property属性

笔记本：	Python提高-2		
创建时间：	2018/5/8 10:56	**更新时间：**	2018/5/8 20:08
作者：	ly		

property语法

作用：定义的时候是 方法，调用的时候通过属性的方式去调用
使用场景：使用私有属性更加快捷，不用调用get和set的方法来使用私有属性

语法格式：

```
class Foo01(object):
    def test(self):
        return 100


# 未使用property时
f = Foo01()
val = f.test()
print(val)


###############


class Foo02(object):
    @property               # 使用property时
    def test(self):
        return 100


f = Foo02()
print(f.test)         # 直接调用f的属性test

# 输出结果都为 100
```

property属性的定义和调用：

- 定义时，在实例化方法的基础上添加@property装饰器；并且仅有一个self参数
- 调用时，无需括号 (当属性来使用)

```
方法：foo01.test()
property属性：foo02.test
```

property属性的完整使用

三种@property装饰器

定义一个类
    第一种装饰器:@property # 获取方法的返回值
    第二种装饰器:@property方法名.setter # 设置方法
    第三种装饰器:@property方法名.deleter # 调用删除方法

```
class Good(object):
    def __init__(self):
        self.org_price = 1000
        self.discount = 0.7

    @property
    def price(self):
        val = self.org_price * self.discount
        return val     # 返回值

    @price.setter   # 此处price必须和上面的函数名相同
    def price(self, new_val):
        self.org_price= new_val

    @price.deleter   # 这个修饰器叫删除   并不会直接删除，作用取决于里面的方法
    def price(self):
        del self.discount


g = Good()

print(g.price)

g.price = 2000
print(g.price)

del g.price
print(g.price)
```

经典类中的属性只有一种访问方式，其对应被 @property 修饰的方法
新式类中的属性有三种访问方式，并分别对应了三个被@property、@方法名.setter、@方法名.deleter修饰的方法

property属性的第二种使用方式:类属性方式

```
class Money(object):
    def __init__(self):
        self.__money = 0

    def getMoney(self):
        return self.__money

    def setMoney(self, value):
        if value >= 0:
          self.__money = value
          print("修改成功")
        else:
            print("error:请存放正确的数值")

    def print_price(self):
        print("调用删除")

    money = property(getMoney, setMoney, print_price)
```

```
                (@property  @getMoney.setter  @getMoney.deleter)

# 正常没有property时，正常调用：
# m = Money()
# val = m.getMoney()
# print(val)
#
# m.setMoney(100)

m = Money()

print(m.money)

m.money = 100


del m.money
```

总结

定义property属性共有两种方式，分别是【装饰器】和【类属性】，而【装饰器】方式针对经典类和新式类又有所不同。
通过使用property属性，能够**简化调用者在获取数据的流程**