

Module and Project

TEAMLAB director
최성철

PYTHON

JAVA



파이썬은 대부분의 라이브러리가 이미
다른 사용자에 의해서 구현되어 있음

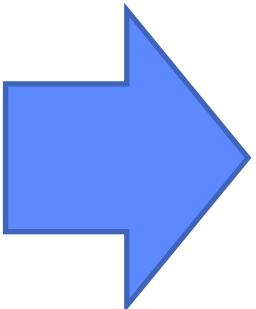
그런데 어떻게 쓰나요?

남이 만든 프로그램 쓰는 법

객체 < 모듈

모듈과 패키지

- 어떤 대상의 부분 혹은 조각
예) 레고 블록, 벽돌, 자동차 부품들



- 프로그램에서는 작은 프로그램 조각들,
모듈들을 모아서 하나의 큰 프로그램을 개발함
- 프로그램을 모듈화 시키면 다른 프로그램이 사용하기 쉬움
예) 카카오톡 게임을 위한 카카오톡 접속 모듈

Facebook의 새로운 제품을 살펴보세요.

<https://developers.facebook.com/docs/>

AI 도구

연구 및 프로덕션을 위한 딥 러닝 프레임워크입니다.

[개요](#)

Spark AR Studio

코드를 사용하거나 사용하지 않고 인터랙티브 증강 현실 경험을 만들어보세요.

[개요](#) [문서](#)

인스턴트 게임

Messenger 및 Facebook 뉴스피드에서 플레이할 수 있는 HTML5 게임입니다.

[개요](#) [문서](#)

인공 지능

Wit.ai



게임

Facebook Gameroom



게시

Audience Network



증강 현실

Spark AR Studio



Game Payments



Instagram Platform



Games on Facebook



Instant Articles



Instant Games



Live Video API

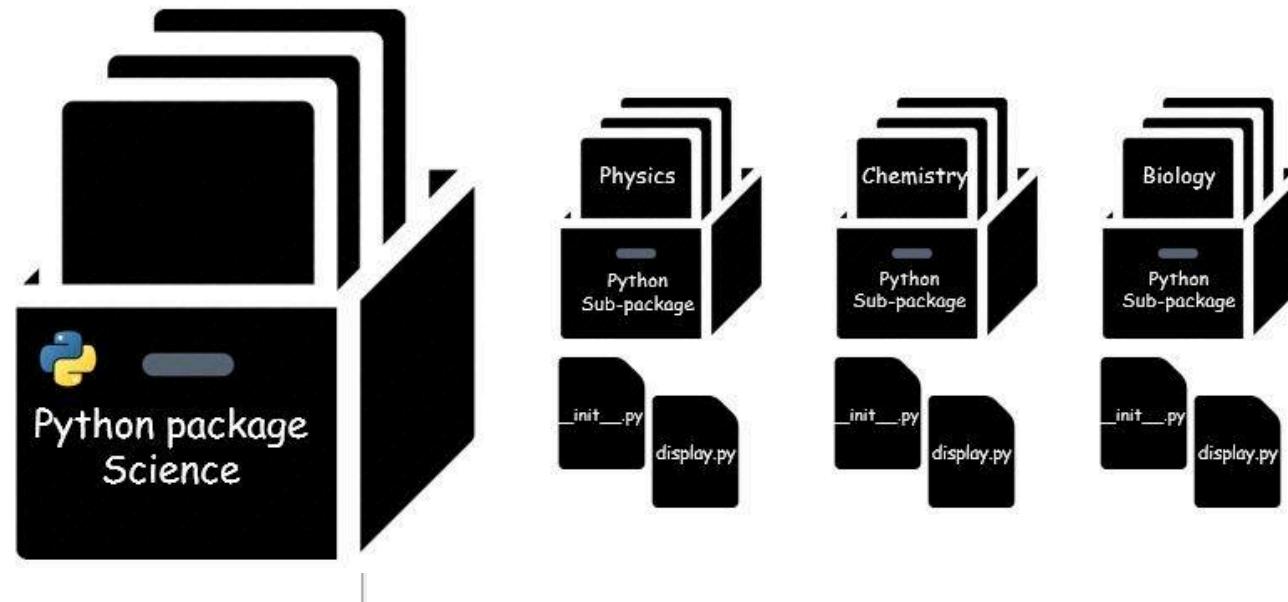
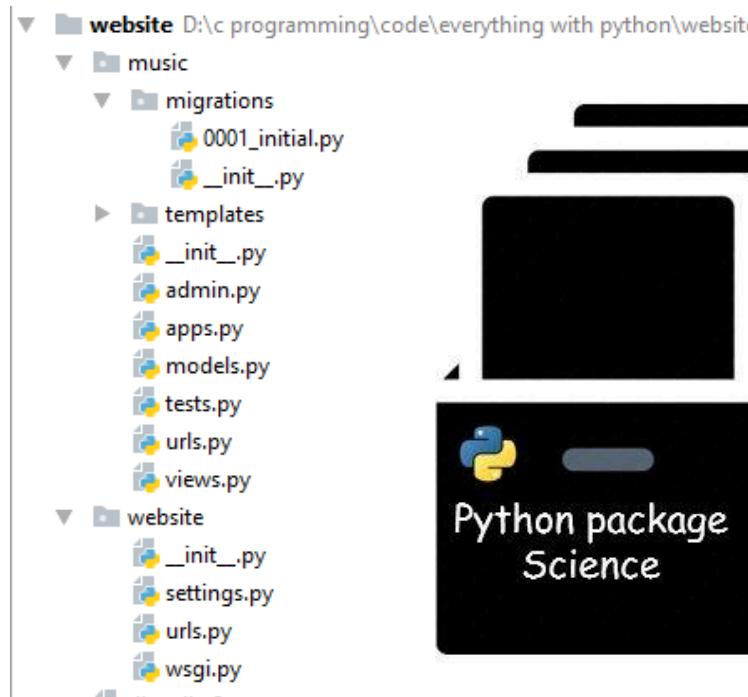


