Python Module Structure I

ESM2014-41

객체지향프로그래밍 및 실습

SKKU 시스템경영공학과

조영일

Terminology

- Module : 코드의 재사용성을 위한 최소 단위, 파이썬 파일(*.py)로 이루어져 있으며 이미 작성된 클래스, 함수 들의 집합이다.

- Package : Module 들의 논리적인 집합. 하나의 주제와 관련된 문제들을 해결하기 위해 관련된 하위 Module들이 존재한다.

- Script : 실행 가능한 모든 Python 파일(*.py)

- Standard Library : Python Interpreter와 함께 기본적으로 배포되는 Module과 Package 들의 집합

Importing from a Module(from, import, as)

- import module1[, module2[, … moduleN] : 특정 module들을 사용할 수 있도록 불러온다.

- import module as nickname : 특정 모듈을 import하면서, Name(Namespace)를 변경한다.

- from module import name : 특정 module 의 특정 기능만을 사용할 수 있도록 불러온다.

Importing from a Module(from, import, as)

```
# import module1[, module2[, … moduleN]
import turtle, random, math

# import / as
import math as m
m.sqrt(16)

# from / import
import math
sqrt(16) # ERRROR!

from math import sqrt
sqrt(16)

# math 모듈 내의 모든 함수 import
# ** 추천하지 않는 방식
from math import *
sin(radians(90))
cos(radians(180))

pi = "Raspberry Pie"
from math import *
print(pi) # 3.141592653589793
```

Package Structures

```
movie_theater/
├── __init__.py
├── theaters/
│   ├── __init__.py
│   ├── projector.py
│   ├── screen.py
│   └── seat.py
├── employees/
│   ├── __init__.py
│   ├── manager.py
│   ├── employee.py
│   └── alba.py
└── movies/
    ├── __init__.py
    └── movie.py
```

- 모든 Package 는 디렉토리와 디렉토리에 속한 파일들로 구성됨
- __init__.py : 단순 디렉토리가 아니라 Python Package 임을 나타내기 위한 방법
- Package : movie_theater
- Sub Package : theaters, employees, movies

Package Structures

```
movie_theater/
├── __init__.py
├── theaters/
│   ├── __init__.py
│   ├── projector.py
│   ├── screen.py
│   └── seat.py
├── employees/
│   ├── __init__.py
│   ├── manager.py
│   ├── employee.py
│   └── alba.py
└── movies/
    ├── __init__.py
    └── movie.py
```

```python
import movie_theater.theaters.projector
movie_theater.theaters.projector.play_movie("Joker")

from movie_theater.theaters import projector
projector.play_movie("Joker")

from movie_theater.theaters.projector import play_movie
play_movie("Joker")
```

Importing conventions

- import 구문은 파일의 최상단에 위치 시킨다. : 명확한 의존성 표시, 특정 조건에 따라 import 하는 것은 가능한 회피한다.

- 가능한 from … import … 구문보다는 import … 구문을 사용한다. : module/package 에 따른 명확한 namespace 구분, namespace 충돌의 방지

- from … import * 구문은 사용하지 않는다. : 무엇이 import 되는지 명확히 알 수 없으며, 여러 module/package에서 import 하는 경우 namespace 충돌이 발생한다.

Importing Priority

- Import 할 Module/Package 를 찾는 우선 순위(Namespace 개념과 유사함)

    1. Built-in Module
    2. 실행하는 Script 가 존재하는 현재 Directory
    3. 시스템 변수 상의 $PYTHONPATH
    4. Default Python Installation Path

Python Standard Library

- Concept of Python : "Python" is a "batteries-included“

- 범용적으로 많이 사용되는 기능은 이미 module/package 형태로 개발되어 “Standard Library” 형태로 Python Interpreter에 기본 탑재되어 배포됨

Python Standard Library

- collections – container datatypes: namedtuple / defaultdict / counter
- math – default mathematics tools
- re – regular expressions: patterns for strings
- itertools – iterators for efficient looping
- json – encode and decode structured JSON data
- random – generate pseudo-random numbers
- sys – interact with system
- pathlib – intelligent filesystem navigation
- subprocess/shlex – spawn processes
- debugging – breakpoint(), pprint, timeit
- miscellaneous- turtle, unicodedata, this, antigravity

Python Standard Library

- https://docs.python.org/3/py-modindex.html

Summary

- Terminology : Module/Package/Script/Standard Library
- Importing from module/package
- Importing conventions
- Importing priority
- Python Standard Library