TURING 图灵程序设计丛书

Apress®

Foundations of Python Network Programming, Third Edition

Python网络编程（第3版）

从应用开发角度介绍网络编程基本概念、模块以及第三方库
利用Python轻松快速打造网络应用程序
Python 3示例讲解

[美] Brandon Rhodes　John Goerzen　著
诸豪文　译

中国工信出版集团

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

Foundations of Python Network Programming, Third Edition

Python网络编程

（第3版）

[美] Brandon Rhodes John Goerzen 著
诸豪文 译

人民邮电出版社
北 京

图书在版编目（CIP）数据

Python网络编程 : 第3版 / (美) 布兰登·罗德(Brandon Rhodes), (美) 约翰·格岑(John Goerzen)著 ; 诸豪文译. -- 北京 : 人民邮电出版社, 2016.9
(图灵程序设计丛书)
ISBN 978-7-115-43350-3

Ⅰ. ①P… Ⅱ. ①布… ②约… ③诸… Ⅲ. ①软件工具－程序设计 Ⅳ. ①TP311.561

中国版本图书馆CIP数据核字(2016)第191199号

内 容 提 要

本书针对想要深入理解使用 Python 来解决网络相关问题或是构建网络应用程序的技术人员，结合实例讲解了网络协议、网络数据及错误、电子邮件、服务器架构和 HTTP 及 Web 应用程序等经典话题。具体内容包括：全面介绍 Python 3 中最新提供的 SSL 支持，异步 I/O 循环的编写，用 Flask 框架在 Python 代码中配置 URL，跨站脚本以及跨站请求伪造攻击网站的原理及保护方法，等等。

本书适合 Web 应用程序的开发者、系统集成者、系统管理员以及所有 Python 程序员。

◆ 著 [美] Brandon Rhodes John Goerzen
译 诸豪文
责任编辑 朱 巍
执行编辑 温 雪
责任印制 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
北京昌平百善印刷厂印刷

◆ 开本：800×1000 1/16
印张：22.5
字数：532千字 2016年9月第1版
印数：1－4 000册 2016年9月北京第1次印刷
著作权合同登记号 图字：01-2015-1056号

定价：79.00元

读者服务热线：(010)51095186转600 印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广字第8052号

站在巨人的肩上

Standing on Shoulders of Giants

iTuring.cn

站在巨人的肩上

Standing on Shoulders of Giants

iTuring.cn

版权声明

献　词

献给我可爱的外甥女们：Avery、Savannah和Aila。
记得我们一起骑着自行车飞奔，她们总是在拐弯处勇猛地冲下山坡。
我希望无论她们将来会不会做网络编程的工作，都能永远保持这种对待生活的无畏。

引　言

经过20年来的严谨创新，Python在引入了诸如上下文管理器（context manager）、生成器（generator）以及推导式（comprehension）等特性的同时，仍然保持了其语法及概念一贯的简洁。Python终于开始大放异彩，这对于Python社区来说，分外激动人心。

在有些人眼中，Python是一门只能被Google和NASA这种一流编程机构冒险使用的精品语言，但事实恰好相反，Python正在被广泛使用。它不仅应用于传统的编程任务，如Web应用程序设计；也被大量“被赶鸭子上架的程序员”——科学家、数据专员以及工程师所采用，他们编程并非出于兴趣，而是必须靠编程才能在自己的领域中更进一步。我认为，一门简单的编程语言为业余编程人员提供的便利是不容小觑的。

Python 3

自2008年问世以来，Python 3一直在不断修订和精简，以期承担起Python 2的角色。如今，Python 3迎来了面世以来的第二个5年，已然成为了Python社区内进行创新的首选平台。

Python 3提供给网络程序员的编程平台几乎在方方面面都有所改进，无论是基础性的（如将Unicode文本设为Python 3的默认字符串类型），还是特有的（如对SSL的正确支持、内置的用于异步编程的`asyncio`框架，以及对标准库中大大小小的模块的细微调整）。这是一个显著的进步。要知道，Python 2就已经是程序员在现代互联网环境中用来快速高效工作的最佳语言之一了。