- Built-in Module인 Random을 사용,

난수를 쉽게 생성할 수 있음

```
>>> import random  
>>> random.randint(1,1000)  
315  
>>> random.randint(1,1000)  
840  
>>> random.randint(1,1000)  
780  
>>> random.randint(1,1000)  
57  
>>>
```

- 모듈을 모아놓은 단위, 하나의 프로그램



**직접 구현을 해봐야
알 수 있음**

모듈

- 파이썬의 Module == py 파일을 의미
- 같은 폴더에 Module에 해당하는 .py 파일과 사용하는 .py을 저장한 후
- import 문을 사용해서 module을 호출

fah_converter.py

```
def covert_c_to_f(celcius_value):
    return celcius_value * 9.0 / 5 + 32
```

module_ex.py

```
import fah_converter
```

```
print ("Enter a celsius value: "),
celsius = float(input())
fahrenheit = fah_converter.covert c to f(celsius)
print ("That's ", fahrenheit, " degrees Fahrenheit")
```

- 모듈을 호출할 때 범위 정하는 방법
- 모듈 안에는 함수와 클래스 등이 존재 가능
- 필요한 내용만 골라서 호출 할 수 있음
- from 과 import 키워드를 사용함

Alias 설정하기 - 모듈명을 별칭으로 써서

```
import fah_converter as fah      fah_converter를 fah라는 이름으로  
print(fah.covert_c_to_f(41.6))    그 안에 covert_c_to_f 함수를 쓴다
```

모듈에서 특정 함수 또는 클래스만 호출하기

```
from fah_converter import covert_c_to_f  
print(covert_c_to_f(41.6))      covert_c_to_f 함수만 호출함
```

모듈에서 모든 함수 또는 클래스를 호출하기

```
from fah_converter import *  
print(covert_c_to_f(41.6))      전체 호출
```

- 파이썬이 기본 제공하는 라이브러리
- 문자처리, 웹, 수학 등 다양한 모듈이 제공됨
- 별다른 조치없이 import 문으로 활용 가능

Built-in Module Example

모듈

#난수

```
import random
print (random.randint (0,100)) # 0~100사이의 정수 난수를 생성
print (random.random()) # 일반적인 난수 생성
```

#시간

```
import time
print(time.localtime()) # 현재 시간 출력
```

#웹

```
import urllib.request
response = urllib.request.urlopen("http://thetemlab.io")
print(response.read())
```

- 수 많은 파이썬 모듈은 어떻게 검색할 것인가?

