程序员指南丛书

Python 编程基础

- Python 可用于编写独立程序、快速脚本和复杂应用的原型，它已经在各种平台上实现了移植，包括 UNIX、Linux、Windows 9x、Windows NT 和 Mac OS。
- 本书在简单介绍 Python 的基本原理和组成之后，详细介绍了 Python 程序的基本构件，然后循序渐进地讲述了 Python 的核心内容、应用开发及相关细节。
- 本书是基于 Windows 环境运行 Python 实例程序的，但由于 Python 强大的跨平台功能，本书所介绍的内容在 Linux、Mac OS 等其他平台也完全适用。

肖 建　林海波　等编著

清华大学出版社
http://www.tup.tsinghua.edu.cn

Python 编程基础

肖建　林海波　等　编著

清华大学出版社
北　京

内 容 简 介

Python 是一种功能十分强大的面向对象编程语言，可以用于编写独立程序、快速脚本和复杂应用的原型。作为一种开放源码的软件，Python 可以自由获取，而且易学易用。它已经在各种平台上实现了移植，其中包括 Unix、Linux、Windows 9x、Windows NT 和 MacOS。

本书详细讲述了 Python 语言的各个方面，在简单介绍 Python 的基本原理和组成之后，详细介绍了 Python 程序的基本构件：类型、操作符、语句、函数、模块、类以及异常，接着给出大量实例，循序渐进、深入浅出地讲述了 Python 的核心内容、应用开发及相关细节。本书适合作为 Python 初学者的入门读物。

本书是基于 Windows 环境运行 Python 实例程序的，但是事实上，由于 Python 强大的跨平台功能，本书所介绍的内容在 Linux、MacOS 等其他平台也完全适用。

图书在版编目(CIP)数据

Python 编程基础/肖建等编著. —北京：清华大学出版社，2003.6
(程序员指南丛书)
ISBN 7-302-06555-1

Ⅰ.P… Ⅱ.肖… Ⅲ.软件工具—程序设计 Ⅳ.TP311.56

中国版本图书馆 CIP 数据核字(2003)第 027469 号

出 版 者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
http://www.tup.tsinghua.edu.cn
http://www.tup.com.cn
责任编辑：胡先福
印 刷 者：北京鑫丰华彩印有限公司
发 行 者：新华书店总店北京发行所
开 本：787×1092 1/16 印张：18 字数：450 千字
版 次：2003 年 6 月第 1 版 2003 年 6 月第 1 次印刷
书 号：ISBN 7-302-06555-1/TP·4912
印 数：0001～3000
定 价：25.00 元

前　言

Python是一种功能十分强大的面向对象编程语言，可以用于编写独立程序、快速脚本和复杂应用的原型。Python是日益引起业界及市场广泛关注的开放源代码语言。它功能强大、易学易用、编码快速，目前已经有越来越多的公司和个人开始使用Python语言。因此，Python语言的学习将成为未来几年内的一个热点。

目前，Python语言的教材还比较少，而且多为外文教材。尽管网上有大量的关于Python语言的学习资料，但是往往缺乏系统性和完整性，而且由于目前网络访问个别地方尚不普及，费用较高，稳定性差，所以读者难于从中得到系统的学习。

因此，我们编写了本书。本书主要面向Python的初学者，且由于Python是一种极易学习的语言，所以本书并不要求读者具有相应的编程经验。本书通俗易懂，实例丰富，是步入Python世界的敲门砖。

Python具有很多与众不同的优点，例如：

- 面向对象
- 适应性强
- 可扩展性强
- 可移植性强
- 可阅读性强
- 健壮性强
- 快速建模
- 自动内存管理等

本书详细讲述了Python语言的各个方面，适合作为学习Python的入门书籍。在简单介绍Python的基本原理和组成之后，又详细介绍了Python程序的基本构件：类型、操作符、语句、函数、模块、类以及异常，接着给出大量实例，循序渐进、深入浅出地讲述了Python的核心内容、应用开发及相关细节。

全书共分为9章。第1章主要介绍Python的一些入门知识，包括Python的基本原理，如何使用不同方式运行Python等。

第2章介绍了Python的数据类型以及一些相关的基础知识，其中读者需要重点理解和掌握Python特有的数据类型如列表、元组等概念，并能够在后面的学习中深入理解这些数据类型的使用。本章所举的例子基本上都是比较简短的几个语句，而且都是在交互模式下运行。

第3章介绍了Python的程序结构，从这一章开始就进入了真正的Python编程阶段。在这一章里面所介绍的各种流程语句、异常处理、函数、模块等概念，都将为后面编写Python应用脚本程序打下坚实的基础。

