

# 파이썬 가상환경의 이해

# 가상환경의 이해

## 가상환경(Virtual Environments)

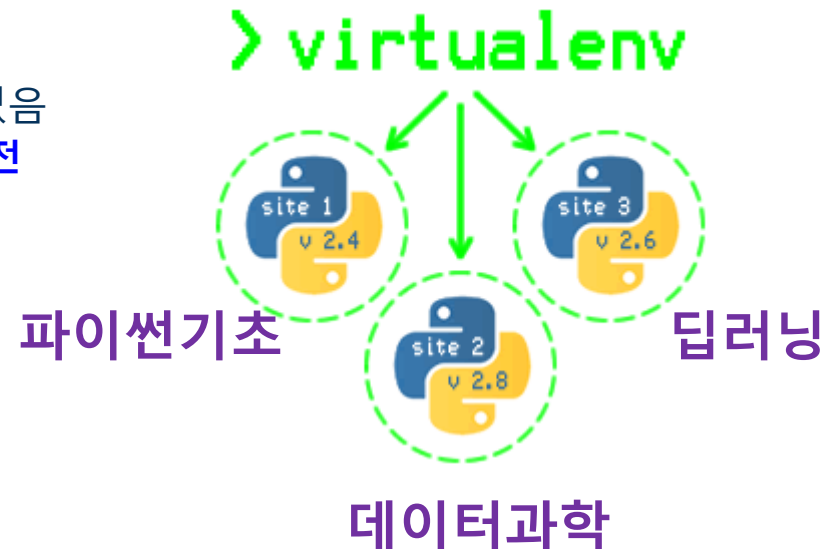
- 격리된 별도의 라이브러리 설치 폴더
  - 원하는 Python 환경을 구축하기 위해 필요한 모듈만 담아 놓는 바구니
- 같은 모듈을 사용한다고 하더라도 다른 버전이 필요한 경우 발생
  - Tensorflow 1.15.0
    - Session 사용
  - Tensorflow 2.0.0
    - Session 미사용

## 가상환경의 필요성

- 하나의 환경에는 여러 버전의 설치를 할 수 없음
  - 여러 환경을 만들고 각각의 환경에 여러 버전 설치 가능
- python 프로그램을 실행하기 위한 다양한 환경을 마련

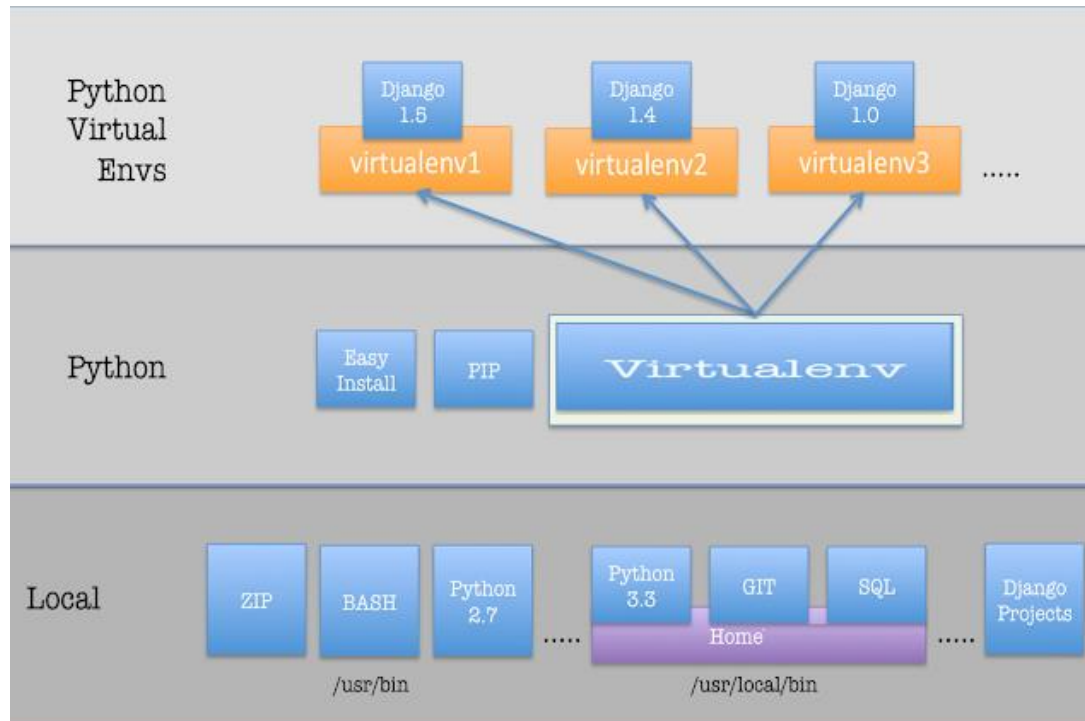
## 가상환경의 주요 목적

- Python 프로젝트를 위한 격리된 개발환경을 만드는 것
  - 다른 모든 프로젝트의 종속성에 관계없이 각 프로젝트마다 고유한 종속성을 가질 수 있음을 의미



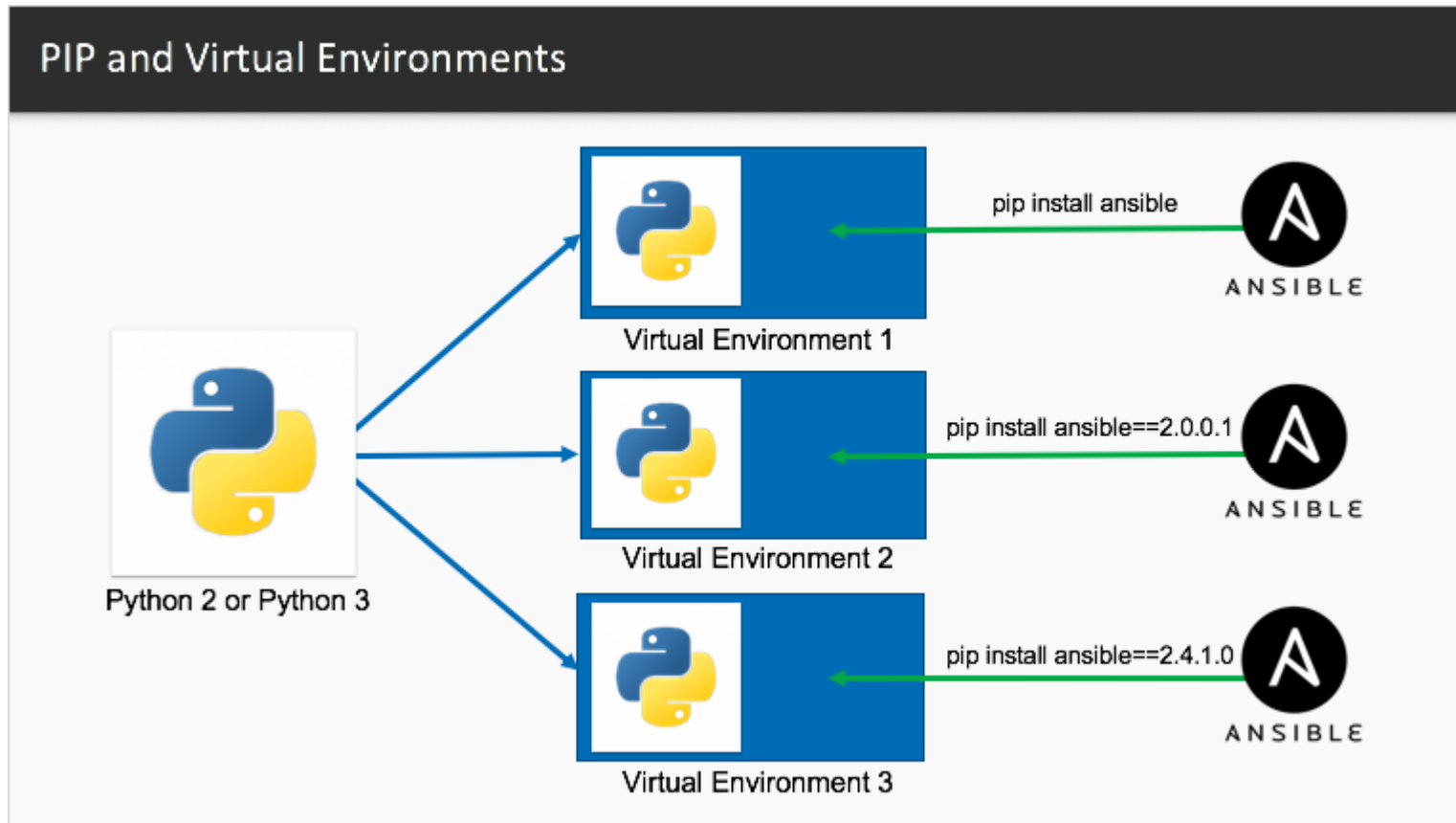
# 가상환경 생성 방법

- Python에서 가상환경을 만드는 방법
  - 크게 2가지로 virtualenv와 conda를 사용
    - venv, pipenv 등 매우 다양



