



普通高等教育
软件工程

“十三五”规划教材



工业和信息化普通高等教育
“十三五”规划教材

13th Five-Year Plan Textbooks
of Software Engineering

Python 3 程序设计

唐永华 刘德山 李玲 主编

*Python 3
Programming*



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS



普通高等教育
软件工程

“十三五”规划教材



工业和信息化普通高等教育
“十三五”规划教材

13th Five-Year Plan Textbooks
of Software Engineering



Python 3 程序设计

唐永华 刘德山 李玲 ◎ 主编

Python 3
Programming

RFID

人民邮电出版社

北京

图书在版编目(CIP)数据

Python 3程序设计 / 唐永华, 刘德山, 李玲主编

— 北京: 人民邮电出版社, 2019.2

普通高等教育软件工程“十三五”规划教材

ISBN 978-7-115-49879-3

I. ①P… II. ①唐… ②刘… ③李… III. ①软件工
具—程序设计—高等学校—教材 IV. ①TP311.561

中国版本图书馆CIP数据核字(2018)第242069号

内 容 提 要

本书以通俗易懂的语言、翔实的示例、新颖的内容诠释了 Python 这门“简单”“优雅”“易学”的计算机语言。全书共 14 章, 第 1 章~第 10 章介绍了 Python 语言基础, 覆盖了全国计算机等级考试 Python 语言的主要内容; 第 11 章和第 12 章介绍了 Python 语言的应用, 包括图形用户界面、数据库编程等内容; 第 13 章和第 14 章介绍了 Python 第三方库的应用。本书内容以应用为核心展开, 力求以知识的最小集来实现最大范围的应用。

本书难度适中, 主要面向普通本科院校非计算机专业的学生, 也可作为初学者学习 Python 程序设计课程的教材, 或者作为参加全国计算机等级考试的 Python 教材。

-
- ◆ 主 编 唐永华 刘德山 李 玲
责任编辑 邹文波
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>
山东百润本色印刷有限公司印刷
 - ◆ 开本: 787×1092 1/16
印张: 16.75 2019 年 2 月第 1 版
字数: 438 千字 2019 年 2 月山东第 1 次印刷
-

定价: 49.80 元

读者服务热线: (010)81055256 印装质量热线: (010)81055316

反盗版热线: (010)81055315

前言

近年来,人工智能(AI)已经上升为国家战略。Python丰富的AI库、机器学习库、自然语言和文本处理库,使其成为最适用于人工智能的语言之一。此外,Python还可应用于数据分析、组件集成、图像处理、科学计算等众多领域。

为适应跨界创新的需求,不同层次、不同专业的读者迫切需要一种可以更多专注于解决的问题,而不必更多考虑细节的计算机语言,让计算机语言回归服务的功能,Python就是最佳的选择。

Python以其“简单”“优雅”“明确”“易学”的特性成为学习编程的入门级语言。超过十几万的第三方库,形成了Python的“计算生态”,推动了Python的发展。

Python在业界得到了广泛的应用,几乎所有大中型互联网企业都在使用Python,如YouTube、豆瓣、知乎、Google、Yahoo、Facebook、百度、腾讯、美团等。

面对诸多的应用需求,以及Python适合于所有专业学生学习的特点,2018年,教育部将Python纳入了全国计算机等级考试范围,相信在未来的几年里,Python将得到更好的普及与发展。

编者从教学实践中精选了大量的示例,让读者能全面地了解和学习这门“简单”“易学”的语言。编写本书的各位教师曾主讲C、Java、Python等课程,他们从“实用、易用、有效”的角度组织内容,以应用为核心展开,力求以知识的最小集来实现最大范围的应用。

本书主要具有以下特色。

(1) 内容重点突出。在保证内容科学、完整的前提下,由浅入深地安排章节次序。考虑到Python语言的应用特色,与其他程序设计教材相比,本书更强调应用思维。

(2) 案例资源丰富。全书设计了240个示例,内容基本覆盖Python的所有知识要点。还提供相应的教学课件、程序源码,有需要的读者可前往人邮教育社区(www.ryjiaoyu.com)下载。

(3) 在教材内容上,协调了与全国计算机等级考试和Python应用需求的关系。书中的知识点基本覆盖了等级考试的核心内容,并删减了部分使用频率较低的内容。

本书建议教学的组织形式是“示例—分析—练习—总结”。从应用的角度介绍语言,通过示例来说明编程的方法和过程。建议授课48学时,第11~14章的内容可根据需要选讲,书中标注*号的章节可以略讲,这部分内容不影响Python的学习和参加全国计算机等级考试。

