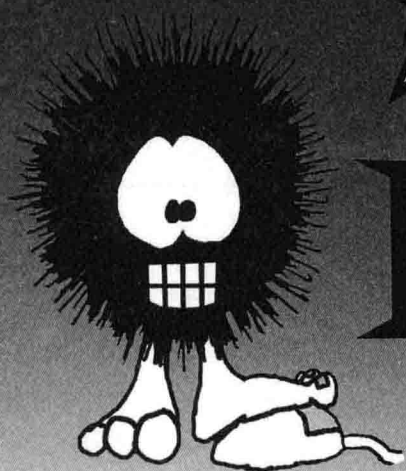




Python

[澳] Anthony Briggs 著
王威 袁国忠 译

易学



Python

[澳] Anthony Briggs 著
王威 袁国忠 译

人民邮电出版社
北京

图书在版编目 (CIP) 数据

易学Python / (澳) 布里格斯 (Briggs, A.) 著 ; 王威, 袁国忠译. — 北京 : 人民邮电出版社, 2014.4
ISBN 978-7-115-33522-7

I. ①易… II. ①布… ②王… ③袁… III. ①软件工具—程序设计 IV. ①TP311.56

中国版本图书馆CIP数据核字 (2013) 第254251号

版权声明

Original English language edition, entitled Hello! Python by Anthony Briggs, published by Manning Publications Co., 209 Bruce Park Avenue, Greenwich, CT 06830.

Copyright ©2012 by Manning Publications Co.

Simplified Chinese-language edition copyright ©2013 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由 Manning Publications Co. 授权人民邮电出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。
版权所有, 侵权必究。



-
- ◆ 著 [澳] Anthony Briggs
 - 译 王威 袁国忠
 - 责任编辑 傅道坤
 - 责任印制 程彦红 焦志炜
 - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
 - 邮编 100164 电子邮件 315@ptpress.com.cn
 - 网址 http://www.ptpress.com.cn
 - 北京鑫正大印刷有限公司印刷
 - ◆ 开本: 800×1000 1/16
 - 印张: 19.5
 - 字数: 417千字 2014年4月第1版
 - 印数: 1—3 500册 2014年4月北京第1次印刷

著作权合同登记号 图字: 01-2011-7809 号

定价: 45.00 元

读者服务热线: (010)81055410 印装质量热线: (010)81055316

反盗版热线: (010)81055315

广告经营许可证: 京崇工商广字第 0021 号

内容提要

本书采用简洁、有趣、易学的方式对 Python 编程语言进行了讲解，其风格与通篇介绍编程特性、罗列语言功能的大多数编程图书不同，而是引导读者带着好奇，带着问题去学习、掌握 Python 编程语言，继而编写真实而有用的程序。

本书总共分为 3 部分，共 12 章，第 1 部分介绍了为何使用 Python 来编程。从第 2 部分起，通过编写 Hunt the Wumpus 游戏带领读者认识并解决编程中的一些实际问题。例如，如何管理复杂的程序，确保其清晰易懂；如何在程序中使用 Python 标准库，以节省编程时间，同时让程序更容易理解；如何对编写好的程序进行测试；如何进行面向对象的程序设计。第 3 部分则使用框架对书中的程序进行完善，让读者对 Python 的强大功能有一个更深入的认识。本书最后还提供了一些 Python 资源，供读者深入学习 Python 时参考。

无论您是零基础的 Python 初学人员，还是具有其他语言编程经验，但是想从事 Python 开发的人员，本书都将带领您踏上有趣的 Python 学习之路。

序

Anthony 邀请我为本书作序时，我的第一反应是，“又是负担，我可不干”。然而，我总得先瞄一眼吧。这一看不打紧，我很快发现了点缀其中的 *User Friendly* 卡通人物(我敢肯定，像我一样从穿孔卡和磁带时代起就一直从事计算机工作的人，都对 *User Friendly* 钟爱有加)。我想我得看一看手稿，结果发现它包含短短 12 章，可让您对 Python 及其最常见的应用有足够认识，进而踏入编程殿堂，或让您确定做程序员并非您想要的生活。

即便最终的结论是编程并非您想要的生活，为购买本书所做的投资也物超所值。如果您连使用 Python 进行编程都不喜欢，那您很可能根本就不喜欢编程。在这种情况下，阅读本书可避免您入错行，白白浪费数年光阴。