第4章则从面向对象的角度来看待Python的编程，主要介绍了Python中如何使用类和

对象。要注意到，在 Python 中，所能接触到的一切几乎都是对象。

第 5 章介绍了如何使用 Tkinter 以及 Pmw 进行窗口程序的开发。许多用过 Delphi 或 Visual C/C ++ 的读者肯定对窗口应用程序的开发怀有强烈的兴趣，Python 提供了强大的功能来帮助用户开发窗口应用程序。

第 6 章所介绍的正则表达式对于一些读者可能比较新鲜。然而在 Unix 世界中，正则表达式被大量使用，它具有很强大的功能。

后面的 3 章则分别从多线程编程、图形编程以及多媒体编程的角度出发，开辟各自的专题进行讲解，其中介绍了一些相关模块的使用，并给出了许多实用的实例，帮助读者更好地掌握如何使用 Python 编写实用、高效的应用程序。

随着 Python 语言的普及，相信关于 Python 语言培训方面的书籍也会拥有越来越多的读者。但考虑到 Python 语言在我国刚刚起步，本书主要定位于对 Python 感兴趣的初学者。而关于 Python 语言的高级编程，将在后续的书目中进行介绍。

限于文章篇幅和作者水平，有很多其他功能没能介绍到，而且在所介绍的知识中，肯定还存在不足和漏误之处，希望有心的读者能够予以指正。

作　者

2003 年 3 月

目　录

第 1 章　Python 入门

Python 是一种容易学习又功能强大的程序语言，既可用于独立的程序，也可用于脚本程序，且适用于各种领域。它含有高效率的数据结构，也是一种简单但效果非常明显的面向对象编程语言（object-oriented programming）。Python 优雅的语法及动态类型识别（dynamic typing），加上解释性（intepretion）的本质，使它成为一种能在多种功能、多种平台上撰写脚本（scripts）及快速开发的理想语言。

Python 的解释器可以很容易地延伸，可以加入新的由 C 或 C++ 所编写的函数或数据类型。Python 也很适合用作其他应用程序的扩展语言，例如可以用 Python 来扩展 CAD、DBMaker 等的功能。

本书主要向读者介绍 Python 语言以及系统的基本概念和特性。但本书并不试图完整地介绍 Python 的每一个特性，甚至也不试图介绍每一个常用的功能。相反地，本书将介绍许多 Python 值得认识的功能，并且让读者对这个语言的大致风貌有一个了解。读完此书后，读者应该可以阅读并编写 Python 的模块和应用程序，并且可以自学在 Python 自带文档 Python Library Reference 中所介绍的各种模块了。

Python 是一种自由的开放源码的解释性语言，简单易学而且不失严谨，因此得到了初学者和编程专家的共同称赞。它类似于 Basic 语言的交互式特点，使它非常容易被学习和了解，很适合于进行快速原型开发，也同样适合于教学和程序设计的初学者。另外，它严谨的模块和对象体系使它同样适合于大型软件的开发。

在本书编写过程中，Python 2.3 Alpha 版已经发布（2002 年 12 月 31 日），但是从稳定性方面考虑，本书仍以 Python 2.2 作为讲解的基础。

1.1　Python 的特点

如果读者曾经写过大型的 Shell Script，大概就能了解这种感觉：想要新增加一个功能，但这个 Script 实在已经够大够慢够复杂了；或者，想要加入的新功能需要调用系统功能或其他函数，但是这些功能/函数只有 C 才能调用，但是要解决的问题好像并没有严重到要重新用 C 来编写整个程序。或者有些问题因为要用到可变长度的字符串或特殊的数据结构，例如用已排序的数据名称组成列表（list），用 C 来编写要比 Shell 麻烦得多，又或者是读者根本对 C 不是很熟。

另外一种情况是：也许要使用好几个 C 程序库，但是标准开发 C 程序的过程（写→编译→测试→重新编译）实在太花时间，而需要能快速地开发软件。又或者已经写好一个应用程序，这个程序可以使用一个扩展的语言来控制，但却不想创造一种语言，然后还得编写这个语言的编译器，还得把这个编译器跟程序放在一起。

在这些情况之下，Python 也许正是所需要的语言。Python 虽然简单，但却是不折不扣的程序语言。对大型程序来说，它能提供比 Shell 更多的结构性及支援。另一方面，它也提供了比 C 语言更强大的错误检查功能。由于 Python 是一种非常高级的语言，所以它有许多内建的数据类型，如有弹性的阵列及字典(dictionary)等，但如果这些用 C 来做的话得花上大半天的时间。正因为 Python 有较为一般的数据类型，所以它可以应用的范围比起 awk 甚或是 Perl 来要广得多。最起码，Python 跟这些语言一样容易开发。

