

属性和方法

笔记本： Python提高-2

创建时间： 2018/5/8 17:00

更新时间： 2018/5/8 17:35

作者： ly

类属性、实例属性

```
class Province(object):
    # 类属性
    country = '中国'

    def __init__(self, name):
        # 实例属性
        self.name = name

# 创建一个实例对象
obj = Province('山东省')

# 直接访问实例属性
print(obj.name)

# 直接访问类属性
Province.country
```

它们在定义和使用中有所区别，而最本质的区别是**内存中保存的位置**不同

- 实例属性属于对象 实例属性需要通过对象来访问 类属性在内存中只保存一份
- 类属性属于类 类属性通过类访问 实例属性在每个对象中都要保存一份

实例方法、静态方法和类方法

方法包括：实例方法、静态方法和类方法，三种方法在内存中都归属于**类**，区别在于**调用方式**不同。

- 实例方法：由对象调用；至少一个self参数；执行实例方法时，自动将调用该方法的对象赋值给self；
- 类方法：由类调用；至少一个cls参数；执行类方法时，自动将调用该方法的类赋值给cls；
- 静态方法：由类调用；无默认参数；

```
class Foo(object):
    def __init__(self, name):
        self.name = name

    def ord_func(self):
        """ 定义实例方法，至少有一个self参数 """
        # print(self.name)
        print('实例方法')
```

```

@classmethod
def class_func(cls):
    """ 定义类方法，至少有一个cls参数 """
    print('类方法')

@staticmethod
def static_func():
    """ 定义静态方法，无默认参数 """
    print('静态方法')

f = Foo("中国")
# 调用实例方法
f.ord_func()

# 调用类方法
Foo.class_func()

# 调用静态方法
Foo.static_func()

```

对比

- 相同点：对于所有的方法而言，均属于类，所以在内存中也只保存一份
- 不同点：方法调用者不同、调用方法时自动传入的参数不同。

以内存的方式去理解：

对象：在内存地址中实际存在的具体的事物

类对象：存放类相关的数据的空间

实例对象：存放实例相关的数据的空间

类属性：所有实例对象都可以用到的属性
定义在类的内部，方法的外部。类属性为全部类所共有

实例属性：每个实例化私有的属性

类方法：可以修改类属性的方法-----类对象所拥有的方法
在类中直接写的方法，默认都是实例方法，应该都用实例对象去调用
cls---类对象的传递
可以用类对象去调用，也可以用实例化对象去调用

实例方法：可以修改实例属性的方法

静态方法：谁都能修改
方法内部即不用类对象，也不用实例对象

类对象：存放类相关的数据的空间

实例对象：存放实例相关的数据的空间

类属性：所有的实例对象都可以用到的属性

实例属性：每个实例私有的属性

类方法：可以修改类属性的方法

实例方法：可以修改实例属性的方法

静态方法：谁都给改

