PYTHON
编程基础

@property装饰器

- 类中的属性可以直接访问和赋值，这为类的使用者提供了方便，但也带来了问题：类的使用者可能会给一个属性赋上超出有效范围的值。

- 为了解决这个问题，Python提供了@property装饰器，可以将类中属性的访问和赋值操作自动转为方法调用，这样可以在方法中对属性值的取值范围做一些条件限定。

- 直接使用@property就可以定义一个用于获取属性值的方法（即getter）。

- 如果要定义一个设置属性值的方法（setter），则需要使用名字“@属性名.setter”的装饰器。

- 如果一个属性只有用于获取属性值的getter方法，而没有用于设置属性值的setter方法，则该属性是一个只读属性，只允许读取该属性的值、而不能设置该属性的值。

- 例：通过@property装饰器使得学生成绩的取值范围必须在0~100之间。

```
1       import datetime
2       class Student: #定义Student类
3           @property
4           def score(self): #用@property装饰器定义一个用于获取score值的方法
5               return self._score
```

注意：在类的setter和getter方法中使用self访问属性时，需要在属性名前加上下划线，否则系统会因不断递归调用而报错。

```
6       @score.setter
7       def score(self, score): #用score.setter定义一个用于设置score值的方法
8           if score<0 or score>100: #不符合0~100的限定条件
9               print('成绩必须在0~100之间！')
10          else:
11              self._score=score
12      @property
13      def age(self): #用@property装饰器定义一个用于获取age值的方法
14          return datetime.datetime.now().year-self.birthyear
```

```
15      if __name__=='__main__':
16          stu=Student() #创建Student类对象stu
17          stu.score=80 #将stu对象的score属性赋值为80
18          stu.birthyear=2000 #将stu对象的birthyear属性赋值为2000
19          print('年龄：%d,成绩：%d'%(stu.age,stu.score))
```

```
年龄：18,成绩：80
```

- 20 #stu.age=19 #取消前面的注释符则会报错
- 21 stu.score=105 #将stu对象的score属性赋值为105
- 22 print('年龄：%d,成绩：%d'%(stu.age,stu.score))

成绩必须在0~100之间！

年龄：18,成绩：80