**Python - venv — 가상 환경 생성**

<https://docs.python.org/ko/3.8/library/venv.html>

<https://dgkim5360.tistory.com/entry/python-virtualenv-on-linux-ubuntu-and-windows>

<https://blog.ugonfor.kr/138>

### **왜 사용해야 하는 가?**

파이썬을 사용하다보면, pip 명령어로 모듈을 굉장히 많이 사용하게 된다. 그런데, github에 올라와있는 코드들이 특정 모듈에 대해서만 작동한다던가 혹은 본인이 사용하던 모듈을 다운그레이드, 업그레이드 해야하는 상황이 생기면 호환성이 망가지게 된다. 그래서 파이썬 가상환경(Virtualenv)를 제공하고, 깃헙 repository에는 보통 requirement.txt를 두고서 파이썬 모듈에 대한 정보들이 나타나게 된다.

* virtualenv는 파이썬 가상환경을 설정하는 모듈이다.
* requirement.txt는 파이썬 모듈 정보(버전, 리스트)를 담은 파일이다.

### **virtualenv**

$ pip install virtualenv

$ virtualenv venv

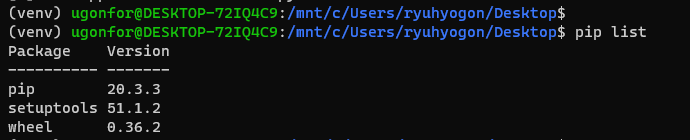
$ source venv/bin/activate

* 위 명령어를 실행하게 되면, venv라는 폴더가 생성되고 아래 사진처럼 앞에 (venv)가 붙게 된다. 이때, 2번째 줄의 venv를 원하는 대로 바꿔주면, 가상환경의 이름을 바꿔줄 수 있다.



**source venv/bin/activate**를 통해서 가상환경을 실행할 수 있고, 나가고 싶을 때는 **deactivate** 를 입력하면 나가진다.

이렇게 가상환경을 설정하게 되면, 아래처럼 pip list를 했을 때 나타나는 모듈이 초기상태로 바뀌게 되고, 원하는 모듈들을 설치하여 사용하면 된다.



참고로 venv에서 사용되는 파이썬 버전을 설정하고 싶을 때는 다음과 같이 하면 된다.

# python 2

$ python -m virtualenv venv

$ virtualenv venv --python=python

$ virtualenv venv --python=python2.7

# python 3

$ python3 -m virtualenv venv

$ virtualenv venv --python=python3

$ virtualenv venv --python=python3.5

출처: https://dgkim5360.tistory.com/entry/python-virtualenv-on-linux-ubuntu-and-windows [개발새발로그]

위처럼 하면, 각 버전에 맞게 파이썬이 생기게 된다.

### **requirement.txt**

requirement.txt를 생성해야하는 경우는 본인이 사용하고 있는 모듈 리스트를 알려주고 싶을 때이다.

위 파일이 없다면 git clone을 했을 때, user는 버전에 맞는 모듈들을 모두 설치해야 하기에 엄청난 고생을 해야 한다...

#### requirement.txt가 주어졌을 때

(venv) $ pip install -r requirements.txt

위와 같이 명령어를 입력하면 requirement.txt에 해당하는 모듈들이 모두 정상적으로 설치된다.

#### requirement.txt를 생성할 때

(venv) $ pip freeze > requirements.txt

위와 같이 명령어를 입력하면 requirement.txt에 본인이 사용중인 모듈을 저장하게 된다.

### 마무리

virtualenv를 통해서 모듈들을 잘 관리를 하도록 하자. pip list로 local에다가 깔아버리면 충돌이 나는 경우가 있기에 venv를 잘 사용해야 한다.

나중에 git에 올리려는 프로젝트들은 venv로 모듈들을 잘 관리해서 requirement.txt를 생성해놓는 것도 중요하다.

**Python3 가상환경**

# python3 버전 확인

python3 -V

# Or python3 설치

# python3 -m venv ./(생성하고자 하는 가상환경 이름)

python3 -m venv ./myenv

# source (가상환경 이름)/bin/activate

source myenv/bin/activate

# 가상환경 비활성화

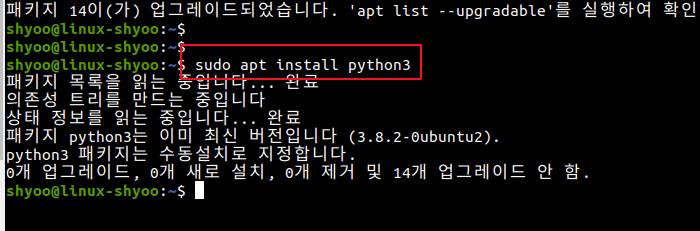
deactivate

# 가상환경 삭제 : 가상환경 비활성화 및 가상환경 디렉토리 삭제

rm -rf (가상환경 이름)

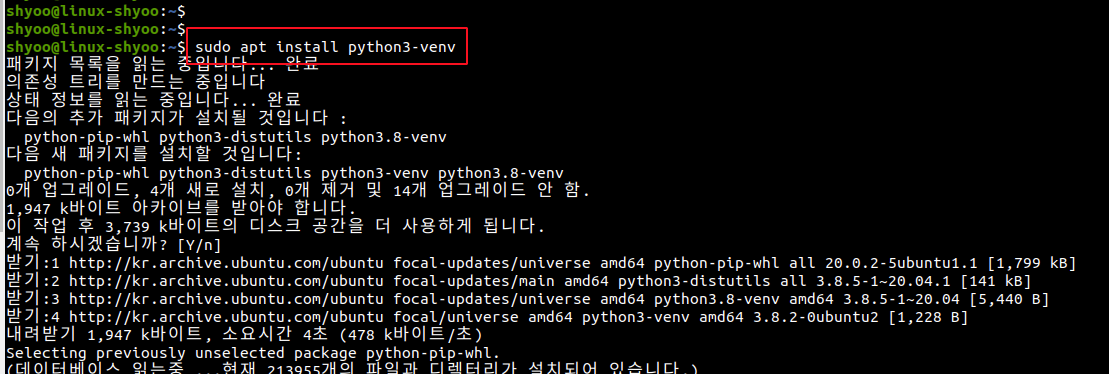
1. Python3 설치

**sudo apt install python3**



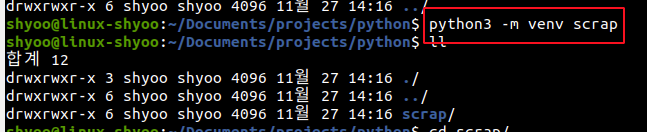
2. Python3-venv 설치

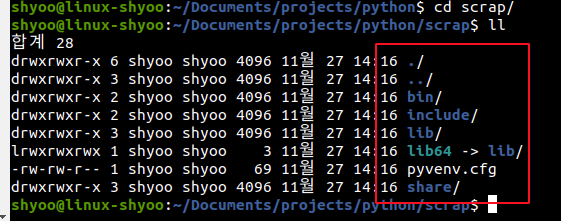
**sudo apt install python3-venv**



3. 가상 환경 세팅

**python3 -m venv scrap(가상 공간 이름)**





세팅이 잘 되는지를 확인하기 위해 생성된 가상 공간 이름의 디렉토리로 접근했을 때, 위와 같은 바이너리, 헤더, 라이브러리 등이 존재하면 세팅이 잘 진행된 것이다.

4. 가상 환경 활성화

**source 가상 환경 경로/bin/activate**