本书由唐永华、刘德山、李玲主编,若书中存在疏漏和不足之处,恳请读者批评指正。48学时不长,稍纵即逝;48学时不短,您可以学习和发挥Python的“神奇”所在。

注意,软件版本与下载页面在不断更新,读者打开的下载界面和看到的软件可下载版本可能与本书的不一样,但下载与安装的方法类似。

编者

2018年12月

目 录

第 1 章 初识 Python.....1

1.1 程序设计语言.....1

1.1.1 程序设计语言的概念.....1

1.1.2 编译与解释.....2

1.2 Python 语言.....2

1.2.1 Python 的历史.....2

1.2.2 Python 的特点.....3

1.2.3 Python 的应用.....3

1.3 Python 的开发环境.....4

1.3.1 下载和安装 Python.....4

1.3.2 内置的 IDLE 开发环境.....6

1.3.3 PyCharm 集成开发环境.....7

1.4 Python 程序的运行.....10

1.4.1 Python 程序的运行原理.....10

1.4.2 建立和运行 Python 程序.....11

1.5 Python 编程方法与应用.....13

1.5.1 程序设计方法.....13

1.5.2 程序示例.....13

1.5.3 Python 的帮助文档.....17

本章小结.....18

习题 1.....18

第 2 章 Python 基础知识.....20

2.1 程序的书写规范.....20

2.1.1 Python 的语句.....20

2.1.2 代码块与缩进.....21

2.1.3 注释.....21

2.2 标识符和关键字.....22

2.2.1 标识符.....22

2.2.2 关键字.....22

2.3 Python 的数据类型.....23

2.4 Python 的变量.....25

2.5 Python 的运算符.....25

2.5.1 算术运算符.....26

2.5.2 比较运算符.....26

2.5.3 逻辑运算符.....26

2.5.4 赋值运算符.....27

2.5.5 位运算符.....27

2.6 运算符的优先级.....28

本章小结.....29

习题 2.....29

第 3 章 Python 中的字符串.....31

3.1 字符串的表示.....31

3.2 字符串的格式化.....32

3.2.1 用%操作符格式化字符串.....32

3.2.2 format()方法.....33

3.3 字符串的操作符.....35

3.4 字符串处理函数.....36

3.5 输入/输出语句.....39

3.5.1 输入语句.....40

3.5.2 输出语句.....40

本章小结.....41

习题 3.....41

第 4 章 Python 程序的流程.....43

4.1 程序设计流程.....43

4.1.1 程序流程图.....43

4.1.2 结构化程序设计的基本流程.....43

4.2 分支结构.....44

4.3 循环结构.....46

4.3.1 遍历循环: for 语句.....46

4.3.2 条件循环: while 语句.....47

4.3.3 循环的嵌套.....48

4.4 流程控制的其他语句.....49

4.4.1 跳转语句.....49

4.4.2 pass 语句.....50

4.4.3 循环结构中的 else 语句.....50

4.5 流程控制语句的应用	51	6.5 Python 的内置函数	83
本章小结	52	6.5.1 数学运算函数	83
习题 4	53	6.5.2 字符串运算函数	84
第 5 章 Python 的组合数据类型	55	6.5.3 转换函数	84
5.1 序列类型	55	6.5.4 序列操作函数	84
5.2 列表	56	6.5.5 Python 操作相关函数	88
5.2.1 列表的基本操作	56	本章小结	89
5.2.2 列表的方法	57	习题 6	90
5.2.3 遍历列表	58	第 7 章 用类实现抽象和封装*	92
5.3 元组	58	7.1 面向对象编程概述	92
5.3.1 元组的基本操作	59	7.1.1 面向对象编程的概念	92
5.3.2 元组与列表的转换	59	7.1.2 面向对象编程的特点	93
5.4 字典	60	7.2 创建类与对象	93
5.4.1 字典的基本操作	60	7.2.1 创建类	94
5.4.2 字典的常用方法	61	7.2.2 创建对象	94
5.5 集合	63	7.3 构造方法和析构方法	95
5.5.1 集合的基本操作	64	7.3.1 构造方法	95
5.5.2 集合运算	65	7.3.2 析构方法	96
5.6 组合数据类型的应用	66	7.3.3 self 参数	97
本章小结	68	7.3.4 成员变量和类变量	98
习题 5	68	7.3.5 类方法和静态方法	98
第 6 章 用函数实现代码复用	70	7.4 类的继承	101
6.1 函数的定义和调用	70	7.4.1 继承的实现	101
6.1.1 函数的定义	70	7.4.2 方法重写	103
6.1.2 函数的调用	71	7.4.3 Python 的多继承	104
6.1.3 函数的嵌套	71	7.5 类的多态	105
6.2 函数的参数和返回值	72	7.6 运算符重载	106
6.2.1 函数的参数	72	7.7 面向对象编程的应用	109
6.2.2 默认参数	74	本章小结	114
6.2.3 可变参数	75	习题 7	115
6.2.4 函数的返回值	76	第 8 章 使用模块和库编程	117
6.2.5 lambda 函数	77	8.1 模块	117
6.3 闭包和递归函数	78	8.1.1 模块的概念	117
6.3.1 闭包	78	8.1.2 导入模块	117
6.3.2 递归函数	79	8.1.3 执行模块	118
6.4 变量的作用域	80	8.1.4 模块搜索路径	120
6.4.1 局部变量	80	8.1.5 __name__ 属性	121
6.4.2 全局变量	81	8.2 包	121
6.4.3 global 语句	83	8.3 Python 的标准库	122