# 가상환경의 개념

- 다른 모듈, 동일 모듈의 다른 버전 등 설치



# 가상환경 생성 절차와 모듈 설치 pip 명령

# 가상환경 생성 4단계 절차

- 새로운 가상환경을 생성
  - virtualenv, venv, pipenv, conda 등 매우 다양
  - venv만 설치 불필요
    - `pip install virtualenv`
    - `pip install pipenv`
    - 아나콘다 설치
- 생성된 가상환경을 활성화(들여가기)
  - `activate`
- 활성화된 가상환경 내에서 필요한 라이브러리를 설치하고, 개발을 진행
  - virtualenv, venv
    - `pip install 모듈명`
  - pipenv
    - `pipenv install 모듈명`
      - `pip install 모듈명` 도 가능
  - conda
    - `conda install 모듈명`
- 현재 활성화된 가상환경을 비활성화(빠져나오기)
  - `deactivate`

# 모듈 설치 명령 pip

- pip 명령

pip 명령	설명
pip search {키워드}	키워드로 관련 패키지를 검색합니다.
pip install {패키지명}	지정한 패키지를 설치합니다.
pip uninstall {패키지명}	지정한 패키지를 삭제합니다.
pip install --upgrade {패키지}	지정한 패키지를 업데이트합니다.
pip show {패키지}	지정한 패키지의 버전을 출력합니다.
pip list	전체 패키지 목록을 출력합니다.

# pip로 패키지 목록 관리하기

- pip는 파이썬 환경의 모든 라이브러리를 조회하여 출력 기능을 제공
  - 파일 “requirements” 파일을 읽어서 기술된 모든 패키지를 설치하는 기능도 제공

```
$ pip freeze > requirements.txt
```

“requirements” 파일에는 현재 설치된 모든 라이브러리 목록이 저장됩니다. “requirements” 파일은 다음과 같은 형태의 정보를 포함합니다.

```
ipython==6.1.0
ipython-genutils==0.2.0
ipywidgets==7.0.0
jedi==0.10.2
Jinja2==2.9.6
jsonschema==2.6.0
jupyter==1.0.0
jupyter-client==5.1.0
jupyter-console==5.2.0
jupyter-core==4.3.0
MarkupSafe==1.0
mistune==0.7.4
nbconvert==5.3.1
nbformat==4.4.0
notebook==5.0.0
pandocfilters==1.4.2
pexpect==4.2.1
pickleshare==0.7.4
prompt-toolkit==1.0.15
```

“requirements” 파일에 기술된 모든 라이브러리를 설치하는 pip 명령은 다음과 같습니다.

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



virtualenv, venv, pipenv  
활용 가상환경 생성

# 지정한 폴더에 각각의 도구로 가상환경 생성

## • 상위 폴더 ve 하부에 각각 생성

- virtualenv 로 생성
  - **venv**
- venv 로 생성
  - **venv\_test**
- pipenv 로 생성
  - **penv**
    - 이 폴더는 가상환경이 지정된 프로젝트 폴더 의미

```

C:\Windows\System32\cmd.exe
D:\we>dir
D 드라이브의 볼륨: DATADRIE0
볼륨 일련 번호: 0AEE-91E2

D:\we 디렉터리

2020-01-15 오전 10:32 <DIR>      .
2020-01-15 오전 10:32 <DIR>      ..
2020-01-14 오후 06:37 <DIR>      mini
2020-01-15 오전 10:42 <DIR>      penv
2020-01-14 오후 01:12 <DIR>      venv
2020-01-14 오후 06:10 <DIR>      venv_test
                                0개 파일              0 바이트
                                6개 디렉터리  1,985,650,483,200 바이트 남음

D:\we>
  
```

# virtualenv 사용 방법

## • 모듈 virtualenv 설치 필요

- pip install virtualenv

## • 가상환경 venv 만들기

- 적당한 폴더로 이동

- D:
- md ve
- cd ve

- 가상환경 venv 생성

- virtualenv venv

- 가상환경 활성화

- venv\Scripts\activate

- 가상환경에 필요 모듈 설치

- pip install numpy

- 가상환경에서 프로젝트 개발

- Python

- 가상환경에서 나오기

- deactivate

```
C:\Windows\System32\cmd.exe
D:\>md ve
D:\>cd ve
D:\ve>pip install virtualenv
Collecting virtualenv
  Using cached https://files.pythonhosted.org/packages/05/f1/2e07e8ca50e047b9cc9ad56cf4291f4e041fa73207d000a095fe478abf84/virtualenv-16.7.9-py2.py3-none-any.whl
Installing collected packages: virtualenv
Successfully installed virtualenv-16.7.9
WARNING: You are using pip version 19.2.3, however version 19.3.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
D:\ve>python -m pip install --upgrade pip
Collecting pip
  Using cached https://files.pythonhosted.org/packages/00/b6/9cfa56b4081ad13874b0c6f96af8ce16cfc1cb06bedf8e9164ce551ec1/pip-19.3.1-py2.py3-none-any.whl
Installing collected packages: pip
Found existing installation: pip 19.2.3
Uninstalling pip-19.2.3:
  Successfully uninstalled pip-19.2.3
Successfully installed pip-19.3.1
D:\ve>
```

가상환경이름

python -m pip install --upgrade pip

```
C:\Windows\System32\cmd.exe
Successfully uninstalled pip-19.2.3
Successfully installed pip-19.3.1
D:\ve>virtualenv venv
Using base prefix 'd:\python38-32'
New python executable in D:\ve\venv\Scripts\python.exe
Installing setuptools, pip, wheel...
done.
D:\ve>dir
D 드라이브의 볼륨: DATADRVIVE
볼륨 일련 번호: 0AEE-91E2

D:\ve 디렉터리

2020-01-14 오후 01:12 <DIR>          .
2020-01-14 오후 01:12 <DIR>          ..
2020-01-14 오후 01:12 <DIR>          venv
               0개 파일              0 바이트
               3개 디렉터리 1,986,222,137,344 바이트 남음

D:\ve>venv\scripts\activate
'venv'은(는) 내부 또는 외부 명령, 실행할 수 있는 프로그램, 또는
배치 파일이 아닙니다.
D:\ve>venv\scripts\activate
(venv) D:\ve>pip install numpy
Collecting numpy
  Downloading https://files.pythonhosted.org/packages/0e/c3/be53614c4e3490778050e1df48fd463837297d5dd402dae3b500f2050eba/numpy-1.18.1-cp38-cp38-win32.whl (10.8MB)
Installing collected packages: numpy
Successfully installed numpy-1.18.1
(venv) D:\ve>
```

Python

# 가상환경 **venv**에서 프로젝트 개발

- **numpy 테스트**
  - 브로드캐스팅과 행렬 모양 변환

C:\Windows\System32\cmd.exe - python

```
(venv) D:\#ve>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> a = np.array([[1, 2]])
>>> a
array([[1, 2]])
>>> b = np.array([[1], [2]])
>>> print(b)
[[1]
 [2]]
>>> c = a + b
>>> print(c)
[[2 3]
 [3 4]]
>>> x = np.array(np.random.random(10))
>>> x
array([0.62964497, 0.20982989, 0.39815468, 0.41462358, 0.41903129,
        0.11562117, 0.54325543, 0.16417797, 0.7962333 , 0.77974707])
>>> y = x.reshape(2, 5)
>>> y
array([[0.62964497, 0.20982989, 0.39815468, 0.41462358, 0.41903129],
       [0.11562117, 0.54325543, 0.16417797, 0.7962333 , 0.77974707]])
>>> print(y)
[[0.62964497 0.20982989 0.39815468 0.41462358 0.41903129]
 [0.11562117 0.54325543 0.16417797 0.7962333 0.77974707]]
>>> -
```

