lesson_4

August 23, 2022

1 Lesson 4

1.1 Decorators

1.1.1 First Class Functions

[1]:
```
def foo():
    print("Here I am")
```

[2]:
```
foo()
```

```
Here I am
```

[3]:
```
def bar(func):
    print("Executing function")
    func()
    print("Done!")
```

[4]:
```
bar(foo)
```

```
Executing function
Here I am
Done!
```

[5]:
```
foo
```

[5]:
```
<function __main__.foo()>
```

[6]:
```
baz = foo
```

[7]:
```
baz
```

[7]:
```
<function __main__.foo()>
```

[8]:
```
baz()
```

```
Here I am
```

[9]:
```
foo()
```

```
Here I am
```

[10]:
```
def foo():
    def bar():
        print("I'm here now")

    return bar
```

[11]:
```
f = foo()
```

[12]:
```
f
```

[12]:
```
<function __main__.foo.<locals>.bar()>
```

[13]:
```
f()
```

```
I'm here now
```

[14]:
```
def foo():
    def bar(name):
        print(f"My name is: {name}")

    return bar
```

[15]:
```
f = foo()
```

[16]:
```
f
```

[16]:
```
<function __main__.foo.<locals>.bar(name)>
```

[18]:
```
f("Henry")
```

```
My name is: Henry
```

[19]:
```
def foo(name):
    def bar():
        print(f"My name is: {name}")

    return bar
```

[20]:
```
f = foo("Henry")
```

[21]:
```
f()
```

```
My name is: Henry
```

[22]:
```
f()
```

```
My name is: Henry
```

[23]:
```
def greeter():
    print("Welcome to our program")
```

[24]:
```
greeter()
```

```
Welcome to our program
```

[29]:
```
def func_logger(func):
    print(f"Executing the function: {func.__name__}")
    func()
    print("Done!")
```

[30]:
```
func_logger(greeter)
```

```
Executing the function: greeter
Welcome to our program
Done!
```

[32]:
```
greeter()
```

```
Welcome to our program
```

[33]:
```
def func_logger(func):
    def wrapper():
        print(f"Executing the function: {func.__name__}")
        func()
        print("Done!")

    return wrapper
```

[35]:
```
f = func_logger(greeter)
```

[36]:
```
f
```

[36]:
```
<function __main__.func_logger.<locals>.wrapper()>
```

[37]:
```
f()
```

```
Executing the function: greeter
Welcome to our program
Done!
```

[38]:
```
greeter()
```

```
Welcome to our program
```

[39]:
```
greeter = func_logger(greeter)
```

[41]:
```
greeter()
```

```
Executing the function: greeter
Welcome to our program
Done!
```

[42]:
```
def func_logger(func):
    def wrapper():
        print(f"Executing the function: {func.__name__}")
        func()
        print("Done!")

    return wrapper
```

[46]:
```
def greeter():
    print("Welcome")

greeter = func_logger(greeter)

# =========== SAME AS =============

@func_logger
def greeter():
    print("Welcome")
```

[47]:
```
@func_logger
def greeter():
    print("Welcome")
```

[48]:
```
greeter()
```

```
Executing the function: greeter
Welcome
Done!
```

[64]:
```
def func_logger(func):
    def wrapper(*args, **kwargs):
        print(f"Executing the function: {func.__name__}")
        func(*args, **kwargs)
        print("Done!")

    return wrapper
```

[65]:
```
@func_logger
def greeter(name):
```

```
    print(f"Welcome {name}!")
```

[66]:
```
greeter("Henry")
```

```
Executing the function: greeter
Welcome Henry!
Done!
```

[67]:
```
@func_logger
def introducer(name, age):
    print(f"Hello! I'm {name} and I am {age} years old")
```

[68]:
```
introducer("Adam", 18)
```

```
Executing the function: introducer
Hello! I'm Adam and I am 18 years old
Done!
```

[81]:
```
def func_logger(func):
    def wrapper(*args, **kwargs):
        print(f"Executing the function: {func.__name__}")
        res = func(*args, **kwargs)
        print("Done!")
        return res

    return wrapper
```

