信息科学与技术丛书

主　编　张燕妮
副主编　丁维才
参　编　张秀凤

Python 即学即用

- 科技人员的编程语言
- 跨平台的开发环境
- 快速开发 GUI 程序
- 处理大数据的利器
- 提供针对 Python 2/3 的代码

机械工业出版社
CHINA MACHINE PRESS

信息科学与技术丛书

Python 即学即用

主　编　张燕妮
副主编　丁维才
参　编　张秀凤

机械工业出版社

本书从易用性角度介绍了 Python 编程，分为 Python 基本内容和高级话题两大部分。其中，基本内容主要包括 Python 数据类型、控制流程、文件、类、模块、网络编程、正则表达式、GUI 和数据库访问等；在每一章的基本内容基础上加以延伸，引出对应的高级话题，分别介绍了 Matplotlib、NumPy、SciPy、Flask、PyQt、ORM 等优秀的 Python 软件包。最后介绍了大数据常用工具（JSON、XML、HDF5、pandas）。本书是以即学即用的方式进行讲解的，读者可在每章学习之后应用该章的知识解决实际工作中的问题。

本书适合 Python 初学者、Web 软件开发人员及数据分析工程师，也适合高等院校的计算机教学。

书中的实例代码（分别针对 Python 2 和 Python 3）可免费下载。

图书在版编目（CIP）数据

Python 即学即用 / 张燕妮主编. —北京：机械工业出版社，2016.5
（信息科学与技术丛书）
ISBN 978-7-111-53989-6

Ⅰ. ①P… Ⅱ. ①张… 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（2016）第 128049 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
责任编辑：车 忱
责任校对：张艳霞
责任印制：李 洋

三河市宏达印刷有限公司印刷

2016 年10月第 1 版 • 第 1 次印刷
184mm×260mm • 16.75 印张 • 398 千字
0001—3000 册
标准书号：ISBN 978-7-111-53989-6
定价：50. 00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
服务咨询热线：（010）88361066
读者购书热线：（010）68326294
（010）88379203

网络服务
机 工 官 网：www.cmpbook.com
机 工 官 博：weibo.com/cmp1952
教育服务网：www.cmpedu.com
金 书 网：www.golden-book.com

出 版 说 明

随着信息科学与技术的迅速发展，人类每时每刻都会面对层出不穷的新技术和新概念。毫无疑问，在节奏越来越快的工作和生活中，人们需要通过阅读和学习大量信息丰富、具备实践指导意义的图书来获取新知识和新技能，从而不断提高自身素质，紧跟信息化时代发展的步伐。

众所周知，在计算机硬件方面，高性价比的解决方案和新型技术的应用一直备受青睐；在软件技术方面，随着计算机软件的规模和复杂性与日俱增，软件技术不断地受到挑战，人们一直在为寻求更先进的软件技术而奋斗不止。目前，计算机和互联网在社会生活中日益普及，掌握计算机网络技术和理论已成为大众的文化需求。由于信息科学与技术在电工、电子、通信、工业控制、智能建筑、工业产品设计与制造等专业领域中已经得到充分、广泛的应用，所以这些专业领域中的研究人员和工程技术人员越来越迫切需要汲取自身领域信息化所带来的新理念和新方法。

针对人们了解和掌握新知识、新技能的热切期待，以及由此促成的人们对语言简洁、内容充实、融合实践经验的图书迫切需要的现状，机械工业出版社适时推出了“信息科学与技术丛书”。这套丛书涉及计算机软件、硬件、网络和工程应用等内容，注重理论与实践的结合，内容实用、层次分明、语言流畅，是信息科学与技术领域专业人员不可或缺的参考书。

目前，信息科学与技术的发展可谓一日千里，机械工业出版社欢迎从事信息技术方面工作的科研人员、工程技术人员积极参与我们的工作，为推进我国的信息化建设做出贡献。

机械工业出版社

前　言

Python 是一种面向对象、解释型计算机程序设计语言，其语法简洁清晰、易于学习，几乎可以在所有的操作系统下运行。Python 常被称为“胶水”语言，因为它能够把不同语言编写的各个模块轻松地组织在一起，例如将众多优秀的 Fortran 和 C 语言库集成到 Pyhton 环境下，帮助开发者处理各种工作。Python 的优秀特性决定了其在实际应用中的广泛性，在很多领域如快速原型开发、网络服务器脚本、科学计算、文本处理、数据库编程、嵌入开发、GUI 开发、游戏开发和移动开发中均有广泛应用。