# 일반 리스트 더하기

```
m = [1, 2, 3]
```

```
n = [4, 5, 6]
```

```
print(m + n)
```

```
# print(m - n)
```

**import numpy as np**

# 브로드캐스팅

```
a = np.array([[1, 2]])
```

```
print(a)
```

```
b = np.array([[1], [2]])
```

```
print(b)
```

```
c = a + b
```

```
print(c)
```

# 행렬 모양 바꾸기

```
x = np.array(np.random.random(10))
```

```
print(x)
```

```
y = x.reshape(2, 5)
```

```
print(y)
```

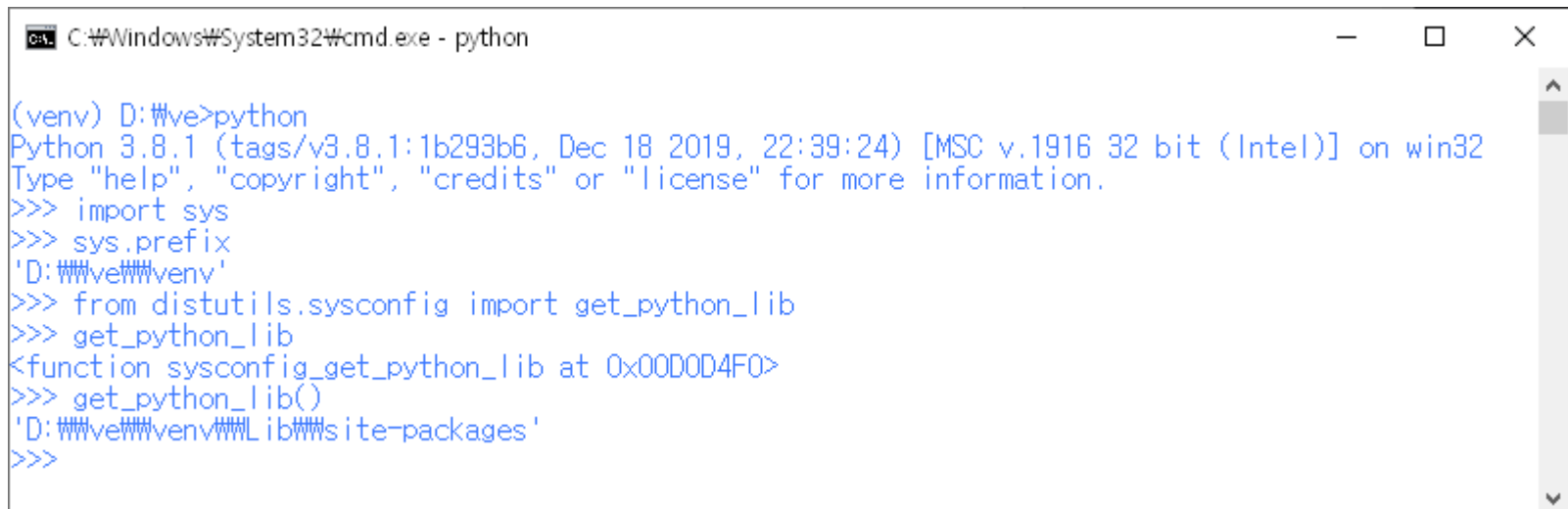
# 가상환경 확인 방법

## 가상환경의 인터프리터에서 다음 확인

- 가상환경 폴더 확인
- 외부모듈 설치폴더 확인

```
>>> import site
>>> site.getsitepackages()
```

```
>>> import sys
>>> sys.prefix
'D:\\ve\\venv'
>>> from distutils.sysconfig import get_python_lib
>>> print(get_python_lib())
D:\\ve\\venv\\Lib\\site-packages
```



```
C:\Windows\System32\cmd.exe - python

(venv) D:\ve>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>> sys.prefix
'D:\\ve\\venv'
>>> from distutils.sysconfig import get_python_lib
>>> get_python_lib
<function sysconfig_get_python_lib at 0x00D0D4F0>
>>> get_python_lib()
'D:\\ve\\venv\\Lib\\site-packages'
>>>
```

# venv 사용 방법

- 모듈 venv 설치는 불필요
  - Python 3.4 이상 사용 가능
- 가상환경 venv 만들기
  - 적당한 폴더로 이동
    - D:
    - cd ve
  - 가상환경 venv 생성
    - `python -m venv venv_test`
  - 가상환경 활성화
    - `venv_test\Scripts\activate`
  - 가상환경에 numpy 설치
    - `pip install numpy`
  - 설치 확인
    - `pip show numpy`
  - 가상환경에서 나오기
    - `deactivate`

```

C:\Windows\System32\cmd.exe

(venv_test) D:\ve>pip list
Package Version
-----
pip      19.2.3
setuptools 41.2.0

(venv_test) D:\ve>python -m pip install --upgrade pip
Collecting pip
  Using cached https://files.pythonhosted.org/packages/00/b6/9cfa56b4081ad13874b0c6f96af8ce16cfbc1cb06bedf8e91844c3d1ec1/pip-19.3.1-py2.py3-none-any.whl
Installing collected packages: pip
  Found existing installation: pip 19.2.3
  Uninstalling pip-19.2.3:
    Successfully uninstalled pip-19.2.3
  Successfully installed pip-19.3.1

(venv_test) D:\ve>pip install numpy
Collecting numpy
  Using cached https://files.pythonhosted.org/packages/0e/c3/be53614c4e3490778050e1df48fd463837297d5dd402dae3b500f2050eba/numpy-1.18.1-cp38-cp38-win32.whl
Installing collected packages: numpy
Successfully installed numpy-1.18.1

(venv_test) D:\ve>pip show numpy
WARNING: Package(s) not found: numpy

(venv_test) D:\ve>pip show numpy
Name: numpy
Version: 1.18.1
Summary: NumPy is the fundamental package for array computing with Python.
Home-page: https://www.numpy.org
Author: Travis E. Oliphant et al.
Author-email: None
License: BSD
Location: d:\ve\venv_test\lib\site-packages
Requires:
Required-by:

(venv_test) D:\ve>
  
```

가상환경이름

# 가상환경 확인 방법

## • 가상환경의 인터프리터에서 다음 확인

- 가상환경 폴더 확인
- 외부모듈 설치폴더 확인

```
>>> import site
>>> site.getsitepackages()
```

```
>>> import sys
>>> sys.prefix
'D:\\ve\\venv'
>>> from distutils.sysconfig import get_python_lib
>>> print(get_python_lib())
D:\\ve\\venv\\Lib\\site-packages
```

```

C:\Windows\System32\cmd.exe - python

(venv_test) D:\#ve>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>> sys.prefix
'D:\\ve\\venv_test'
>>> from distutils.sysconfig import get_python_lib
>>> get_python_lib()
'D:\\ve\\venv_test\\Lib\\site-packages'
>>>
  
```

# pipenv 사용 방법

- 가장 먼저 모듈 pipenv 설치 필요
  - pip install pipenv

```
C:\Windows\System32\cmd.exe
virtualenv 16.7.9

D:\>pip install pipenv
Collecting pipenv
  Downloading https://files.pythonhosted.org/packages/13/b4/3ffa55f77161cff9a5220f162670f7c5eb00df52e00939e203f601b0f579/pipenv-2018.11.26-py3-none-any.whl (5.2MB)
    |████████████████████| 5.2MB 6.4MB/s
Requirement already satisfied: setuptools>=36.2.1 in d:\python38-32\lib\site-packages (from pipenv) (41.2.0)
Collecting virtualenv<=0.2.5
  Downloading https://files.pythonhosted.org/packages/ba/f8/50c2b7dbc99e05fce5e5b9d9a31f37c988c99acd4e8dedd720b7b8d4011d/virtualenv_clone-0.5.3-py2.py3-none-any.whl
Collecting certifi
  Using cached https://files.pythonhosted.org/packages/b9/63/df50cac98ea0d5b006c55a399c3bf1db9da7b5a24de7890bc9cf5dd9e99/certifi-2019.11.28-py2.py3-none-any.whl
Requirement already satisfied: pip>=9.0.1 in d:\python38-32\lib\site-packages (from pipenv) (19.3.1)
Requirement already satisfied: virtualenv in d:\python38-32\lib\site-packages (from pipenv) (16.7.9)
Installing collected packages: virtualenv-clone, certifi, pipenv
Successfully installed certifi-2019.11.28 pipenv-2018.11.26 virtualenv-clone-0.5.3

D:\>pip list
Package            Version
-----
certifi            2019.11.28
pip               19.3.1
pipenv            2018.11.26
setuptools         41.2.0
virtualenv        16.7.9
virtualenv-clone  0.5.3

D:\>
```