- 1) 구글신에게 물어본다
- 2) 모듈을 import 후 구글 검색 또는 Help 쓰기
- 3) 공식 문서를 읽어본다

<https://docs.python.org/3/library/>

- 실습: 1 부터 100까지 특정 난수를 뽑고 싶다!

패키지

- 하나의 대형 프로젝트를 만드는 코드의 묶음
- 다양한 모듈들의 합, 폴더로 연결됨
- __init__, __main__ 등 키워드 파일명이 사용됨
- 다양한 오픈 소스들이 모두 패키지로 관리됨

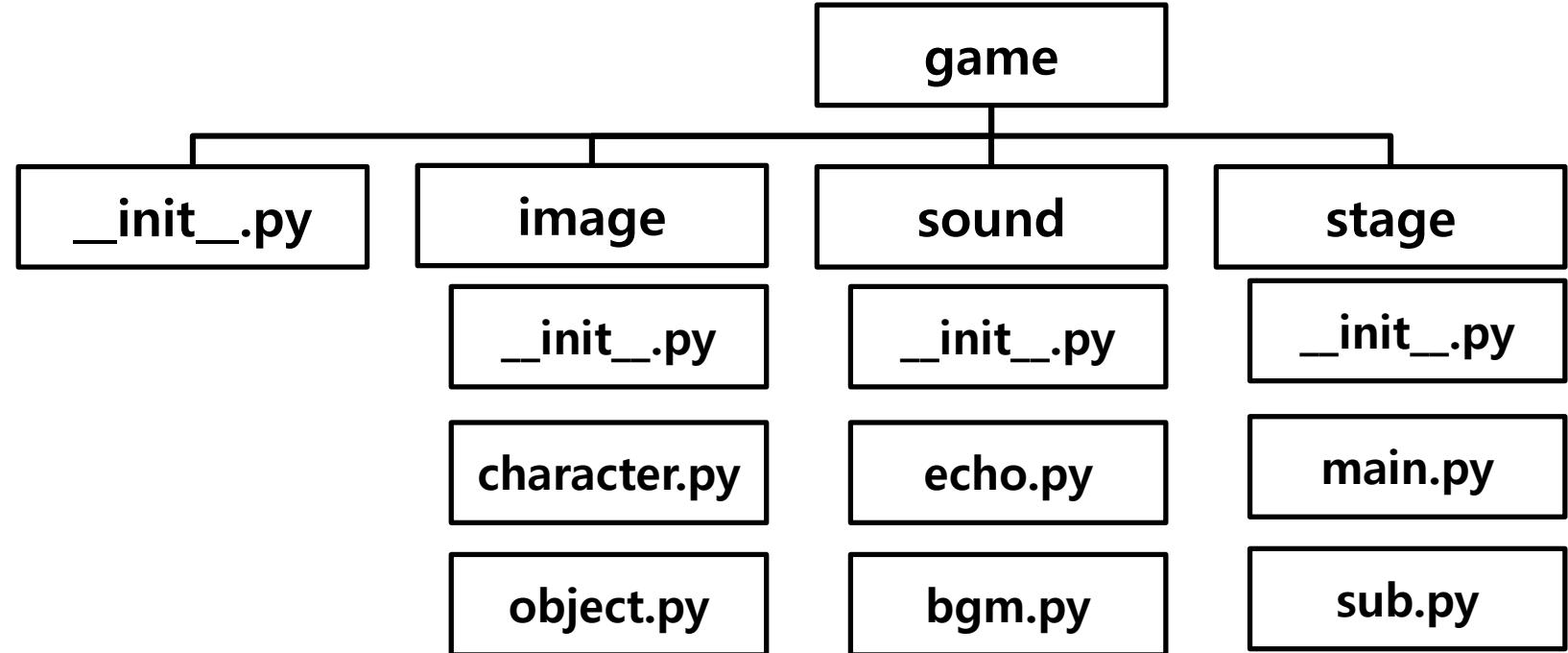
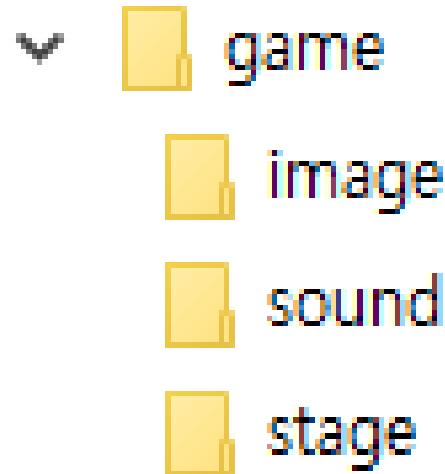
<https://wikidocs.net/1418>