本书到处都是实用建议，作者从不装大尾巴狼，试图发表一些不可信的看法。我深信，这部脚踏实地的著作将帮助大量非程序员读者熟悉 Python 语言。

但愿本书能让广大读者对编程和迷人的信息技术有新的认识。当前，美国很多州教育资金都捉衿见肘，难以在中学开展丰富的计算机教育，而本书足以吸引中学生，让他们迷恋编程。等政府洞悉教育的目的后，学生们将被编程课本深深地吸引，很快消化全书的内容。

Steve Holden
The Open Bastion 总裁

关于作者

从 2000 年初开始，Anthony Briggs 就一直在使用 Python 开发程序，当前正为墨尔本的 Ramble Communications 开发 Web 发布系统。他还曾为澳大利亚和加拿大的一家旅游公司开发预订系统，并最终成为首席开发人员，负责整个项目。

致谢

首先，感谢我美丽的妻子 Lyndall 给予支持，让我能抽出时间编写本书。本书的编写时间比预想的长得多，但她的热情始终如一，即便无数个周末我为了本书而深居简出。

其次，感谢 Manning 出版社的团队：感谢编辑 Sebastian Stirling 的建议和经验；感谢 June Eding 和 Tiffany Taylor 做最后的编辑和校对，确保本书按时付梓；感谢 Karen Tegtmeier 的协调和组织工作；感谢 Michael Stephens 帮助制订最初的写作计划。

第三，感谢 J.D. Fraser 允许 Manning 出版社在 Hello 系列丛书中使用 *User Friendly* 中的卡通形象，并允许我在本书中给这些卡通形象撰写对白。

第四，感谢本书 beta 版本的所有测试人员帮助找出错误，他们是 Daniel Hadson、Eldar Marcussen、William Taylor、David Hepworth 和 Tony Haig；还要感谢 MEAP 项目的所有成员，感谢他们提出建议和批评以及指正错误。

最后，要感谢以下评审人员在各个阶段提供宝贵的反馈：Tray Skates、Curtis Miller、Joe Hoover、Michael R. Bain、Francesco Goggi、Mike Stok、Michael R. Head、Cheryl M. Davis、Daniel Bretoi、Amos Bannister、Rob Allen、Dr. John Grayson、William Z. Taylor、Munch Paulson、David Hepworth、Eldar Marcussen、Daniel Hadson、Tony Niemann、Paolo Corti、Edmon Begoli、Lester Lobo、Robby O’ Connor 和 Sopan Shewale。特别感谢 Marion Newlevant 对终稿做详尽的技术审阅，还要特别感谢 Steve Holden 欣然为本书作序。

自序

受邀编写本书时，我无意再编写一本介绍性图书，而想采取不同的做法。我阅读过的编程图书无不罗列功能：列表可用于存储信息，您可调用 `len(mylist)` 来获悉它包含多少项信息，调用 `pop()` 来删除末尾的元素，调用 `append()` 来添加元素，等等。这些就是您需要知道的有关列表的全部知识，接下来介绍下一项功能。在这样的图书中，即便有示例程序，也要么只包含寥寥数行代码，要么出现在最后几章，作为对全书内容的总结。

我自己在最初学习编程时，并不是先从头到尾阅读整本编程图书，等对一切都熟悉后再开始编写程序。相反，我带着问题（要做的事情）去阅读，并在阅读过程中搞懂这些问题。我经常阅读编程图书，但都是旨在搞清楚我遇到的问题。程序编写出来后，它们也许不是特别优雅，运行速度也可能不是特别快，但它们毕竟出自我的手——我知道其工作原理，还解决了我面临的实际问题。

就这样锻炼到今天，我编写的程序在很大程度上都可以说既优雅，速度也快。在我认识的优秀程序员中，大多也这样学习编程。在本书中，我竭尽所能地重现这个过程，但步伐更快，涵盖了我学到的所有编程知识，还有我曾遇到过的陷阱。除第 1 章和第 12 章外，每章都将一个实用程序作为核心，旨在演示特定 Python 功能或库——通常是多个。有些章节很有趣，有些很实用，但不再有乏味的铺垫章节，详尽地讲述烦人的细节——列表或字典的各种功能，甚至是如何使用 Python 将数字相加。