# pipenv 도움말

- pipenv -h

```
C:\Windows\System32\cmd.exe
D:\>pipenv -h
Usage: pipenv [OPTIONS] COMMAND [ARGS]...

Options:
  --where          Output project home information.
  --venv           Output virtualenv information.
  --py             Output Python interpreter information.
  --envs           Output Environment Variable options.
  --rm             Remove the virtualenv.
  --bare           Minimal output.
  --completion     Output completion (to be eval'd).
  --man            Display manpage.
  --support        Output diagnostic information for use in GitHub issues.
  --site-packages  Enable site-packages for the virtualenv. [env var:
                    PIPENV_SITE_PACKAGES]
  --python TEXT    Specify which version of Python virtualenv should use.
  --three / --two  Use Python 3/2 when creating virtualenv.
  --clear          Clears caches (pipenv, pip, and pip-tools). [env var:
                    PIPENV_CLEAR]
  -v, --verbose    Verbose mode.
  --pyi-mirror TEXT Specify a PyPI mirror.
  --version        Show the version and exit.
  -h, --help       Show this message and exit.

Usage Examples:
  Create a new project using Python 3.7, specifically:
  $ pipenv --python 3.7

  Remove project virtualenv (inferred from current directory):
  $ pipenv --rm

  Install all dependencies for a project (including dev):
  $ pipenv install --dev

  Create a lockfile containing pre-releases:
  $ pipenv lock --pre

  Show a graph of your installed dependencies:
  $ pipenv graph

  Check your installed dependencies for security vulnerabilities:
  $ pipenv check

  Install a local setup.py into your virtual environment/Pipfile:
  $ pipenv install -e .

  Use a lower-level pip command:
  $ pipenv run pip freeze

Commands:
  check    Checks for security vulnerabilities and against PEP 508 markers
           provided in Pipfile.
  clean    Uninstalls all packages not specified in Pipfile.lock.
  graph    Displays currently-installed dependency graph information.
  install  Installs provided packages and adds them to Pipfile, or (if no
           packages are given), installs all packages from Pipfile.
  lock     Generates Pipfile.lock.
  open     View a given module in your editor.
  run      Spawns a command installed into the virtualenv.
  shell    Spawns a shell within the virtualenv.
  sync     Installs all packages specified in Pipfile.lock.
  uninstall Uninstalls a provided package and removes it from Pipfile.
  update   Runs lock, then sync.
```

# pipenv 사용 방법

- 가상환경 penv 만들기
  - 프로젝트 디렉터리에 위치한 상태로 가상환경을 생성
- 적당한 폴더로 이동과 폴더 생성
  - D:
  - cd ve
  - 프로젝트 폴더 생성과 이동
    - md penv
    - cd penv
- 가상환경 생성
  - 명령문
    - pipenv shell
  - 가상환경 이름
    - penv-Snxa-AB5
  - 가상환경 폴더
    - C:\Users\217\virtualenvs\penv-Snxa-AB5\Scripts\python.exe

C:\Windows\System32\cmd.exe - pipenv shell

D:\>cd ve

D:\ve>dir

D 드라이브의 볼륨: DATADRIEVO  
볼륨 일련 번호: 0AEE-91E2

D:\ve 디렉터리

```
2020-01-14 오후 06:37 <DIR> .
2020-01-14 오후 06:37 <DIR> ..
2020-01-14 오후 06:37 <DIR> mini
2020-01-14 오후 01:12 <DIR> venv
2020-01-14 오후 06:10 <DIR> venv_test
                    0개 파일              0 바이트
                    5개 디렉터리  1,985,989,898,240 바이트 남음
```

D:\ve>md penv

D:\ve>cd penv

D:\ve\penv>pipenv shell

Creating a virtualenv for this project...

Pipfile at D:\ve\penv\Pipfile

Using d:\python38-32\python.exe (3.8.1) to create virtualenv...

[...] Creating virtual environment - Already using interpreter d:\python38-32\python.exe

Using base prefix 'd:\python38-32'

New python executable in C:\Users\217\virtualenvs\penv-Snxa-AB5\Scripts\python.exe

Installing setuptools, pip, wheel...

done.

Successfully created virtual environment!

Virtualenv location: C:\Users\217\virtualenvs\penv-Snxa-AB5

Creating a Pipfile for this project...

Launching subshell in virtual environment...

Microsoft Windows [Version 10.0.17763.914]

(c) 2018 Microsoft Corporation. All rights reserved.

(penv-Snxa-AB5) D:\ve\penv>dir

D 드라이브의 볼륨: DATADRIEVO  
볼륨 일련 번호: 0AEE-91E2

D:\ve\penv 디렉터리

```
2020-01-15 오전 10:32 <DIR> .
2020-01-15 오전 10:32 <DIR> ..
2020-01-15 오전 10:33                138 Pipfile
                    1개 파일              138 바이트
                    2개 디렉터리  1,985,989,898,240 바이트 남음
```

(penv-Snxa-AB5) D:\ve\penv>

가상환경과 연결되는  
프로젝트 폴더

# pipenv 모듈 설치

- 모듈 설치
  - 가상환경에 필요 모듈 matplotlib 설치
    - `pipenv install matplotlib`
  - 설치 확인
    - `pip show matplotlib`
- 프로젝트 폴더 생성파일
  - Pipfile
  - Pipfile.lock

```
C:\Windows\System32\cmd.exe - pipenv shell

(penv-Snxa-AB5) D:\ve\#pipenv install matplotlib
Installing matplotlib...
Adding matplotlib to Pipfile's [packages]...
Installation Succeeded
Pipfile.lock not found, creating...
Locking [dev-packages] dependencies...
Locking [packages] dependencies...
Success!
Updated Pipfile.lock (312932)!
Installing dependencies from Pipfile.lock (312932)...
===== 7/7 - 00:00:01

(penv-Snxa-AB5) D:\ve\#pipenv type pipfile
[[source]]
name = "pypi"
url = "https://pypi.org/simple"
verify_ssl = true

[dev-packages]

[packages]
matplotlib = "*"

[requires]
python_version = "3.8"

(penv-Snxa-AB5) D:\ve\#pipenv dir
D 드라이브의 볼륨: DATADRIVE0
볼륨 일련 번호: 0AEE-91E2

D:\ve\#pipenv 디렉터리

2020-01-15 오전 10:42 <DIR> .
2020-01-15 오전 10:42 <DIR> ..
2020-01-15 오전 10:40          155 Pipfile
2020-01-15 오전 10:42      8,408 Pipfile.lock
                2개 파일          8,563 바이트
                2개 디렉터리 1,985,989,758.976 바이트 남음

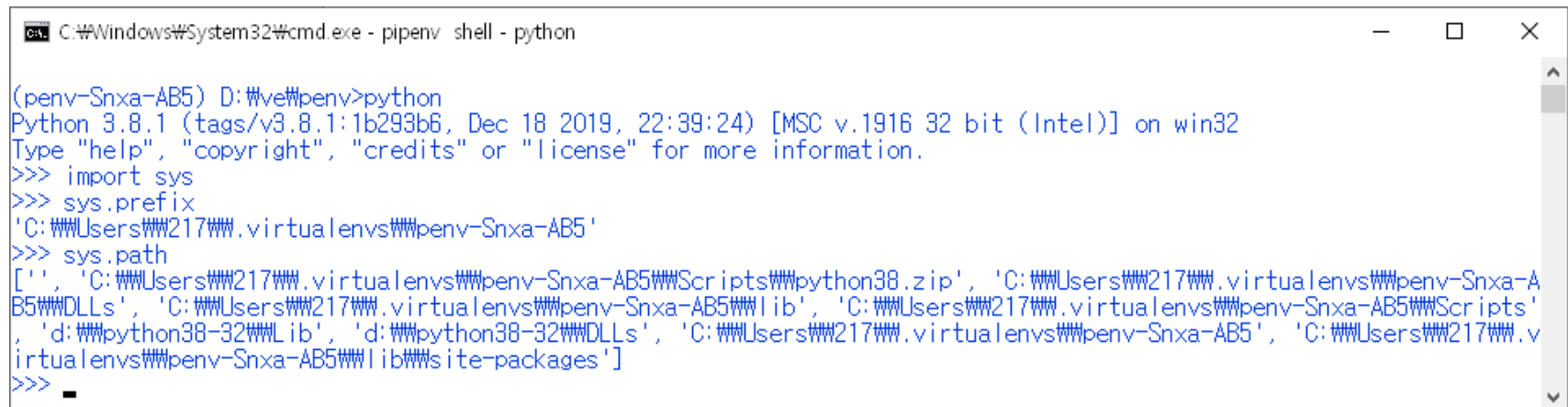
(penv-Snxa-AB5) D:\ve\#pipenv type pipfile.lock
{
  "_meta": {
    "hash": {
      "sha256": "8f16afa364d6ea39f2e3fe69621a42a3a1c60d012cb7c20ee810b918a9312932"
    },
    "pipfile-spec": 6,
    "requires": {
      "python_version": "3.8"
    },
    "sources": [
      {