8.3.1 math 库	122	10.1.2 异常示例	163
8.3.2 random 库	123	10.2 Python 的异常类	164
8.3.3 datetime 库	124	10.3 异常处理机制	167
8.4 Python 的第三方库	128	10.3.1 try...except 语句	167
8.4.1 第三方库简介	128	10.3.2 else 语句和 finally 语句	168
8.4.2 使用 pip 工具安装第三方库	128	10.3.3 捕获所有的异常	169
8.4.3 Python 常用的第三方库	130	10.4 抛出异常	171
8.4.4 使用 pyinstaller 库打包文件	131	10.4.1 raise 语句	171
8.5 turtle 库的应用	132	10.4.2 抛出异常示例	172
8.6 jieba 库的应用	135	10.5 断言与上下文管理	173
8.6.1 jieba 库简介	136	10.5.1 断言	173
8.6.2 jieba 库的分词函数	136	10.5.2 上下文管理	174
8.6.3 添加单词和自定义词典	137	10.6 自定义异常	175
8.6.4 基于 TF-IDF 算法的关键词抽取	138	本章小结	176
8.6.5 中文文本的词频统计	138	习题 10	176
8.6.6 打包词频统计程序	142		
本章小结	143	第 11 章 tkinter GUI 编程	179
习题 8	143	11.1 tkinter 编程概述	179
第 9 章 Python 的文件操作	145	11.1.1 第一个 tkinter GUI 程序	179
9.1 文件的概念	145	11.1.2 设置窗口和组件的属性	180
9.2 文件的打开与关闭	146	11.2 tkinter GUI 的布局管理	181
9.3 文件的读/写操作	147	11.2.1 使用 pack()方法的布局	181
9.3.1 读取文件数据	148	11.2.2 使用 grid()方法的布局	183
9.3.2 向文件写数据	149	11.2.3 使用 place()方法的布局	184
9.3.3 文件的定位读/写	150	11.2.4 使用框架的复杂布局	184
9.3.4 读/写二进制文件	151	11.3 tkinter 的常用组件	185
9.4 文件和目录操作	153	11.3.1 Label 组件	186
9.4.1 常用的文件操作函数	153	11.3.2 Button 组件	187
9.4.2 文件的复制、删除及重命名操作	154	11.3.3 Entry 组件	188
9.4.3 文件的目录操作	155	11.3.4 Listbox 组件	189
9.5 使用 CSV 文件格式读/写数据	155	11.3.5 Radiobutton 组件	191
9.5.1 CSV 文件介绍	155	11.3.6 Checkbutton 组件	193
9.5.2 读/写 CSV 文件	156	11.3.7 Text 组件	194
9.6 文件操作的应用	159	11.3.8 Spinbox 组件	195
本章小结	161	11.4 tkinter 的事件处理	197
习题 9	162	11.4.1 使用 command 参数实现 事件处理	197
第 10 章 异常处理	163	11.4.2 使用组件的 bind()方法实现 事件处理	198
10.1 异常处理概述	163	11.5 tkinter GUI 的应用	199
10.1.1 异常的概念	163		

本章小结	202
习题 11	202
第 12 章 Python 的数据库编程	204
12.1 数据库的基础知识	204
12.1.1 数据库的概念	204
12.1.2 关系型数据库	205
12.1.3 Python 的 SQLite3 模块	206
12.2 SQLite 数据库	206
12.2.1 SQLite 数据库简介	206
12.2.2 下载和安装 SQLite 数据库	206
12.2.3 SQLite3 的常用命令	207
12.2.4 SQLite3 的数据类型	208
12.2.5 SQLite3 模块中的对象	208
12.2.6 SQLite3 的函数	209
12.2.7 创建 SQLite3 数据库	210
12.3 关系数据库语言 SQL	210
12.3.1 数据表的建立和删除	211
12.3.2 向表中添加列	212
12.3.3 向表中插入数据	212
12.3.4 修改表中的数据	213
12.3.5 删除数据	213
12.3.6 查询数据	213
12.4 Python 的 SQLite3 编程	214
12.4.1 访问数据库的步骤	214
12.4.2 创建数据库和表	216
12.4.3 数据库的插入、更新和 删除操作	216
12.5 SQLite 编程的应用	217
本章小结	220
习题 12	221

第 13 章 科学计算与图表绘制

13.1 用于科学计算的 numpy 库	223
13.1.1 numpy 简介	223
13.1.2 numpy 数组的创建	223
13.1.3 访问 numpy 数组的元素	226
13.1.4 numpy 数组的算术运算	227
13.1.5 numpy 数组的形状操作	229
13.2 用于绘制图表的 matplotlib 库	230
13.2.1 matplotlib 简介	230
13.2.2 matplotlib.pyplot 库中的函数	231
13.2.3 绘制直方图、条形图、饼状图	235
本章小结	238
习题 13	238

