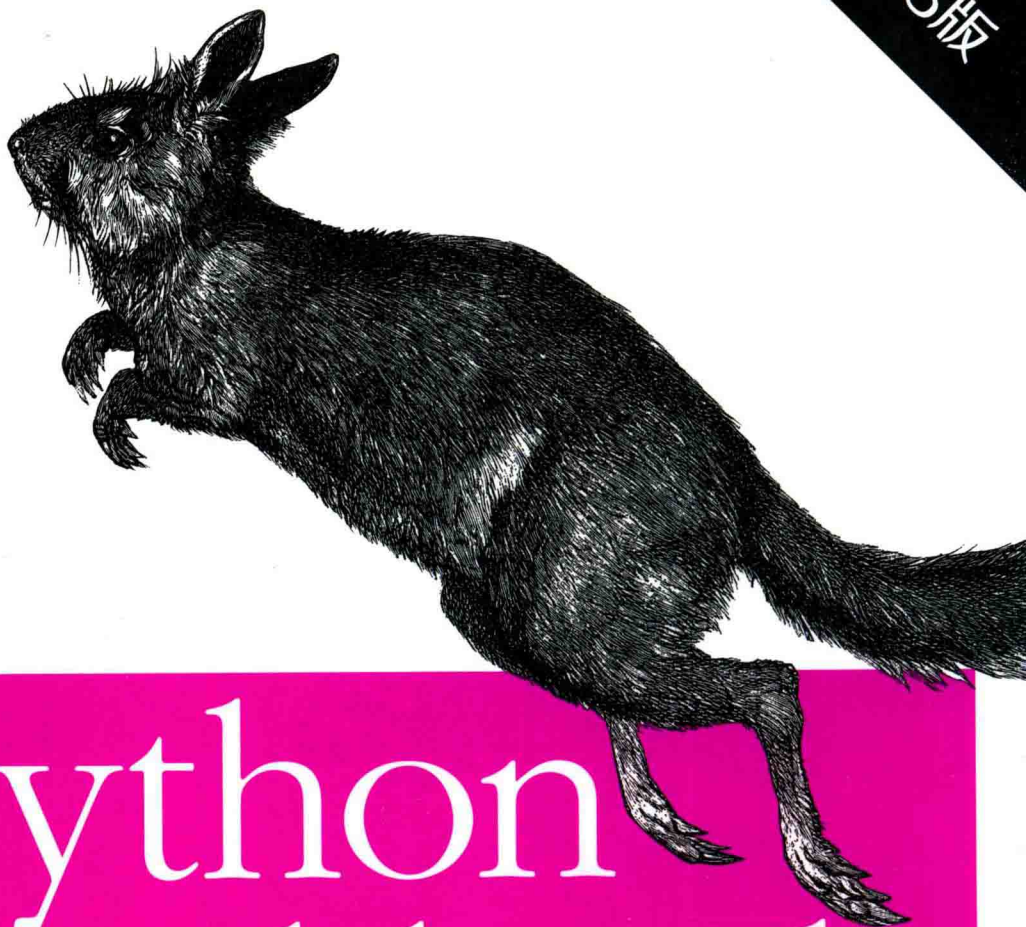


*Python Cookbook*  
*Recipes for Mastering Python 3*

第3版



# Python Cookbook™

中文版

[美] *David Beazley & Brian K. Jones* 著  
陈舸 译

O'REILLY®

人民邮电出版社  
POSTS & TELECOM PRESS

O'REILLY®

---

# Python Cookbook

## (第3版) 中文版

[美] David Beazley Brian K. Jones 著  
陈 舸 译

人民邮电出版社

北京

## 图书在版编目 (C I P) 数据

Python Cookbook : 中文版 : 第3版 / (美) 比斯利 (Beazley, D.), (美) 琼斯 (Jones, B. K.) 著 ; 陈舸译  
— 北京 : 人民邮电出版社, 2015. 5  
ISBN 978-7-115-37959-7

I. ①P… II. ①比… ②琼… ③陈… III. ①软件工  
具—程序设计—英文 IV. ①TP311.56

中国版本图书馆CIP数据核字(2015)第004926号

## 版权声明

Copyright © 2013 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and Posts & Telecom Press, 2015. Authorized translation of the English edition, 2011 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