[82]:
```
def square(x):
    return x**2
```

[83]:
```
y = square(2)
```

[84]:
```
y
```

[84]:
```
4
```

[85]:
```
@func_logger
def square(x):
    return x**2
```

[86]:
```
y = square(2)
```

```
Executing the function: square
Done!
```

[87]:
```
y
```

[87]: 4

1.2 Unpacking

[88]:
```
t = (42, "Henry", True)
```

[89]:
```
age, name, is_teacher = t
```

[90]:
```
age
```

[90]: 42

[91]:
```
name
```

[91]: 'Henry'

[92]:
```
is_teacher
```

[92]: True

[94]:
```
age, *_ = t
```

[95]:
```
age
```

[95]: 42

[96]:
```
def student_list(students):
    for i, name in enumerate(students):
        print(f"{i + 1}: {name}")
```

[97]:
```
student_list(["Henry", "Adam", "Jack", "John"])
```

```
1: Henry
2: Adam
3: Jack
4: John
```

[98]:
```
student_list(("Henry", "Adam", "Jack", "John"))
```

```
1: Henry
2: Adam
3: Jack
4: John
```

[101]:
```
def student_list(*students):
    print(students)
    print(type(students))
```

```
    for i, name in enumerate(students):
        print(f"{i + 1}: {name}")
```

[103]:
```
student_list("Henry", "Adam", "Jack", "John", "Caroline")
```

```
('Henry', 'Adam', 'Jack', 'John', 'Caroline')
<class 'tuple'>
1: Henry
2: Adam
3: Jack
4: John
5: Caroline
```

[145]:
```
def student_info(name, **kwargs):
    print(f"Student {name} \n {', '.join([f'{key}: {val}' for key, val in␣
 ↪kwargs.items()])}")
```

[146]:
```
student_info("Adam", age=42, gender="Male", city="Amsterdam")
```

```
Student Adam
 age: 42, gender: Male, city: Amsterdam
```

1.3 Function Names with Decorators

[147]:
```
def func_logger(func):
    def wrapper(*args, **kwargs):
        print(f"Executing the function: {func.__name__}")
        res = func(*args, **kwargs)
        print("Done!")
        return res

    return wrapper
```

[151]:
```
def foo():
    """This is a simple function."""
    print("welcome")
```

[152]:
```
foo.__name__
```

[152]:
```
'foo'
```

[153]:
```
foo.__doc__
```

[153]:
```
'This is a simple function.'
```

[154]:
```
help(foo)
```

```
Help on function foo in module __main__:

foo()
    This is a simple function.
```

[156]:
```python
@func_logger
def bar():
    "This is another simple function"
    print("Hello!")
```

[157]:
```python
bar.__name__
```

[157]:
```
'wrapper'
```

[158]:
```python
bar.__doc__
```

[159]:
```python
help(bar)
```

```
Help on function wrapper in module __main__:

wrapper(*args, **kwargs)
```

[165]:
```python
def func_logger(func):
    def wrapper(*args, **kwargs):
        print(f"Executing the function: {func.__name__}")
        res = func(*args, **kwargs)
        print("Done!")
        return res

    wrapper.__name__ = func.__name__
    wrapper.__doc__ = func.__doc__

    return wrapper
```

[166]:
```python
@func_logger
def baz(name):
    """Another simple function with parameter"""
    print("Hello", name)
```

[167]:
```python
baz.__name__
```

[167]:
```
'baz'
```

[168]:
```python
baz.__doc__
```

[168]:
```
'Another simple function with parameter'
```

[169]:
```
help(baz)
```

```
Help on function baz in module __main__:

baz(*args, **kwargs)
    Another simple function with parameter
```

[171]:
```
from functools import wraps
```

[172]:
```
def func_logger(func):
    @wraps(func)
    def wrapper(*args, **kwargs):
        print(f"Executing the function: {func.__name__}")
        res = func(*args, **kwargs)
        print("Done!")
        return res

    return wrapper
```

[180]:
```
@func_logger
def greeter(name):
    """Example"""
    print("hello", name)
```

[181]:
```
greeter.__name__
```

[181]:
```
'greeter'
```

[182]:
```
greeter.__doc__
```

[182]:
```
'Example'
```

[183]:
```
help(greeter)
```

```
Help on function greeter in module __main__:

greeter(name)
    Example
```

[ ]:
```

```