相反，您将编写一个个程序，并在需要时学习相关的 Python 功能，而不是预先学习它们。有几章建立在前几章的基础之上，因此您将学习如何扩展既有程序，以添加新功能并避免设计失控——无论您要编写的程序规模如何，都必须这样做。本书还探讨了多种编程风格：从简单脚本到面向对象程序，再到基于事件的游戏。

我旨在向您呈现一部与众不同的作品，让您从第 1 章开始就编写程序，并在实际应用中学习 Python 功能的用法。但愿这种写作方式能助您真正明白如何使用 Python。

关于本书

本书旨在帮助读者学习 Python 和如何使用 Python 编程。无论您在编程方面完全是门外汉还是有一定经验，本书都将引领您踏上编程之路，最终编写出网络游戏和 Web 应用程序。

本书的风格不同于大多数编程图书：不详尽地列举各种功能，而向读者展示一幅更真实的画卷。从第 2 章起，您就将跟随笔者的脚步，编写真实而实用的程序——既说优点也谈缺点。编程语言的每项功能都有其用途，但如果不见识因误用导致的 bug、有问题的代码和糟糕的程序，就很难对其用途有真正的认识。

在本书中，将逐步改进和扩展一些程序，让读者明白函数、类和模块等 Python 功能有助于您控制规模不断增大的代码。需要扩展程序时，这些功能还有助于减轻您的负担。

虽然没有明确指出，但本书实际上大致分三部分。第 1 章介绍 Python 基本语法、库的用法和一些常见概念，还有为理解程序的工作原理而需要知道的各种知识；第二部分介绍高级功能以及让您能够直接用来实现更多功能的库；最后一部分使用框架编写了一些完整的程序，这对您学习 Python 帮助更大。

阅读完本书后，您依然将受惠于它。本书的所有程序都是可扩展的，您可在编写自己的程序时重用它们。大多数经验丰富的程序员都有代码库，其中包含他以前编写的所有代码；本书的代码为您开发自己的项目打下了基础。

本书组织结构

第 1 章简要地介绍了 Python、编程以及编程的根本目的：为何编写程序、可使用程序做什么。本章还详细地阐述了如何在 Windows、Mac 和 Linux 系统上安装 Python，以及在安装过程中可能遇到的一些常见问题。

第 2 章介绍程序的基本组成部分，而您将编写第一个程序——Hunt the Wumpus。

在此过程中，您将亲身体验程序员面临的一些问题。例如，如何管理复杂的程序，确保它们清晰易懂。

第 3 章介绍著名的 Python 标准库，还有如何导入标准库代码以及其他程序员为执行常见任务而编写的代码。您将学习如何在程序中使用这些代码，以节省大量时间并让程序更容易理解。

第 4 章介绍如何测试程序，包括单元测试和系统测试以及一些常见的测试问题及其解决方案。在此过程中，您将编写一个简单且易于扩展的待办事项清单应用程序。

第 5 章介绍如何使用 Python 进行面向业务的编程：下载网页、分析其中的信息并使用这些信息来创建电子邮件和 CSV 文件。本章还介绍了如何让程序更健壮，能够应对错误的信息格式和其他错误。

第 6 章将编写一个冒险游戏，其中有洞穴，有怪物，还有财宝。在此过程中，您将学习类的工作原理以及如何设计面向对象的程序。

第 7 章介绍如何使用混合类、`__getattr__` 和特性等高级功能来扩展类。本章还简要地介绍了 Python 的其他高级功能，如迭代器、生成器、正则表达式和函数式编程。

第 8 章介绍 Django 并帮助您建立一个待办事项网站。您将学习 Django 模板、数据库处理功能、表单和管理功能。本章还介绍了一些常见的 Web 开发模式，包括 REST 式 URL 设计以及使用正确的 HTTP 方法。

第 9 章介绍如何使用 Pyglet 库编写类似于《行星撞击地球》和《月球着陆器》的街机游戏，您将学习几何学、基于事件的编程和定时器。

第 10 章对第 6 章编写的冒险游戏进行扩展，让您和朋友能够使用 Telnet 通过网络玩这款游戏。为此，我们将使用 Python 网络库 Twisted 来处理连接、定义协议以及添加日志功能。

第 11 章修改第 8 章编写的待办事项清单应用程序，让每位用户都有自己的待办事项清单。您将学习如何处理登录、使用 Django 创建用户、使用 Django 通用视图、确保 Web 应用程序的安全、将 Web 应用程序部署到 Apache 或 Nginx 等服务器。