本书中文简体字版由 O'Reilly Media, Inc. 授权人民邮电出版社出版。未经出版者书面许可, 对本书的任何部分不得以任何方式复制或抄袭。

版权所有, 侵权必究。

- 
- ◆ 著 [美] David Beazley Brian K. Jones
  - 译 陈 舸
  - 责任编辑 傅道坤
  - 责任印制 张佳莹 彭志环
  - ◆ 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>  
三河市中晟雅豪印务有限公司印刷
  - ◆ 开本: 787×1 000 1/16  
印张: 43.75  
字数: 914 千字 2015 年 5 月第 1 版  
印数: 1-3 000 册 2015 年 5 月河北第 1 次印刷

著作权合同登记号 图字: 01-2013-7656 号

定价: 108.00 元

读者服务热线: (010)81055410 印装质量热线: (010)81055316  
反盗版热线: (010)81055315

---

# 内容提要

本书介绍了 Python 应用在各个领域中的一些使用技巧和方法，其主题涵盖了数据结构和算法，字符串和文本，数字、日期和时间，迭代器和生成器，文件和 I/O，数据编码与处理，函数，类与对象，元编程，模块和包，网络和 Web 编程，并发，实用脚本和系统管理，测试、调试以及异常，C 语言扩展等。

本书覆盖了 Python 应用中的很多常见问题，并提出了通用的解决方案。书中包含了大量实用的编程技巧和示例代码，并在 Python 3.3 环境下进行了测试，可以很方便地应用到实际项目中去。此外，本书还详细讲解了解决方案是如何工作的，以及为什么能够工作。

本书非常适合具有一定编程基础的 Python 程序员阅读参考。

---

# O'Reilly Media, Inc. 介绍

O'Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始，O'Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 Make 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项 O'Reilly 的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

## 业界评论

“O'Reilly Radar 博客有口皆碑。”

——Wired

“O'Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference 是聚集关键思想领袖的绝对典范。”

——CRN

“一本 O'Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去 Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

---

# 前言

自 2008 年以来，我们已经目睹了整个 Python 世界正缓慢向着 Python 3 进化的事实。众所周知，完全接纳 Python 3 要花很长的时间。事实上，就在写作本书时（2013 年），大多数 Python 程序员仍然坚持在生产环境中使用 Python 2。关于 Python 3 不能向后兼容的事实也已经做了许多努力来补救。的确，向后兼容性对于任何已经存在的代码库来说是个问题。但是，如果你着眼于未来，你会发现 Python 3 带来的好处绝非那么简单。

正因为 Python 3 是着眼于未来的，本书在之前的版本上做了很大程度的修改。首先也是最重要的一点，这是一本积极拥抱 Python 3 的书。所有的章节都采用 Python 3.3 来编写并进行了验证，没有考虑老的 Python 版本或者“老式”的实现方式。事实上，许多章节都只适用于 Python 3.3 甚至更高的版本。这么做可能会有风险，但是最终的目的是要编写一本 Python 3 的秘籍，尽可能基于最先进的工具和惯用法。我们希望本书可以指导人们用 Python 3 编写新的代码，或者帮助开发人员将已有的代码升级到 Python 3。

无需赘言，以这种风格来编写本书给编辑工作带来了一定的挑战。只要在网络上搜索一下 Python 秘籍，立刻就能在 ActiveState 的 Python 版块或者 Stack Overflow 这样的站点上找到数以千计的使用心得和秘籍。但是，大部分这类资源已经沉浸在历史和过去中了。由于这些心得和秘籍几乎完全是针对 Python 2 所写的，其中常常包含各种针对 Python 不同版本（例如 2.3 版对比 2.4 版）之间差异的变通方法和技巧。此外，这些网上资源常常使用过时的技术，而这些技术现在成了 Python 3.3 的内建功能。想寻找专门针对 Python 3 的资源会比较困难。

本书并非搜寻特定于 Python 3 方面的秘籍将其汇集而成，本书的主题都是在创作中由现有的代码和技术而产生出的灵感。我们将这些思想作为跳板，尽可能采用最现代化的 Python 编程技术来写作，因此本书的内容完全是原创性的。对于任何希望以现代化的风格来编写代码的人，本书都可以作为参考手册。