第 14 章 爬取与分析网页中的数据

14.1 爬取网页的 urllib 和 requests 库	240
14.1.1 爬取网页的基础知识	240
14.1.2 urllib 库	241
14.1.3 requests 库	243
14.2 解析网页的 BeautifulSoup4 库	245
14.2.1 BeautifulSoup4 库概述	245
14.2.2 BeautifulSoup4 库的对象	247
14.2.3 BeautifulSoup4 库操作解析 文档树	249
14.3 网页爬取技术的应用	253
14.3.1 爬取单一网页页面的信息	253
14.3.2 爬取来自多个页面的信息	256
本章小结	259
习题 14	259

参考文献

第1章

初识 Python

Python 是一种面向对象的、解释型的计算机编程语言，可应用于 Web 开发、科学计算、游戏程序设计、图形用户界面等领域。那么，什么是编程语言？解释型语言有什么特点？Python 语言有什么特点？本章将帮助我们认识 Python，了解 Python 程序的开发环境，理解 Python 程序的执行过程。

1.1 程序设计语言

1.1.1 程序设计语言的概念

让计算机按照用户的目的完成相应的操作，需要使用程序设计语言来编程。程序设计语言也称计算机语言，是用于描述计算机所执行的操作的语言。从计算机产生到现在，作为软件开发工具的程序设计语言经历了机器语言、汇编语言、高级语言等几个阶段。

(1) 机器语言

机器语言是采用计算机指令格式并以二进制编码表达各种操作的语言。计算机能够直接理解和执行机器语言程序。

机器语言能够被计算机直接识别，它执行速度快，占用存储空间小，但难读、难记，编程难度大，调试修改麻烦，而且不同型号的计算机具有不同的机器指令系统。

(2) 汇编语言

汇编语言是一种符号语言，它用助记符来表达指令功能。

汇编语言程序较机器语言程序易读、易写，并保持了机器语言执行速度快、占用存储空间小的优点。汇编语言的语句功能简单，但程序的编写较复杂，而且程序难以移植，因为汇编语言和机器语言都是面向机器的语言，都是为特定的计算机系统而设计的。汇编语言程序不能被计算机直接识别和执行，需要由一种起翻译作用的程序（称为汇编程序），将其翻译成机器语言程序（称为目标程序），计算机才能执行，这一翻译过程称之为“汇编”。

机器语言和汇编语言都被称为低级语言。

(3) 高级语言

高级语言是面向问题的语言，它比较接近于人类的自然语言。因为高级语言是与计算机结构无关的程序设计语言，它具有更强的表达能力，因此，可以方便地表示数据的运算和程序控制结构，能更有效地描述各种算法，使用户容易掌握。

Python 是一种高级语言，例如，计算 $5+11$ 的 Python 语言程序如下。

```
>>> print(5+11)
16      #运算结果
```

用高级语言编写的程序（称为源程序）并不能被计算机直接识别和执行，需要经过翻译程序翻译成机器语言程序后才能执行，高级语言的翻译程序有编译程序和解释程序两种。下面分别介绍编译程序和解释程序。

1.1.2 编译与解释

不同的高级语言，计算机程序的执行方式是不同的。这里所说的执行方式是指计算机执行一个程序的过程。按照计算机程序的执行方式，可以将高级语言分成静态语言和脚本语言两类。静态语言采用编译执行的方式，脚本语言采用解释执行的方式。无论哪种执行方式，用户执行程序的方法都是一致的，例如，都可以通过鼠标双击执行一个程序。

(1) 编译

编译是将源代码转换成目标代码的过程。源代码是计算机高级语言的代码，而目标代码则是机器语言的代码。执行编译的计算机程序称为编译器（Compiler）。

(2) 解释

解释是将源代码逐条转换成目标代码，同时逐条运行目标代码的过程。执行解释的计算机程序称为解释器（Interpreter）。

编译和解释的区别：编译是一次性地翻译，程序被编译后，运行时就不再需要源代码了；解释则是在每次程序运行时都需要解释器和源代码。这两者的区别类似于外语资料的笔译和实时的同声传译。

编译的过程只进行一次，所以编译过程的速度并不是关键，关键是生成目标代码的执行速度。因此，编译器一般都会集成尽可能多的优化技术，使生成的目标代码有更好的执行效率；而解释器反而因为执行速度的原因不会集成太多的优化技术。

1.2 Python 语言

1.2.1 Python 的历史

Python 的作者 Guido van Rossum 是荷兰人。Guido 理想中的计算机语言，是能够方便调用计算机的各项功能，如打印、绘图、语音等，而且程序可以轻松地编辑与运行，适合所有人学习和使用。1989 年，Guido 开始编写这种理想的计算机语言的脚本解释程序，并将其命名为 Python。Python 语言的目标是成为功能全面、易学易用、可拓展的语言。

第一个 Python 的公开版本于 1991 年发布。它是用 C 语言实现的，能够调用 C 语言的库文件，具有类、函数、异常处理等功能，包含表和词典等核心数据类型，以模块为基础的拓展系统。

之后，在 Python 的发展过程中，形成了 Python 2.x 和 Python 3.x 两个不同系列的版本，这两个版本之间不兼容。为了满足不同 Python 用户的需求，目前是 Python 2.x 和 Python 3.x 两个版本并存。Python 2.x 的最高版本是 Python 2.7，Python 官网宣布，直到 2020 年，都不再为 Python 2.x

