16장. 모듈과 패키지

파이썬 정복

Contents

❖ 목차

1. 모듈

❖ 모듈의 작성 및 사용

▪ 모듈

- 파이썬 코드를 저장하는 기본 단위
- 편의상 스크립트를 여러 개의 파일로 나눈 하나
- .py 빼고 파일명으로 불림
- 파이썬에서 자주 사용하는 기능은 표준 모듈로 제공됨
- 직접 제작 가능

util

```
INCH = 2.54

def calcsum(n):
    sum = 0
    for num in range(n + 1):
        sum += num
    return sum
```

1. 모듈

utiltest

```
import util

print("1inch =", util.INCH)
print("~10 =", util.calcsum(10))
```

실행결과

```
1inch = 2.54
~10 = 55
```

1. 모듈

❖ 테스트 코드

- 모듈에 간단한 테스트 코드를 작성할 수 있음

util

```
INCH = 2.54

def calcsum(n):
    sum = 0
    for num in range(n + 1):
        sum += num
    return sum

print("인치 =", INCH)
print("합계 =", calcsum(10))
```

실행결과

```
인치 = 2.54
합계 = 55
```

1. 모듈

- 타 모듈에 함수 제공하는 모듈에서는 테스트 코드를 조건문으로 감쌈

util2

```
INCH = 2.54

def calcsum(n):
    sum = 0
    for num in range(n + 1):
        sum += num
    return sum

if __name__ == "__main__":
    print("인치 =", INCH)
    print("합계 =", calcsum(10))
```

utiltest2

```
import util2

print("1inch =", util2.INCH)
print("~10 =", util2.calcsum(10))
```

1. 모듈

❖ 모듈 경로

- 모듈은 임포트하는 파일과 같은 디렉토리에 있어야 함

```
Traceback (most recent call last):
  File "C:\PyStudy\utiltest2.py", line 1, in <module>
    import util2
ModuleNotFoundError: No module named 'util2'
```

- 모듈을 특정 폴더에 두려면 임포트 패스에 추가

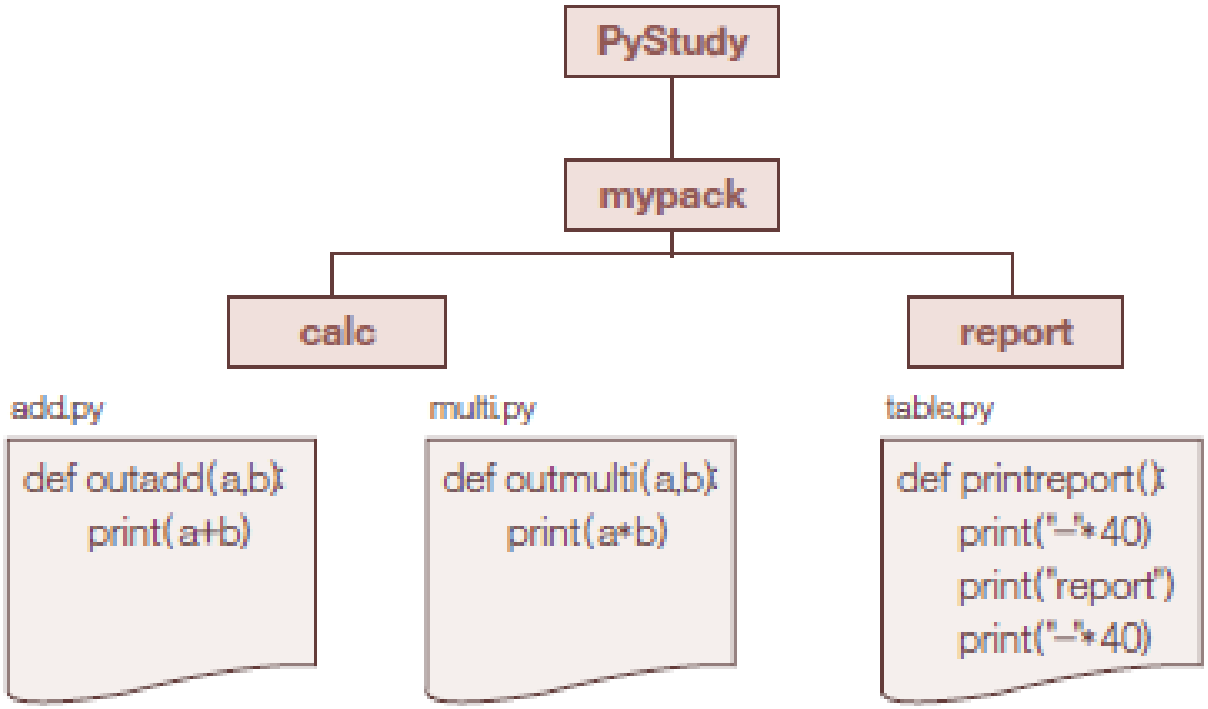
```
>>> import sys
>>> sys.path
['C:\\PyStudy', 'C:\\Python\\Lib\\idlelib', 'C:\\Python\\python36.zip', 'C:\\
Python\\DLLs', 'C:\\Python\\lib', 'C:\\Python', 'C:\\Python\\lib\\site-packages']
```

```
>>> sys.path.append("C:\\Temp")
>>> import util2
>>> util2.calcsum(10)
55
```