在选择应该包含哪些章节时，我们有一个共识。那就是根本不可能编写一本涵盖了每种 Python 用途的书。因此，我们在主题上优先考虑 Python 语言核心方面的内容，以及能够广泛适用于各种应用领域的常见任务。此外，有许多秘籍是用来说明在 Python 3 中新增的功能，这对许多人来说比较陌生，甚至对于那些使用老版 Python 经验丰富的程序员也是如此。我们也会优先选择普遍适用的编程技术（即，编程模式）作为主

题，而不会选择那些试图解决一个非常具体的实际问题但适用范围太窄的内容。尽管在部分章节中也提到了特定的第三方软件包，但本书绝大多数章节都只关注语言核心和标准库。

## 本书适合谁

本书的目标读者是希望加深对 Python 语言的理解以及学习现代化编程惯用法的有经验的程序员。本书许多内容把重点放在库、框架和应用中使用的高级技术上。本书假设读者已经有了理解本书主题的必要背景知识（例如对计算机科学的一般性知识、数据结构、复杂度计算、系统编程、并发、C 语言编程等）。此外，本书中提到的秘籍往往只是一个框架，意在提供必要的信息让读者可以起步，但是需要读者自己做更多的研究来填补其中的细节。因此，我们假设读者知道如何使用搜索引擎以及优秀的 Python 在线文档。

有一些更加高级的章节将作为读者耐心阅读的奖励。这些章节对于理解 Python 底层的工作原理提供了深刻的见解。你将学到新的技巧和技术，可以将这些知识运用到自己的代码中去。

## 本书不适合谁

这不是一本用来给初学者首次学习 Python 编程而使用的书。事实上，本书已经假设读者通过 Python 教程或者入门书籍了解了基本知识。本书同样不能用来作为快速参考手册（即，快速查询特定模块中的某个函数）。相反，本书的目标是把重点放在特定的编程主题上，展示可能的解决方案并以此作为跳板引导读者学习更加高级的内容。这些内容你可能会在网上或者参考书中遇到过。

## 本书中的约定



### 提示

这个图标用来强调一个提示、建议或一般说明。



### 警告

这个图标用来说明一个警告或注意事项。

## 在线代码示例

本书中几乎所有的代码示例都可以在 <http://github.com/dabeaz/python-cookbook> 上找到。作者欢迎读者针对代码示例提供 bug 修正、改进以及评论。

## 使用代码示例

本书的目的是为了帮助读者完成工作。一般而言，你可以在你的程序和文档中使用本书中的代码，而且也没有必要取得我们的许可。但是，如果你要复制的是核心代码，则需要和我们打个招呼。例如，你可以在无需获取我们许可的情况下，在程序中使用本书中的多个代码块。但是，销售或分发 O'Reilly 图书中的代码光盘则需要取得我们的许可。通过引用本书中的示例代码来回答问题时，不需要事先获得我们的许可。但是，如果你的产品文档中融合了本书中的大量示例代码，则需要取得我们的许可。

在引用本书中的代码示例时，如果能列出本书的属性信息是最好不过。一个属性信息通常包括书名、作者、出版社和 ISBN。例如：Python Cookbook, 3rd edition, by David Beazley and Brian K. Jones (O'Reilly). Copyright 2013 David Beazley and Brian Jones, 978-1-449-34037-7。

在使用书中的代码时，如果不确定是否属于正常使用，或是否超出了我们的许可，请通过 [permissions@oreilly.com](mailto:permissions@oreilly.com) 与我们联系。

## 联系方式

如果你想就本书发表评论或有任何疑问，敬请联系出版社。

美国：

O'Reilly Media Inc.  
1005 Gravenstein Highway North  
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室 (100035)  
奥莱利技术咨询 (北京) 有限公司

我们还为本书建立了一个网页，其中包含了勘误表、示例和其他额外的信息。你可以通过链接 [http://oreil.ly/python\\_cookbook\\_3e](http://oreil.ly/python_cookbook_3e) 来访问页面。

关于本书的技术性问题或建议，请发邮件到：

[bookquestions@oreilly.com](mailto:bookquestions@oreilly.com)

