PACKT PUBLISHING

Expert Python Programming

Python 高级编程

[法] Tarek Ziadé 著
姚军 夏海轮 王秀丽 译

旧金山湾区Python社区主持人
Shannon –jj Behrens倾情作序

人民邮电出版社
POSTS & TELECOM PRESS

Python高级编程

[法] Tarek Ziadé 著
姚军 夏海轮 王秀丽 译

人民邮电出版社
北京

图书在版编目（CIP）数据

Python高级编程 /（法）莱德著 ；姚军，夏海轮，王秀丽译. -- 北京 ：人民邮电出版社，2010.1
ISBN 978-7-115-21703-5

Ⅰ. ①P… Ⅱ. ①莱… ②姚… ③夏… ④王… 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(2009)第202900号

Python 高级编程

◆ 著 [法] Tarek Ziadé
译 姚 军 夏海轮 王秀丽
责任编辑 刘映欣

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子函件 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：20
字数：422 千字 2010 年 1 月第 1 版
印数：1－3 000 册 2010 年 1 月北京第 1 次印刷

著作权合同登记号 图字：01-2009-3160 号
ISBN 978-7-115-21703-5

定价：45.00 元

读者服务热线：(010)67132705 印装质量热线：(010)67129223
反盗版热线：(010)67171154

内 容 提 要

本书通过大量的实例，介绍了 Python 语言的最佳实践和敏捷开发方法，并涉及整个软件生命周期的高级主题，诸如持续集成、版本控制系统、包的发行和分发、开发模式、文档编写等。本书首先介绍如何设置最优的开发环境，然后以 Python 敏捷开发方法为线索，阐述如何将已被验证的面向对象原则应用到设计中。这些内容为开发人员和项目管理人员提供了整个软件工程中的许多高级概念以及专家级的建议，其中有些内容的意义甚至超出了 Python 语言本身。

本书针对具备一定 Python 基础并希望通过在项目中应用最佳实践和新的开发技术来提升自己的 Python 开发人员。

序　言

Python 已经出现很长时间了。

曾几何时，我坚持使用 Python，许多公司都认为我疯了。现在，Python 编码人员已经供不应求了。诸如 Google、YouTube、VMware 和 DreamWorks 等重要的公司都在不断地争夺能找到的 Python 人才。

Python 过去一贯落后于 Perl，因为 Perl 拥有 CPAN。而现在，setuptools 和 PyPI 已经引发了高可用的、高质量的第三方 Python 程序库的大爆发。Python 也曾经落后于 Java Servlets 和 Ruby on Rails，因为没有标准的用于与 Web 服务器交互的 API。现在，Web 服务器网关接口（WSGI）引领了 Python Web 世界的复兴。有了 Google App Engine，我想我们还将看到更多。

Python 似乎对很固执并对简洁性有真正品味的编程人员具有吸引力。很少有人因为学校里的学习任务或者大公司都在使用 Python 而成为 Python 编程人员。人们只有在发现了 Python 的内在美才会沉迷于它。因此，Python 的书多得令人吃惊。我没有足够的统计数字来证明，但是，似乎 Python 的编程书籍要多于其他语言。然而，一直没有出现足够高级的 Python 书籍，直至本书的出现。

本书介绍了一系列有趣的主题。将介绍 Python 的一组特性，以及以意想不到的方式使用它们的方法。此外，还介绍了一组精选的、有趣的第三方程序库和工具，以及使用 Python 工具和程序库的敏捷编程方法。这包括基于 nose 的测试驱动开发，基于 doctest 的文档驱动开发，使用 Mercurial 进行源代码控制，使用 Buildbot 实现持续集成，以及使用 Trac 完成项目管理。最后，介绍了一些更传统的主题，如剖析、优化以及诸如 Alex Martelli 的 Borg 方法，还介绍了诸如单例之类的设计模式。

如果你正打算从了解 Python 进步到精通 Python，那么本书正适合你。实际上，这正是 5 年前我所希望拥有的书。我花费了数年，通过踏踏实实地参加 PyCon 和本地的 Python 用户组而得到的一切，现在已经都在这一本简洁的书当中了。

没有什么比成为 Python 编程人员更激动人心的了！

Shannon-jj Behrens

旧金山湾区 Python 兴趣团体主持人

关于作者

Tarek Ziadé 是巴黎 Ingeniweb 公司的 CTO，其工作方向为 Python、Zope、Plone 技术和质量保证。他参与 Zope 社区已经有 5 年了，并且曾经为 Zope 自身贡献过代码。

Tarek 创建了 Afpy，这是法国的 Python 用户组，并且编写了两本法语的 Python 书籍。他还在诸如 Solutions Linux、Pycon、OSCON 和 EuroPython 等法国及国际会议上发表了许多演讲，并且主持了许多课程。