目前 Python 语言越来越受到重视，并已有大量成功的案例，如 YouTube（视频分享网站）、豆瓣（社区网站）、OpenStack（云计算平台）和 Tornado（Web 服务器）等都是基于 Python 开发的。

本书既介绍了 Python 的基础知识，也介绍了很多 Python 的高级话题，并附有实例，是一本即学即用的书。本书首先介绍了 Python 的数据类型、编程语法、函数、类和模块等基础知识，然后介绍了 Python 在网络、数据库、正则表达式和大数据方面的应用。每一章的最后都介绍了与该章内容相关的高级话题，这些高级话题可直接在数据处理、网站开发和数据库管理等领域使用，使得读者每学习一章即可通过该章内容解决工作、科研中的实际问题，充分体现了即学即用特点，突破了以往必须将书读完才能用于实战的思路。高级话题涵盖了大数据分析用的 NumPy、SciPy、PyTables 和 pandas 等工具，讲解了如何采集数据以及如何为调研报告生成漂亮的图表等内容。书中的案例采用实际项目使用的小测试案例，具有极强的实用性。

下面是各章内容简要介绍：

第 1 章是 Python 的基础介绍，讲解了 Python 的特点，如何安装和使用 Python，介绍了 Python 的常用绘图工具——Matplotlib。

第 2～4 章分别讲解了 Python 的数据类型、程序结构和函数，并在此基础上介绍了 NumPy 和 SciPy 的用法，NumPy 是开源的 Python 科学计算库，对数组和矩阵的支持使其成为 Python 科学计算软件的基础。SciPy 基于 NumPy，提供了科学计算的工具集软件，涉及统计、优化、积分、线性代数模块、傅里叶变换、信号和图像处理、常微分方程求解器等。书中介绍了 NumPy 的数据类型、通用函数及如何利用 SciPy 进行科学运算。

第 5 章讲解了 Python 的类文件以及文件系统操作，还介绍了读写 Excel 文件的三个 Python 库：xlwt、xlrd、xlutils。

第 6 章首先讲解了 Python 的模块与包。模块与包是 Python 代码的组织基础，良好的组织结构可增强代码的健壮性。然后讲解了如何通过包和模块发布 Python 程序。

第 7 章讲解了 Python 中面向对象编程的知识。首先讲解了类、实例、属性和继承多态等用法，通过 Python 实现模板、代理设计模式的方式介绍了类的使用，使读者能够更加清晰地了解如何应用类，如何设计自己的程序框架，最后讲解了面向对象的高级话题——抽象基类。

第 8 章和第 9 章采用相同的数据模型（todo 表）进行讲解。第 8 章首先介绍了 Python 的

DB-API 2.0 接口，接着介绍了 PostgreSQL 以及 MySQL 数据库的读写操作，最后讲解了 ORM 的使用。第 9 章首先从 socket 开始，介绍了如何构建 HTTP 访问程序，进而扩展到 CGI 程序的开发，为使用 Web 框架提供了基本概念，最后介绍了微型的 Web 框架 Flask，Flask 一节使用了与第 8 章相同的 todo 数据库。

第 10 章讲解了正则表达式，并介绍了 Beautiful Soup 包。

第 11 章首先讲解了 Tkinter，Tkinter 作为 Python 标准自带的 GUI 设计程序，可用于小型的 GUI 设计。然后讲解了专业级的 GUI 设计程序 PyQt，PyQt 是对 Qt 的 Python 包装，不仅包含跨平台的 Qt GUI 设计程序，而且包含了视频、网络、数据库等功能模块。

第 12 章是关于 Python 在数据分析领域中的应用。首先讲解了常用的数据采集方式（JSON、XML）。目前 JSON 作为一种基于文本的数据交换格式，广泛应用于互联网领域，所以掌握了 JSON 就很容易获取互联网领域的数据。XML 被设计用来传输和存储数据，不仅互联网领域使用 XML，而且大量软件也支持 XML，例如微软的 Office 现在已经使用 Open Office XML（以 XML 和 ZIP 为基础构建）代替原有的二进制格式文件。12.2 节讲解了如何使用 Python 处理 XML 数据。12.3 节介绍了 HDF5 接口，HDF5 是一种不同于数据库存储方式的数据存储，用于存储和分发科学数据的一种自我描述、多对象文件格式。12.4 节介绍了 pandas，pandas 已经成为 Python 数据分析的标准工具，广泛用于时间序列数据领域。

附录讲解了如何安装/编译相应的 Python 版本，以及如何通过 virtualenv 实现多个独立的 Python 运行环境，并给出了 Python 在一些学科领域的优秀软件包介绍，读者可根据需要选用相应的软件包。