<https://github.com/scikit-learn/scikit-learn/tree/master/sklearn>

1) 기능들을 세부적으로 나눠 폴더로 만듦

- ▼  game
 -  image
 -  sound
 -  stage

2) 각 폴더별로 필요한 모듈을 구현함



<https://wikidocs.net/1418>

2) 각 폴더별로 필요한 모듈을 구현함

```
# echo.py
def echo_play(echo_number):
    print ("echo {} number start".format(echo_number))
```

3) 1차 Test – python shell

```
>>> import echo
>>> echo.echo_play(10)
echo 10 number start
```

4) 폴더별로 __init__.py 구성하기

- 현재 폴더가 패키지임을 알리는 초기화 스크립트
- 없을 경우 패키지로 간주하지 않음 (3.3+ 부터는 X)
- 하위 폴더와 py 파일(모듈)을 모두 포함함
- import와 __all__ keyword 사용

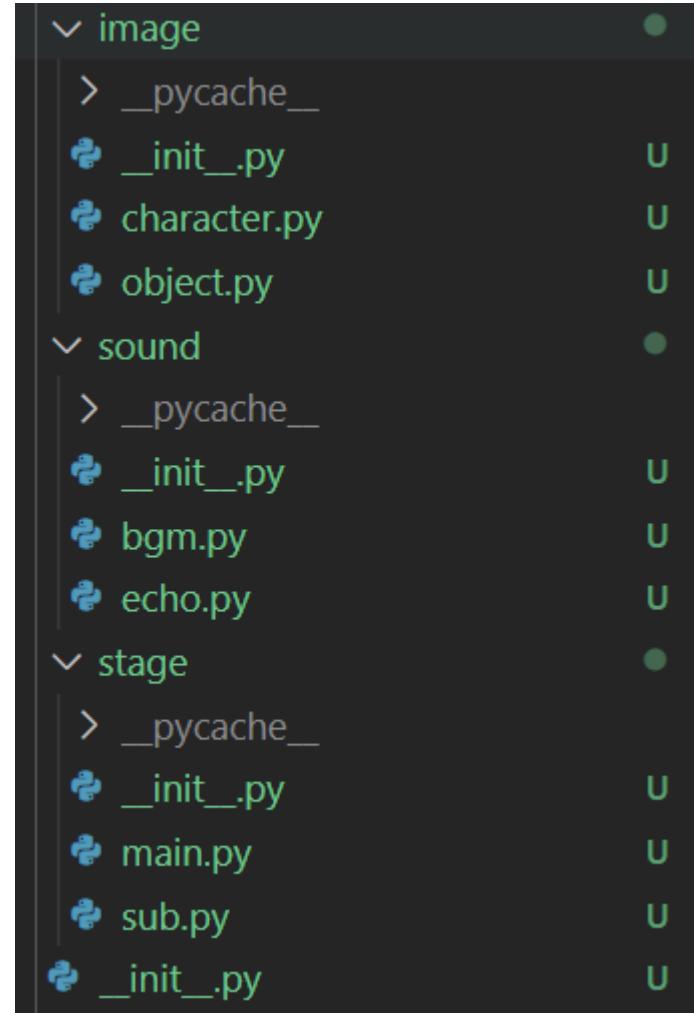
4) 폴더별로 __init__.py 구성하기

```
__all__= ['image', 'stage', 'sound']
```

```
from . import image
```

```
from . import stage
```