我要感谢在编写本书时帮助过我的所有人。

首先感谢整个 Python 社区、AFPY 用户组，感谢 Stefan Schwarzer 关于优化的讲义以及他的引用和了不起的反馈和评论，感谢 Georg Brandl 对第 10 章中 Sphinx 部分的评审，Peter Bulychev 对 CloneDigger 部分的协助，Ian Bicking 对 minimock 部分的协助，Logilab 团队对 PyLint 部分的协助，感谢 Gael Pasgrimaud、Jean-François Roche 和 Kai Lautaportti 在 collective.buildbot 之上的工作，感谢 Cyrille Lebeaupin、Olivier Grisel、Sebastien Douche 和 St é fane Fermigier 对本书的审阅。感谢 OmniGroup 和他们了不起的 OmniGraffle 工具，本书中的所有图都是用它制作出来的（参见 http://www.omnigroup.com/ applications/OmniGraffle）。

特别感谢 Shannon "jj" Behrens 对本书的深入评审。

关于审校人员

Shannon -jj Behrens 是旧金山湾区 Python 兴趣团体主持人。在对 Python 书籍进行技术评审和不断忙碌工作之余，他享受着和 4 个孩子的游乐时光。

我要感谢 Tarek 耐心地听取我的批评。我还要感谢我可爱的妻子 Gina-Marie Behrens，她使我有足够的时间完成本书的编辑而免受孩子的搅扰。

Paul Kennedy 是 Sydney 科技大学工程和信息技术系的高级讲师。他还是 Quantum 计算和智能系统公司 UTS 中心的知识架构实验室主任。Kennedy 博士从 1989 年开始其跨越工业界和学术界的职业生涯，专注于开发软件。他曾经使用包括 C/C++ 和 Python 在内的多种语言完成了不同领域的软件开发，诸如计算机图形、人工智能、信息生物学及数据挖掘。在最近 10 年中，他主要的工作是教授软件工程和数据挖掘的大学及研究生课程。他于 1998 年完成其计算机科学的博士课程，并且常常作为工业界数据挖掘项目的技术顾问。他是 2006-2008 年澳洲数据挖掘协会的主席，曾经积极向国际编程委员会投稿，参与国际杂志的评审，并且著有 30 种出版物。

Wendy Langer 最早在玩 Hunt the Wumpus 和 Colossal Caves'游戏的间隙学习了 Microbee Basic 编程，这是很久以前的事了。许多年后，她在大学的物理系里学习了 Fortran。最终，在长期徘徊于黑暗中之后，她终于发现了完美的编程语言——Python。尽管现在很多时间还是花在 C++编程上，但是她的心始终属于 Python。

作为一名 Web 开发人员，她使用过 Python、Zope、Django、MySQL 和 PostgreSQL 等技术。她也是 Packt 公司出版的由 Ayman Hourieh 编写的 *Learning Website Development with Django* 一书的技术评审。

我要感谢我的母亲及小狗 Jesse，他们保护我在评审本书时免遭许多本地危险物种（如负鼠、猫和邮差）的攻击。

前　言

Python 很棒！

从 20 世纪 80 年代末出现的最早版本到当前的版本，它一直遵循着相同的理念不断发展：提供一个强调可读性和生产力的多范式语言。

人们曾经将 Python 看作一种新的脚本语言，认为不应该用它来建立大型系统。但是随着岁月流逝，在一些公司的努力下，显然，Python 可以用于构建几乎所有类型的系统。

实际上，许多其他语言的开发人员也醉心于 Python，并将其作为第一选择。

本书展现了作者多年构建各种 Python 应用程序的经验，包括从一两个小时就完成的很小的系统脚本，到许多开发人员历经数年编写的很大的应用程序。

它描述了开发人员使用 Python 的最佳实践。

本书名为《Python 高级编程》，这是因为它包含了一些不关注于语言本身，而更多关注于利用它的工具和技术。

换句话说，本书描述了高级的 Python 开发人员每天的工作方式。

本书内容

第 1 章介绍如何安装 Python，以确保所有读者有最接近的标准化环境。因为本书不是针对初学者的，所以本章差点被删除。但是，因为有些有经验的 Python 开发人员没有意识到这里提到的一些事情，所以最终仍然还是将它保留下来了。如果读者已经很了解这些内容，不要感到失望，因为本书其他的部分应该能够满足你的需要。