本书的目的并非提供从Python 2迁移到Python 3的全面指南。本书不会讲述如何给老版本的`print`语句添加括号，也不会介绍如何对导入的标准库模块进行重命名，更不会讲解如何对Python 2中从字节字符串到Unicode字符串的自动转换（这一转换通常基于粗略的猜测）这一危险特性引入的缺陷代码进行深度调试。关于如何从Python 2迁移到Python 3，以及如何仔细编写能够同时支持这两个平台的代码，已经有很多优质的资源提供了相应的指导。

本书的重点在于网络编程，并且在所有示例脚本及代码片段中都使用Python 3来阐释。这些例子的目的是帮助读者全面了解使用这门语言提供的工具构建网络客户端、网络服务器以及网络工具的最佳实践。读者可以将这些例子与第2版中各章使用的脚本进行比较，以此来学习从Python 2到Python 3的迁移。两个版本的代码都可以从https://github.com/brandon-rhodes/fopnp/tree/m/获取。这要感谢Apress出版社，让我们能够从网络上获取源代码。接下来各个章节的目的就是介绍如何最大化地使用Python 3提供的功能来解决现代网络编程的问题。

本书直接把关注点放在了如何使用Python 3正确完成工作上，希望以此帮助准备从头开始编写新应用程序的程序员和准备将老版本代码移植到新规范的程序员。他们都应该了解如何使用Python 3编

写出正确的网络程序代码，并清楚他们应该努力去编写的目标代码的样式和特点。

这一版的改进

这一版对以前的版本进行了更新，除了将目标语言向Python 3迁移，以及对过去5年中标准库和第三方Python模块进行了许多更新外，还在如下方面进行了改进。

- 这一版列出的每个Python程序都编写成了一个模块。换言之，每个程序都会导入其依赖的模块，定义其函数或类，然后通过一个if语句来确保所有导入行为。只有在模块__name__为特殊字符串值'__main__'时，该if语句对应的代码块才会执行。模块__name__为'__main__'，表示该模块作为主程序执行。这是在本书之前版本中被彻底忽略的一个Python最佳实践，使得读者无法很方便地将示例代码应用到真实的代码库中解决问题。老版本的代码清单在左边空白区域列出了执行逻辑，而没有将其放在if语句中。这样做可能会少写一两行代码，但是初学Python的程序员在部署实际代码时，能从中得到的实践指导却少了很多。
- 老版本中的脚本临时使用原始的sys.argv字符串列表来解析命令行选项和参数，而这一版中的大多数脚本使用的则是标准库的argparse模块。这不仅阐明并记录了每个脚本被调用时表示的语义，还允许每个脚本的用户使用-h或者--help查询选项，在Windows或Unix的命令行中获取交互式的帮助文档。
- 这一版的程序通过在with控制语句中打开文件来进行合理的资源控制。with语句包含的代码块完成的时候，打开的文件会自动关闭。在之前的版本中，大多数程序并不会这样做，而是依赖于C Python的运行时服务。Python的官方网站提供的C Python通过严格的引用计数机制，通常可以确保打开的文件能够及时关闭。
- 大多数程序在进行字符串插值时已经转而使用现代的format()方法，以前则使用string % tuple的方法。后者在20世纪90年代有一定的意义，因为那时大多数程序员都通晓C语言。但对于现在进入这个领域的新人程序员来说，这种方法可读性较差，而且由于自定义的Python类不能对百分号格式符进行操作符重载，因此提供的功能也不够强大。
- 重写了关于HTTP和万维网的3章（第9章至第11章），侧重于更清晰地解释协议，并介绍Python所提供的大部分用于编写Web应用的现代工具。这一版在解释HTTP协议时使用Requests库进行客户端操作，它提供的API相当实用。第11章提供了Flask和Django框架的示例。
- Python 3大量改进了为编写安全的应用程序所提供的支持，所以这一版彻底重写了关于SSL/TLS（第6章）的内容。Python 2的ssl模块使用的是一个折中的方法。该方法功能较弱，甚至没有验证服务器的证书是否与Python连接的主机名对应。因此，Python 3的ssl模块提供了一个设计更严谨、功能更丰富的API，以便用户安全方便地使用其特性。