Python 的另外一个特点就是可以将程序切成小模块，然后这些模块还可以应用在其他的程序之中。Python 本身也有一个相当大的标准模块库可以使用，或者当作学习 Python 程序设计的范例。在 Python 中也有内建的模块可以提供许多功能，例如：数据 I/O、系统调用、Sockets，甚至是与 Tk 之类的 GUI 工具交互的界面。

Python 是一种解释性语言，可以省掉不少在开发程序时编译及连结程序所用的时间。这个 Python 的解释器甚至可以交互式地使用，可以编写一些小程序来试验 Python 语言的特性，或是测试程序时可以节省不少时间。

用 Python 可以编写出非常精练而且可读性比较高的程序。用 Python 编写出的程序通常比用 C 或 C++ 编写的程序要短得多，其理由如下：

- Python 的高级数据类型使用户可以用很简单的一个声明表达复杂的运作过程
- Python 使用缩排来代替 C/C++ 中常见的前后括号{}
- Python 不需要声明变量以及参数

下面具体讨论 Python 还有哪些重要的特性。

1.1.1 面向对象

面向对象程序设计为结构化和过程化程序设计语言增添了新的活力，这些语言中的数据和逻辑关系都是程序设计中不可分割的元素。面向对象程序设计允许将特定的行为、特性和功能与将要处理的数据或它们所代表的数据关联在一起。

Python 语言面向对象的特性是与生俱来的。在 Python 程序员看来，凡事都是对象。对象，在编程语法中是部分暴露在表面的代码的封装单元。它们可以被再次使用而且是可移植的。面向对象语言不用实际了解任何关于对象内部如何工作的问题，就可以很容易扩展和使用其他程序的内部函数。

1.1.2 适应性强

人们通常会把 Python 语言和批处理或者 Unix 操作系统下的 Shell 脚本语言相提并论。简单的 Shell 脚本程序处理的是简单的任务，它们在长度上可以无限制地增长，但是在功能方面总有一个极限。Shell 脚本程序的代码很少能够被重复使用，因此也就把它的应用范围局限在了小型项目上。而事实上，即使是一个小型的项目也有可能导致代码冗长而又不知所云。

Python 语言就不会出现这种情况，我们可以根据项目的不同再增减代码，添加新的或者现有的 Python 元素，按照自己的想法重复使用编写好的代码。Python 语言鼓励简洁的代码

设计风格、高水平的结构设计、把多个组件捆绑到一起等做法。这些特点能够在软件开发项目向广度和深度扩展的同时有效地保证灵活性、一致性和更短的开发时间。

1.1.3　可扩展性强

Python 具有很强大的扩展性。如果知道如何写 C 语言程序，就可以很容易在 Python 的解释器中加入新的内建函数(function)或是模块。这样做的好处是可以让程序中关键部分的速度调到最快，或者是连结 Python 到二进制的程序库(例如第三方图形程序库)去。我们也可以把 Python 解释器嵌入到用 C 编写的应用程序里面去，然后 Python 就变成该应用程序的扩展或是商业化的语言了。

代码中最关键的部分，例如数据处理过程中经常会出现的热点或者那些必然会执行到的代码部分，最适合用于语言扩展。而通过 Python 接口对底层代码进行打包则更有利于用户建立二进制模块，而这些打包操作将要用到接口与纯 Python 模块所使用的完全一样，代码和用户的访问方法是一模一样的，根本不需要对代码进行任何修改。从代码方面讲，需要注意的惟一区别就是代码执行性能上的改善。很自然，这将取决于用户的应用程序以及对资源的要求情况。把应用程序中的瓶颈转换为编译后的代码肯定是有好处的，因为它会决定性地改善应用程序的整体性能。

1.1.4　可移植性强

能够运行 Python 语言的计算机平台相当广泛，例如下面这些平台是比较常用的：

- Unix(Solaris、Linux、FreeBSD、AIX、HP/UX、SunOS 以及 IRIX 等)
- Win9x/2000/NT/XP
- Macintoch
- OS/2
- DOS
- Windows 3.x
- PalmOS
- Windows CE
- Acorn/RISC OS
- BeOS
- Amiga
- VMS/OpenVMS
- ONX
- VxWorks
- Psion

