

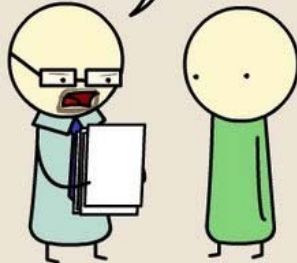
## Module & Package

출처: 최성철 교수



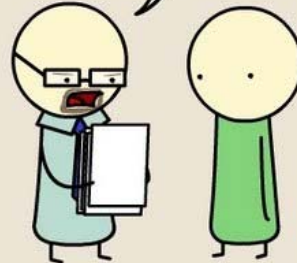
## PYTHON

THIS IS PLAGIARISM.  
YOU CAN'T JUST "IMPORT" ESSAY."



## JAVA

I'M TWO PAGES IN AND I STILL  
HAVE NO IDEA WHAT YOU'RE SAYING.



파이썬은 대부분의 라이브러리가 이미  
다른 사용자에게 의해서 구현되어 있음

---

그런데 어떻게  
쓰나요?

---

남이 만든 프로그램 쓰는 법

① 함수 ② 객체 ③ 모듈

# 모듈과 패키지

## 모듈 (Module)

- 어떤 대상의 부분 혹은 조각

예) 레고 블록, 벽돌, 자동차 부품들



<https://goo.gl/tlv3UO>

# 모듈 (Module)

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

# 모듈 (Module)

**KakaoDevelopers** | API 소개 | **개발가이드** | 개발자포럼 | 기술블로그 | Search | 로그인

## 개발가이드

- iOS 개발가이드
- iOS 레퍼런스
- Android 개발가이드
  - 시작하기
  - 사용자 관리
  - 카카오스토리
  - 카카오톡
  - 카카오링크
  - 푸시 알림
  - 오픈로그 분석
  - Troubleshooting
- Android 레퍼런스
- JavaScript 개발가이드
- JavaScript 레퍼런스
- REST API 개발가이드
- 소셜 플러그인
- REST API 도구
- Cache Tool
- SDK 다운로드

### 시작하기 [\[URL\]](#)

카카오 플랫폼 서비스는 카카오톡, 카카오스토리와 같은 카카오 서비스 및 카카오 플랫폼 기술과 관련된 API를 제공합니다. 해당 기능을 쉽게, 올바르게 사용하기 위해서는 카카오에서 제공하는 Kakao SDK가 필요합니다.

본 문서는 카카오 플랫폼 서비스를 사용하기 위한 기본 개발환경 구성 및 Kakao SDK 사용법뿐만 아니라, 카카오 플랫폼 서비스에서 제공하는 각각의 기능별 상세 설명을 포함합니다.

앞으로 제공될 상세기능에 대한 설명을 참고하거나 해당 기능을 실행해 보기전에, 반드시 아래의 기본 과정을 숙지하기를 권장합니다.

- 개발환경 구성
- 샘플앱 실행
- 앱 생성
- 문제해결

본 문서에서 제공되는 상세기능은 다음과 같습니다.

- 사용자 관리  
카카오계정을 통한 간편 로그인을 제공합니다. 이외에도 사용자들의 개인 정보를 손쉽게 관리해주는 기능을 포함합니다. 카카오 플랫폼 서비스에서 제공하는 기능 중 로그인이 필요한 기능을 사용하기 위해서는 반드시 실행되어야 할 내용들을 다룹니다.
- 카카오스토리  
카카오스토리에서 제공하는 API를 앱에서 직접 사용할 수 있습니다.
- 카카오톡  
카카오톡에서 제공하는 API를 앱에서 직접 사용할 수 있습니다.
- 카카오링크  
카카오톡 앱을 이용하여 다양한 메시지를 보낼 수 있습니다.
- 푸시 알림  
카카오계정에서 로그인한 앱에서 푸시 알림을 사용하기 위한 과정에 대해 설명합니다.
- 오픈로그 분석  
앱의 시작/종료, 사용자의 세션, 특정 이벤트 등 앱의 활동성 등을 분석할 수 있는 기능입니다.

아래 문서는 다음의 환경을 기준으로 작성되었으며, 개발자의 시스템 환경에 따라 구성이 조금씩 다를 수 있습니다.

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

---

## 모듈 (Module) in Python

- 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
>>>
```

---

## 패키지

- 모듈을 모아놓은 단위, 하나의 프로그램

```
Speech/           패키지의 최 상위 레벨
  __init__.py
SignalProcessing/ 신호 처리 하위 패키지
  __init__.py
  LPC.py
  Cepstrum.py
  FFT.py
  FilterBank.py
Recognition/      음성 인식 하위 패키지
  __init__.py
  HMM.py
  NN.py
  DTW.py
Synthesis/       음성 합성 하위 패키지
  __init__.py
  Tagging.py
  ProsodyContro.py
  DBAccess.py
```

Source: 파이썬 3바이블, 이강성

---

# Module 만들기

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

---

# Module 만들기

## 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")
```

---

# Namespace

- 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

---

# Namespace 사용 예시

## 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
```

---

## Built-in Modules

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

---

## Built-in Modules Examples

*#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://www.jnu.ac.kr")
print(response.read())
```



---

## Built-in Modules

- 수 많은 파이썬 모듈은 어떻게 검색할 것인가?
  - 1) 구글신에게 물어본다
  - 2) 모듈을 import 후 구글신 검색 또는 Help 쓰기
  - 3) 공식 문서를 읽어본다 <https://docs.python.org/3/library/>
- 실습: 1 부터 100까지 특정 난수를 뽑고 싶다!

