

Ask Django

장식자 (Decorator)

장식자 (Decorator) #doc

- 어떤 함수를 감싸는 (Wrapping) 목적의 함수
- (잠깐) 1급 함수 : 함수를 동적으로 생성 가능, 반환값으로 전달 가능

```
def base_10(fn):  
    def wrap(x, y):  
        return x + y + 10  
    return wrap
```

```
def mysum(x, y):  
    return x + y  
mysum = base_10(mysum)
```

```
>>> mysum(1, 2)  
13
```

```
@base_10
```

```
def mysum(x, y):  
    return x + y
```

```
>>> mysum(1, 2)  
13
```

Example: memoize

```
import time

def memoize(fn):
    cached = {}
    def wrap(x, y):
        key = (x, y)
        if key not in cached:
            cached[key] = fn(x, y)
        return cached[key]
    return wrap
```

```
def long_mysum1(x, y):  
    time.sleep(1)  
    return x + y
```

@memoize

```
def long_mysum2(x, y):  
    time.sleep(1)  
    return x + y
```

```
for i in range(3):  
    print(long_mysum1(1, 2))
```

실제 소요시간 약 3초
memoize 미적용

```
for i in range(3):  
    print(long_mysum2(1, 2))
```

실제 소요시간 약 1초
memoize 적용

장식자에 인자 지원

```
def base(base_i):  
    def outer(fn):  
        def wrap(x, y):  
            return x + y + base_i  
        return wrap  
    return outer
```

```
@base(20)  
def mysum2(x, y):  
    return x + y
```

```
@base(30)  
def mysum3(x, y, z):  
    return x + y + z
```

```
>>> mysum2(1, 2)  
23  
>>> mysum3(1, 2, 3)  
36
```

Quiz: 지정된 조건의 인자만 처리하기

- **filter_fn**을 통과하지 못하는 인자는 **alter_value** 값으로 대체하기


```
def myfilter(filter_fn, alter_value):  
    def wrap(fn):  
        def inner(*args):  
            raise NotImplementedError( ' 구현해주세요. ' )    # TODO  
        return inner  
    return wrap
```

```
@myfilter(lambda i: i%2==0, 0)
def mysum(a, b, c, d, e):
    return a + b + c + d + e
```

```
@myfilter(lambda i: i%2==0, 1)
def mymultiply(a, b, c, d, e):
    return a + b + c + d + e
```

```
>>> mysum(1, 2, 3, 4, 5)
6
```

```
>>> mymultiply(1, 2, 3, 4, 5)
8
```

*Life is short,
use Python3/Django.*