```

# Pipenv로 설치된 python 확인

## • Python 실행

```
>>> import sys
>>> sys.prefix
'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5'
>>> sys.path
['', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\Scripts\\python38.zip',
'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\DLLs',
'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\lib',
'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\Scripts', 'd:\\python38-32\\Lib',
'd:\\python38-32\\DLLs', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5',
'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\lib\\site-packages']
```



```

C:\Windows\System32\cmd.exe - pipenv shell - python

(penv-Snxa-AB5) D:\ve\penv>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import sys
>>> sys.prefix
'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5'
>>> sys.path
['', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\Scripts\\python38.zip', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\DLLs', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\lib', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\Scripts', 'd:\\python38-32\\Lib', 'd:\\python38-32\\DLLs', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5', 'C:\\Users\\217\\.virtualenvs\\penv-Snxa-AB5\\lib\\site-packages']
>>> _
  
```

# Pipenv로 설치된 matplotlib 확인 및 실행

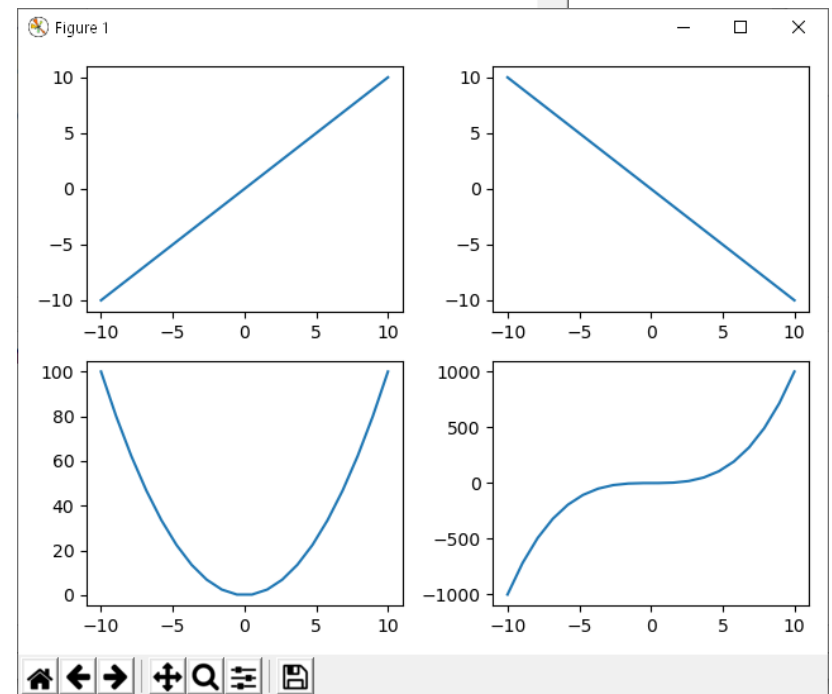
## • Python 실행

```

C:\Windows\System32\cmd.exe - pipenv shell - python

(penv-Snxa-AB5) D:\ve\#pipenv>python
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> import matplotlib.pyplot as plt
>>> x = np.linspace(-10, 10, 20)
>>> plt.subplot(2, 2, 1)
<matplotlib.axes._subplots.AxesSubplot object at 0x037C3910>
>>> plt.plot(x, x)
[<matplotlib.lines.Line2D object at 0x10815340>]
>>> plt.subplot(2, 2, 2)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'plit' is not defined
>>> plt.subplot(2, 2, 2)
<matplotlib.axes._subplots.AxesSubplot object at 0x10806BB0>
>>> plt.plot(x, -x)
[<matplotlib.lines.Line2D object at 0x00DB9178>]
>>> plt.subplot(2, 2, 3)
<matplotlib.axes._subplots.AxesSubplot object at 0x00DB9AA8>
>>> plt.plot(x, x**2)
[<matplotlib.lines.Line2D object at 0x00DD2700>]
>>> plt.subplot(2, 2, 4)
<matplotlib.axes._subplots.AxesSubplot object at 0x00DD2DD8>
>>> plt.plot(x, pow(x, 3))
[<matplotlib.lines.Line2D object at 0x012D7388>]
>>> plt.show()

```



# Matplotlib의 서브플롯

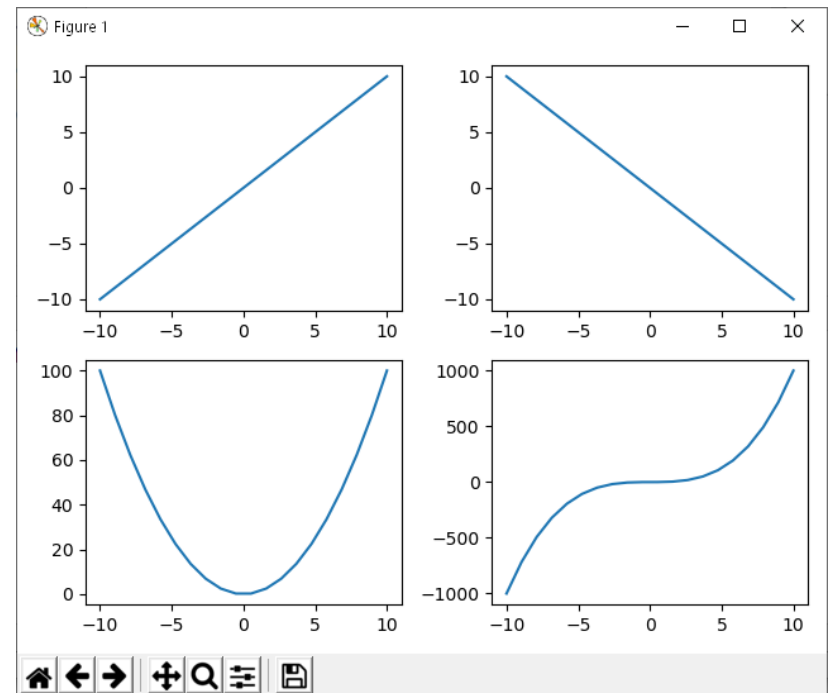
- 여러 그림을 하나의 캔버스에 그리는 방법

```
import numpy as np
import matplotlib.pyplot as plt

# -10에서 10까지 20등분한 자료
x = np.linspace(-10, 10, 20)

# 2행 2열의 부분 그림
plt.subplot(2, 2, 1) # 첫 번째 부분 그림
plt.plot(x, x)
plt.subplot(2, 2, 2) # 두 번째 부분 그림
plt.plot(x, -x)
plt.subplot(2, 2, 3) # 세 번째 부분 그림
plt.plot(x, x*x)
plt.subplot(2, 2, 4) # 네 번째 부분 그림
plt.plot(x, pow(x, 3))

plt.tight_layout() # 적절한 공간 배치
plt.show() # 그리기
```