欢迎登录我们的网站 (<http://www.oreilly.com>)，查看更多我们的书籍、课程、会议和最新动态等信息。

Facebook: <http://facebook.com/oreilly>

Twitter: <http://twitter.com/oreillymedia>

YouTube: <http://www.youtube.com/oreillymedia>



## 致谢

我们要感谢本书的技术校审人员，他们是 Jake Vanderplas、Robert Kern 以及 Andrea Crotti。感谢他们非常有用的评价，也要感谢整个 Python 社区的支持和鼓励。我们也要感谢本书第 2 版的编辑 Alex Martelli、Anna Ravenscroft 以及 David Ascher。尽管本书的第 3 版是新创作的，但之前的版本为本书提供了挑选主题以及所感兴趣的秘籍的初始框架。最后也是最重要的是，我们要感谢本书早期版本的读者，感谢你们为本书的改进做出的评价和建议。

### David Beazley 的致谢

写一本书绝非易事。因此，我要感谢我的妻子 Paula 以及我的两个儿子，感谢你们的耐心以及支持。本书中的许多素材都来自于我过去 6 年里所教的与 Python 相关的训练课程。因此，我要感谢所有参加了我的课程的学生，正是你们最终促成了本书的问世。我也要感谢 Ned Batchelder、Travis Oliphant、Peter Wang、Brain Van de Ven、Hugo Shi、Raymond Hettinger、Michael Foord 以及 Daniel Klein，感谢他们飞到世界各地去教学，而让我可以留在芝加哥的家中完成本书的写作。感谢来自 O'Reilly 的 Meghan Blanchette 以及 Rachel Roumeliotis，你们见证了本书的创作过程，当然也经历了那些无法预料到的延期。最后也是最重要的是，我要感谢 Python 社区不间断的支持，以及容忍我那不着调的胡思乱想。

David M.Beazley

<http://www.dabeaz.com>

<https://twitter.com/dabeaz>

### Brain Jones 的致谢

我要感谢我的合著者 David Beazley 以及 O'Reilly 的 Meghan Blanchette 和 Rachel Roumeliotis，感谢你们和我一起完成了本书的创作。我也要感谢我的妻子 Natasha，感谢你在我写作本书时给予的耐心和鼓励，也要谢谢你对于我所有追求的支持。我尤其要感谢 Python 社区。虽然我已经在多个开源项目和编程语言中有所贡献，但与 Python 社区长久以来所做的如此令人欣慰和富有意义的工作相比，我做的算不上什么。

Brain K.Jones

<http://www.protocolostomy.com>

<https://twitter.com/bkjones>

# 目录

第 1 章 数据结构和算法 .....	1
1.1 将序列分解为单独的变量 .....	1
1.2 从任意长度的可迭代对象中分解元素 .....	3
1.3 保存最后 N 个元素 .....	5
1.4 找到最大或最小的 N 个元素 .....	7
1.5 实现优先级队列 .....	9
1.6 在字典中将键映射到多个值上 .....	11
1.7 让字典保持有序 .....	13
1.8 与字典有关的计算问题 .....	14
1.9 在两个字典中寻找相同点 .....	15
1.10 从序列中移除重复项且保持元素间顺序不变 .....	17
1.11 对切片命名 .....	18
1.12 找出序列中出现次数最多的元素 .....	20
1.13 通过公共键对字典列表排序 .....	22
1.14 对不原生支持比较操作的对象排序 .....	23
1.15 根据字段将记录分组 .....	25
1.16 筛选序列中的元素 .....	26
1.17 从字典中提取子集 .....	29
1.18 将名称映射到序列的元素中 .....	30
1.19 同时对数据做转换和换算 .....	33
1.20 将多个映射合并为单个映射 .....	34
第 2 章 字符串和文本 .....	37
2.1 针对任意多的分隔符拆分字符串 .....	37
2.2 在字符串的开头或结尾处做文本匹配 .....	38
2.3 利用 Shell 通配符做字符串匹配 .....	40
2.4 文本模式的匹配和查找 .....	42
2.5 查找和替换文本 .....	45
2.6 以不区分大小写的方式对文本做查找和替换 .....	47
2.7 定义实现最短匹配的正则表达式 .....	48
2.8 编写多行模式的正则表达式 .....	49
2.9 将 Unicode 文本统一表示为规范形式 .....	50
2.10 用正则表达式处理 Unicode 字符 .....	52

