EDU

CSDN学院　IT实战派

第一章 Python基础

第11节 Python面向对象

讲师：张 涛

本节目标

1. 面向对象概述
2. 类和对象
3. 构造函数
4. 属性(类变量)和方法
5. 继承与重写

1. 面向对象概述

Python从设计之初就已经是一门面向对象的语言，正因为如此，在Python中创建一个类和对象是很容易的.

面向对象技术简介:

- 类(Class): 用来描述具有相同的属性和方法的对象的集合。它定义了该集合中每个对象所共有的属性和方法。对象是类的实例。
- 类变量： 类变量在整个实例化的对象中是公用的。类变量定义在类中且在函数体之外。类变量通常不作为实例变量使用。
- 数据成员： 类变量或者实例变量用于处理类及其实例对象的相关的数据。
- 方法重写： 如果从父类继承的方法不能满足子类的需求，可以对其进行改写，这个过程叫方法的覆盖（override），也称为方法的重写。
- 实例变量： 定义在方法中的变量，只作用于当前实例的类。
- 继承： 即一个派生类（derived class）继承基类（base class）的字段和方法。继承也允许把一个派生类的对象作为一个基类对象对待。例如，有这样一个设计：一个Dog类型的对象派生自Animal类，这是模拟“是一个（is-a）”关系（例图，Dog是一个Animal）。
- 实例化： 创建一个类的实例，类的具体对象。
- 方法： 类中定义的函数。
- 对象： 通过类定义的数据结构实例。对象包括两个数据成员（类变量和实例变量）和方法。

2. 类和对象

- 类的定义格式：

```
class ClassName:
    <statement-1>
    ...
    ...
    <statement-N>
```

- 类实例化成对象:

```
对象 = ClassName（[参数列表]）
```

- 对象的使用:

```
对象.属性名
对象.方法( )
```

3. 构造函数

- 构造函数：__init__()

  类定义了 __init__() 方法的话，类的实例化操作会自动调用 __init__() 方法。

  __init__() 方法可以有参数，参数通过 __init__() 传递到类的实例化操作上。

- Self 说明：

  self 代表的是类的实例，代表当前对象的地址，而 self.class 则指向类。

  self 不是 python 关键字，我们可以换成其他变量名

4. 属性(类变量)和方法

- 使用 def 关键字来定义一个方法.与一般函数定义不同，类方法必须包含参数 self, 且为第一个参数，self 代表的是类的实例

  self 的名字并不是规定死的，也可以使用 this，但最好按照约定是self

- 类的私有属性：private_attrs

  两个下划线开头，声明该属性为私有，不能在类地外部被使用或直接访问。在类内部的方法中使用self调用

- 类的私有方法：private_method

  两个下划线开头，声明该方法为私有方法，只能在类的内部调用（self），不能在类地外部调用

5. 继承与重写

Python 同样支持类的继承，如果一种语言不支持继承，类就没有什么意义。派生类的定义如下所示:

```
class 子类（父类）:
    <statement-1>
    ……
    <statement-N>
```

方法重写（重载）：

如果你的父类方法的功能不能满足你的需求，你可以在子类重写你父类的方法

多继承：

需要注意圆括号中父类的顺序，若是父类中有相同的方法名，而在子类使用时未指定，python从左至右搜索 即方法在子类中未找到时，从左到右查找父类中是否包含方法。

小结

1. 面向对象概述
2. 类和对象
3. 构造函数
4. 属性(类变量)和方法
5. 继承与重写

布置作业

- 跟着老师自己动手练一练
- 使用Python的面向对象去定义一些生活中的类。


EDU
CSDN学院 IT实战派