发布新的版本。Python 3.x 是从 2008 年开始发布的, 本书中的程序是在 Python 3.6 版本下实现的。

存在 Python 2.x 和 Python 3.x 两个不同版本的原因是, Python 3.0 发布时, 就不支持 Python 2.0 的版本, 但 Python 2.0 拥有大量用户, 这些用户无法正常升级使用新版本, 所以之后才发布了一个 Python 2.7 的过渡版本, 并且 Python 2.7 将会被支持到 2020 年。

1.2.2 Python 的特点

Python 是目前最流行且发展最迅速的计算机语言, 它具有如下几个特点。

(1) 简单、易学

Python 以“简单”“易学”的特性成为编程的入门语言。一个良好的 Python 程序像一篇英文文档, 非常接近于人的自然语言, 用户在应用 Python 的过程中, 可以更多地专注于解决的问题, 而不必考虑计算机语言的细节, 从而回归语言的服务功能。

(2) 开源, 拥有众多的开发群体

用户可以查看 Python 源代码, 研究其代码细节或进行二次开发。用户不需要为使用 Python 支付费用, 也不涉及版权问题。因为开源, 越来越多的优秀程序员加入到 Python 开发中, Python 的功能也会愈加丰富和完善。

(3) Python 是解释型语言

使用 Python 语言编写的程序可以直接从源代码运行。在计算机内部, Python 解释器先把源代码转换成字节码的中间形式, 然后再把它翻译成计算机使用的机器语言并运行。Python 是解释型语言, 用户可以将一些代码行在交互方式下直接测试执行, 使得 Python 的学习更加简单。

(4) 良好的跨平台性和可移植性

Python 的开源本质, 决定了它可以被移植到多个平台。如果用户的 Python 程序使用了依赖于系统的特性, Python 程序可能需要修改与平台相关的代码。Python 的应用平台包括 Linux、Windows、Macintosh、Solaris、OS/2、FreeBSD、Amiga、Android、iOS 等。

(5) 面向对象

Python 既支持面向过程的编程, 也支持面向对象的编程。在“面向过程”的语言中, 程序是由过程或仅仅是可重用代码的函数构建起来的。在“面向对象”的语言中, 程序是由数据和功能组合而成的对象构建起来的。与其他主要的语言(如 C++ 和 Java) 相比, Python 以一种非常强大又简单的方式实现面向对象编程, 为大型程序的开发提供了方便。

(6) 可扩展性和丰富的第三方库

Python 中可以运行 C/C++ 编写的程序, 以便某段关键代码可以运行得更快或者希望某些算法不公开。用户也可以把 Python 程序嵌入到 C/C++ 程序中, 提高 C/C++ 程序的脚本能力, 使其具有良好的可扩展性。

Python 还有功能强大的开发库。Python 标准库可以处理各种工作, 包括正则表达式、文档生成、单元测试、线程、数据库、HTML、WAV 文件、密码系统、GUI (图形用户界面) 和其他与系统有关的操作。除了这些标准库, 它还有大量高质量的第三方库, 如 wxPython、Twisted 和 Python 图像库等。

1.2.3 Python 的应用

Python 的应用领域覆盖了 Web 开发、科学运算、系统运维、GUI 编程、数据库编程等诸多方面。

(1) Web 开发

Python 包含标准的 Internet 模块, 可用于实现网络通信及应用。Python 的第三方框架包括

Django、Web2py、Zope 等,可以让程序员方便地开发 Web 应用程序。典型的 Web 应用,如 Google 爬虫、Google 广告、世界上最大的视频网站 YouTube、豆瓣、知乎等都是使用 Python 开发的。

(2) 科学运算

Python 广泛应用于人工智能与深度学习领域,典型的第三方库包括 NumPy、SciPy、Matplotlib 等。随着众多程序库的开发,使得 Python 越来越适合于进行科学计算、绘制高质量的 2D 和 3D 图像。例如,美国航天局(NASA)多使用 Python 进行数据分析和运算。

(3) 云计算

Python 是云计算方面应用最广的语言,其典型应用 OpenStack 就是一个开源的云计算管理平台项目。

(4) 系统运维

Python 是运维人员必备的语言。Python 标准库包含多个调用操作系统功能的库。通过第三方软件包 pywin32,Python 能够访问 Windows API。使用 Ironpython,Python 能够直接调用 Net Framework。一般而言,使用 Python 编写的系统管理脚本在可读性、性能、代码重用度、扩展性等方面都优于普通的 Shell 脚本。

(5) GUI 编程

Python 可以非常简单、快捷地实现 GUI 程序。Python 内置了 Tkinter 的标准面向对象接口 TkGUIAPI,可以非常方便地开发图形应用程序,还可以使用其他一些扩展包(如 WxPython、PyQT、Dabo 等)在 Python 中创建 GUI 应用。

1.3 Python 的开发环境

1.3.1 下载和安装 Python

Python 是一个轻量级的软件,读者可以在其官网下载 Python 安装程序(软件版本与下载页面在不断更新,读者打开的下载界面和看到的软件可下载版本可能会与本书的不一样,但下载与安装的方法类似)。