2.11	从字符串中去掉不需要的字符	53
2.12	文本过滤和清理	54
2.13	对齐文本字符串	57
2.14	字符串连接及合并	59
2.15	给字符串中的变量名做插值处理	62
2.16	以固定的列数重新格式化文本	64
2.17	在文本中处理 HTML 和 XML 实体	66
2.18	文本分词	67
2.19	编写一个简单的递归下降解析器	70
2.20	在字节串上执行文本操作	80
<b>第 3 章</b>	<b>数字、日期和时间</b>	<b>83</b>
3.1	对数值进行取整	83
3.2	执行精确的小数计算	85
3.3	对数值做格式化输出	87
3.4	同二进制、八进制和十六进制数打交道	89
3.5	从字节串中打包和解包大整数	90
3.6	复数运算	92
3.7	处理无穷大和 NaN	94
3.8	分数的计算	96
3.9	处理大型数组的计算	97
3.10	矩阵和线性代数的计算	101
3.11	随机选择	103
3.12	时间换算	105
3.13	计算上周 5 的日期	107
3.14	找出当月的日期范围	108
3.15	将字符串转换为日期	110
3.16	处理涉及到时区的日期问题	112
<b>第 4 章</b>	<b>迭代器和生成器</b>	<b>114</b>
4.1	手动访问迭代器中的元素	114
4.2	委托迭代	115
4.3	用生成器创建新的迭代模式	116
4.4	实现迭代协议	118
4.5	反向迭代	121
4.6	定义带有额外状态的生成器函数	122
4.7	对迭代器做切片操作	123
4.8	跳过可迭代对象中的前一部分元素	124
4.9	迭代所有可能的组合或排列	127

4.10	以索引-值对的形式迭代序列	129
4.11	同时迭代多个序列	131
4.12	在不同的容器中进行迭代	133
4.13	创建处理数据的管道	134
4.14	扁平化处理嵌套型的序列	137
4.15	合并多个有序序列, 再对整个有序序列进行迭代	139
4.16	用迭代器取代 while 循环	140
<b>第 5 章</b>	<b>文件和 I/O</b>	<b>142</b>
5.1	读写文本数据	142
5.2	将输出重定向到文件中	145
5.3	以不同的分隔符或行结尾符完成打印	145
5.4	读写二进制数据	146
5.5	对已不存在的文件执行写入操作	149
5.6	在字符串上执行 I/O 操作	150
5.7	读写压缩的数据文件	151
5.8	对固定大小的记录进行迭代	152
5.9	将二进制数据读取到可变缓冲区中	153
5.10	对二进制文件做内存映射	155
5.11	处理路径名	157
5.12	检测文件是否存在	158
5.13	获取目录内容的列表	159
5.14	绕过文件名编码	161
5.15	打印无法解码的文件名	162
5.16	为已经打开的文件添加或修改编码方式	164
5.17	将字节数据写入文本文件	166
5.18	将已有的文件描述符包装为文件对象	167
5.19	创建临时文件和目录	169
5.20	同串口进行通信	171
5.21	序列化 Python 对象	172
<b>第 6 章</b>	<b>数据编码与处理</b>	<b>177</b>
6.1	读写 CSV 数据	177
6.2	读写 JSON 数据	181
6.3	解析简单的 XML 文档	186
6.4	以增量方式解析大型 XML 文件	188
6.5	将字典转换为 XML	192
6.6	解析、修改和重写 XML	194
6.7	用命名空间来解析 XML 文档	196

