# Contents

# Part I

# 类和面向对象编程（OOP）

# Chapter 1

# OOP：宏伟蓝图

## 1.1 概览 OOP

### 1.1.1 属性继承搜索

在 Python 对象模型中，类和通过类产生的实例是两种不同的对象模型：

**类** 类是实例工厂。类的属性提供了行为（数据以及函数），所有从类产生的实例都继承了该类的属性。

**实例** 代表程序领域中具体的元素。实例的属性记录了每个实例自己的数据。

### 1.1.2 方法调用

方法可以通过实例 bob.giveRaise() 或类 Employee.giveRaise(bob) 来调用，这两种形式都可以在我们的脚本中发挥作用。这些调用还说明了 OOP 中的两个关键思想：运行 bob.giveRaise() 方法调用，Python：

1. 通过继承搜索从 bob 中查找 GiveRaise
2. 将 bob 传递给位于特殊 self 参数中的 GiveRaise 函数

当你调用 Employee.giveRaise(bob) 时，你只需自己执行这两个步骤。从技术上讲，此描述是默认情况（Python 还有其他方法类型），但它适用于用该语言编写的绝大多数 OOP 代码。

### 1.1.3 编写树类

从操作的角度来看，当 def 出现在类的内部时，通常称为方法。而且会自动接收第一个特殊参数（按照惯例称为 self），这个参数提供了被处理的实例的引用。所有你自己向方法中传入的参数都被赋给了 self 后面的参数。

# Chapter 2

# 类代码编写基础

类有三个主要的不同之处。从最底层来看，类几乎就是命名空间，很像 研究过的模块。但是，和模块不同的是，类也支持多个对象的产生、命名空间继承以及运算符重载。

## 2.1 类产生多个实例对象

要了解多个对象的概念是如何工作的，得先了解 Python 的 OOP 模型中的两种对象：类对象和实例对象。类对象提供默认行为，是实例对象的工厂。实例对象是程序处理的实际对象：各自都有独立的命名空间，但是继承（可自动存取）创建该实例的类中的变量名。类对象来自于语句，而实例来自于调用。每次调用一个类，就会得到这个类的新的实例。

### 2.1.1 类对象提供默认行为

以下是 Python 类主要特性的要点：

**class 语句创建类对象并将其赋值给变量名** 就像函数 def 语句，Python class语句也是可执行语句。执行时，会产生新的类对象，并将其赋值给 class 头部的变量名。此外，就像def 应用，class 语句一般是在其所在文件导入时执行的。

**class 语句内的赋值语句会创建类的属性** 就像模块文件一样，class 语句内的顶层的赋值语句（不是在 def 之内）会产生类对象中的属性。从技术角度来讲，class 语句的作用域会变成类对象的属性的命名空间，就像模块的全局作用域一样。执行 class 语句后，类的属性可由变量名点号运算获取 object.name。

**类属性提供对象的状态和行为** 类对象的属性记录状态信息和行为，可由这个类所创建的所有实例共享。

### 2.1.2 实例对象是具体的元素

以下是类的实例内含的重点概要：

**像函数那样调用类对象会创建新的实例对象** 每次类调用时，都会建立并返回新的实例对象。实例代表了程序领域中的具体元素。

**每个实例对象继承类的属性并获得了自己的命名空间** 由类所创建的实例对象是新命名空间。一开始是空的，但是会继承创建该实例的类对象内的属性。

**在方法内对 self 属性做赋值运算会产生每个实例自己的属性** 在类方法函数内，第一个参数（按惯例称为 self）会引用正处理的实例对象。对 self 的属性做赋值运算，会创建或修改实例内的数据，而不是类的数据。

## 2.2 类通过继承进行定制

除了作为工厂来生成多个实例对象之外，类也可引入新组件（子类）来进行修改，而不对现有组件进行原地的修改。

在 Python 中，实例从类中继承，而类继承于超类。以下是属性继承机制的核心观点：

- 超类列在了类开头的括号中
- 类从其超类中继承属性
- 实例会继承所有可访问类的属性
- 每个 object.attribute 都会开启新的独立搜索
- 逻辑的修改是通过创建子类，而不是修改超类

## 2.3 类可以截获 Python 运算符

运算符重载就是让用类写成的对象，可截获并响应用在内置类型上的运算：加法、切片、打印和点号运算等。这只是自动分发机制：表达式和其他内置运算流程要经过类的实现来控制。这里也和模块没有什么相似之处：模块可以实现函数调用，而不是表达式的行为。

- 以双下划线命名的方法是特殊钩子。Python 运算符重载的实现是提供特殊命名的方法来拦截运算。Python 语言替每种运算和特殊命名的方法之间，定义了固定不变的映射关系。
- 当实例出现在内置运算时，这类方法会自动调用。
- 类可覆盖多数内置类型运算。
- 运算符覆盖方法没有默认值，而且也不需要。
- 新式类有一些默认的运算符重载方法，但是不属于常见运算。
- 运算符可让类与 Python 的对象模型相集成。

# Chapter 3

# 运算符重载