本书使用 Anaconda Python 作为开发环境。Anaconda 是 Python 的科学技术包的合集，包含了大量的科学计算包，如 NumPy、SicPy 和 Matplotlib 等，并支持 Windows、Linux、OS X 环境。相比其他 Python 集成开发环境，Anaconda 不仅支持 Python 2.X，而且支持 Python 3.X 的科学计算包。可从 Anaconda 的官网（https://www.continuum.io/downloads）下载相应版本的 Anaconda。如果 Anaconda 未包含书中所用的模块，可参考第 1 章介绍的 pip 和 easy_install 的方法安装相应模块。本书以 Python 2 为主进行讲解，但同时提供了 Python 2 和 Python 3 下的代码，便于读者学习。

本书的第 6 章由张秀凤编写，第 10 章由丁维才编写，其余内容由本人编写。写书过程中，经常忽视女儿的好玩天性，没能很好地陪伴女儿，心有愧疚。谨以此书献给我的女儿和所有关心支持我的人。

张燕妮

目　录

第1章 绪　论

软件发展到今天，人们可以不断借助于各种已有软件完成科研、商务和日常工作。软件的发展和软件开发技术的普及也促使开发模式的转变：原先接到项目工程后，需集中软件开发人员封闭开发，但现在人人都可以根据自己需要编写小段程序进行数据处理，软件开发从最开始的阳春白雪转变成大众化的工作。

现在专门从事软件开发的人员越来越多，多数情况下数据及信息的处理，如处理从实验室、工业现场获得的数据以及进行社会调研、信息发布等，都可以委托专业软件人员去做。但有时各种客观条件限制，需要数据采集者能够对这些数据、信息根据自己的思想去筛选并完成初步处理，这就需要非软件开发人员自己动手编写程序。

在众多程序设计语言中，C/C++应用面广、目标程序效率高，但不容易上手，其艰辛的学习过程，使得很多学习者迷失在前进的道路上，难以使用 C/C++完成项目开发，更谈不上使用 C/C++实现自己的想法。Java 在开发难度上虽然有所降低，但相对于脚本语言，当仅仅需要对一个微分方程进行求值时，Java 又显高大了。MATLAB 在数据处理、原型开发、科学研究中具有独到之处，也适合科研以及模型处理，但从版权考虑，它又过于昂贵。微软的 Visual Basic 以及 C#虽然方便易用，但具有平台局限性。现在更多的开发者希望选用不受操作系统限制的开发语言。

在这种情况下，越来越多的程序员和科研人员把目光投向了 Python。Python 不仅具有平台无关性，还具有简单易用、开发效率高的优点，并且具有面向对象编程的特点，相关开发工具唾手可得，在很多科学领域、互联网领域均有相应的库支持。

1.1 Python 的特点

在 Python 的环境中输入“import this”，可看到 Python 的设计哲学，即 Tim Peters 写的 The Zen of Python。在此可看到 Python 的简洁性、可读性、明确性等设计思想。Python 不像许多语言那样使用“{}”(例如 C/C++、C#、Java)或者“Sub…End Sub”（Visual Basic.net）的组合去标注语法，它使用了缩进方式，而且这个缩进要求很严格。但也正因为缩进的要求，使得代码可以时隔多日，仍然清晰明了，而不用去猜测自己当初写的这段代码的功能。

虽然 Python 的缩进功能很强大，但实际操作是很简单的。行首的空白（可以是空格和制表符）即为缩进，空白的长度用于确定逻辑行的缩进层次，从而决定语句的分组。用于表示缩进等级的空格个数不可随意改变，并且空格与制表符不可混用。在 PEP8（Python 编码规范）中推荐使用 4 个空格的缩进风格。除非使用记事本之类的工具编写 Python 程序，否则只要是支持 Python 语法输入的编辑器均有快捷输入 4 个空格（不需要连续输入 4 次空格）的方法。

1.1.1 为何适应各种用户需求

1．效率高

相对于 C、C++和 Java 等编译/静态类型语言，Python 的开发效率提高了数倍。要完成同样的工作，Python 代码的长度往往只有 C++或者 Java 代码的 1/5～1/3，这意味着可以录入更少的代码、调试更少的代码并在开发完成后维护更少的代码。并且 Python 程序可以编辑后立即执行，无需传统编译/静态语言所必需的编译及链接等步骤，进一步提高了程序开发效率。

2．可移植性好

Python 很重视程序的可移植性，可以设置包括程序启动和文件夹处理等操作系统接口。绝大多数 Python 程序不做任何改变即可在多数计算机平台上运行。例如在 Linux 和 Windows 之间移植 Python 代码，只需要简单地在计算机间复制代码即可。

