```
  $ /usr/bin/python3.9 -m venv venv
  ```

- **For Python 2.7:**

  ```
  $ virtualenv -p /usr/bin/python2.7 venv
  ```

The above commands create a directory named **venv** in the current directory with a local copy of files. While working on this website, you should activate the local environment in order to make sure you're working with the right versions of your tools and packages.

Step 3 – Activate Virtual Environment

To work with a Python virtual environment, you need to activate the environment. After that, you can install a required module for your Python project as well as run your Python application in an isolated environment.

Use the following command to active the Python environment:

```
$ source venv/bin/activate
```

Any package that you install using pip is now placed in the virtual environments project folder, isolated from the global Python installation.

Use pip3 to install a module. To install the most commonly used ‘requests’ module, type:

```
(venv) rahul@tecadmin:~/myPythonApp$ pip3 install requests
```

```
Output
```

```
Collecting requests
  Downloading requests-2.26.0-py2.py3-none-any.whl (62 kB)
     |████████████████████████████████| 62 kB 1.8 MB/s
Collecting idna<4,>=2.5
  Downloading idna-3.2-py3-none-any.whl (59 kB)
     |████████████████████████████████| 59 kB 12.0 MB/s
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.7-py2.py3-none-any.whl (138 kB)
     |████████████████████████████████| 138 kB 35.3 MB/s
Collecting charset-normalizer~=2.0.0
  Downloading charset_normalizer-2.0.6-py3-none-any.whl (37 kB)
Collecting certifi>=2017.4.17
  Downloading certifi-2021.5.30-py2.py3-none-any.whl (145 kB)
     |████████████████████████████████| 145 kB 36.4 MB/s
Installing collected packages: urllib3, idna, charset-normalizer, certifi,
Successfully installed certifi-2021.5.30 charset-normalizer-2.0.6 idna-3.2
```

All the installed modules files are placed at **venv/lib/python3.9/site-packages** directory.

Step 4 – Deactivate Python Virtual Environment

After finishing your work inside the virtual environment, just type the “deactivate command to exit from the isolated environment prompt.

```
(venv) rahul@tecadmin:~/myPythonApp$ deactivate
```

You will get the default system prompt.

Step 5 – Delete the Virtual Environment

To Delete the Python virtual environment from your application. Simply delete the **venv** directory from your application folder.

```
$ rm -rf venv
```

Conclusion

In this tutorial, you have learned to create Python virtual environment on Ubuntu, Debian and other Debian derivative Linux systems. The Python virtual environment helps us to deploy multiple Python application on single server without making conflicts for modules between each other.

env Python virtualenv

‹ PREVIOUS ARTICLE
How to Install Gradle on Ubuntu 18.04

NEXT ARTICLE ›
How to Install and Configure Gitlab on CentOS 8

RELATED POSTS

How to Switch Python Version in Ubuntu & Debian

October 5, 2021

Working with Python Functions

October 4, 2021

How To Install PIP on Debian 11 Linux

October 1, 2021

2 COMMENTS

Mohammad javad on February 9, 2021 11:49 am

thank you very much, sir

REPLY ›

Pedro Riva on June 20, 2020 7:25 am

The official recommendation is to use venv instead of virtualenv. What do you think about?

REPLY ›

LEAVE A REPLY

Your Comment

Name *

Email *

Website

□ Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

Privacy - Terms

LATEST ARTICLES

How To Install and Switch Desktop Environments in Fedora

October 15, 2021

How to Git Remove File but Keep Local Version

August 9, 2021

How to Clone a Git Repository Into a Specific Folder

October 14, 2021

How To Install NVIDIA Drivers on Fedora

October 13, 2021

TAGS

Apache aws bash CentOS 8 command cpanel database date debian Debian 9 Debian 10 Docker fedora file git Java javascript js lamp Laravel linux mail mariadb mongodb MySQL nginx node node.js NodeJs npm nvm PHP php framework PostgreSQL Python

Python3 security SSH SSL string tomcat Ubuntu Ubuntu 18.04 Ubuntu 20.04 windows