最后，第 12 章提供了一些资源，供您继续学习 Python 时参考，这包括邮件列表和用户组、可供您阅读和探索的程序以及您可能想研究的其他库。

目录

第 1 章 为何学习 Python 1

1.1 学习编程 2

1.1.1 告诉计算机做什么 2

1.1.2 编程是创意 4

1.1.3 编程是设计 4

1.2 是什么让 Python 如此杰出 5

1.2.1 Python 简单易学 5

1.2.2 Python 是真正的语言 5

1.2.3 Python “开箱即用” 6

1.2.4 Python 社区规模庞大 6

1.3 在 Windows 系统上安装 Python 7

1.3.1 安装 Python 7

1.3.2 在 Windows 系统上运行 Python 程序 9

1.3.3 从命令行运行 Python 程序 11

1.4 Linux 14

1.4.1 在 Linux 系统上安装 Python 14

1.4.2 Linux GUI 14

1.4.3 Linux 命令行 16

1.5 Macintosh 16

1.5.1 更新 Shell 配置文件 17

1.5.2 设置默认应用程序 17

1.6 排除故障 18

1.6.1 语法错误 18

1.6.2 文件扩展名不对 (Windows) 19

1.6.3 Python 的安装位置不对 (Linux) 19

1.7 文本编辑器和 IDE 19

1.8 总结 20

第 2 章 Hunt the Wumpus 21

2.1 程序是什么 22

2.1.1 在屏幕上显示 23

2.1.2 使用变量存储信息 23

2.1.3 询问玩家想做什么 24

2.1.4 做出决策 24

2.1.5 循环 25

2.1.6 函数 26

2.2 您的第一个程序 27

2.2.1 Hunt the Wumpus 的第一个版本 27

2.2.2 调试 29

2.3 捣鼓程序 29

2.3.1 调整洞穴数量 30

2.3.2 更友好的 wumpus 30

2.3.3 多个 wumpus 30

2.4 创建洞穴 31

2.4.1 列表 31

2.4.2 for 循环 33

2.4.3 构造洞穴网络 33

2.5 修复微妙的错误 35

2.5.1 问题 36

2.5.2 解决方案 36

2.5.3 打造连通的洞穴网络 36

2.6 使用函数让代码更整洁 39

2.6.1 函数的基本知识 39

2.6.2 变量作用域 40

2.6.3 共享状态 41

2.7 使用函数组织游戏

Hunt the Wumpus 42

- 2.7.1 与洞穴交互 42
- 2.7.2 创建洞穴 43
- 2.7.3 与玩家交互 44
- 2.7.4 程序的其他部分 45
- 2.8 弓和箭 47
- 2.9 进一步美化 49
- 2.10 接下来如何做 51
 - 2.10.1 蝙蝠和深渊 52
 - 2.10.2 让 Wumpus 移动 52
 - 2.10.3 不同的洞穴网络 52
- 2.11 总结 52

3

第3章 与外部交互 53

- 3.1 开箱即用: Python 库 54
 - 3.1.1 Python 标准库 54
 - 3.1.2 其他库 54
 - 3.1.3 使用库 55
 - 3.1.4 库到底是什么 55
- 3.2 另一种提问方式 58
 - 3.2.1 使用命令行参数 58
 - 3.2.2 使用模块 sys 58
- 3.3 读写文件 59
 - 3.3.1 路径和目录 (我的文件在哪里) 59
 - 3.3.2 路径 61
 - 3.3.3 打开文件 61
- 3.4 比较文件 63
 - 3.4.1 采集文件的指纹 63
 - 3.4.2 将文件的指纹存储到字典中 64
- 3.5 综合应用 65
- 3.6 测试程序 68
- 3.7 改进脚本 70
 - 3.7.1 按顺序排列结果 70
 - 3.7.2 比较目录 72
- 3.8 接下来如何做 73
- 3.9 总结 73

4

第4章 组织有序 74

- 4.1 规范程序 74
- 4.2 如何确定程序是正确的 75
 - 4.2.1 手工测试太烦 75
 - 4.2.2 功能测试 76
 - 4.2.3 单元测试: 让计算机去做 76