2. 패키지

❖ 패키지

- 모듈을 담는 디렉토리
- 디렉토리로 계층 구성하면 모듈을 기능 등에 따라 분류 가능



2. 패키지

usepackage

```
import sys
sys.path.append("C:/PyStudy")

import mypack.calc.add
mypack.calc.add.outadd(1,2)

import mypack.report.table
mypack.report.table.outreport()
```

실행결과

```
3
---------------------------------------
report
---------------------------------------
```

- 함수명이나 모듈명에 충돌 발생하지 않음
- 단일 모듈에 비해 호출문 길어지는 불편함 있음
  - from 패키지 import 모듈

```
from mypack.calc import add
add.outadd(1,2)
```

2. 패키지

❖ __init__.py

- 모든 모듈 한꺼번에 불러올 때는 어떤 모듈 대상인지 밝혀두어야
- 임포트할 대상 모듈 리스트 명시
- 패키지 로드될 때의 초기화 코드 작성해 둠

importall

```
import sys
sys.path.append("C:/PyStudy")

from mypack.calc import *
add.outadd(1,2)
multi.outmulti(1,2)
```

실행결과

```
Traceback (most recent call last):
  File "C:\PyStudy\Test.py", line 5, in <module>
    add.outadd(1,2)
NameError: name 'add' is not defined
```

2. 패키지

__init__

```
__all__ = ["add", "multi"]

print("add module imported")
```

- import * 로 읽을 때 add 및 multi 모듈 모두 읽어 옴
- __init__.py의 초기화 코드 실행

- __init__.py 목록에 어떤 모듈 작성할 것인지는 패키지 개발자 재량

- 한 번 임포트한 모듈은 컴파일 상태로 캐시에 저장
  - 확장자 pyc
  - 한 모듈을 각각 다른 파이썬 버전끼리 공유 가능

3. 서드 파티 모듈

❖ 모듈의 내부

- 각 모듈은 기능별로 나누어져 다수 함수 포함
- dir 내장 함수
  - 모듈에 있는 함수나 변수 목록 조사

```
>>> import math
>>> dir(math)
['__doc__', '__loader__', '__name__', '__package__', '__spec__', 'acos',
'acosh', 'asin', 'asinh', 'atan', 'atan2', 'atanh', 'ceil', 'copysign',
'cos', 'cosh', 'degrees', 'e', 'erf', 'erfc', 'exp', 'expm1', 'fabs',
'factorial', 'floor', 'fmod', 'frexp', 'fsum', 'gamma', 'gcd', 'hypot',
'inf', 'isclose', 'isfinite', 'isinf', 'isnan', 'ldexp', 'lgamma', 'log',
'log10', 'log1p', 'log2', 'modf', 'nan', 'pi', 'pow', 'radians', 'sin',
'sinh', 'sqrt', 'tan', 'tanh', 'tau', 'trunc']
```

3. 서드 파티 모듈

❖ 외부 모듈의 목록

- 서드 파티 모듈 (Third Party Module)
  - 파이썬 외 회사 및 단체가 제작하여 배포하는 모듈

모듈	설명
Django, Flask	웹 프레임워크
BeautifulSoup	HTML, XML 파서
wxPython, PyGtk	그래픽 툴킷
pyGame	게임 제작 프레임워크
PIL	이미지 처리 라이브러리
pyLibrary	유틸리티 라이브러리

3. 서드 파티 모듈

❖ pip (Python Package Index)

- 외부 모듈 관리 용이
- pip 명령 패키지명

명령	설명
install	패키지를 설치한다.
uninstall	설치한 패키지를 삭제한다.
freeze	설치한 패키지의 목록을 보여 준다.
show	패키지의 정보를 보여 준다.
search	pyPI에서 패키지를 검색한다.

wxtest

```
import wx

app = wx.App()
frame = wx.Frame(None, 0, "파이썬 만세")

frame.Show(True)
app.MainLoop()
```

Thank You !

파이썬 정복