第 2 章是关于类级别以下的语法最佳实践。它将以高级的方式介绍迭代程序、生成器和描述符。

第 3 章也是关于语法的最佳实践，但是它将关注于类级别之上。

第 4 章是关于如何选择好名称的。这是用命名最佳实践对 PEP8 的扩展，还给出了一些如何设计良好 API 的提示。

第 5 章说明了编写包和使用模板的方法，然后关注于发行和分发代码的方法。

第 6 章是第 5 章的扩展，描述了编写完整应用程序的方法。它通过一个小的 Atomisator

案例进行示范。

第 7 章的主题是 zc.buildout，这是一个用于管理开发环境和发行应用程序的系统，其广泛地用于 Zope 和 Plone 社区，现在也开始在 Zope 世界之外使用了。

第 8 章介绍了对项目代码库管理的一些深入观察，并说明了建立持续集成的方法。

第 9 章介绍通过迭代和增量方法管理软件生命周期的方法。

第 10 章的主题是文档，并且给出了一些关于技术协作和 Python 项目文档的提示。

第 11 章阐述了测试驱动开发及其所用的工具。

第 12 章是关于优化的，给出了剖析技术和优化策略指南。

第 13 章是对第 12 章的扩展，提供了一些使程序运行更快的解决方案。

第 14 章用一组有用的设计模式结束本书。

最后，大家请密切关注http://atomisator.ziade.org，这是为本书英文版构建的网站。它拥有本书中所有的代码，以及勘误表和其他额外软件。

本书所需环境

本书是为在 Linux、Mac OS X 或 Windows 之下工作的开发人员编写的。第 1 章中介绍了所有的先决条件，以确保系统能够启用 Python 并满足基本需求。

这对于 Windows 开发人员很重要，因为他们需要确保拥有与 Mac OS X 和 Linux 用户所拥有的相近的命令行环境。一般来说，所有示例都可以在任何平台上工作。

最后请记住，本书不是用于代替在线资源的，而是用于补充它们。所以，需要通过所提供的互联网链接来完成某些方面的延伸阅读。

本书读者

本书是为希望进一步精通 Python 的开发人员编写的。本书的某些部分（如持续集成）是面对项目领导者的。

本书是对讲解“如何进行 Python 编程”的常规参考书和在线资源的补充，并且更深入地讲解了语法的使用。

本书还说明了敏捷编码的方法。虽然这适用于任何语言，但本书更聚焦于 Python 实例。所以，如果没有实施测试或者使用版本控制系统，将可能通过本书学到许多甚至在其他语言上都有帮助的内容。

从测试驱动开发到分布式版本控制系统和持续集成，读者将学习到大项目上有经验的

Python 开发人员所使用的最新编程技术。

虽然这些主题都在快速发展着，但是本书不会过时，因为它关注于为什么而不是具体怎么做。

所以，即使书中介绍的特定工具已经不再使用，但仍能理解它为什么有用，并能够以专业的眼光选择一个正确的工具。

阅读须知

本书中有许多用于区分不同信息的文本样式。以下是一些样式的示例及其意义的解释。

代码文本如下所示。这个环境可以用 buildout 命令建立。

```
>>> from script_engine import run
>>> print run('a + b', context={'a': 1, 'b':3})
4
```

命令行输入和输出如下所示：

```
$ python setup.py --help-commands
```


警告或重要的注释。


技巧和决窍。

下载本书的示例代码

本书示例代码的下载链接是http://www.packtpub.com/files/code/4947_Code.zip。

所下载的文件中提供了使用说明。

作者的网站http://atomisator.ziade.org上也有本书中提到的代码。

目　录

第 1 章

准备工作

Python 是一种对开发人员很有价值的语言。

不管你或你的客户使用的是什么操作系统，它都能够正常工作。除非使用平台相关的功能，或者使用了特定平台的程序库，否则就可以跨平台使用。例如，可以将在 Linux 环境下开发的程序部署到安装了其他操作系统的平台上。不过，这并不是其独有的特性（Ruby、Java 等许多语言都能够做到）。综合考虑本书介绍的其他功能，Python 将是公司首选开发语言之一。

本章将介绍开始使用 Python 之前必须掌握的知识，不管使用的是哪种运行环境。主要包括：

- 如何安装 Python；
- 如何使用并增强其命令行；
- 如何通过安装 setuptools 扩展 Python；
- 如何配置开发环境，包括老款或新型方法。

如果你对 Python 很熟悉，并且安装了自己喜欢的代码编辑器，那么建议你跳过本章的 1.1 节，直接阅读其他章节，在那里可以找到一些增强编程环境的小技巧。不过，请一定不要跳过关于 setuptools 的小节，对于本书其他章节而言它是必须安装的工具。

如果使用的是 Windows 操作系统，那么必须确保安装了本章中介绍的所有软件，运行本书中提供的所有示例都将依赖于这些环境。

1.1 安装 Python

Python 语言能够在各种操作系统中使用，包括 Linux、Macintosh 和 Windows。在 Python

网站的主下载页面 http://www.python.org/download 中可以找到由 Python 核心开发团队提供的各种版本。针对其他平台的版本是由社区中的其他开发人员维护的，在“贡献（dedicated）”页面中有相关信息（参见 http://www.python.org/download/other）。在此，甚至可以找到早年所用操作系统的版本。