- 4.2.4 测试驱动的开发 76
- 4.3 编写程序 77
- 4.4 整合程序 80
 - 4.4.1 测试用户界面 80
 - 4.4.2 用输入做什么 81
 - 4.4.3 执行命令 82
 - 4.4.4 运行程序 85
- 4.5 状况评估 85
 - 4.5.1 接下来做什么 86
 - 4.5.2 我既繁忙又重要 88
 - 4.5.3 列表解析 89
 - 4.5.4 发现 bug 92
- 4.6 存储待办事项清单 95
- 4.7 编辑和删除 98
 - 4.7.1 修复一个小问题 99
 - 4.7.2 删除待办事项 101
 - 4.7.3 编辑待办事项 103
- 4.8 接下来如何做 106
 - 4.8.1 帮助命令 106
 - 4.8.2 撤销 106
 - 4.8.3 不同的界面 107
 - 4.8.4 时间管理和估算 107
 - 4.8.5 研究一个单元测试框架 107
- 4.9 总结 107

5

第5章 面向业务的编程 109

- 5.1 让程序相互交流 110
 - 5.1.1 CSV 来救场 110
 - 5.1.2 其他格式 111
- 5.2 准备工作 112
 - 5.2.1 安装 Beautiful Soup 112
 - 5.2.2 安装 Firefox 和 Firebug 113
 - 5.2.3 查看网页 113
- 5.3 使用 Python 下载网页 115
 - 5.3.1 提取想要的信息 116
 - 5.3.2 进一步提取数据 117
 - 5.3.3 网页抓取的注意事项 119
- 5.4 写入 CSV 文件 119
- 5.5 通过电子邮件发送 CSV 文件 121
 - 5.5.1 电子邮件的结构 121
 - 5.5.2 创建电子邮件 122
 - 5.5.3 发送电子邮件 123
 - 5.5.4 其他电子邮件模块 124

5.6 一个简单脚本——哪些地方可能出问题 125

- 5.6.1 未连接到网络 126
- 5.6.2 数据无效 126
- 5.6.3 数据出乎意料 126
- 5.6.4 无法写入数据 126
- 5.6.5 无法访问邮件服务器 126
- 5.6.6 您不必修复这些问题 127

5.7 如何处理有问题的脚本 127

- 5.7.1 交流 127
- 5.7.2 对故障的承受力 127
- 5.7.3 一开始就做好 128
- 5.7.4 失败要赶早，还要大张旗鼓 128
- 5.7.5 双保险 128
- 5.7.6 压力测试和性能测试 129
- 5.7.7 以后再试 129

5.8 异常 131

- 5.8.1 为何使用异常 131
- 5.8.2 程序崩溃是什么意思 131
- 5.8.3 捕获异常 134
- 5.8.4 模块 traceback 135

5.9 接下来如何做 136

5.10 总结 136

第6章 类与面向对象编程 137

6.1 类是什么 137

- 6.1.1 类包含数据 138
- 6.1.2 类是类型 138
- 6.1.3 类的工作原理 138
- 6.1.4 您的第一个类 138

6.2 面向对象设计 140

6.3 玩家输入 143

6.4 财宝 146

- 6.4.1 方法该放在什么地方 146
- 6.4.2 寻宝 147
- 6.4.3 捡宝 148

6.5 在洞穴迷宫中漫步 151

6.6 怪物出没 156

- 6.6.1 创建怪物 156
- 6.6.2 一些面向对象的技巧 157
- 6.6.3 组合起来 158

6.7 危险与刺激 161

6.8 接下来如何做 163

- 6.8.1 增加怪物和财宝 164
- 6.8.2 扩展格斗方式和物品 164

6.8.3 添加更多冒险元素 164

6.8.4 尝试动词和名词 164

6.8.5 研究类的高级功能 164

6.9 总结 164

第7章 高级技术 166

7.1 面向对象 166

- 7.1.1 混合类 166
- 7.1.2 super()及注意事项 169

7.2 定制类 169

- 7.2.1 __getattr__ 170
- 7.2.2 __setattr__ 170
- 7.2.3 __getattribute__ 172
- 7.2.4 特性 (property) 173
- 7.2.5 模拟其他类型 175

