Python Workshop

- Raspberry Pi -

2017년 8월

서강대학교

Module

https://wikidocs.net/29

```
# mod1.py
def sum(a, b):
    return a + b
```

```
def safe_sum(a, b):
    if type(a) != type(b):
        print("더할수 있는 것이 아닙니다.")
        return
    else:
        result = sum(a, b)
    return result
```

```
>>> import mod1
>>> print(mod1.sum(3,4))
7
```

```
>>> from mod1 import sum
>>> sum(3, 4)
7
```

```
from mod1 import sum, safe_sum
```

```
from mod1 import *
```

if __name__ == "__main__":

```python
# mod1.py
def sum(a, b):
    return a+b

def safe_sum(a, b):
    if type(a) != type(b):
        print("더할수 있는 것이 아닙니다.")
        return
    else:
        result = sum(a, b)
    return result

print(safe_sum('a', 1))
print(safe_sum(1, 4))
print(sum(10, 10.4))
```

```
C:\Python>python mod1.py
```

```python
if __name__ == "__main__":
    print(safe_sum('a', 1))
    print(safe_sum(1, 4))
    print(sum(10, 10.4))
```

Access Module's Internal Elements

```python
# mod2.py
PI = 3.141592

class Math:
    def solv(self, r):
        return PI * (r ** 2)

def sum(a, b):
    return a+b

if __name__ == "__main__":
    print(PI)
    a = Math()
    print(a.solv(2))
    print(sum(PI , 4.4))
```

```
>>> import mod2
>>> print(mod2.PI)
3.141592
```

```
>>> a = mod2.Math()
>>> print(a.solv(2))
12.566368
```

Free Module's Location

1. sys.path.append(모듈을 저장한 디렉터리) 사용하기

```
>>> import sys
>>> sys.path.append("C:/Python/Mymodules")
```

2. PYTHONPATH 환경 변수 사용하기

```
C:\Users\home>set PYTHONPATH=C:\Python\Mymodules
```

Package

https://wikidocs.net/1418

가상의 game 패키지 예

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

Package: Testing

```
C:/Python/game/__init__.py
C:/Python/game/sound/__init__.py
C:/Python/game/sound/echo.py
C:/Python/game/graphic/__init__.py
C:/Python/game/graphic/render.py
```

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

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

```
C:\> set PYTHONPATH=C:/Python
```

```
>>> import game.sound.echo
>>> game.sound.echo.echo_test()
echo
```

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

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

__init__.py

```
>>> from game.sound import *
>>> echo.echo_test()
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
NameError: name 'echo' is not defined
```

```
# C:/Python/game/sound/__init__.py
__all__ = ['echo']
```

Package 내에서 모듈 액세스

```python
# render.py
from game.sound.echo import echo_test
def render_test():
    print ("render")
    echo_test()
```

```python
# render.py
from ..sound.echo import echo_test

def render_test():
    print ("render")
    echo_test()
```

Exception

```
try:
    ...
except [발생 오류[as 오류 메시지 변수]]:
    ...
```

```
try:
    ...
except:
    ...
```

```
try:
    ...
except 발생 오류:
    ...
```

```
try:
    ...
except 발생 오류 as 오류 메시지 변수:
    ...
```

```
try:
    f = open('foo.txt', 'r')
except FileNotFoundError as e:
    print(str(e))
else:
    data = f.read()
    f.close()
```

```
f = open('foo.txt', 'w')
try:
    # 무언가를 수행한다.
finally:
    f.close()
```

```
try:
    4 / 0
except ZeroDivisionError as e:
    print(e)
```

Exception: Multiple Error

```
try:
    a = [1,2]
    print(a[3])
    4/0
except ZeroDivisionError:
    print("0으로 나눌 수 없습니다.")
except IndexError:
    print("인덱싱 할 수 없습니다.")
```

```
try:
    a = [1,2]
    print(a[3])
    4/0
except (ZeroDivisionError, IndexError) as e:
    print(e)
```

```
try:
    a = [1,2]
    print(a[3])
    4/0
except ZeroDivisionError as e:
    print(e)
except IndexError as e:
    print(e)
```