Python 的应用平台如此广泛，使它能在计算机领域中快速地发展和壮大。事实上，Python 是用 C 语言编写的，由于 C 语言的可移植性，所以 Python 能够工作在具有 C 语言编译器和通用操作系统接口的任何类型的系统上。

虽然会有一些针对某种系统特别开发的模块，但在一种系统上开发的通用 Python 应用程序只需很少或者甚至不需要修改就能够运行在另外一种系统上，它的可移植性可以跨越多种体系结构和多种操作系统。

1.1.5 可阅读性强

Python 语言的语法和其他语言有一个非常明显的差异，那就是没有其他语言中通常用来完成存取变量、定义代码段、查找匹配模式等操作时所使用的符号。这些符号常见的有：美元符号“$ ”、分号“;”、波浪号“~”等。没有了这些符号，Python 代码的定义更加整齐，也更适合于阅读。

此外，与其他语言相比，用 Python 语言不太容易写出重叠嵌套的复杂语句，这使其他人能够很容易读懂你所编写的程序，你也可以很容易读懂他人所编写的程序。这同时也意味着它比较易于学习。

1.1.6 健壮性强

没有什么能够比允许程序员识别出某些错误条件，并且在这些错误发生时提供一个软件句柄(software handler)更有效果了。Python 在错误发生时提供了“安全而且理智”的退出机制，使程序员能够控制住局面。如果是因为致命错误退出时，Python 会通过一个完整的堆栈跟踪记录指明什么地方出现了错误，错误又是怎样发生的。Python 错误会产生例外(exception)，堆栈跟踪记录会指出例外出现的名称和类型。Python 还向程序员提供了识别例外并采取相应的必要措施的功能。这些例外句柄(exception handler)可以用来编写例外发生时采取的措施，在结束应用程序之前可以采取忽略该错误、重定向程序流、执行清理操作以及其他补救措施等做法。无论哪一种情况，开发周期中的调试纠错工作都会大大减轻，因为 Python 所具备的健壮性对软件设计人员和用户都有好处。如果某些特定的错误发生后处理不当，它还具备一定的统计定位功能。作为某个错误的结果而生成的堆栈跟踪记录功能不仅能够指示该错误的类型和位置，还能够指出错误代码位于哪个模块中。

1.1.7 快速建模

前面提到过 Python 语言非常容易学习和阅读，但用过 Basic 语言的读者可能会觉得这一点 Basic 语言也能做到。那么两者相比，Python 语言又多了什么优势呢？

Python 语言与那些自我包容和灵活性较少的编程语言不同，它有许多联系其他系统的不同接口，而且在功能上足够强大，也足够健壮，因此即使单独使用 Python 语言也完全可以建立起一个系统的整个模型。

当然，用传统的编译语言也能够建立同样的系统模型，但 Python 语言在工程方面的简单性使我们在完成同样的工作时显得游刃有余。此外，人们已经为 Python 语言开发出很多外部开发库，因此无论应用程序准备干什么用，都可能已经有人在这一条路走过了，这时候，只需要将这些外部开发库拿过来自行调配一番，就可以实现所需要的功能了。这些开发库所

涉及的范围非常广，其中包括：图形用户接口（GUI）、Internet 应用、图形图像处理（PIL）、数学计算分析（Numpy）、数据库处理、超文本、操作系统扩展以及音频/视频程序的开发工具等。

1.1.8　自动内存管理

使用 C 或 C++ 语言编写程序的一个最大弊病就是内存管理是由软件开发人员负责的。即使一个很少涉及内存访问、内存修改和内存管理的应用程序，程序员也必须在基本任务的基础上尽到为之管理内存使用情况的义务。这就给开发人员增添了不必要的负担和责任，也往往会因此而影响程序质量。

但在 Python 中就不同了。因为内存管理是由 Python 语言的解释器来完成的，所以应用程序的开发人员就可以撇开内存方面的问题，专心地按事先计划来编写应用程序，而这将带来更少的程序错误、更健壮的应用程序以及更短的整体开发时间。

