

Special Object Methods

在 Inheritance and Composition 中我们已经知道下面这一事实：

Python 中几乎所有数据类型的基类都是 Object (对象)

我们可以通过 `dir(object)` 来获取一个对象的所有属性列表，下面将讲讲其中几种。

1) __str__() 返回人类可读的对象字符串表示

`one_third = 1/3`

`one_third.__str__()`

`>>> '0.3333333333'`

两者都是可行的

`float.__str__(one_third)`

但后者更规范

这一方法在调用 `print()`, `str()`, 打印字符串等多种函数和方法时都会用到。

值得注意的是在自定义的类中我们可以修改 `__str__()` 的定义

对这个类的实例调用上述函数都会采取新定义的 `__str__()` 方法

2) __repr__() 返回字符串计算具有相同值的对象表示

`from fractions import Fraction`

`one_half = Fraction(1, 2)`

我们可以用 `eval()` 函数对字符串表达式求值。

`Fraction.__repr__(one_half) >>> 'Fraction(1, 2)'`

相当于返回一段代码，运行后的结果为输入的对象。

调用 `repr()` 函数时，实际上就是调用了 `__repr__()` 方法。两者返回值相同。

交互式会话实际上也是调用 `__repr__()` 的过程。

同样，可以修改 `__repr__()` 的定义以便在使用交互式会话调试时取得更好效果。

3) getattr() 调用方法为 `getattr(object, name[, default])`

在 `object` 中查找名为 `name` 的属性。

需要将变量名以字符串的形式

如果有，则返回 `object.name`，如果没有定义，则返回 `default`。

传入，否则将在全局快中查找名

如果没有给出 `default`，则报错，错误类型 `AttributeError`。

为 `name` 的变量（然后报错）

--getattribute__() 不管是直接用点表达式还是调用 `getattr()` 函数

都使用了 `__getattribute__()` 方法。

该方法的返回值与 `getattr()` 函数相似。

一个检查属性是否存在内置函数为 `hasattr(object.name)`

如果有，则返回 `True`，否则返回 `False`。

4) __ setattr__(): 给属性赋值时会调用的方法.

对于上述几种方法在Python中对所有变量类型都可以进行调用.
它们都继承自object这一基类.