6.8	同关系型数据库进行交互	198
6.9	编码和解码十六进制数字	201
6.10	Base64 编码和解码	202
6.11	读写二进制结构的数组	203
6.12	读取嵌套型和大小可变的二进制结构	207
6.13	数据汇总和统计	218
<b>第 7 章</b>	<b>函数</b>	<b>221</b>
7.1	编写可接受任意数量参数的函数	221
7.2	编写只接受关键字参数的函数	223
7.3	将元数据信息附加到函数参数上	224
7.4	从函数中返回多个值	225
7.5	定义带有默认参数的函数	226
7.6	定义匿名或内联函数	229
7.7	在匿名函数中绑定变量的值	230
7.8	让带有 N 个参数的可调用对象以较少的参数形式调用	232
7.9	用函数替代只有单个方法的类	235
7.10	在回调函数中携带额外的状态	236
7.11	内联回调函数	240
7.12	访问定义在闭包内的变量	242
<b>第 8 章</b>	<b>类与对象</b>	<b>246</b>
8.1	修改实例的字符串表示	246
8.2	自定义字符串的输出格式	248
8.3	让对象支持上下文管理协议	249
8.4	当创建大量实例时如何节省内存	251
8.5	将名称封装到类中	252
8.6	创建可管理的属性	254
8.7	调用父类中的方法	259
8.8	在子类中扩展属性	263
8.9	创建一种新形式的类属性或实例属性	267
8.10	让属性具有惰性求值的能力	271
8.11	简化数据结构的初始化过程	274
8.12	定义一个接口或抽象基类	278
8.13	实现一种数据模型或类型系统	281
8.14	实现自定义的容器	287
8.15	委托属性的访问	291
8.16	在类中定义多个构造函数	296
8.17	不通过调用 <code>init</code> 来创建实例	298

8.18	用 Mixin 技术来扩展类定义 .....	299
8.19	实现带有状态的对象或状态机 .....	305
8.20	调用对象上的方法，方法名以字符串形式给出 .....	311
8.21	实现访问者模式 .....	312
8.22	实现非递归的访问者模式 .....	317
8.23	在环状数据结构中管理内存 .....	324
8.24	让类支持比较操作 .....	327
8.25	创建缓存实例 .....	330
<b>第 9 章</b>	<b>元编程 .....</b>	<b>335</b>
9.1	给函数添加一个包装 .....	335
9.2	编写装饰器时如何保存函数的元数据 .....	337
9.3	对装饰器进行解包装 .....	339
9.4	定义一个可接受参数的装饰器 .....	341
9.5	定义一个属性可由用户修改的装饰器 .....	342
9.6	定义一个能接收可选参数的装饰器 .....	346
9.7	利用装饰器对函数参数强制执行类型检查 .....	348
9.8	在类中定义装饰器 .....	352
9.9	把装饰器定义成类 .....	354
9.10	把装饰器作用到类和静态方法上 .....	357
9.11	编写装饰器为被包装的函数添加参数 .....	359
9.12	利用装饰器给类定义打补丁 .....	362
9.13	利用元类来控制实例的创建 .....	364
9.14	获取类属性的定义顺序 .....	367
9.15	定义一个能接受可选参数的元类 .....	370
9.16	在 *args 和 **kwargs 上强制规定一种参数签名 .....	372
9.17	在类中强制规定编码约定 .....	375
9.18	通过编程的方式来定义类 .....	378
9.19	在定义的时候初始化类成员 .....	382
9.20	通过函数注解来实现方法重载 .....	384
9.21	避免出现重复的属性方法 .....	391
9.22	以简单的方式定义上下文管理器 .....	393
9.23	执行带有局部副作用的代码 .....	395
9.24	解析并分析 Python 源代码 .....	398
9.25	将 Python 源码分解为字节码 .....	402
<b>第 10 章</b>	<b>模块和包 .....</b>	<b>406</b>
10.1	把模块按层次结构组织成包 .....	406
10.2	对所有符号的导入进行精确控制 .....	407