Exception: Intentional Raise

```
try:
    f = open("나없는파일", 'r')
except FileNotFoundError:
    pass
```

```
class Bird:
    def fly(self):
        raise NotImplementedError
```

```
class MyError(Exception):
    pass
```

```
def say_nick(nick):
    if nick == '바보':
        raise MyError()
    print(nick)
```

```
say_nick("천사")
say_nick("바보")
```

```
try:
    say_nick("천사")
    say_nick("바보")
except MyError:
    print("허용되지 않는 별명입니다.")
```

Exception: Custom Exception

```
class MyError(Exception):
    def __init__(self, msg):
        self.msg = msg

    def __str__(self):
        return self.msg


def say_nick(nick):
    if nick == '바보':
        raise MyError("허용되지 않는 별명입니다.")
    print(nick)

try:
    say_nick("천사")
    say_nick("바보")
except MyError as e:
    print(e)
```

Built-In Function

https://wikidocs.net/32

- abs
- all
- any
- chr
- dir
- divmod
- enumerate
- eval
- filter
- hex
- id
- input

```
>>> for i, name in enumerate(['body', 'foo', 'bar']):
...     print(i, name)
...
0 body
1 foo
2 bar
```

```
def positive(x):
    return x > 0

print(list(filter(positive, [1, -3, 2, 0, -5, 6])))
```

```
>>> print(list(filter(lambda x: x > 0, [1, -3, 2, 0, -5, 6])))
```

Built-In Function (Cont'd)

- int
- isinstance
- lambda
- len
- list
- max
- min
- oct
- open
- ord
- pow
- range

lambda 인수1, 인수2, ... : 인수를 이용한 표현식

```
>>> sum = lambda a, b: a+b
>>> sum(3,4)
7
```

```
>>> myList = [lambda a,b:a+b, lambda a,b:a*b]
>>> myList[0](3,4)
7
>>> myList[1](3,4)
12
```

Built-In Function (Cont'd)

- sorted
- str
- tuple
- type
- zip

```
>>> sorted([3, 1, 2])
[1, 2, 3]
>>> sorted(['a', 'c', 'b'])
['a', 'b', 'c']
>>> sorted("zero")
['e', 'o', 'r', 'z']
>>> sorted((3, 2, 1))
[1, 2, 3]
```

```
>>> a = [3, 1, 2]
>>> result = a.sort()
>>> print(result)
None
>>> a
[1, 2, 3]
```

```
>>> list(zip([1, 2, 3], [4, 5, 6]))
[(1, 4), (2, 5), (3, 6)]
>>> list(zip([1, 2, 3], [4, 5, 6], [7, 8, 9]))
[(1, 4, 7), (2, 5, 8), (3, 6, 9)]
>>> list(zip("abc", "def"))
[('a', 'd'), ('b', 'e'), ('c', 'f')]
```

Built-In Function (Cont'd)

- map

map(f, iterable)은 함수(f)와 반복 가능한(iterable) 자료형을 입력으로 받는다. map은 입력받은 자료형의 각 요소가 함수 f에 의해 수행된 결과를 묶어서 리턴하는 함수이다.

```
>>> def two_times(x): return x*2
...
>>> list(map(two_times, [1, 2, 3, 4]))
[2, 4, 6, 8]
```

```
>>> list(map(lambda a: a*2, [1, 2, 3, 4]))
[2, 4, 6, 8]
```

External Functions

- sys

```
C:/User/home>python test.py abc pey guido
```

```
# argv_test.py
import sys
print(sys.argv)
```

```
C:/Python/Mymodules>python argv_test.py you need python
['argv_test.py', 'you', 'need', 'python']
```

```
>>> sys.exit()
```

```
# path_append.py
import sys
sys.path.append("C:/Python/Mymodules")
```

External Functions (Cont'd)

- pickle

```
>>> import pickle
>>> f = open("test.txt", 'wb')
>>> data = {1: 'python', 2: 'you need'}
>>> pickle.dump(data, f)
>>> f.close()
```

```
>>> import pickle
>>> f = open("test.txt", 'rb')
>>> data = pickle.load(f)
>>> print(data)
{2:'you need', 1:'python'}
```