---

## 패키지(Package)

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

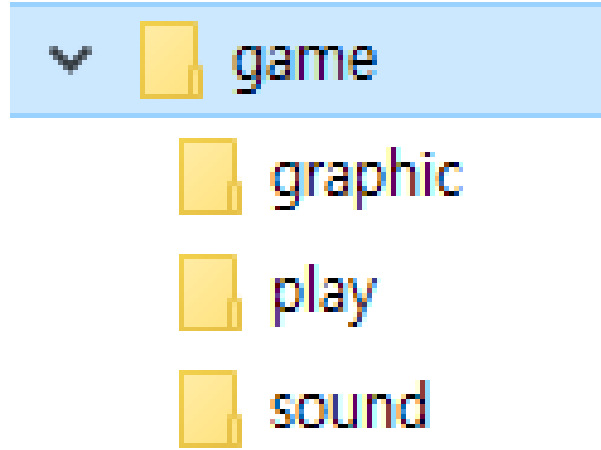
<https://wikidocs.net/1418>

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

---

# Package 만들기

## 1) 기능들을 세부적으로 나눠 폴더로 만듦

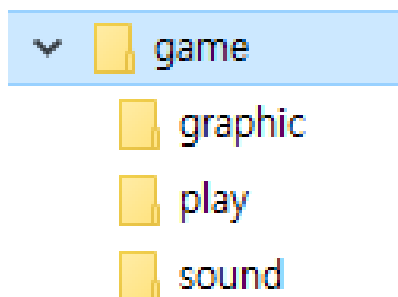


<https://wikidocs.net/1418>

---

# Package 만들기

## 2) 각 폴더별로 필요한 모듈을 구현함



```
game/  
  __init__.py  
  sound/  
    __init__.py  
    echo.py  
    wav.py  
  graphic/  
    __init__.py  
    screen.py  
    render.py  
  play/  
    __init__.py  
    run.py  
    test.py
```

<https://wikidocs.net/1418>

---

# Package 만들기

## 2) 각 폴더별로 필요한 모듈을 구현함

```
# echo.py
def echo_test():
    print ("echo")
```

```
# render.py
def render_test():
    print ("render")
```

## 3) 1차 Test – python shell

```
>>> from game.sound.echo import echo_test
>>> echo_test()
echo
>>> from game.graphic import render
>>> from game.graphic import render as rd
>>> rd.render_test()
render
>>> _
```

---

# Package 만들기

## 4) 폴더별로 `_init_.py` 구성하기

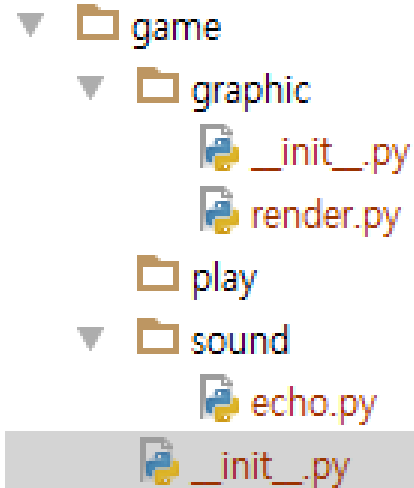
- 현재 폴더가 패키지임을 알리는 초기화 스크립트
- 없을 경우 패키지로 간주하지 않음 (3.3+ 부터는 X)
- 하위 폴더와 py 파일(모듈)을 모두 포함함
- import와 `_all_` keyword 사용

# Package 만들기

## 4) 폴더별로 `__init__.py` 구성하기

```
__all__=['graphic', 'play', 'sound']
```

```
import graphic  
import play  
import sound
```

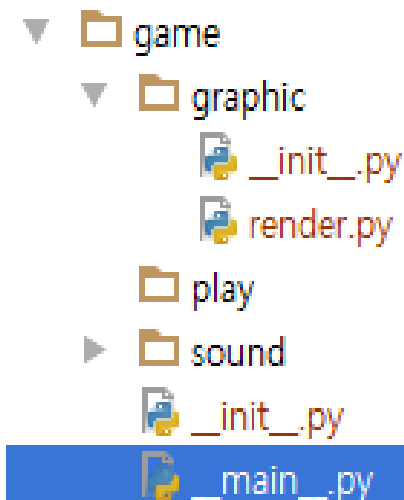


# Package 만들기

## 5) `__main__.py` 파일 만들기

```
from graphic.render import render_test  
from sound.echo import echo_test
```

```
if __name__ == '__main__':  
    render_test()  
    echo_test()
```