# 가상환경 생성 정리

- **virtualenv, venv**
  - 하부 폴더에 지정한 이름의 가상환경이 생성됨
    - **virtualenv** 가상환경
    - **python -m venv** 가상환경
- **pipenv**
  - pipenv shell이 실행되는 폴더를 기본으로 지정된 폴더에 가상환경이 생성됨
    - **pipenv shell**
      - **C:\Users\217\virtualenvs\실행폴더이름-0000**
  - activate가 따로 있지 않고 생성하면서 진입

# 가상환경 생성 실습(20분)

- **지정한 폴더에 각각의 도구로 가상환경 생성**
  - 상위 폴더 myve 하부에 각각 생성
    - **virtualenv** 로 생성, **numpy** 설치
      - **ve\_np**
    - **venv** 로 생성, **matplotlib** 설치
      - **ve\_mp**
    - **pipenv** 로 생성, **pandas** 설치
      - **ve-pd**



# 아나콘다의 conda 명령어

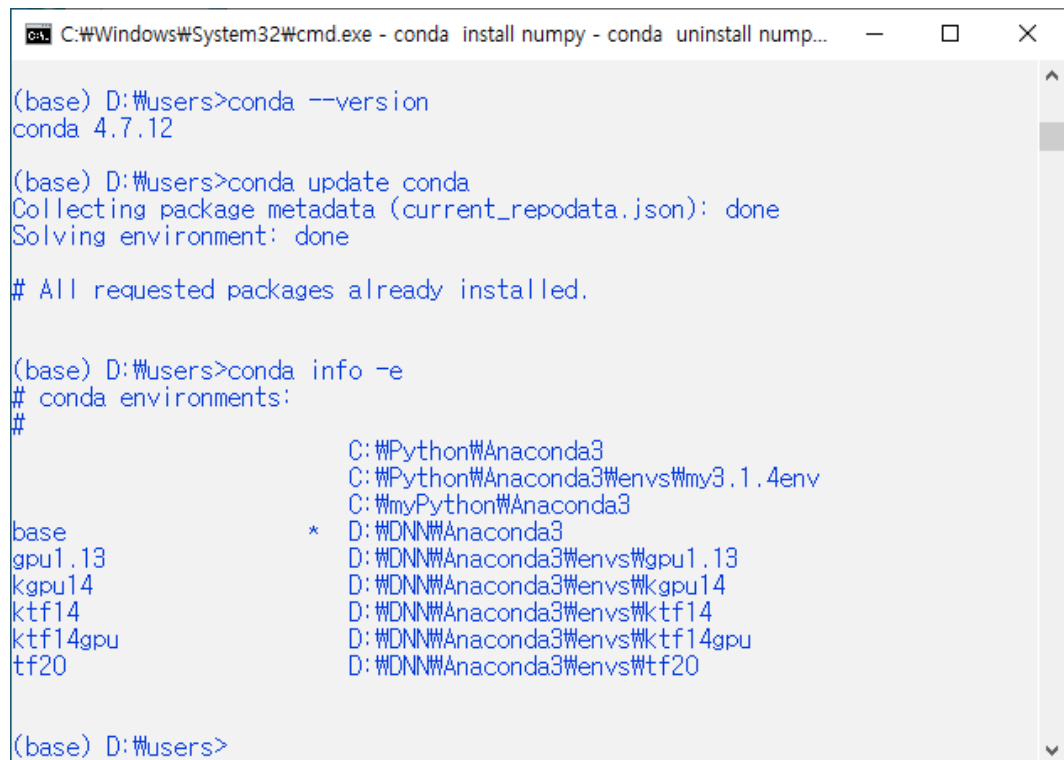
# 아나콘다 conda

- 아나콘다가 제공하는 명령 프롬프트에서 실행되는 명령어
  - conda 프롬프트 열기
    - 설치 메뉴 Anaconda Prompt(Anaconda3) 실행
      - %windir%\System32\cmd.exe "/K" D:\Anaconda3\Scripts\activate.bat D:\Anaconda3
- 가상환경이나 모듈을 관리하는 명령어
  - 항상 conda로 시작

명령어	설명
conda --version	설치된 아나콘다 버전 확인
conda clean	설치된 패키지를 모두 삭제
conda create	새로운 가상환경 생성
conda config	설정 보기, 신규 설정, 수정등
conda info	설치된 아나콘다 정보
conda install	패키지 설치
conda list	설치된 패키지 정보(환경 별로 다름)
conda remove	설치된 패키지 삭제
conda search	설치된 패키지 조회
conda uninstall	Alias for conda remove
conda update	최신 버전으로 업데이트
conda upgrade	Alias for conda update

# 아나콘다의 conda 명령 기본

- **conda --version**
  - 현재 콘다 버전
- **conda update conda**
  - 콘다 최신 버전으로
- **conda info -e**
  - 현재 가상환경 목록



```

C:\Windows\System32\cmd.exe - conda install numpy - conda uninstall nump...
(base) D:\Users>conda --version
conda 4.7.12

(base) D:\Users>conda update conda
Collecting package metadata (current_repodata.json): done
Solving environment: done

# All requested packages already installed.

(base) D:\Users>conda info -e
# conda environments:
#
base                    * C:\Python\Anaconda3
gpu1.13                 C:\Python\Anaconda3\envs\my3.1.4env
kgpu14                  C:\Python\Anaconda3
ktf14                   D:\DNN\Anaconda3
ktf14gpu                D:\DNN\Anaconda3\envs\gpu1.13
tf20                    D:\DNN\Anaconda3\envs\kgpu14
                        D:\DNN\Anaconda3\envs\ktf14
                        D:\DNN\Anaconda3\envs\ktf14gpu
                        D:\DNN\Anaconda3\envs\tf20

(base) D:\Users>
  
```

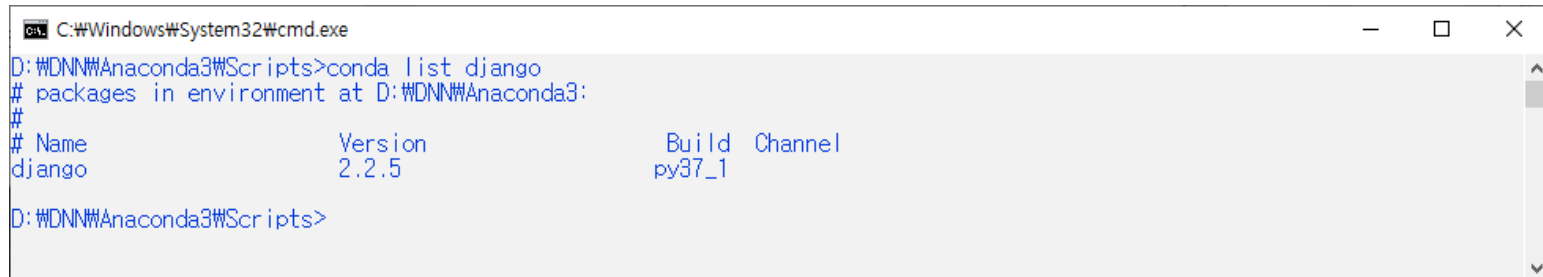
# 아나콘다의 conda 명령어

- **conda list**
  - conda list 패키지명
    - 패키지 관련 설치 모듈과 버전 정보 제공
- **conda install**
- **conda search 패키지명**
  - 저장소에 있는 패키지 검색
  - 설치전에 설치할 패키지를 한번 찾아본 후 버전 등을 보고 설치

명령어	설명
conda --version	설치된 아나콘다 버전 확인
conda clean	설치된 패키지를 모두 삭제
conda create	새로운 가상환경 생성
conda config	설정 보기, 신규 설정, 수정등
conda info	설치된 아나콘다 정보
conda install	패키지 설치
conda list	설치된 패키지 정보(환경 별로 다름)
conda remove	설치된 패키지 삭제
conda search	설치된 패키지 조회
conda uninstall	Alias for conda remove
conda update	최신 버전으로 업데이트
conda upgrade	Alias for conda update