因此，对于渴望不断学习的程序员来说，且不论Python 3本身所做出的改进，单论代码清单以及示例的构建，这一版相较以前的版本也绝对是更好的资源。

网络实验环境

本书给出程序的源代码可在网上获取，因此本书所有者及想要阅读本书的读者均可访问网络资源

进行学习。读者可以从如下网址找到本书各章的代码目录：

https://github.com/brandon-rhodes/fopnp/tree/m/py3

然而对于对网络编程充满好奇的学生来说，本书给出的代码清单所能提供的帮助仍然是有限的。如果只在单机上进行实验，那么网络编程的很多特性都无法得到体现。因此，本书的程序库也提供了一个由12台机器组成的网络实验环境，每台机器通过一个Docker容器实现。程序库同样包含了一个安装脚本，用来构建、启动以及连接各容器的镜像。读者可从如下地址找到该安装脚本以及容器镜像：

https://github.com/brandon-rhodes/fopnp/tree/m/playground

图0-1展示了这12台机器以及它们的连接架构。该网络的设计目的是模拟一个小型的互联网。

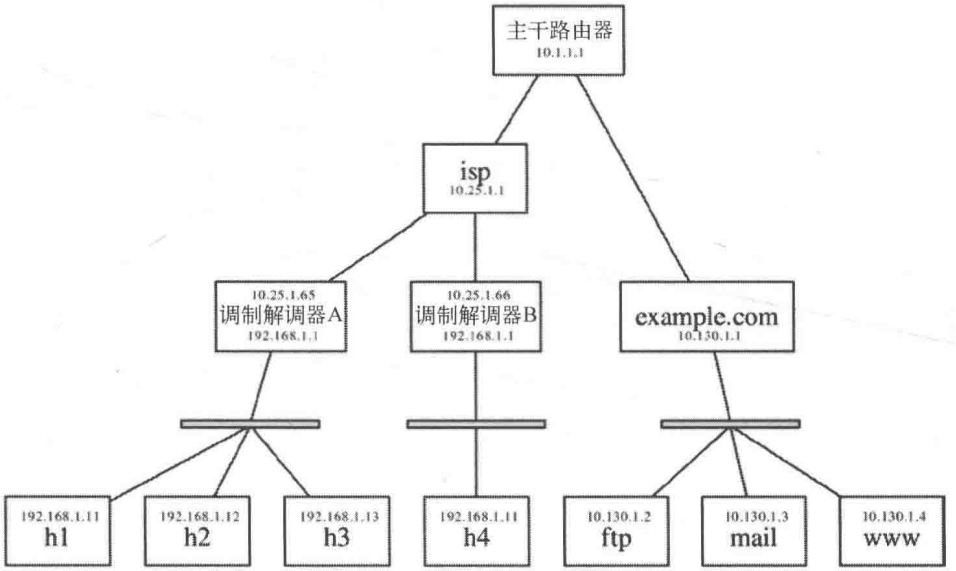

图0-1　网络实验环境的拓扑结构

- 调制解调器A和B下面的客户机（h1到h4）表示典型的客户端场景，如家庭或咖啡店。这些客户机不向互联网提供服务。事实上，它们对外部网络是完全不可见的。它们与同一家庭或咖啡店环境内的其他主机共享子网，只拥有在这些子网内才有意义的本地IP地址。当这些客户机连接外网时，其实都是通过调制解调器的IP地址进行连接的。
- 调制解调器可通过一个ISP网关与更广域的网络直接相连。实验环境中的广域网由一个主干路由器表示，该路由器负责将数据包发送至与之相连的网络。
- example.com及与之相连接的机器表示一组简单的面向服务的机房配置。这里没有进行任何网络地址转换或伪装。互联网上的各个客户端可随意访问example.com后的3个服务器提供的服务端口。
- ftp服务器、mail服务器和www服务器中的任意一个都运行着正确配置的守护进程。因此，本书中的Python脚本也可以运行在其他机器上，并成功连接到上述服务。
- 所有服务器均已成功安装TLS证书（见第6章），而所有客户机则有example.com的签名认证，并安装了受信证书。这意味着要求TLS认证的Python脚本可以成功获取认证。

随着Python和Docker的不断优化，我们也会持续维护网络实验环境。关于如何下载并在本机上运行该网络实验环境的使用说明会不断更新，也会根据用户的使用报告进行微调，以保证读者可以在Linux、Mac OS X以及Windows机器上运行该实验环境的虚拟机。