# Package 만들기

## 5) 실행하기 - 패키지 이름만으로 호출하기

python game

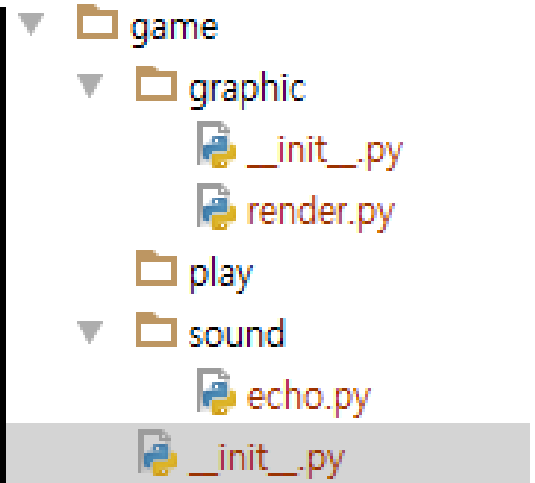
```
D: #workspace#Gachon_CS50_Python_KM00C#code#10#package>dir
D 드라이브의 볼륨: Data
볼륨 일련 번호: 2C24-D8FE

D: #workspace#Gachon_CS50_Python_KM00C#code#10#package 디렉터리

2016-10-28 오후 12:37 <DIR>      .
2016-10-28 오후 12:37 <DIR>      ..
2016-10-28 오후 01:06 <DIR>      game
                0개 파일              0 바이트
                3개 디렉터리  347,535,712,256 바이트 남음

D: #workspace#Gachon_CS50_Python_KM00C#code#10#package>python game
render
echo

D: #workspace#Gachon_CS50_Python_KM00C#code#10#package>
```



# Package namespace

## Package 내에서 다른 폴더의 모듈을 부를 때

### 상대 참조로 호출하는 방법

```
from game.graphic.render import render_test()
    . 현재 디렉토리 기준

from .render import render_test()

from ..sound.echo import echo_test()
    .. 부모 디렉토리 기준
```

---

**진짜 프로젝트를 한다**  
**내 PC에 패키지를 설치한다.**

---

**두 개의 프로젝트**  
**웹 과 데이터 분석**  
**패키지는 둘다 설치?**

# 가상환경 설정하기

## Virtual Environment

---

### 가상환경 (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 도구**를 사용함

# 가상환경 (Virtual Environment)

- 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  
**Windows에서 장점**

## Conda 가상환경

```
conda create -n my project python=3.4
```

가상환경 새로 만들기

가상환경 이름

파이썬 버전

```
Fetching package metadata: .....
Solving package specifications: .....
Package plan for installation in environment C:\Users\wnhkim\Anaconda3\envs\my_p
object:

The following packages will be downloaded:

  package | build | size
  -----|-----|-----
  setuptools-27.2.0 | py34_1 | 762 KB

The following NEW packages will be INSTALLED:

  pip: 8.1.2-py34_0
  python: 3.4.5-0
  setuptools: 27.2.0-py34_1
  vs2010_runtime: 10.00.40219.1-2
  wheel: 0.29.0-py34_0

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



# Conda 가상환경

## 가상환경 호출

```
activate my_project
```

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

## 가상환경 해제

```
deactivate
```

# 패키지 설치

```
conda install <패키지명>
```

설치하고자하는 패키지명 입력

```
conda install matplotlib
```

The following packages will be downloaded:

| package          | build       |          |
|------------------|-------------|----------|
| icu-57.1         | vc10_0      | 34.3 MB  |
| jpeg-8d          | vc10_2      | 177 KB   |
| openssl-1.0.2j   | vc10_0      | 4.5 MB   |
| numpy-1.11.2     | py34_0      | 3.2 MB   |
| pytz-2016.7      | py34_0      | 170 KB   |
| qt-5.6.0         | vc10_0      | 51.1 MB  |
| sip-4.18         | py34_0      | 236 KB   |
| pyqt-5.6.0       | py34_0      | 4.2 MB   |
| matplotlib-1.5.3 | np111py34_1 | 6.2 MB   |
| Total:           |             | 104.0 MB |

---

**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       | [vc10] |
| jpeg:            | 8d-vc10_2         | [vc10] |
| libpng:          | 1.6.22-vc10_0     | [vc10] |
| matplotlib:      | 1.5.3-np111py34_0 |        |
| mkl:             | 11.3.3-1          |        |
| numpy:           | 1.11.2-py34_0     |        |
| openssl:         | 1.0.2j-vc10_0     | [vc10] |
| pyarsing:        | 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      | [vc10] |
| sip:             | 4.18-py34_0       |        |
| six:             | 1.10.0-py34_0     |        |
| tk:              | 8.5.18-vc10_0     | [vc10] |
| zlib:            | 1.2.8-vc10_3      | [vc10] |

---

# Conda 가상환경 예시

## matplotlib 활용한 그래프 표시

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

<http://matplotlib.org/>

---

# Conda 가상환경 예시

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