# 장고 설치 및 삭제

- `conda list django`
- `conda search django`
- `conda install django`
- `conda remove django`



```
C:\Windows\System32\cmd.exe
D:\DNN\Anaconda3\Scripts>conda list django
# packages in environment at D:\DNN\Anaconda3:
#
# Name          Version      Build    Channel
django          2.2.5        py37_1
D:\DNN\Anaconda3\Scripts>
```

# 가상환경 생성 및 필요 모듈 설치

## • conda create -n 가상환경이름 설치모듈명

- conda info -e
  - 현재 가상환경 이름 확인
- conda create -n kang flask
  - 가상환경 kang을 만들고 모듈 flask 설치
- conda info -e
  - 가상환경 kang 확인
- conda activate kang
  - 가상환경 활성화
- conda list

```

C:\Windows\System32\cmd.exe

# $ conda activate kang
# To deactivate an active environment, use
# $ conda deactivate

(base) C:\Windows\system32>conda info -e
# conda environments:
#
base                  * C:\Python\Anaconda3
gpu1.13              C:\Python\Anaconda3\envs\my3.1.4envs
kang                  C:\Python\Anaconda3
kgpu14               D:\DNN\Anaconda3
ktf14                D:\DNN\Anaconda3\envs\kgpu1.13
ktf14gpu             D:\DNN\Anaconda3\envs\kang
tf20                 D:\DNN\Anaconda3\envs\kgpu14
                    D:\DNN\Anaconda3\envs\ktf14
                    D:\DNN\Anaconda3\envs\ktf14gpu
                    D:\DNN\Anaconda3\envs\tf20

(base) C:\Windows\system32>conda activate kang
(kang) C:\Windows\system32>conda list
# packages in environment at D:\DNN\Anaconda3\envs\kang:
#
# Name                      Version                      Build      Channel
ca-certificates             2019.10.16                   0          py38_0
certifi                     2019.9.11                    py38_0
click                       7.0                          py_0
flask                       1.1.1                        py_0
itsdangerous                1.1.0                        py_0
jinja2                      2.10.3                       py_0
markupsafe                  1.1.1                        py38he774522_0
openssl                     1.1.1d                       he774522_3
pip                          19.3.1                       py38_0
python                      3.8.0                        hff0d562_2
setuptools                  41.6.0                       py38_0
sqlite                      3.30.1                       he774522_0
vc                          14.1                         h0510ff6_4
vs2015_runtime              14.16.27012                  hf0eaf9b_0
werkzeug                    0.16.0                       py_0
wheel                       0.33.6                       py38_0
wincertstore                0.2                          py38_0

(kang) C:\Windows\system32>

```

# 모듈과 가상환경 삭제

- **conda uninstall numpy**
- **conda deactivate**
- **conda env remove -n kang**
  - 가상환경명 kang 삭제
- **conda info -e**

```

C:\Windows\System32\cmd.exe - conda install numpy - conda uninstall nump...
(kang) D:\Users>conda list numpy
# packages in environment at D:\DNN\Anaconda3\envs\kang:
#
# Name                               Version                               Build Channel
(kang) D:\Users>conda deactivate
(base) D:\Users>conda env remove -n kang
Remove all packages in environment D:\DNN\Anaconda3\envs\kang:

(base) D:\Users>conda info -e
# conda environments:
#
base                                * C:\Python\Anaconda3
gpu1.13                             D:\DNN\Anaconda3\envs\gpu1.13
kgpu14                             D:\DNN\Anaconda3\envs\kgpu14
ktf14                               D:\DNN\Anaconda3\envs\ktf14
ktf14gpu                           D:\DNN\Anaconda3\envs\ktf14gpu
tf20                               D:\DNN\Anaconda3\envs\tf20

(base) D:\Users>

```

# conda 사용 가상환경 생성



# Conda로 가상환경 만들기

## • conda 프롬프트 열기

- 설치 메뉴 Anaconda Prompt(Anaconda3) 실행
  - %windir%\System32\cmd.exe "/K" D:\Anaconda3\Scripts\activate.bat D:\Anaconda3
  - cmd /k cd %homepath% 로 열어서 다음 실행
    - D:\Anaconda3\Scripts\activate.bat
    - 또는 conda activate

## • 가상환경 생성 및 모듈 설치 동시

- conda create -n 가상환경 설치모듈1 설치모듈2 설치모듈3 ...

## • 가상환경 활성화

- conda activate 가상환경

## • 필요하면 모듈 추가 설치

- conda install 설치모듈4 설치모듈5 설치모듈6

## • 가상환경 비활성화

- conda deactivate

> echo %windir%  
C:\windows

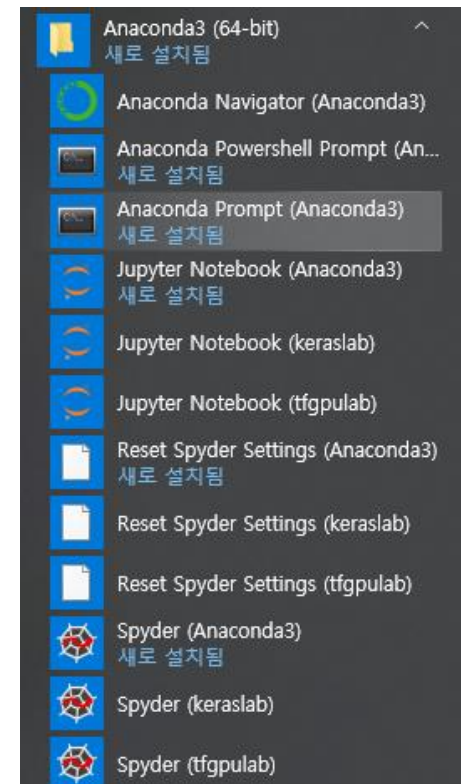
윈도 기본 기본변수

# 수업에서의 가상환경 3개

- 가상환경 mynmp
  - 설치 모듈
    - matplotlib(numpy), pandas
    - jupyter, spyder
- 가상환경 ktf1.15
  - 설치 모듈
    - tensorflow-gpu, keras
      - 1.15.0
    - matplotlib, pandas
    - Jupyter, spyder
- 가상환경 ktf2.0
  - 설치 모듈
    - tensorflow-gpu=2.0.0, keras
      - 2.0.0
    - matplotlib, pandas
    - jupyter, spyder

# 아나콘다 프롬프트에서 가상환경 생성

- **도스 창 실행됨**
  - 프롬프트 앞에 '(가상환경이름)'이 있음
    - 기본적으로 (base)
  - (base) ...>
- **가상환경 생성**
  - 이름
    - mynmp
  - 설치 모듈
    - matplotlib(numpy), pandas
    - jupyter, spyder
  - 명령어
    - conda create -n 가상환경 설치모듈1 설치모듈2



# 가상환경 mynmp

- 모듈 matplotlib, mumpy 설치
  - 모듈 jupyter, spyder도 함께 설치
- 현재 가상환경 확인
  - conda info -e
- 가상환경 생성과 모듈 matplotlib pandas jupyter spyder 설치 명령
  - conda create -n mynmp matplotlib pandas jupyter spyder
- 다시 가상환경 확인, 만들어진 가상환경 mynmp 확인 필요
  - Conda info -e
    - 지금 새로 생성된 가상환경 mynmp 확인
- 가상환경으로 진입, 활성화
  - conda activate mynmp
    - 가상환경 활성화해 내부로 진입, 가상환경 이름이 맨 앞에 표시

설치할 모듈 이름 목록

```

Anaconda Prompt (Anaconda3)
(mynmp) C:\Users\W217>
  
```

- 새로이 생성된 가상환경 내부에서 설치된 모듈 점검
  - conda list numpy
  - conda list matplotlib
  - conda list pandas
  - conda list jupyter
  - conda list spyder

# 가상환경 ktf2.0

## • 모듈 tensorflow-gpu 설치

- 모듈 keras, matplotlib, pandas, jupyter, spyder도 함께 설치

## • 필요 명령

- conda info -e
- conda create -n ktf2.0 tensorflow-gpu keras pandas matplotlib jupyter spyder
- conda info -e
  - 지금 새로 생성된 가상환경 mynmp 확인
- conda activate ktf2.0
  - 가상환경 활성화해 내부로 진입
  - 가상환경 이름이 맨 앞에 표시