7.3 生成器和迭代器 177

- 7.3.1 迭代器 177
- 7.3.2 生成器 179
- 7.3.3 生成器表达式 180

7.4 使用迭代器 180

- 7.4.1 读文件 181
- 7.4.2 处理日志行 183
- 7.4.3 提取字段 184

7.5 函数式编程 188

- 7.5.1 副作用 188
- 7.5.2 map 和 filter 188
- 7.5.3 传递和返回函数 189

7.6 接下来如何做 191

7.7 总结 191

第8章 Django 192

8.1 使用 Django 编写 Web 应用 193

- 8.1.1 安装 Django 193
- 8.1.2 设置 Django 193

8.2 编写应用 197

- 8.2.1 最简单的待办事项清单 197
- 8.2.2 使用模板 198

8.3 使用模型 201

- 8.3.1 设置数据库 201
- 8.3.2 创建模型 201
- 8.3.3 Django 模块 admin 203
- 8.3.4 添加管理界面 204

8.4 使用数据 206

- 8.4.1 使用模型 206
- 8.4.2 设计 URL 208

- 8.4.3 提交表单 210
- 8.4.4 处理待办事项 213
- 8.5 最后的优化 216
- 8.6 接下来如何做 217
- 8.7 总结 217

第 9 章 使用 Pyglet 开发

游戏 218

- 9.1 安装 Pyglet 218
- 9.2 起步 220
- 9.3 简单的宇宙飞船 222
 - 9.3.1 让事情发生 224
 - 9.3.2 回到学校：牛顿第一定律及
矢量 226
- 9.4 引力 229
 - 9.4.1 计算引力 229
 - 9.4.2 小心行星 232
- 9.5 不能缺了枪炮 234
- 9.6 邪恶的外星人 237
- 9.7 接下来如何做 242
 - 9.7.1 添加新元素 242
 - 9.7.2 改成别的类型 242
 - 9.7.3 重构 243
 - 9.7.4 获取反馈 243
- 9.8 总结 243

第 10 章 Twisted 网络编程 244

- 10.1 安装 Twisted 244
- 10.2 第一个应用程序 245
- 10.3 MUD 初步 249
- 10.4 让游戏更有趣 254
 - 10.4.1 可恨的怪物 254
 - 10.4.2 回到聊天服务器 256
- 10.5 要求玩家登录 260
 - 10.5.1 探索不熟悉的代码 260
 - 10.5.2 整合 261
 - 10.5.3 编写自定义状态机 265
- 10.6 保存玩家状态 269
- 10.7 接下来如何做 273
- 10.8 总结 273

第 11 章 再谈 Django 274

- 11.1 身份验证 274
 - 11.1.1 登录 275
 - 11.1.2 添加用户 277
- 11.2 只列出当前用户的待办
事项 278
 - 11.2.1 修复数据库 279
 - 11.2.2 言归正传 282
 - 11.2.3 滴水不漏 283
 - 11.2.4 更新接口 284
- 11.3 测试 285
 - 11.3.1 单元测试 285
 - 11.3.2 功能测试 287
 - 11.3.3 运行测试 288
- 11.4 图像和样式 289
 - 11.4.1 使用 Django 提供媒体
内容 289
 - 11.4.2 由另一个服务器提供
媒体 291
 - 11.4.3 最后一步 292
- 11.5 接下来如何做 292
- 11.6 总结 293

第 12 章 接下来如何做 294

- 12.1 再阅读一些代码 294
 - 12.1.1 Python 标准库 295
 - 12.1.2 Python 秘诀 295
 - 12.1.3 开源项目 295
 - 12.1.4 加入 Python 社区 295
 - 12.1.5 加入邮件列表 295
 - 12.1.6 寻找当地用户组 296
 - 12.1.7 给开源项目帮忙 296
- 12.2 解决自己遇到的
问题 296
- 12.3 其他 Python 库 297
 - 12.3.1 代码剖析 297
 - 12.3.2 日志 297
 - 12.3.3 子进程和多任务 297
 - 12.3.4 更复杂的分析 297
 - 12.3.5 PIL 和图像处理 298
 - 12.3.6 XML、ElementTree 和
JSON 298
- 12.4 总结 298

第 1 章 为何学习 Python

本章介绍如下内容：

- 计算机和程序简介以及编写程序的原因；
- Python 简介及其如此杰出的原因；
- 安装 Python。

既然购买了本书，您很可能想学习编程技术。祝贺您！打算学编程的人不是很多，但编程很有趣，自学编程的回报也很高。编程是衡量您是不是文盲的新标准；如果不会编写简单程序（如批处理文件、邮件过滤器、电子表格公式），将在会这样做的人面前处于劣势。编程还是工具，可帮助您将点子付诸实施。