Python 开发包下载页面如图 1-1 所示,本书是在 Windows 10 操作系统下,应用 Python 3.6.5 版,读者也可以下载 Linux、iOS、Android 等操作系统的 Python 开发包,或选择其他的 Python 版本。

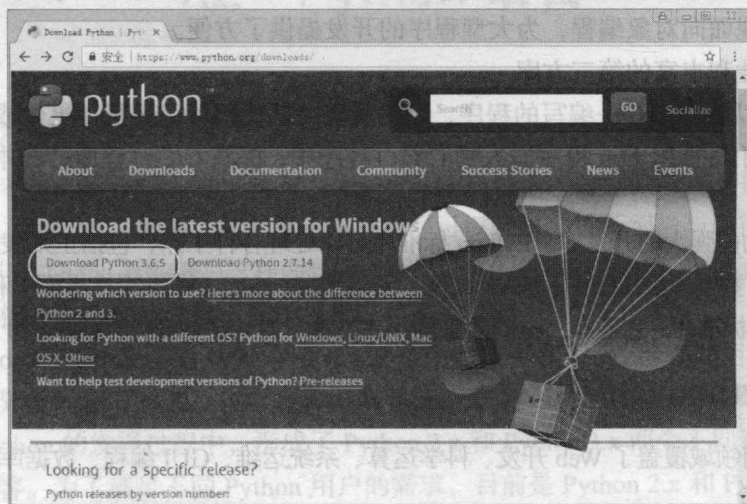


图 1-1 Python 官网下载页面

双击打开下载的 Python 安装程序 Python 3.6.5.exe，将启动安装向导，接下来用户按提示操作即可。在图 1-2 所示的安装程序页面中，选中“Add Python 3.6 to PATH”复选框，将 Python 的可执行文件路径添加到 Windows 操作系统的环境变量 PATH 中，以方便在将来的开发中启动各种 Python 工具。

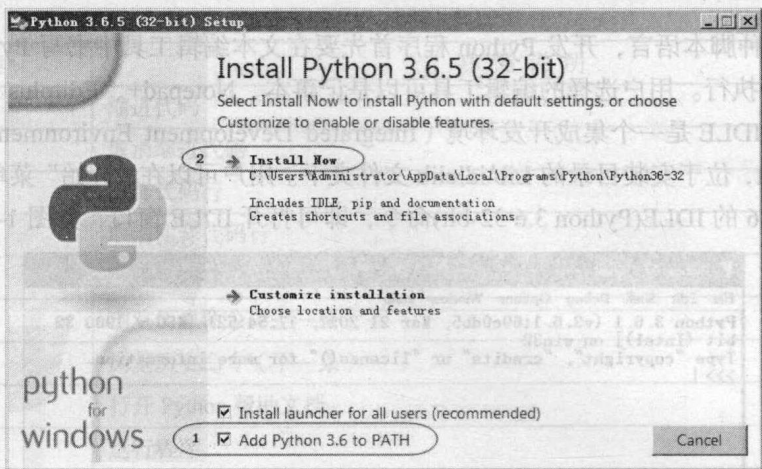


图 1-2 安装程序界面

安装成功后的界面如图 1-3 所示，并且会在 Windows 系统的“开始”菜单中显示图 1-4 所示的 Python 命令。这些命令的具体含义如下。

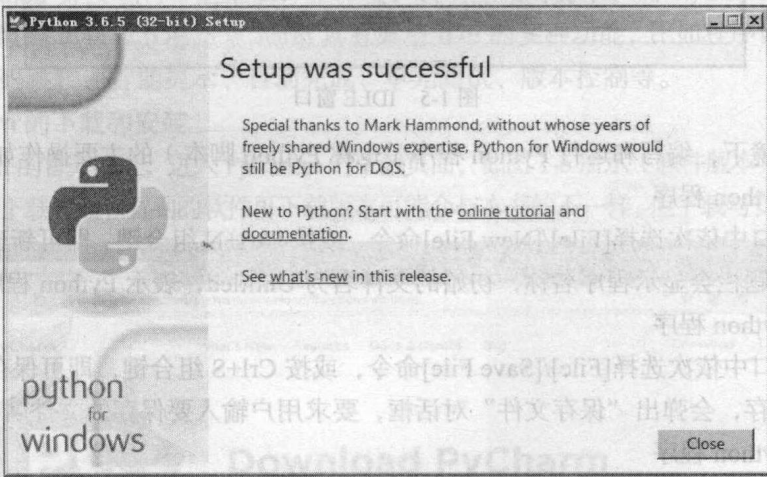


图 1-3 Python 安装成功界面

- “IDLE (Python 3.6 32-bit)”：启动 Python 自带的集成开发环境 IDLE。
- “Python 3.6 (32-bit)”：将以命令行的方式启动 Python 的解释器。
- “Python 3.6 Manuals (32-bit)”：打开 Python 的帮助文档。
- “Python 3.6 Module Docs (32-bit)”：将以内置服务的方式打开 Python 模块的帮助文档。

