<<Python源码剖析>>

图书基本信息

书名：<<Python源码剖析>>

13位ISBN编号：9787121068744

10位ISBN编号：7121068745

出版时间：2000-1

出版时间：电子工业出版社

作者：陈儒

页数：480

<<Python源码剖析>>

内容概要

作为主流的动态语言，Python不仅简单易学、移植性好，而且拥有强大丰富的库的支持。
此外，Python强大的可扩展性，让开发人员既可以非常容易地利用C/C++编写Python的扩展模块，还能将Python嵌入到C/C++程序中，为自己的系统添加动态扩展和动态编程的能力。

为了更好地利用Python语言，无论是使用Python语言本身，还是将Python与C/C++交互使用，深刻理解Python的运行原理都是非常重要的。
本书以CPython为研究对象，在C代码一级，深入细致地剖析了Python的实现。
书中不仅包括了对大量Python内置对象的剖析，更将大量的篇幅用于对Python虚拟机及Python高级特性的剖析。
通过此书，读者能够透彻地理解Python中的一般表达式、控制结构、异常机制、类机制、多线程机制、模块的动态加载机制、内存管理机制等核心技术的运行原理，同时，本书所揭示的动态语言的核心技术对于理解其他动态语言，如Javascript、Ruby等也有较大的参考价值。

本书适合于Python程序员、动态语言爱好者、C程序员阅读。

<<Python源码剖析>>

书籍目录

章节摘录

第1章 Python对象初探　　对象是Python中最核心的一个概念，在Python的世界中，一切都是对象，一个整数是一个对象，一个字符串也是一个对象。
更为奇妙的是，类型也是一种对象，整数类型是一个对象，字符串类型也是一个对象。
换句话说，面向对象理论中的“类”和“对象”这两个概念在Python中都是通过Python内的对象来实现的。
在Python中，已经预先定义了一些类型对象，比如int类型、string类型、dict类型等，这些我们称之为内建类型对象。
这些类型对象实现了面向对象中“类”的概念；这些内建类型对象通过“实例化”，可以创建内建类型对象的实例对象，比如int对象、string对象、dict对象。
类似的，这些实例对象可以视为面向对象理论中“对象”这个概念在Python中的体现。
同时，Python还允许程序员通过class A（object）这样的表达式自己定义类型对象。
基于这些类型对象，同样可以进行“实例化”的操作，创建的对象称为“实例对象”。
Python中不光有着这些千差万别的对象，这些对象之间还存在着各种复杂的关系，从而构成了我们称之为“类型系统”或“对象体系”的东西。
Python中的对象体系是一个庞大而复杂的体系，如果说在本书的第一章我就试图将这个体系阐释清楚，这只能说明我是个疯子。
在本章中，我们的重点将放在了解对象在Python内部是如何表示的，更确切地说，因为Python是由C实现的，所以我们首先要弄清楚的一个问题就是：对象，这个神奇的东西，在C的层面，究竟长得是个什么模样，究竟是三头六臂，还是烈焰红唇。
除此之外，我们还将了解到类型对象在C的层面是如何实现的，并初步认识类型对象的作用及它与实例对象的关系。
总之，本章对Python对象体系的介绍力求简洁，但是并不肤浅，有的地方甚至会相当深入。
因此，在本章的阅读中，如果有什么疑难的地方，没有关系，先放下，只要有一个直观的感觉就可以了，这并不妨碍你阅读接下来的内容。
本章的目的是为能够顺利而快速地进入对内建对象的剖析打下必要的基础，至于对Python对象体系的详细剖析，会在第2部分的最后一章中介绍到。
只有到了那个时候，我们才有足够的能力将这个体系看个明白。
1.1　Python内的对象　　从1989年Guido在圣诞节揭开Python的大幕开始，一直到现在，Python经历了一次一次的升级，但是其实现语言一直都是ANSI C。
我们知道，C并不是一个面向对象的语言，那么在Python中，它的对象机制是如何实现的呢？
对于人的思维来说，对象是一个比较形象的概念，而对于计算机来说，对象却是一个抽象的概念。
它并不能理解这是一个整数，那是一个字符串，对于计算机来说，它所知道的一切都是字节。
通常的说法是，对象是数据以及基于这些数据的操作的集合。
在计算机中，一个对象实际上就是一片被分配的内存空间，这些内存可能是连续的，也可能是离散的，这都不重要，重要的是这片内存在更高的层次上可以作为一个整体来考虑，这个整体就是一个对象。
在这片内存中，存储着一系列的数据以及可以对这些数据进行修改或读取操作的一系列代码。
在Python中，对象就是为C中的结构体在堆上申请的一块内存，一般来说，对象是不能被静态初始化的，并且也不能在栈空间上生存。
唯一的例外就是类型对象，Python中所有的内建的类型对象（如整数类型对象，字符串类型对象）都是被静态初始化的。
在Python中，一个对象一旦被创建，它在内存中的大小就是不变的了。
这就意味着那些需要容纳可变长度数据的对象只能在对象内维护一个指向一块可变大小的内存区域的指针。

为什么要设定这样一条特殊的规则呢，因为遵循这样的规则可以使通过指针维护对象的工作变得非常的简单。
一旦允许对象的大小可在运行期改变，我们就可以考虑如下的情形。
在内存中有对象A，并且其后紧跟着对象B。
如果运行期某个时刻，A的大小增大了，这意味着必须将A整个移动到内存中的其他位置，否则A增大的部分将覆盖原本属于B的数据。
只要将A移动到内存中的其他位置，那么所有指向A的指针就必须立即得到更新，光是想一想，就知道这样的工作是多么的繁琐。

<<Python源码剖析>>

编辑推荐

《Python源码剖析：深度探索动态语言核心技术》适合于Python程序员、动态语言爱好者、C程序员阅读。

作为主流的动态语言，Python不仅简单易学、移植性好，而且拥有强大丰富的库的支持。此外，Python强大的可扩展性，让开发人员既可以非常容易地利用C/C++编写Python的扩展模块，还能将Python嵌入到C/C++程序中，为自己的系统添加动态扩展和动态编程的能力。

为了更好地利用Python语言，无论是使用Python语言本身，还是将Python与C/C++交互使用，深刻理解Python的运行原理都是非常重要的。

版权说明