Python 对象

要搞懂元类，我们还是先从对象说起。

对象（Object）

Python 一切皆对象，一个数字是对象，一个字符串是对象，一个列表是对象，一个字典是对象，例如：

```
>>> i = 10
>>> s = "abc"
>>> nums = [1,2,3]
>>> dicts = {"name":"zhang"}
```

等号右边是对象，左边是给这些对象取的名字，任何对象都有3个关键属性：**标识、值、类型。**

标识

标识就和人的身份证ID一样，每个对象有唯一ID标识，在整个生命周期中都不会变，你可以认为标识是这个对象在计算机内存中的地址。通过函数 `id()` 可以查看对象的ID标识。

```
>>> id(i)
40592592
>>> id(s)
44980584
```

对象值

对象的第二个属性是值，值很好理解，比如 i 的值是 10，s 的值是 abc，nums 的值就是 1,2,3。

类型

对象还有一个很重要的属性就是类型，任何对象都有属于自己的类型，**对象就是由它的类型构造出来的。**

比如上面 i 的类型是 int 类型，s 类型是字符串类型，nums 的类型是列表类型，dicts 的类型是字典类型，它们都是由对应的类型构建出来的。

通过 `type()` 可以查看对象的类型。

```
>>> type(i)
<class 'int'>
>>> type(s)
<class 'str'>
>>> type(nums)
<class 'list'>
>>> type(dicts)
<class 'dict'>
```

对象的类型也和ID标识一样不会改变。唯一可能变的就是值。

类与（实例）对象

除了系统已经定义好了的整数类型，字符串类型，列表等类型之外，我们还可以创建自己的类型，用关键字 `class` 来定义。例如：

```
>>> class Person:
        # 这里的 self 指某个实例对象自己
...     def __init__(self, name):
            # name 是实例的属性
...         self.name = name
        # live 是类的属性
        live = True
```

这里的 Person 就是自定义类，类是一个抽象的模版，既不是指张三也不是李四等具体的人，现在我们可以通过调用这个类来构造（实例化）出一个具体的，实在的，有名字的**对象**出来，这个对象称之为实例对象（Instance）。

```
>>> p1 = Person("zhangsan")
>>> p1.name
'zhangsan'
>>>
>>> p2 = Person("lisi")
>>> p2.name
'lisi'
```

这里的 p1、p2 就是实例化之后的（实例，instance）对象，这两个对象的类型都是 Person 类，类与（实例）对象的关系就像一个车辆模具与一辆被造出来的真实车的关系一样。

```
>>> p1
<__main__.Person object at 0x0195AA30>
>>> type(p1)
<class '__main__.Person'> # 这里的__main__是模块名称
```

类也是对象（又叫类对象）

刚刚我们说了一切都是对象，实例（真实的车）是对象，类（模具车）当然也是对象，因为它也是实实在在存在的东西。

当 Python 解释器执行到关键字 `class` 这个指令的时候，在内部就会创建一个名为 “Person” 的类，这个类也是个对象，我们称之为类对象（注意区别实例对象），它一样有ID标识、有类型、有值。例如：

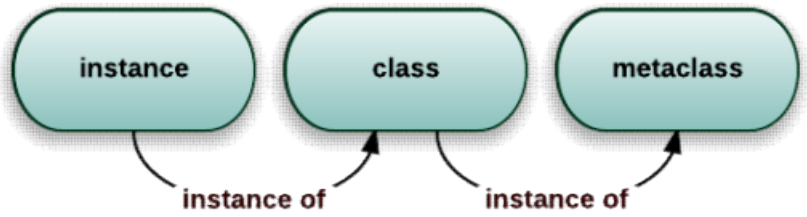
```
>>> id(Person)
26564024
>>> type(Person)
<class 'type'>
>>> Person
<class '__main__.Person'>
```

我们注意到这个 Person 这个类对象的类型叫 “type”，也就是说 Person 类是由 type 创建出来的，现在你要记住，p1，p2 是实例对象，而 Person 是类对象。另外，这个 type 是什么鬼?

我们来回顾一下，实例对象 p1 的类型是类对象 Person，Person 的类型 type

```
>>> nums = [1,2,3]
>>> type(nums)
<class 'list'>
>>> type(list)
<class 'type'>
```

nums 的类型是 list，list 的类型也是 type，字典类（dict）的类型也是 type，所有类的类型都是 type，也就是说所有的类都是由type 创建的。**这个 type 就是元类（metaclass），元类是用于创建类的类，道生一，一生二，三生万物，元类就是 Python 中的造物主。**（元类自己也是对象）



现在我们都知道类（对象）可以使用 `class` 关键字创建，我们还知道类（对象）的类型是 type，既然知道了它的类型是 type，那么肯定可以通过 type（元类）来创建。

用元类创建类

前面讲到过，`type` 有一个作用是用于检查对象的类型，其实它还有另外一个作用就是作为元类动态地创建类（对象）。

```
>>> Person = type("Person", (), {"live":True})
>>> Person
<class '__main__.Person'>
```

Person 就是一个类，它等价于：

```
>>> class Person:
...     live = True
...
>>> Person
<class '__main__.Person'>
```

用元类 `type` 创建类的语法是：

`type(类名,父类元组(可以为空)，属性字典)`

那么元类到底有什么用处呢？我们真的需要元类吗？请关注下回讲解，（留给大家多些时间消化，O(∩_∩)O）

小结

Python中一切皆为对象，类是对象，元类也是对象，元类是用于创建类的类。

By 刘志军 Python之禅