由于可以在网络实验环境的任意一台机器上连接并运行命令，读者可对网络中的任意一个点进行数据包追踪，查看客户端和服务器之间的网络数据传输情况。通过文档中的示例代码以及本书中的例子和使用说明，读者可以深刻、生动地理解客户端与服务器是如何通过网络进行通信的。

致　谢

2009年，当我开始修订本书第1版，重新编写第2版时，我就对自2004年出版第1版以来的5年之内Python语言发生的巨大变化震惊不已。CherryPy和Django这样全新的Web框架彻底改变了程序员编写Web应用的方式，而mechanize和lxml这样的库也大大简化了从HTTP服务器获取信息的步骤。

当我完成第3版的初稿时，Python社区又完成了一次革新！大大小小的改进让我觉得有必要对一些文字进行修改，并且彻底重写某些章节。这个新版首要体现了整个Python语言社区的辛勤工作，无论是编写Python 3的语言核心开发者，还是Flask、Tornado等让编写HTTP服务更方便更安全的新Web框架的作者，又或者是维护标准库和第三方库的程序员；说真的，他们对整个生态系统进行了优化，加入了大量工具和库，帮助我们更好地编写互联网程序。

自第2版出版以来，许多读者联系了我，提出了各种问题和想法，或是指出某个代码清单无法成功运行或是不再正确。我希望他们知道，这些反馈对于新版本的编写是至关重要的，我想他们将在这本书中看到基于这些反馈所做出的重大改进。

非常感谢Apress的编辑和审稿人，他们阅读了我的初稿（有些地方写得特别挫），发现代码清单中的bug以及错误，在编写到完成的过程中给予了帮助及指导。我要特别感谢文字编辑，我慢慢地从他们那里学到不要在每句话里都说very和actually。也要感谢他们，让我在人生中慢慢学会如何区分在连接从句时该使用that还是which。

最后，感谢在我编写本书时所有等着我回邮件、改bug或是和我见面的人，特别感谢我的妻子Jackie一直给我鼓励。

站在巨人的肩上

Standing on Shoulders of Giants

iTuring.cn

站在巨人的肩上
Standing on Shoulders of Giants

iTuring.cn

目　录

第1章 客户端/服务器网络编程简介

本书将带领读者使用Python语言探索网络编程。全书涵盖了网络编程的基本概念、模块以及第三方库，这些第三方库在使用流行互联网通信协议与远程机器进行通信时可能会用到。

本书并不适用于未使用过Python语言甚至未编写过计算机程序的读者，因为书中并不教授如何使用Python语言进行编程。本书假设读者已经通过相关资料对Python编程有所了解。我希望书中的Python示例能够帮助读者了解如何组织并编写自己的代码，但我会不加解释地大量使用各种高级的Python特性。不过，在遇到一些我认为尤其有趣或巧妙的用法时，我也会指出并加以说明。

另一方面，本书并不预设读者有任何网络知识！只要使用过浏览器或者发送过电子邮件，就完全可以从头开始阅读本书并循序渐进地学习计算机网络。本书会从应用开发者的角度来介绍网络：一是从实现一个网络连接服务的角度，例如网站、邮箱服务器或者网络游戏；二是从编写使用服务的客户端程序的角度。

然而，需要注意的是，本书并不教授如何建立或配置网络。网络设计、服务器管理以及自动化配置都是内容丰富的主题，并且与本书涉及的计算机编程内容没有重复。随着在OpenStack、SaltStack和Ansible中的大量应用，Python确实已经在自动化配置领域得到了广泛的应用。尽管如此，如果读者想深入学习配置运维相关的技术，还是应该寻找专门介绍这方面技术的书籍和文档。

1.1 基础：协议栈与库

在学习Python网络编程的过程中，会不断遇到以下两个概念。

- 协议栈（protocol stack）。复杂的网络服务建立在简单网络服务的基础之上。
- 经常需要使用之前编写过的Python库（library），包括Python内置的标准库以及下载安装的第三方库，来解析要使用的网络通信协议。