10.3	用相对名称来导入包中的子模块	408
10.4	将模块分解成多个文件	410
10.5	让各个目录下的代码在统一的命名空间下导入	413
10.6	重新加载模块	415
10.7	让目录或 zip 文件成为可运行的脚本	416
10.8	读取包中的数据文件	417
10.9	添加目录到 sys.path 中	418
10.10	使用字符串中给定的名称来导入模块	420
10.11	利用 import 钩子从远端机器上加载模块	421
10.12	在模块加载时为其打补丁	439
10.13	安装只为自己所用的包	441
10.14	创建新的 Python 环境	442
10.15	发布自定义的包	444
<b>第 11 章</b>	<b>网络和 Web 编程</b>	<b>446</b>
11.1	以客户端的形式同 HTTP 服务交互	446
11.2	创建一个 TCP 服务器	450
11.3	创建一个 UDP 服务器	454
11.4	从 CIDR 地址中生成 IP 地址的范围	456
11.5	创建基于 REST 风格的简单接口	458
11.6	利用 XML-RPC 实现简单的远端过程调用	463
11.7	在不同的解释器间进行通信	466
11.8	实现远端过程调用	468
11.9	以简单的方式验证客户端身份	472
11.10	为网络服务增加 SSL 支持	474
11.11	在进程间传递 socket 文件描述符	481
11.12	理解事件驱动型 I/O	486
11.13	发送和接收大型数组	493
<b>第 12 章</b>	<b>并发</b>	<b>496</b>
12.1	启动和停止线程	496
12.2	判断线程是否已经启动	499
12.3	线程间通信	503
12.4	对临界区加锁	508
12.5	避免死锁	511
12.6	保存线程专有状态	515
12.7	创建线程池	517
12.8	实现简单的并行编程	521
12.9	如何规避 GIL 带来的限制	525

12.10	定义一个 Actor 任务	528
12.11	实现发布者/订阅者消息模式	532
12.12	使用生成器作为线程的替代方案	536
12.13	轮询多个线程队列	544
12.14	在 UNIX 上加载守护进程	547
<b>第 13 章</b>	<b>实用脚本和系统管理</b>	<b>552</b>
13.1	通过重定向、管道或输入文件来作为脚本的输入	552
13.2	终止程序并显示错误信息	553
13.3	解析命令行选项	554
13.4	在运行时提供密码输入提示	557
13.5	获取终端大小	558
13.6	执行外部命令并获取输出	558
13.7	拷贝或移动文件和目录	560
13.8	创建和解包归档文件	562
13.9	通过名称来查找文件	563
13.10	读取配置文件	565
13.11	给脚本添加日志记录	568
13.12	给库添加日志记录	571
13.13	创建一个秒表计时器	573
13.14	给内存和 CPU 使用量设定限制	575
13.15	加载 Web 浏览器	576
<b>第 14 章</b>	<b>测试、调试以及异常</b>	<b>578</b>
14.1	测试发送到 stdout 上的输出	578
14.2	在单元测试中为对象打补丁	579
14.3	在单元测试中检测异常情况	583
14.4	将测试结果作为日志记录到文件中	585
14.5	跳过测试，或者预计测试结果为失败	586
14.6	处理多个异常	587
14.7	捕获所有的异常	589
14.8	创建自定义的异常	591
14.9	通过引发异常来响应另一个异常	593
14.10	重新抛出上一个异常	595
14.11	发出告警信息	596
14.12	对基本的程序崩溃问题进行调试	598
14.13	对程序做性能分析以及计时统计	600
14.14	让你的程序运行得更快	603



第 15 章 C 语言扩展 .....	610
15.1 利用 ctypes 来访问 C 代码 .....	612
15.2 编写简单的 C 语言扩展模块 .....	618
15.3 编写一个可操作数组的扩展函数 .....	622
15.4 在 C 扩展模块中管理不透明指针 .....	625
15.5 在扩展模块中定义并导出 C API .....	628
15.6 从 C 中调用 Python .....	633
15.7 在 C 扩展模块中释放 GIL .....	639
15.8 混合使用 C 和 Python 环境中的线程 .....	639
15.9 用 Swig 来包装 C 代码 .....	640
15.10 用 Cython 来包装 C 代码 .....	646
15.11 用 Cython 来高效操作数组 .....	652
15.12 把函数指针转换为可调用对象 .....	657
15.13 把以 NULL 结尾的字符串传给 C 库 .....	659
15.14 把 Unicode 字符串传递给 C 库 .....	663
15.15 把 C 字符串转换到 Python 中 .....	667
15.16 同编码方式不确定的 C 字符串打交道 .....	669
15.17 把文件名传给 C 扩展模块 .....	672
15.18 把打开的文件传给 C 扩展模块 .....	673
15.19 在 C 中读取文件型对象 .....	674
15.20 从 C 中访问可迭代对象 .....	677
15.21 排查段错误 .....	678
附录 A 补充阅读 .....	680