Python进阶之“属性（property）”详解

2015/01/25 • 基础知识 • property

Python中有一个被称为属性函数(property)的小概念，它可以做一些有用的事情。在这篇文章中，我们将看到如何能做以下几点：

- 将类方法转换为只读属性
- 重新实现一个属性的setter和getter方法

在本文中，您将学习如何以几种不同的方式来使用内置的属性函数。希望读到文章的末尾时，你能看到它是多么有用。

开始

使用属性函数的最简单的方法之一是将它作为一个方法的装饰器来使用。这可以让你将一个类方法转变成一个类属性。当我需要做某些值的合并时，我发现这很有用。其他想要获取它作为方法使用的人，发现在写转换函数时它很有用。让我们来看一个简单的例子：

Python

```python
##################################################################################
class Person(object):
    """"""

    #----------------------------------------------------------------------
    def __init__(self, first_name, last_name):
        """Constructor"""
        self.first_name = first_name
        self.last_name = last_name

    #----------------------------------------------------------------------
    @property
```

```
    def full_name(self):
        """
        Return the full name
        """
        return "%s %s" % (self.first_name, self.last_name)
```

在上面的代码中，我们创建了两个类属性：self.first_name和self.last_name。接下来，我们创建了一个full_name方法，它有一个@property装饰器。这使我们能够在Python解释器会话中有如下的交互：

```python
>>> person = Person("Mike", "Driscoll")
>>> person.full_name
'Mike Driscoll'
>>> person.first_name
'Mike'
>>> person.full_name = "Jackalope"
Traceback (most recent call last):
  File "<string>", line 1, in <fragment>
AttributeError: can't set attribute
```

正如你所看到的，因为我们将方法变成了属性，我们可以使用正常的点符号访问它。但是，如果我们试图将该属性设为其他值，我们会引发一个AttributeError错误。改变full_name属性的唯一方法是间接这样做：

```python
>>> person.first_name = "Dan"
>>> person.full_name
'Dan Driscoll'
```

这是一种限制，因此让我们来看看另一个例子，其中我们可以创建一个允许设置的属性。

使用Python property取代setter和getter方法

让我们假设我们有一些遗留代码，它们是由一些对Python理解得不够好的人写的。如果你像我一样，你之前已经看到过这类的代码：

```python
from decimal import Decimal

##################################################################################
class Fees(object):
    """"""

    #----------------------------------------------------------------------
    def __init__(self):
        """Constructor"""
        self._fee = None

    #----------------------------------------------------------------------
    def get_fee(self):
        """
        Return the current fee
        """
        return self._fee

    #----------------------------------------------------------------------
    def set_fee(self, value):
        """
        Set the fee
        """
        if isinstance(value, str):
```

```
            self._fee = Decimal(value)
        elif isinstance(value, Decimal):
            self._fee = value
```

要使用这个类，我们必须要使用定义的getter和setter方法：

```Python
>>> f = Fees()
>>> f.set_fee("1")
>>> f.get_fee()
Decimal('1')
```

如果你想添加可以使用正常点符号访问的属性，而不破坏所有依赖于这段代码的应用程序，你可以通过添加一个属性函数非常简单地改变它：

```Python
from decimal import Decimal

##########################################################################
class Fees(object):
    """"""

    #----------------------------------------------------------------------
    def __init__(self):
        """Constructor"""
        self._fee = None

    #----------------------------------------------------------------------
    def get_fee(self):
        """
        Return the current fee
        """
        return self._fee

    #----------------------------------------------------------------------
    def set_fee(self, value):
        """
        Set the fee
        """
        if isinstance(value, str):
            self._fee = Decimal(value)
        elif isinstance(value, Decimal):
            self._fee = value

    fee = property(get_fee, set_fee)
```

我们在这段代码的末尾添加了一行。现在我们可以这样做：

```Python
>>> f = Fees()
>>> f.set_fee("1")
>>> f.fee
Decimal('1')
>>> f.fee = "2"
>>> f.get_fee()
Decimal('2')
```

正如你所看到的，当我们以这种方式使用属性函数时，它允许fee属性设置并获取值本身而不破坏原有代码。让我们使用属性装饰器来重写这段代码，看看我们是否能得到一个允许设置的属性值。

```Python
from decimal import Decimal

```

```
3  ################################################################################
4  class Fees(object):
5      """"""
6
7      #----------------------------------------------------------------------
8      def __init__(self):
9          """Constructor"""
10         self._fee = None
11
12     #----------------------------------------------------------------------
13     @property
14     def fee(self):
15         """
16         The fee property - the getter
17         """
18         return self._fee
19
20     #----------------------------------------------------------------------
21     @fee.setter
22     def fee(self, value):
23         """
24         The setter of the fee property
25         """
26         if isinstance(value, str):
27             self._fee = Decimal(value)
28         elif isinstance(value, Decimal):
29             self._fee = value
30
31 #----------------------------------------------------------------------
32 if __name__ == "__main__":
33     f = Fees()
```

上面的代码演示了如何为fee属性创建一个setter方法。你可以用一个名为@fee.setter的装饰器装饰第二个方法名也为fee的方法来实现这个。当你如下所做时，setter被调用：