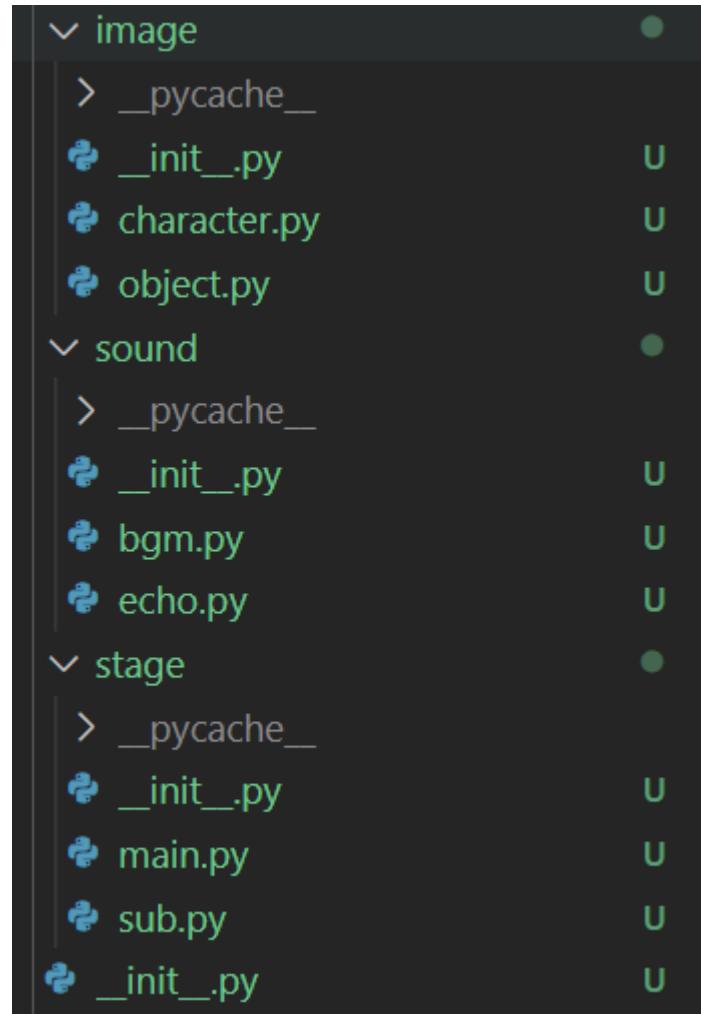
```
from . import sound
```



5) __main__.py 파일 만들기

```
from stage.main import game_start
from stage.sub import set_stage_level
from image.character import show_character
from sound.bgm import bgm_play

if __name__ == '__main__':
    game_start()
    set_stage_level(5)
    bgm_play(10)
    show_character()
```



Package 내에서 다른 폴더의 모듈을 부를 때 상대 참조로 호출하는 방법

절대참조

```
from game.graphic.render import render_test()
```

.현재 디렉토리 기준

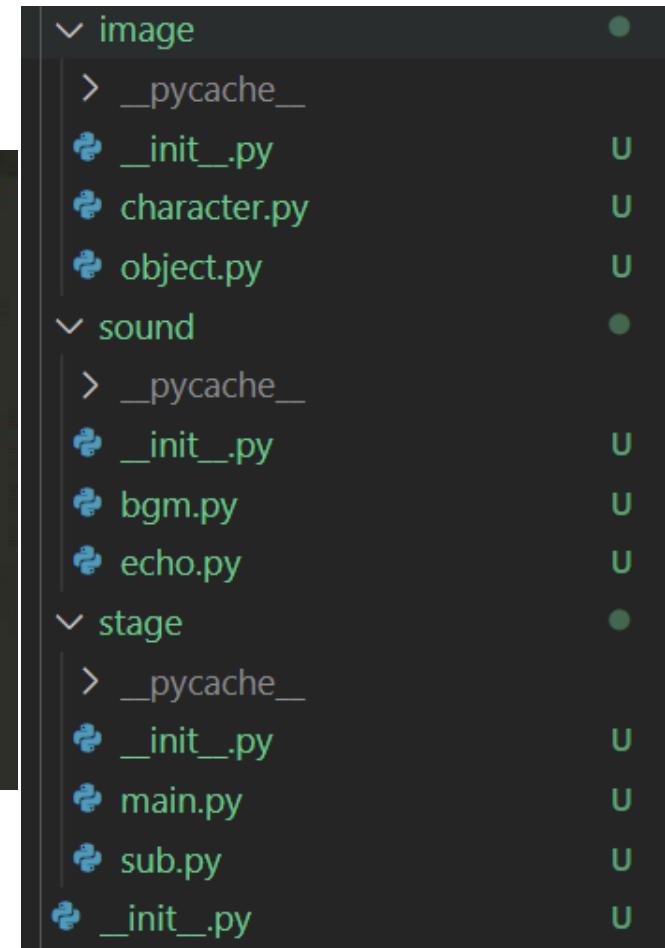
```
from .render import render_test()
```

```
from ..sound.echo import echo_test()
```

.. 부모 디렉토리 기준

6) 실행하기 - 패키지 이름만으로 호출하기

```
D:\workspace\data-academy\python\codes\ch9\package_example  
(base) λ ls  
game/  
  
D:\workspace\data-academy\python\codes\ch9\package_example  
(base) λ python game  
Game Start  
Set stage lavel: 5  
BGM 10 number start  
show_character
```



오픈소스 라이브러리 사용하기

진짜 프로젝트를 한다

내 PC에 패키지를 설치한다.