如果有一台电脑，就可以使用 Python，而不管这台电脑上安装的是什么操作系统。

否则，将其丢在一边吧。

在安装 Python 之前，先来看看各种现有的实现版本。

1.1.1　Python 实现版本

Python 的主实现版本是用 C 语言编写的，被称为 CPython。当人们提到 Python 时，通常指的就是它。随着该语言的不断演化，C 语言实现也随着改变。除了 C 语言实现之外，Python 也提供了其他的实现版本，以保持对主流的跟踪。这些实现版本通常比 CPython 要落后几个里程碑，但也为 Python 在特定环境中的使用和宣传提供了巨大的机会。

1．Jython

Jython 是 Python 语言的 Java 实现版本。它将其代码编译成 Java 字节码，因此开发人员可以在 Python 模块（在 Python 中，存储代码的文件被称为模块）中自由地使用 Java 类。Jython 让大家可以在诸如 J2EE 之类的复杂应用程序上，将 Python 作为顶层脚本语言使用。同时，也可以将 Java 应用程序引入到 Python 应用程序中。使 Apache Jackrabbit（一种基于 JCR 实现的文档仓库 API 能够应用于 Python 程序，就是 Jython 存在价值的一个良好示例。Jython 的当前版本是 2.2.1，Jython 开发团队正在研发 2.5 版本。现在许多诸如 Pylons 之类的 Python Web 框架正在通过 Jython 被移植到 Java 世界中。

2．IronPython

IronPython 将 Python 引入了.NET 环境。该项目是由微软提供支持的，IronPython 的主开发人员供职于该公司。其最新的稳定版本是 1.1（发布于 2007 年 4 月），它是对 Python 2.4.3 版本的实现。它能够在 ASP.NET 环境中使用，开发人员在.NET 应用程序中使用 Python 代码的方法和在 Java 环境中使用 Jython 一样。它对于该语言的推广起着十分重要的作用。.NET 社区和 Java 社区一样，都是最大的开发人员社区之一。TIOBE 社区索引也显示了.NET 语言

是一颗冉冉升起的新星。

3．PyPy

PyPy 或许是各种实现版本中最有趣的一款，其目标是用 Python 语言重写 Python。在 PyPy 中，Python 解释程序本身就是用 Python 编写的。对于 Python 实现版本之一的 CPython 而言，需要一个 C 代码层来承载具体的工作。但在 PyPy 实现版本中，这个 C 代码层将彻底用纯 Python 语言重写。这就意味着可以在运行态时改变 Python 解释程序的行为，以及实现一些在 CPython 实现版本中难以实现的代码模式（参见 http://codespeak.net/pypy/dist/pypy/doc/objspace-proxies.html）。PyPy 的运行速度远低于 CPython，不过在近几年中有了很大改善。随着诸如 JIT 编译器之类的技术的引入，其运行速度的提升是很有希望的。它当前的速度因子大概在 1.7～4 之间，针对的目标版本是 Python 2.4。PyPy 可以被看作汇编程序领域开发的领军项目，它在许多领域的创新都是先驱，其他主流的实现版本必将会从中受益。总体来看，PyPy 的开发更多是出于学术研究的动机，关注该项目的都是那些热衷于深入研究语言内部机制的人。因此，通常不会在具体的产品中使用它。

4．其他实现版本

Python 还有许多其他实现版本和移植版本。例如，Nokia 就在其 S60 系列手机上实现了 Python 2.2.2。Michael Lauer 开发维护了一个应用于 ARM 芯片上的 Linux 环境中的 Python 移植版本，使其能够在诸如 Sharp Zaurus 之类的设备上使用。

这样的例子还有很多，不过本书关注的是在 Linux、Windows 和 Mac OS X 上安装 CPython。

1.1.2 在 Linux 环境下安装

如果使用的是 Linux 操作系统，那么或许已经安装了 Python。因此，可以试着在 shell 环境中调用它，如下所示。

```
tarek@dabox:~$ python
Python 2.3.5 (#1, Jul 4 2007, 17:28:59)
[GCC 4.1.2 20061115 (prerelease) (Debian 4.1.1-21)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

如果找到该命令，那么将进入 Python 提供的交互式 shell 环境，其提示符为>>>。同时还将显示编译器相关的信息，包括编译 Python 的语言（在此是 GCC）以及目标系统环境（在此是 Linux）等信息。如果使用的是 Windows，那么可以用微软公司的 Visual Studio 作为编