python 基础 09：面向对象的进一步拓展

我们熟悉了对象和类的基本概念。我们将进一步拓展，以便能实际运用对象和类。

1、调用类的其它信息

上一讲中提到，在定义方法时，必须有 self 这一参数。这个参数表示某个对象。对象拥有类的所有性质，那么我们可以通过 self，调用类属性。

```
class Human(object):
    laugh = 'hahahaha'
    def show_laugh(self):
        print self.laugh
    def laugh_100th(self):
        for i in range(100):
        self.show_laugh()

li_lei = Human()
li_lei.laugh_100th()
```

这里有一个类属性 laugh。在方法 show_laugh()中，通过 self.laugh，调用了该属性的值。

还可以用相同的方式调用其它方法。方法 show_laugh()，在方法

laugh_100th 中()被调用。

通过对象可以修改类属性值。但这是危险的。类属性被所有同一类及其子类的对象共享。类属性值的改变会影响所有的对象。

2、__init__()方法

__init__()是一个特殊方法(special method)。Python 有一些特殊方法。Python 会特殊的对待它们。特殊方法的特点是名字前后有两个下划线。

如果你在类中定义了__init__()这个方法，创建对象时，Python 会自动调用这个方法。这个过程也叫初始化。

```
class happyBird(Bird):
   def __init__(self,more_words):
        print 'We are happy birds.',more_words


summer = happyBird('Happy,Happy!')
```

这里继承了 Bird 类，它的定义见上一讲。

屏幕上打印：

We are happy birds.Happy,Happy!

我们看到，尽管我们只是创建了 summer 对象，但__init__()方法被自动调用了。最后一行的语句(summer = happyBird...)先创建了对象，然后执行：

summer.__init__(more_words)

'Happy,Happy!' 被传递给了__init__()的参数 more_words

2.1 对象的性质

我们讲到了许多属性，但这些属性是类的属性。所有属于该类的对象会共享这些属性。比如说，鸟都有羽毛，鸡都不会飞。

在一些情况下，我们定义对象的性质，用于记录该对象的特别信息。比如说，人这个类。性别是某个人的一个性质，不是所有的人类都是男，或者都是女。这个性质的值随着对象的不同而不同。李雷是人类的一个对象，性别是男；韩美美也是人类的一个对象，性别是女。

当定义类的方法时，必须要传递一个 self 的参数。这个参数指代的就是类的一个对象。我们可以通过操纵 self，来修改某个对象的性质。比如用类来新建一个对象，即下面例子中的 li_lei, 那么 li_lei 就被 self 表示。我们通过赋值给 self.attribute，给 li_lei 这一对象增加一些性质，比如说性别的男女。self 会传递给各个方法。在方法内部，可以通过引用 self.attribute，查询或修改对象的性质。

这样，在类属性的之外，又给每个对象增添了各自特色的性质，从而能描述多样的世界。

```
class Human(object):
    def __init__(self, input_gender):
        self.gender = input_gender

    def printGender(self):
        print self.gender

li_lei = Human('male')  # 这里，'male'作为参数传递给__init__()方法的
```

input_gender 变量。

print li_lei.gender

li_lei.printGender()

在初始化中 ,将参数input_gender ,赋值给对象的性质 ,即self.gender。

li_lei 拥有了对象性质 gender。gender 不是一个类属性。Python 在建立了 li_lei 这一对象之后，使用 li_lei.gender 这一对象性质，专门储存属于对象 li_lei 的特有信息。

对象的性质也可以被其它方法调用 ,调用方法与类属性的调用相似 ,正如在 printGender()方法中的调用。

3、总结

通过 self 调用类属性

__init__(): 在建立对象时自动执行

类属性和对象的性质的区别