3．原型设计转换方便

可以使用 MATLAB 做一些产品设计和算法设计，但从实验室的设计过渡到产品设计需要一个过程，这个过程多数情况是转换成 C/C++代码。这个转换过程往往是非常艰辛的，有时比 MATLAB 前期设计还费时费力（因为 MATLAB 已包含的算法，在转换成 C/C++时可能需要从头做起）。但如果直接使用 Python 做原型设计，该过程就不一样了，因为 Python 的基础库大部分是基于 C/C++的软件。这些软件例如 OpenCV、VTK 以及后文提到的 Sage，提供了 Python 接口，Python 可以方便地调用它们实现各种功能。因此使用 Python 做原型开发，不仅速度快，而且在转换成 C/C++产品时，也是比较方便的。

1.1.2 胶水特点

Python 很容易连接各种编译库，这是它作为胶水语言而流行多年的重要原因。标准的 Python 称为 CPython，除了可以通过 C 语言直接调用 Python API 进行扩展，还有 Swig、Boost 之类的工具可以对 Python 进行扩展，例如 VTK 就是利用 Swig 实现了 Python 接口，使得 Python 用户可直接调用 VTK。而且 Python 语言本身具有多个实现版本，例如使用 Java 实现的 JPython。这种实现版本使得 Python 具有访问 Java 包、C#库的能力。也正是因为胶水特点，使得 Python 在许多开源软件中具有相同的 Python 调用接口。

Sage 充分体现了 Python 的胶水特点。Sage 是一个基于 GPL 协议的开源数学软件，将现有的许多开源软件包整合在一起，构建一个使用 Python 作为通用接口的统一计算平台，使用高度优化的成熟软件，如 GMP，PARI，GAP 和 NTL，目标是在代数、几何、数论、微积分、数值计算等领域提供可用于探索和尝试的软件。

1.1.3 语言特点

Python 相比 C/C++之类语言，缺少了变量声明、变量定义的过程。Python 在运行过程中可跟踪对象的类型，同一变量名也可直接被赋值为新的数据类型，即 Python 是动态类型的，例如：

```
>>> a=1
>>> a=u"新的数值"
>>> a
```

```
"新的数值"
```

如果是 C 语言，情况就不同。例如 char a='1';a=0x32;实际上 a 就是 0～255 之间的一个数值，同时对应 ASCII 码 0～255 之间的某个字符。即使通过 a=0x32 的赋值，也没有改变 a 的数据类型。

许多语言如 C#、Java 等具有自动垃圾回收机制，Python 也是如此，Python 采用引用方式自动进行对象分配，当对象不再使用时自动执行垃圾回收。而 C++通过 new 创建新的变量之后，必须有对应的 delete，否则会造成内存泄漏。

Python 中万物皆对象，如数值、字符串、数据结构、函数、类、模块等。每个对象都有一个与之相关的属性和方法。例如所有的函数都有一个内置的__doc__属性，它会返回在函数源代码中定义的文档字符串。又如 sys 模块是一个对象，有一个 version 属性，可用来显示 Python 版本信息。

```
>>> import sys
>>> sys.version
'3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)]'
```

即使是 C 语言中的一些基本数据类型（字符型、整型、浮点型数据），在 Python 中也都是对象。例如数值 1，是 int 类的实例，并且有__add__方法。

```
>>> type((1))
<class 'int'>
>>> (1).__class__
<class 'int'>
>>> (1).__add__(2)
3
```

Python 的数据是鸭子类型（Duck Typing），是指对象的类型不是主要的，对象是否包含相应的方法或者属性才是主要的。Python 中的文件对象是典型的鸭子类型，即只要含有 read()或者 write()的方法对象，均可当作文件类型进行数据处理。

相比 C/Java 语言，函数的返回值可以是多个。例如：

```
>>> def f():
...     a=1
...     b=2
...     c=3
...     return a,b,c
>>> f()
(1, 2, 3)
>>> a,b,c=f()
>>> print (a,b,c)
1 2 3
```

1.1.4　语法风格

Python 不需要显式声明变量，变量在第一次被赋值时自动声明。这是与 C/C++、Java 语

言的变量定义的不同之处。

Python 是区分大小写的，标识符的第一个字符必须是字母或者下画线“_”，其余字符可以是字母和数字或者下画线。

Python 中用下画线作为变量前缀和后缀指定特殊变量。_xxx_代表系统定义名字，_xxx 用于类中的私有变量名。因此普通变量不推荐使用下画线做前缀。

Python 的注释分为两种。一种是以#字符开头的注释，注释语句从#字符开始，直到该行结束。注释可以在一行的任何地方开始，解释器会忽略该行#之后的所有内容。例如：

```
#本行是注释
print “hello world”
#打印结束
```