很多情况下，网络编程就是选择并使用一个已经支持所需网络操作的库的过程。本书的主要目的就是向读者介绍Python的一些重要网络库，并讲解这些网络库所使用的底层网络服务。通过了解底层知识，除了可以理解网络库的运行原理外，还能够在底层部分出现错误时知道具体发生了什么，因此对读者是大有裨益的。

让我们从一个简单的例子开始。下面是一个邮箱地址：

```
207 N. Defiance St
Archbold, OH
```

假设我们想知道该物理地址的纬度和经度。谷歌提供的地理编码API就能够完成这种从物理地址

到经纬度的转化。那么如何通过Python来使用这个网络服务呢？

想要使用一个全新的网络服务时，总是可以先查找一下是否有人已经实现了这个服务的通信协议。在本例中，所需要使用的是谷歌地理编码协议（Google Geocoding protocol）。因此可以先查阅Python的标准库文档，看看有没有与地理编码有关的内容。

http://docs.python.org/3/library/

读者查询到有关地理编码的库了吗？我也没有找到。尽管如此，经常查阅标准库目录对于Python程序员来说还是非常重要的，因为每次查询都可以加深对Python提供的服务的了解。Doug Hellmann的博客“Python Module of the Week”是另一个极佳的参考资料，读者也可以从中了解Python标准库提供的功能。

由于标准库没有提供用于本例的包，因此我们可以查询Python包索引（Python Package Index）。这是一个相当优质的资源，可以从中找到全球各地的程序员和机构贡献的各种通用包。读者也同样可以通过所用服务提供商的网站查看其是否提供Python库的接口。除此之外，还可以直接用Python加上要使用的网络服务名称作为关键字，在谷歌上搜索，看看搜索结果中的前几个链接是否值得一试。

在本例中，我搜索了Python Package Index，找到如下链接：

https://pypi.python.org/

然后搜索geocoding，很快就找到了一个叫作pygeocoder的包，这个包提供了简洁的接口来获取谷歌的地理编码服务。（尽管从包名可以注意到，这个包并非由谷歌官方提供，而是第三方开发者开发的。）

http://pypi.python.org/pypi/pygeocoder/

正常情况下，开发者都能找到一个看上去完全满足需求的Python包并想试一试。我要在这里暂停一下，向读者介绍一个用来尝试新Python库的最棒的技术：virtualenv！

以前，Python包的安装令人厌恶，并且不可逆。安装过程要求管理员权限，因此永久性地修改了系统的Python安装环境。经过了几个月的大规模Python开发，系统的Python包安装目录下就包含了大量的无用包，这些包均通过手动安装。更悲剧的是，一些新安装的包可能由于与已安装的包不兼容而无法使用。要知道，使用这些已安装包的项目在几个月前就已结束。

而现在，谨慎的Python程序员再也不会遭这种罪了。通常来说，只要在系统层面上安装一个Python包就够了，那就是virtualenv！只要安装了virtualenv，就可以创建任意数量的小型、独立的“虚拟Python环境”。如果想要做实验的话，可以在这些“虚拟Python环境”中安装或卸载相应的包。这很好地保护了系统Python环境。当一个特定的项目或实验结束之后，只需删除对应的虚拟环境目录即可。过程简单，系统也能保持干净。

在本例中，需要新建一个虚拟环境来测试pygeocoder包。如果读者之前从未在系统上安装过virtualenv，可访问下面的链接来下载并安装：

http://pypi.python.org/pypi/virtualenv

virtualenv安装完成后，便可使用如下命令来新建一个环境。（在Windows环境中，虚拟环境中包含Python二进制可执行文件的目录被命名为Scripts，而不是bin。）

```
$ virtualenv -p python3 geo_env
$ cd geo_env
$ ls
```

```
bin/ include/ lib/
$ . bin/activate
$ python -c 'import pygeocoder'
Traceback (most recent call last):
  File "<string>", line 1, in <module>
ImportError: No module named 'pygeocoder'
```