用户在学习 Python 的过程中，通常使用的是 Python 自带的集成开发环境 IDLE。

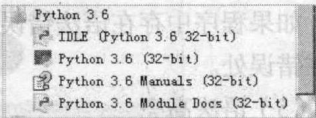


图 1-4 “开始”菜单中的 Python 命令

在 Windows 10 操作系统下, Python 默认的安装路径是 C:\Users\Administrator\AppData\Local\Programs\Python\Python36-32, 如果用户想要自定义 Python 解释器的安装路径, 可以在图 1-2 中选中“Customize installation”选项, 并选择需要安装的部件。

1.3.2 内置的 IDLE 开发环境

Python 是一种脚本语言, 开发 Python 程序首先要在文本编辑工具中书写 Python 程序, 然后由 Python 解释器执行。用户选择的编辑工具可以是记事本、Notepad+、Editplus 等。Python 开发包自带的编辑器 IDLE 是一个集成开发环境 (Integrated Development Environment, IDE), 其启动文件是 idle.bat, 位于安装目录的 Lib\idlelib 文件夹下。用户可以在“开始”菜单的“所有程序”中选择 Python 3.6 的 IDLE(Python 3.6 32-bit)命令, 即可打开 IDLE 窗口, 如图 1-5 所示。

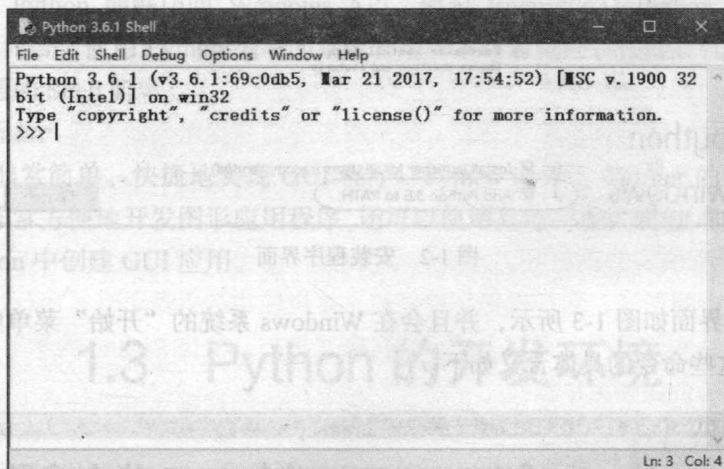


图 1-5 IDLE 窗口

在 IDLE 环境下, 编写和运行 Python 程序 (也称 Python 脚本) 的主要操作如下。

(1) 新建 Python 程序

在 IDLE 窗口中依次选择[File]/[New File]命令, 或按 Ctrl+N 组合键, 即可新建 Python 的脚本程序, 窗口的标题栏会显示程序名称, 初始的文件名为 Untitled, 表示 Python 程序还没有保存。

(2) 保存 Python 程序

在 IDLE 窗口中依次选择[File]/[Save File]命令, 或按 Ctrl+S 组合键, 即可保存 Python 程序。如果是第一次保存, 会弹出“保存文件”对话框, 要求用户输入要保存的文件名。

(3) 打开 Python 程序

在 IDLE 窗口中依次选择[File]/[Open File]命令, 或按 Ctrl+O 组合键, 将会弹出“打开文件”对话框, 要求用户选择要打开的 Python 文件名。

(4) 运行 Python 程序

在 IDLE 窗口中依次选择[Run]/[Run Module]命令, 或按下 F5 键, 即可在 IDLE 中运行当前的 Python 程序。

如果程序中存在语法错误, 则会弹出提示框“invalid syntax”, 并且会有一个浅红色方块定位在错误处。

(5) 语法高亮

IDLE 支持 Python 的语法高亮, 即 IDLE 能够以彩色标识出 Python 语言的关键字, 提醒开发

人员该词的特殊作用。例如，注释以红色显示，关键字以紫色显示，字符串显示为绿色。

(6) 常用快捷键

IDLE 支持撤销、全选、复制、粘贴、剪切等常用快捷键，使用 IDLE 的快捷键能显著提高编程速度和开发效率。IDLE 的常用快捷键及其功能如表 1.1 所示。

表 1.1

IDLE 的常用快捷键及其功能

快 捷 键	功 能 说 明
Ctrl + [	缩进代码
Ctrl +]	取消缩进代码
Alt+3	注释代码行
Alt+4	取消注释代码行
Alt+/	单词自动补齐
Alt+P	浏览历史命令（上一条）
Alt+N	浏览历史命令（下一条）
F1	打开 Python 帮助文档
F5	运行程序
Ctrl+F6	重启 Shell，之前定义的对象和导入的模块全部清除

1.3.3 PyCharm 集成开发环境

IDLE 是 Python 开发包自带的集成开发环境，其功能相对简单；而 PyCharm 则是 JetBrains 公司开发的专业级的 Python IDE，PyCharm 具有典型 IDE 的多种功能，比如程序调试、语法高亮、Project 管理、代码跳转、智能提示、自动完成、单元测试、版本控制等。