Python

```
1 >>> f = Fees()
2 >>> f.fee = "1"
```

如果你看属性函数的说明，它有fget, fset, fdel和doc几个参数。如果你想对属性使用del命令，你可以使用@fee.deleter创建另一个装饰器来装饰相同名字的函数从而实现删除的同样效果。

结束语

现在你知道如何在你的类中使用Python的属性函数。希望你能找到更有用的方式，在你的代码中使用它们。

- Python中的getter和setter方法
- 官方Python文档中对property的介绍
- StackOverflow中对给Python属性函数增加文档字符串的一个讨论

关于作者：Janzou

做一直倦懒的小猫咪，享受午后阳光（新浪微博：@Jan_zou） 个人主页 • 我的文章 • 11

相关文章

- Python 中的 property 属性
- 详解Python中的下划线
- Python中使用内层函数的好处
- 基于Python的测试驱动开发实战
- 理解 Python 字节码
- Python高级特性（1）：Iterators、Generators和itertools
- Python趣文：Import Girlfriend
- 如何在Python中使用static、class、abstract方法（权威指南）
- Python 中 eval 带来的潜在风险
- 像老大一样优化Python

可能感兴趣的话题

- 腾讯2015春招移动客户端开发练习卷(android)
- 2016京东在线笔试题(java) • 6
- Python 匹配指定中文词
- 一枚小小的实习生要不要跳槽呢？ • 4
- 有组队一起去看这周五上映的《谁的青春不迷茫》的吗 • 23
- 2017蘑菇街校招笔试题(Servlet问题) • 7
- 奇虎360 一道笔试题 分享一下 • 32
- 组队看《不二情书》 • 4
- 你最喜欢的粤语歌，求推荐 • 66
- angularJs 入门案例—Nav选项卡(二)

« 另一个Lambda表达式教程
数据科学的完整学习路径（Python版） »

Python小组话题 **我有新话题**

Python 匹配指定中文词
啊呀啊哈哈哈 发起

零基础自学Python感觉很难，不像大...
keepcalm 发起 • 9 回复

怎么写pwm ？
累积未来 发起

如何通过写python代码实现重启DNS...
蒲荷 发起

用Python写一个简单的ide需要学什么？
梦VHP 发起 • 1 回复

如何在Mac上安装jpeglib?不然PI...
workabee 发起

- 本月热门Python文章
- 年度热门
- 热门标签

0 Python爬虫：一些常用的爬虫技巧总结

1 使用python抓取婚恋网用户数据并用决...

2 Python趣味编程：定时给Ta讲笑话

3 如何用Python写一个贪吃蛇AI

4 Python爬虫：抓取One网页上的每...

5 剖析勇士如何成为新赛季夺冠热门：基于Spark...

6 一个人人网python爬虫

7 python测试开发面试题

8 Python 编码风格指南

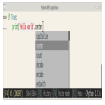
9 为什么会有 Python 3 的存在？

Python工具资源 **更多资源 »**

SciPy库：Python的科学计算工具集
科学计算和数据分析

NumPy：Python科学计算的基石
Python, 科学计算和数据分析

webssh：基于tornado的web linux终端
Python



Python Prompt Toolkit：构建强大交互式命令行的 Pyt...
Python, 开发库

Pythonpy：在命令行中直接执行任何Python指令
Python, 命令行工具 • 1

最新评论

- 
Re: 一个 11 行 Python 代码实...
刚刚算错，还有个问题：每一层的权值和输入样本的个数有关系
- 
Re: 一个 11 行 Python 代码实...
为什么最后预测output和y值还是差很多呢？
- 
Re: Python之美[从菜鸟到高手]-...
最后一个例子，threads会变得很长
- 
Re: 利用 Python 练习数据挖掘
图看起来不一样，实际上加上label之后都是同一个图，但是label是0-77个数，而不是单词，这一...
- 
Re: 利用 Python 练习数据挖掘
我发现比较扯的问题是，networkx每次画的图都不一样，没有label。不知道有没有同样的问题，可...
- 
Re: Python爬虫实战（1）：爬取糗事...
__author__=\'RingoYan\'# -*- coding:utf-8 -*-import ...
- 
Re: 使用python抓取婚恋网用户数据并...
请教下两次请求为什么需要header，我在开发者工具找不到你这么详细的header，不可能是要自己填...
- 
Re: 用Python写一个简单的微博爬虫
楼主问下你用的是什么编辑器，论坛上贴的代码都是这样的，不是sublime吧？

关于 Python 频道

Python频道分享 Python 开发技术、相关的行业动态。

快速链接
问题反馈与求助 »
Python工具资源 »
Python技术话题 »

关注我们

新浪微博：@Python开发者
RSS：订阅地址
推荐微信号

合作联系
Email：bd@Jobbole.com
QQ： 2302462408 （加好友请注明来意）

更多频道

小组 - 好的话题、有启发的回复、值得信赖的圈子
头条 - 分享和发现有价值的内容与观点
相亲 - 为IT单身男女服务的征婚传播平台
资源 - 优秀的工具资源导航
翻译 - 翻译传播优秀的外文文章
文章 - 国内外的精选文章
设计 - UI,网页，交互和用户体验
iOS - 专注iOS技术分享
安卓 - 专注Android技术分享
前端 - JavaScript, HTML5, CSS
Java - 专注Java技术分享
Python - 专注Python技术分享

 首页 博客 资源 小组 相亲 反馈