두 개의 프로젝트

웹과 데이터 분석

패키지는 둘다 설치?

가상환경 설정하기

Virtual Environment

- 프로젝트 진행 시 필요한 패키지만 설치하는 환경
- 기본 인터프리터 + 프로젝트 종류별 패키지 설치
 - ex) 웹 프로젝트, 데이터 분석 프로젝트
각각 패키지 관리할 수 있는 기능
- 다양한 패키지 관리 도구를 사용함

- 대표적인 도구 `virtualenv`와 `conda`가 있음

`virtualenv + pip`

가장 대표적인
가상환경 관리 도구

레퍼런스+패키지 개수

`conda`

상용 가상환경도구
`miniconda` 기본 도구

설치의 용이성
`Windows`에서 장점

```
conda create -n my project python=3.8
```

가상환경 새로 만들기

가상환경 이름

파이썬 버전

```
The following NEW packages will be INSTALLED:
```

certifi	pkgs/main/win-64::certifi-2019.11.28-py36_0
pip	pkgs/main/win-64::pip-19.3.1-py36_0
python	pkgs/main/win-64::python-3.6.9-h5500b2f_0
setuptools	pkgs/main/win-64::setuptools-42.0.2-py36_0
sqlite	pkgs/main/win-64::sqlite-3.30.1-he774522_0
vc	pkgs/main/win-64::vc-14.1-h0510ff6_4
vs2015_runtime	pkgs/main/win-64::vs2015_runtime-14.16.27012-hf0eaf9b_1
wheel	pkgs/main/win-64::wheel-0.33.6-py36_0
wincertstore	pkgs/main/win-64::wincertstore-0.2-py36h7fe50ca_0

```
Proceed ([y]/n)? |
```

가상환경 호출

```
conda activate my_project
```

```
#  
# To activate this environment, use:  
# > activate my_project  
#  
  
C:\Users\nhkim>activate my_project  
Deactivating environment "C:\Users\nhkim\Anaconda3" . . .  
Activating environment "C:\Users\nhkim\Anaconda3\envs\my_project" . . .
```

가상환경 해제

```
conda deactivate
```

```
conda install <패키지명>
```

설치하고자하는패키지명입력

```
conda install matplotlib
```

```
The following packages will be downloaded:
```

package	build	
ca-certificates-2019.11.27	0	124 KB
mkl_fft-1.0.15	py36h14836fe_0	118 KB
numpy-1.17.4	py36h4320e6b_0	5 KB
numpy-base-1.17.4	py36hc3f5095_0	3.8 MB
openssl-1.1.1d	he774522_3	4.8 MB
pyparsing-2.4.5	py_0	62 KB
python-dateutil-2.8.1	py_0	224 KB
six-1.13.0	py36_0	27 KB

	Total:	9.1 MB

Windows에서는
conda

linux, mac에서는
conda or pip

Windows에서는
conda

linux, mac에서는
pip

Windows에서는 컴파일된 C 라이브러리 설치 필요

```
The following NEW packages will be INSTALLED:  
  
cycler:          0.10.0-py34_0  
icu:            57.1-vc10_0  
jpeg:           8d-vc10_2  
libpng:         1.6.22-vc10_0  
matplotlib:    1.5.3-np111py34_  
mkl:            11.3.3-1  
numpy:          1.11.2-py34_0  
openssl:        1.0.2j-vc10_0  
pyparsing:      2.1.4-py34_0  
pyqt:           5.6.0-py34_0  
python-dateutil: 2.5.3-py34_0  
pytz:           2016.7-py34_0  
qt:              5.6.0-vc10_0  
sip:             4.18-py34_0  
six:             1.10.0-py34_0  
tk:              8.5.18-vc10_0  
zlib:            1.2.8-vc10_3
```