我 10 岁左右开始涉足编程，当时使用的是 Commodore 64。在那个时候，除游戏和简单字处理软件外，现成的软件不多。Commodore 等计算机自带了 BASIC，编程很容易，无需学很多知识就能很快得到结果。

然而，这样的情况一去不复返了。当前，要编写程序必须先安装软件；但学会编程后，就可创建各种神奇的程序，替您完成繁琐的工作，向您提供信息，供您娱乐。最重要的是，编程很有趣，每个人都应尝试。

您将发现，本书有一些卡通人物点缀其中。这旨在以有趣的方式提供一些背景知



识，让您知道接下来要讲的内容，或指出一些常见问题。这些卡通人物都来自 *User Friendly*，但内容都是我编写的，如果您不喜欢，就责怪我吧。

下面开始介绍有关编程的基本知识。

1.1 学习编程

这是一本编程书，在详细探讨编程之前，有必要介绍一些基本知识：什么是编程？如何编程？编程的定义很简单：

编程就是告诉计算机做什么。

但与大多数定义一样，这种定义太过简单。与国际象棋一样，学习编程的基本规则很容易，但要精通这些规则并结合使用它们要难得多。编程与人类活动的很多方面相关：从某种意义上说，如果不进行编程，就很难让计算机从事有意义的工作；编程不仅关乎数字和计算，还关乎设计、创意和个人表达。

PROGRAMMING IS
ART, MAN.
THAT'S ALL YOU
NEED TO KNOW.



1.1.1 告诉计算机做什么

下面将前面的编程定义分解为几部分，并分别审视它们。要明白前面的编程定义，需要知道计算机是什么以及“告诉”和“做什么”的准确含义。

1. 计算机

计算机是速度很快的计算器，可按您的指令做简单决策。计算机指令很简单，通常包含计算机要完成的任务，如相加和比较。然而，结合使用一系列指令可创建大型程序，让您能够完成复杂的工作，如撰写文档、玩游戏、计算账户余额以及控制核反应堆。

计算机看似很聪明，实际上很傻，一根筋且缺乏基本常识。计算机毕竟只是机器，您让它做什么，它就做什么，根本不考虑后果。就拿删除整个硬盘的命令来说吧，大多数人明白其后果很严重，可能在遵命行事前向您核实，但计算机什么都不问，直接将硬盘上所有的数据都删除。

注意：您让它做什么它就做什么，这既是计算机的优点，也是计算机的缺点。

如果使用（或编写）的程序行为怪异或无缘无故地崩溃，这不是它的错——它只是按指令行事。

2. 告诉

使用 Python 时，您通常这样向它发出命令：在文本文件中输入程序代码，再让程

序 Python 运行该文件，这将在本章后面介绍。您输入的指令可复杂可简单，执行的任务各种各样：将数字相加、打开其他文件、在屏幕上显示内容等。简单的 Python 程序类似于下面这样：

```
number = "42"
print "Guess my number..."
guess = raw_input(">")
if guess == number:
    print "Yes! that's it!"
else:
    print "No - it's", number

raw_input("hit enter to continue")
```

如果您不明白这个程序的含义，不用太担心；这个示例旨在提供一些背景知识。

3. 做什么

现在事情开始变得有趣起来了。大多数现代计算机都是“图灵完备（Turing complete）”的，即什么都能做——您想得到的任何事情计算机都能做。至少从理论上说如此：所需的时间或复杂程度可能超乎预期，或者需要特殊硬件（如果您希望以特定方式交互），但只要编写的程序没有问题，并提供了足够的数据，计算机就什么都能做。下面是计算机用于完成的一些任务。

- 控制有人或无人航天器和探测器，引导机器人进入其他星球，包括火星探索流浪者 Spirit 和 Opportunity。
- 通过计算机网络（互联网和万维网）将数据传输到世界各地。通过网络可向世界各地传输信息，还可接收来自世界各地的信息，且在一秒钟内就可搞定。
- 打造机器人，从工业机械手到 Roomba 真空吸尘器，再到像人类一样能爬楼并具备基本感情的机器人。
- 模拟现实世界的现象，如重力、光和天气。这包括科学模型及大多数游戏。

您可能没有将机器人探测器送到其他星球所需的硬件，但至少从理论上说，您依然能够运行这些程序。例如，就拿用于控制 Spirit 和 Opportunity 的计算机来说吧，其计算能力根本无法与您的台式机、笔记本电脑和手机相媲美，真是难以置信。