由上可见，pygeocoder包尚未安装。由于运行了activate命令，新建的虚拟环境现在已经在系统路径中了，因此可在虚拟环境中使用pip命令安装pygeocoder包。

```
$ pip install pygeocoder
Downloading/unpacking pygeocoder
  Downloading pygeocoder-1.2.1.1.tar.gz
  Running setup.py egg_info for package pygeocoder

Downloading/unpacking requests>=1.0 (from pygeocoder)
  Downloading requests-2.0.1.tar.gz (412kB): 412kB downloaded
  Running setup.py egg_info for package requests

Installing collected packages: pygeocoder, requests
  Running setup.py install for pygeocoder

  Running setup.py install for requests

Successfully installed pygeocoder requests
Cleaning up...
```

virtualenv中pygeocoder包的二进制可执行文件现在已经可以使用了。

```
$ python -c 'import pygeocoder'
```

安装了pygeocoder后，代码清单1-1所示的程序search1.py就可以运行了。

代码清单1-1 获取经度与纬度

```
#!/usr/bin/env python3
# Foundations of Python Network Programming, Third Edition
# https://github.com/brandon-rhodes/fopnp/blob/m/py3/chapter01/search1.py

from pygeocoder import Geocoder

if __name__ == '__main__':
    address = '207 N. Defiance St, Archbold, OH'
    print(Geocoder.geocode(address)[0].coordinates)
```

在命令行运行上述程序，可得到如下结果：

```
$ python3 search1.py
(41.521954, -84.306691)
```

看，屏幕上已经打印出了该地址的经纬度！这个结果是直接从谷歌的网络服务获取到的。我们的第一个示例程序成功了。

通过上面的过程，你可能会觉得一本讲Python网络编程的书只是教你直接下载安装一个第三方包，然后把一个有趣的网络编程问题变成了3行无聊的Python脚本！你是不是觉得很不爽？淡定！你会发现

90%的编程挑战就是这么解决的——在Python社区中寻找其他已经解决你所面临问题的程序员，然后在他们的解决方案的基础上构建智能又简洁的方案。

但是，本例还没彻底完成。我们已经展示了如何简单地获取复杂的网络服务，但是，优雅的pygeocoder接口背后的原理是怎样的呢？网络服务是如何工作的呢？下面，我们将详细学习如何在一个包含至少6层的网络协议栈的顶层构建这个复杂的服务。

1.2　应用层

第一个示例程序使用了一个从Python包索引下载的第三方Python库来解决问题。这个第三方库完整地包装处理了谷歌地理编码API。但如果没有这样的库怎么办呢？如果需要自己为谷歌地图API编写客户端又该怎么办呢？

让我们通过代码清单1-2所示的search2.py来看看答案。这个程序没有使用直接提供地理编码功能的第三方库，而是使用了更底层的Requests库。Requests库相当流行，较pygeocoder更底层一些。从前面的pip install命令可以看出，Requests库也已经安装在虚拟环境中了。

代码清单1-2　从谷歌地理编码API获取一个JSON文档

```
#!/usr/bin/env python3
# Foundations of Python Network Programming, Third Edition
# https://github.com/brandon-rhodes/fopnp/blob/m/py3/chapter01/search2.py

import requests

def geocode(address):
    parameters = {'address': address, 'sensor': 'false'}
    base = 'http://maps.googleapis.com/maps/api/geocode/json'
    response = requests.get(base, params=parameters)
    answer = response.json()
    print(answer['results'][0]['geometry']['location'])

if __name__ == '__main__':
    geocode('207 N. Defiance St, Archbold, OH')
```

运行这段Python程序的返回结果与第一个脚本的结果颇为相似。

```
$ python3 search2.py
{'lat': 41.521954, 'lng': -84.306691}
```

但可以看到，程序输出并非完全相同。例如，JSON数据将结果封装为一个“对象”，Requests库以Python字典的形式提供了该“对象”。但显而易见的是，这个脚本与第一个脚本基本完成了相同的功能。

读者很容易注意到，本例中的代码并未像pygeocoder库一样提供更高层的地理编码语义。只有仔细分析这段代码才可能知道，其目的是查询一个邮寄地址对应的经纬度。在search1.py中，程序直接请求将地址转化为经纬度，而代码清单1-2则仔细地构造了一个基础URL以及一系列查询参数。程序编写者需要阅读过谷歌的文档才能清楚了解这些查询参数的意义。顺便提一下，如果想阅读该文档，可以从如下链接找到API描述：