一种是叫作文档字符串（docstring）的特殊注释。可以在模块、类或者函数的开头，使用单引号、双引号、三引号（用于多行文字情况）添加一个字符串，起到在线文档的作用，常用于说明如何使用这个包、模块、类、函数（方法），甚至包括使用示例和单元测试。与普通注释不同，文档字符串可以在运行时访问，也可以用来自动生成文档。

文档字符串出现的位置包括以下几种：

（1）包的 docstring 位于包内的 init.py 文件的开头。

（2）模块的 docstring 位于模块所在文件的开头。

（3）函数的 docstring 位于函数名称所在行的下一行，函数体之前。

（4）类的 docstring 位于类的名称所在行的下一行，所有描述之前。

例如：

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
#模块的 docstring
u"""这是模块的文档字符串"""

import os
def test():
#函数的 docstring
u'''函数的文档字符串'''
print " test"
if __name__ == "__main__":
print test.__doc__#输出 test 函数的文档字符串
```

Python 中可通过__doc__特殊变量，获得文档字符串。在模块、类声明、函数声明中的第一个没有赋值的字符串可用 obj.__doc__进行访问。其中 obj 是一个模块、类或者函数的名字。

Python 使用缩进来分割代码组。代码的层次关系是通过不同深度的代码体现的。同一代码组的代码行必须严格左对齐（左边有同样多的空格或者同样多的制表符）。随着缩进深度的增加，代码块的层次也在加深，没有缩进的代码是最高层次的，称作脚本的“主体(main)”部分。缩进推荐使用 4 个空格形式。例如：

```
def cmp(a ,b):
```

```
    if a>b:#第一层缩进
        return  0#第二层缩进
    else : #第一层缩进
        return  1#第二层缩进
cmp(1,2) #最高层
```

1.1.5　多平台

在很多操作系统里，Python 是标准的系统组件。大多数 Linux 发行版以及 NetBSD、OpenBSD 和 Mac OS X 都集成了 Python，可以在终端下直接运行 Python。有一些 Linux 发行版的安装器使用 Python 语言编写，比如 Ubuntu 的 Ubiquity 安装程序、Red Hat Linux 和 Fedora 的 Anaconda 安装程序。Gentoo Linux 使用 Python 来编写它的 Portage 包管理系统。Python 标准库包含了多个调用操作系统功能的库。通过 PyWin32 这个第三方软件包，Python 能够访问 Windows 的 COM 服务及其他 Windows API。使用 IronPython，Python 程序能够直接调用.NET Framework。一般说来，Python 编写的系统管理脚本在可读性、性能、源代码重用度、扩展性几方面均优于普通的 shell 脚本。

Python 除了在 UNIX/Linux、Windows 下有相应实现版本，在 AS/400 (OS/400)、BeOS、MorphOS、MS-DOS、OS/2、OS/390、z/OS、RISC OS、Series 60、Solaris、VMS、Windows CE（或 Pocket PC）、HP-UX 系统下均有相应实现版本。

1.1.6　丰富的支持

Python 是免费、开放的，在 www.python.org 可以下载 Python 源代码。用户也可进行源代码的修改、发布。

当然免费不代表没有技术支持。www.python.org 提供了 Python 的技术支持。除了 Python 语言本身的支持，还有大量有关 Python 的第三方软件。在 1.2 节将给出有关 Python 的版本与集成包内容。如果有人能够将 Python 的源代码进行包装，并加上一些吸引用户的功能，也可做出一个商业化的 Python，还不用担心版权问题。这充分体现了 Python 的开放性。

Python 的标准库是不断发展的。一些原本优秀的第三方库，例如 ctypes,PyUnit（现在改名为 unittest）随着 Python 的发展，逐渐成为标准库。正因为 Python 的开放性才使得 Python 的第三方库非常多，进而促使 Python 越来越强大，应用面越来越广。

1.2　Python 版本与集成包

我们经常讨论的 Python 是指 CPython（即从 www.python.org 上下载的 Python 版本），除了 CPython，还有一些其他的实现版本。

- IronPython 是一种在.NET 和 Mono 上实现的 Python 语言。
- Jython 的原名叫 JPython, 是 Python 编程语言的纯 Java 实现。它可以让用户将 Python 源代码编译成 Java 字节码，并在任何 Java 虚拟机上运行产生的字节码。它是与 Java 的最无缝、最平滑的集成。可以从 Jython 中访问所有 Java 库、构建