## • 새로이 생성된 가상환경 내부에서 설치된 모듈 점검

- conda list tensorflow-gpu
- conda list tensorflow
- conda list keras
- conda list numpy
- conda list matplotlib
- conda list pandas
- conda list jupyter
- conda list spyder

# 가상환경 ktf1.15

- **모듈 tensorflow-gpu 설치, 버전 지정 = 1.15.0**

- 모듈 keras, pandas, matplotlib, jupyter, spyder도 함께 설치

설치할 모듈 이름 목록

- **필요 명령**

- conda info -e
- conda create -n ktf1.15 tensorflow-gpu=1.15.0 keras pandas matplotlib jupyter spyder
- conda info -e

- **지금 새로 생성된 가상환경 mynmp 확인**

- conda activate ktf1.15
  - 가상환경 활성화해 내부로 진입
  - 가상환경 이름이 맨 앞에 표시

현재는 1.15.0으로  
설치되고 있음

- **새로이 생성된 가상환경 내부에서 설치된 모듈 점검**

- conda list tensorflow-gpu
- conda list tensorflow
- conda list keras
- conda list numpy
- conda list matplotlib
- conda list pandas
- conda list jupyter
- conda list spyder

# 설치 모듈 전체 확인

- conda list

```

Anaconda Prompt (Anaconda3)

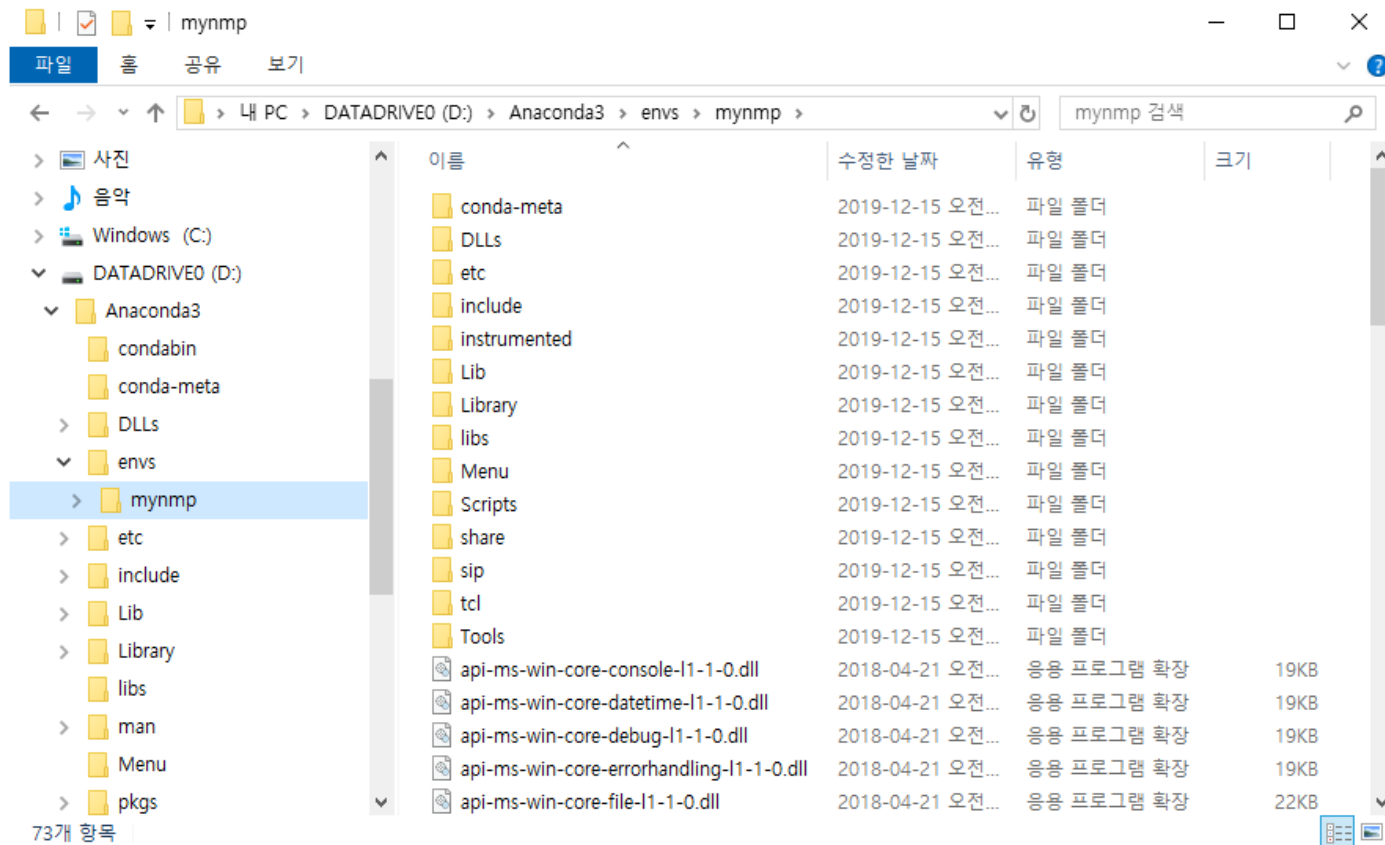
(mymp) C:\Users\217>conda list
# packages in environment at D:\Anaconda3\envs\mymp:
#
# Name                                Version                                Build                                Channel
alabaster                             0.7.12                                py37_0
argh                                  0.26.2                                py37_0
asn1crypto                            1.2.0                                  py37_0
astroid                               2.3.3                                  py37_0
atomicwrites                          1.3.0                                  py37_1
attrs                                 19.3.0                                py_0
autopep8                              1.4.4                                  py_0
babel                                 2.7.0                                  py_0
backcall                              0.1.0                                  py37_0
bcrypt                                3.1.7                                py37he774522_0
blas                                  1.0                                    mkl
bleach                                3.1.0                                py37_0
ca-certificates                       2019.11.27                            0
certifi                               2019.11.28                            py37_0
cffi                                  1.13.2                                py37h7a1dbc1_0
chardet                               3.0.4                                py37_1003
cloudpickle                           1.2.2                                  py_0
colorama                              0.4.1                                  py37_0
cryptography                          2.8                                    py37h7a1dbc1_0
cyclor                                0.10.0                                py37_0
decorator                             4.4.1                                  py_0
defusedxml                            0.6.0                                  py_0
diff-match-patch                      20181111                              py_0
docutils                              0.15.2                                py37_0
entrypoints                           0.3                                    py37_0
flake8                                3.7.9                                  py37_0
freetype                              2.9.1                                ha9979f8_1
future                                0.18.2                                py37_0
icc_rt                                2019.0.0                              h0cc432a_1
icu                                    58.2                                  ha66f8fd_1
jdna                                   2.8                                    py37_0
imagesize                             1.1.0                                  py37_0
importlib_metadata                    1.2.0                                  py37_0
intel-openmp                          2019.4                                245
intervaltree                           3.0.2                                  py_0
ipykernel                             5.1.3                                py37h39e3cac_0
ipython                               7.10.1                                py37h39e3cac_0
ipython_genutils                      0.2.0                                  py37_0
ipywidgets                            7.5.1                                  py_0
isort                                  4.3.21                                py37_0
jedi                                   0.14.1                                py37_0
jinja2                                2.10.3                                py_0
jpeg                                   9b                                    hb83a4c4_2
jsonschema                            3.2.0                                  py37_0
jupyter                                1.0.0                                  py37_7
jupyter_client                        5.3.4                                  py37_0
jupyter_console                       5.2.0                                  py37_1
jupyter_core                          4.6.1                                  py37_0
keyring                               19.2.0                                py37_0
kiwisolver                            1.1.0                                py37ha925a31_0
lazy-object-proxy                     1.4.3                                py37he774522_0
libpng                                1.6.37                                h2a8f88b_0

```

# 가상환경 확인

## 폴더 생성 확인

– [아나콘다설치폴더]/envs/mynmp





# 가상환경 생성 실습(20분)

- 각각 지정한 가상환경 3개를 conda로 생성
  - 가상환경 vnp 생성, numpy 설치
    - `conda create -n vnp numpy`
  - 가상환경 vpd 생성, pandas 설치
    - `conda create -n vpd pandas`
  - 가상환경 vtf 생성, tensorflow 설치
    - `conda create -n vtf tensorflow`