Windows에서는
conda

linux, mac에서는
conda or pip

matplotlib 활용한 그래프 표시

- 대표적인 파이썬 그래프 관리 패키지
- 엑셀과 같은 그래프들을 화면에 표시함
- 다양한 데이터 분석 도구들과 함께 사용됨

conda 가상환경 예시

가상환경

conda install <패키지명>

설치하고자하는패키지명입력

conda install matplotlib
conda install tqdm

The following packages will be downloaded:

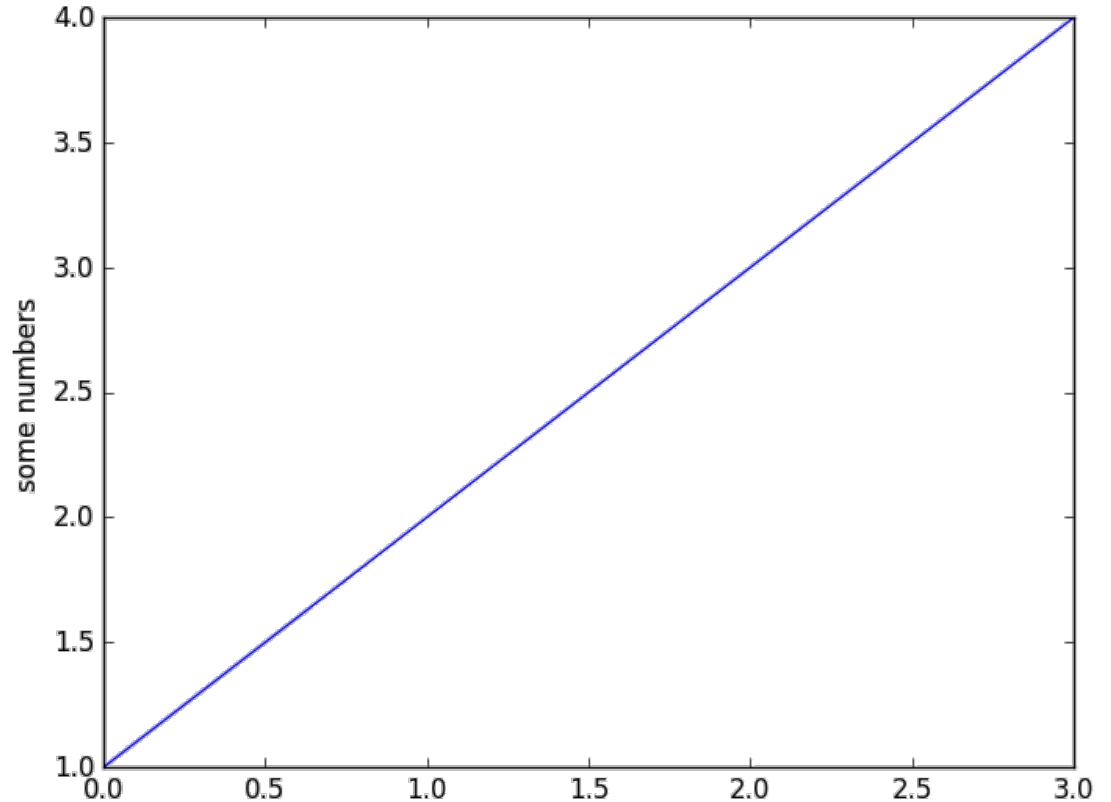
package	build	
ca-certificates-2019.11.27	0	124 KB
mkl_fft-1.0.15	py36h14836fe_0	118 KB
numpy-1.17.4	py36h4320e6b_0	5 KB
numpy-base-1.17.4	py36hc3f5095_0	3.8 MB
openssl-1.1.1d	he774522_3	4.8 MB
pyparsing-2.4.5	py_0	62 KB
python-dateutil-2.8.1	py_0	224 KB
six-1.13.0	py36_0	27 KB

	Total:	9.1 MB

conda 가상환경 예시

가상환경

```
import matplotlib.pyplot as plt  
plt.plot([1,2,3,4])  
plt.ylabel('some numbers')  
plt.show()
```



```
from tqdm import tqdm  
import time  
  
for i in tqdm(range(100000)):  
    if i % 1000 == 0:  
        time.sleep(1)
```

```
(my_project) λ python tqdm_example.py  
100%|██████████| 100000/100000 [00:00<00:00, 3333283.53it/s]
```

End of Document

Thank You.