表 1-1 列举了大部分 Python 语言细节上的特性。

表 1-1　Python 语言细节上的特性

特性	描述
缩排	Python 格式中最有特色的东西，使程序风格统一，而且减少了不必要的块包围符号
多种序列	有列表、元组（tuple）和字典（字符串可以看做元组）
序列乘法	序列可以进行乘法，表示多个序列的组合，如：s = [1]；s * 3 表示[1,1,1]
序列分片	序列可以通过分片进行切割，而且还可以进行序列的复制，如：s[:]
序列的负索引	序列的索引可以为负数，表示从最后向前数，如：s = [1,2,3]；s[-1]表示 3
动态语义	这个特性大部分脚本语言都有，就是可以通过赋值来决定变量的类型；而且可以通过赋值来改变变量的类型
连续比较	可以进行如下比较：2 < x < y < 3
超长整数	在整数后面加上'l'或'L'就可以使用很大的整数，甚至比 C 语言的整数还要大，如 a = 99999999999999999999L
同时赋值	在 Python 中，可以一次给多个变量赋值，如：a,b = 1,2
循环中的 else	在 Python 的循环语句中，可以有 else 子句，表示没有使用 break 语句就退出循环，即循环正常结束时，要执行的语句；在 for，while 中均有
异常中的 else	异常中也可有 else 子句，表示未出现异常时执行的语句；与 finally 不一样，finally 是必须要执行的，而 else 则是有条件执行的
格式输出中的关键字	在 Python 的格式输出中，格式符可以用关键字参数，如："%(table)s" % {'table':'hello'}
模块、包	Python 中可以有模块，还可以有包，可以像 Java 一样导入
语句动态生成	可以写一个字符串，然后运行它，如：exec "x = 2"；还可以执行一个文件（动态生成）execfile()，表达式为：eval()
属性的动态变化	可以使用 getattr()得到一个对象的属性，可以使用 setattr()来设置或修改对象的新属性，使用 delattr()删除对象的属性；有时还可以用a.newattr = attr 来设置新属性

续表

特性	描述
真正的面向对象	可以多态继承(有多个基类)、可以有私有属性,类公共属性
基类的动态改变	可以增加新的基类,从而使一个类拥有新的基类方法,而不用修改原来的基类
默认参数值	Python 的函数可以使用默认参数
关键字参数	支持关键字参数,如 a(name = 'abc')
函数的不定参数	Ython 的函数支持不定参数,可以使用关键字参数,也可以不使用
特殊的类方法	类可以定义特殊的方法,支持逻辑操作,如'+'、'*'、[i]等
单行函数	Python 支持单行函数,如:lambda x:x + 1
列表映射	对列表可以进行映射操作,如:["%d" % x for x in range(10)]
嵌套函数	可以像 pascal 一样写嵌套函数,2.1 版本以上变量的作用域也可以嵌套
对象的持续性	一个对象可以保存起来,在需要时恢复
可编译成字节码	像 Java 一样,可以将 Python 程序编译成字节码

1.2 Python 的安装

由于 Python 是自由的开放源码软件,因此可以直接从网上免费获取 Python 的最新版本。要想获取最新的 Python 源代码、二进制文件、文档以及相关新闻等,可以访问 Python 语言的主站点:http://www.python.org,如图 1-1 所示。

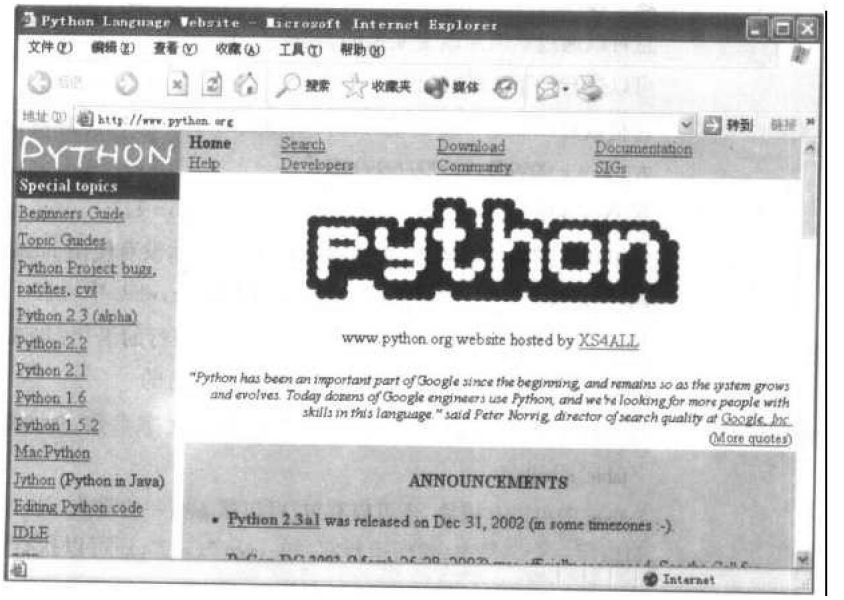

图 1-1 Python 主站点

在这里,如果可以找到对应于你的计算机平台的二进制安装代码,只要把文件下载到本