OS 모듈

```
>>> import os
>>> os.environ
environ({'PROGRAMFILES': 'C:\\Program Files', 'APPDATA': … 생략 …})
>>>
```

```
>>> os.environ['PATH']
'C:\\ProgramData\\Oracle\\Java\\javapath;...생략...'
```

```
>>> os.chdir("C:\WINDOWS")
```

```
>>> os.getcwd()
'C:\WINDOWS'
```

OS 모듈

- 시스템 명령어 호출

```
>>> os.system("dir")
```

```
>>> f = os.popen("dir")
>>> print(f.read())
```

shutil

```
>>> import shutil
>>> shutil.copy("src.txt", "dst.txt")
```

glob

- 디렉토리에 있는 화일들을 리스트로 만들기

```
>>> import glob
>>> glob.glob("C:/Python/q*")
['C:\Python\quiz.py', 'C:\Python\quiz.py.bak']
>>>
```

tempfile

```
>>> import tempfile
>>> filename = tempfile.mktemp()
>>> filename
'C:\WINDOWS\TEMP\~-275151-0'
```

```
>>> import tempfile
>>> f = tempfile.TemporaryFile()
>>> f.close()
```

time

```
>>> import time
>>> time.time()
988458015.73417199
```

```
>>> time.localtime(time.time())
time.struct_time(tm_year=2013, tm_mon=5, tm_mday=21, tm_hour=16,
    tm_min=48, tm_sec=42, tm_wday=1, tm_yday=141, tm_isdst=0)
```

```
>>> time.asctime(time.localtime(time.time()))
'Sat Apr 28 20:50:20 2001'
```

```
>>> time.ctime()
'Sat Apr 28 20:56:31 2001'
```

```
>>> import time
>>> time.strftime('%x', time.localtime(time.time()))
'05/01/01'
>>> time.strftime('%c', time.localtime(time.time()))
'05/01/01 17:22:21'
```

time.sleep

```
#sleep1.py
import time
for i in range(10):
    print(i)
    time.sleep(1)
```

calendar

```
>>> import calendar
>>> print(calendar.calendar(2015))
```

```
>>> calendar.prcal(2015)
```

```
>>> calendar.weekday(2015, 12, 31)
3
```

```
>>> calendar.monthrange(2015,12)
(1, 31)
```

random

```
>>> import random
>>> random.random()
0.53840103305098674
```

```
>>> random.randint(1, 10)
6
```

```
>>> import random
>>> data = [1, 2, 3, 4, 5]
>>> random.shuffle(data)
>>> data
[5, 1, 3, 4, 2]
>>>
```

```
# random_pop.py
import random
def random_pop(data):
    number = random.randint(0, len(data)-1)
    return data.pop(number)

if __name__ == "__main__":
    data = [1, 2, 3, 4, 5]
    while data: print(random_pop(data))
```

```
def random_pop(data):
    number = random.choice(data)
    data.remove(number)
    return number
```

webbrowser

```
>>> import webbrowser
>>> webbrowser.open("http://google.com")
```

```
>>> webbrowser.open_new("http://google.com")
```

Thread

```python
import threading
import time

def say(msg):
    while True:
        time.sleep(1)
        print(msg)

for msg in ['you', 'need', 'python']:
    t = threading.Thread(target=say, args=(msg,))
    t.daemon = True
    t.start()

for i in range(100):
    time.sleep(0.1)
    print(i)
```

virtualenv

- http://docs.python-guide.org/en/latest/dev/virtualenvs/
- virtualenv is a tool to create isolated Python environments. virtualenv creates a folder which contains all the necessary executables to use the packages that a Python project would need.

- Install virtualenv
  $ pip install virtualenv
  $ virtualenv –version

- Create your own virtual environment
  $ cd my_project_folder
  $ virtualenv my_project → *creat a directory named "my_project"*
  $ ls my_project

- Activate specific virtual environment
  $ source my_project/bin/activate
  $ which python

- Install Package
  $ pip install requests