1. PyCharm 的下载和安装

访问 PyCharm 的官方网址，进入 PyCharm 的下载页面，如图 1-6 所示（软件版本与下载页面在不断更新，读者打开的下载界面和看到的软件可下载版本可能会与本书的不一样，但下载与安装的方法类似）。



图 1-6 PyCharm 的下载页面

用户可以根据自己的操作系统平台下载不同版本的 PyCharm。

PyCharm Professional 是需要付费的版本，它提供 Python IDE 的所有功能，除了支持 Web 开发，支持 Django、Flask、Google App 引擎、Pyramid 和 web2py 等框架，还支持远程开发、Python 分析器、数据库和 SQL 语句等。

PyCharm Community 是轻量级的 Python IDE，是一款免费和开源的版本，但它只支持 Python 开发，适合初学者使用。如果是开发 Python 的应用项目，则需要使用 PyCharm Professional 提供更为丰富的功能。

安装 PyCharm 的过程十分简单，用户只要按照安装向导的提示逐步安装即可，图 1-7 是安装过程中选择安装路径的界面。安装完成后的界面如图 1-8 所示。

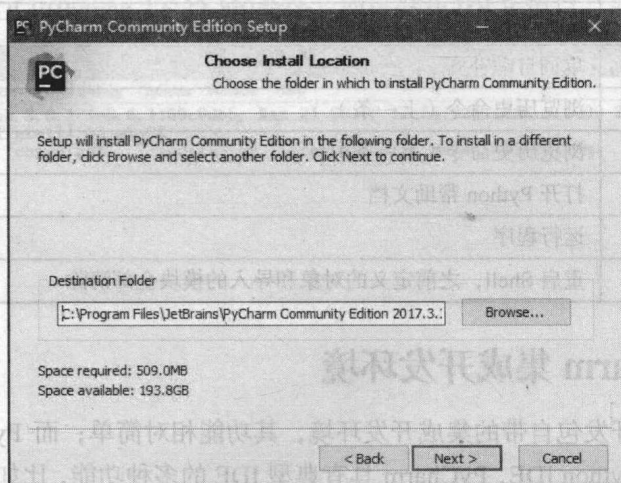


图 1-7 选择 PyCharm 的安装路径

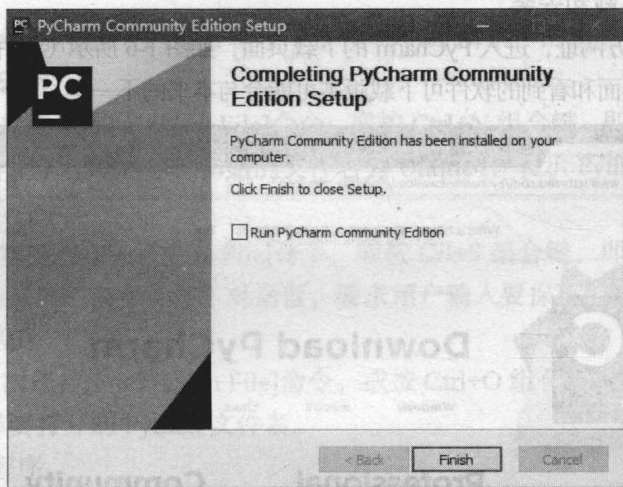


图 1-8 安装成功界面

2. 建立 Python 项目和文件

第一次启动 PyCharm 时，会显示若干初始化的提示信息，保持默认值即可。之后，进入创建项目的界面。如果不是第一次启动 PyCharm，并且以前创建过 Python 项目，则 Python 项目会出现在图 1-9 所示的窗口中，其右侧 3 个选项的含义分别是“创建新项目”“打开已经存在的项目”

“从版本控制中检测项目”。

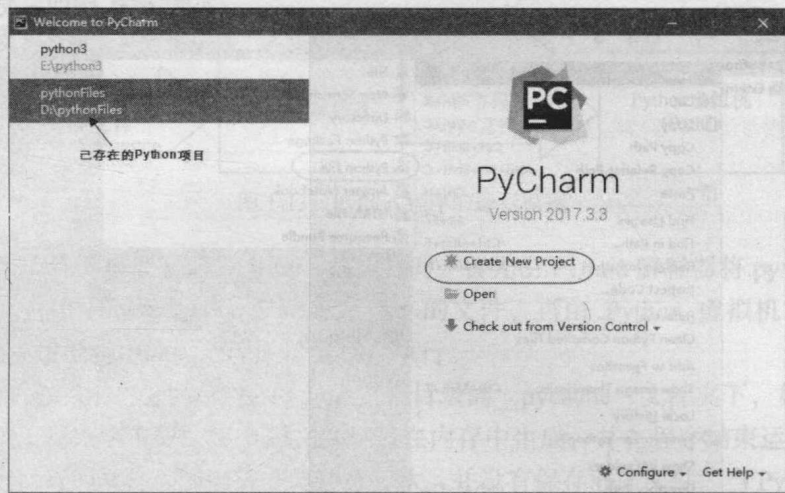


图 1-9 创建 Python 项目界面

(1) 创建项目

选择“Create New Project”选项创建项目后，会出现选择项目存放路径界面，如图 1-10 所示。

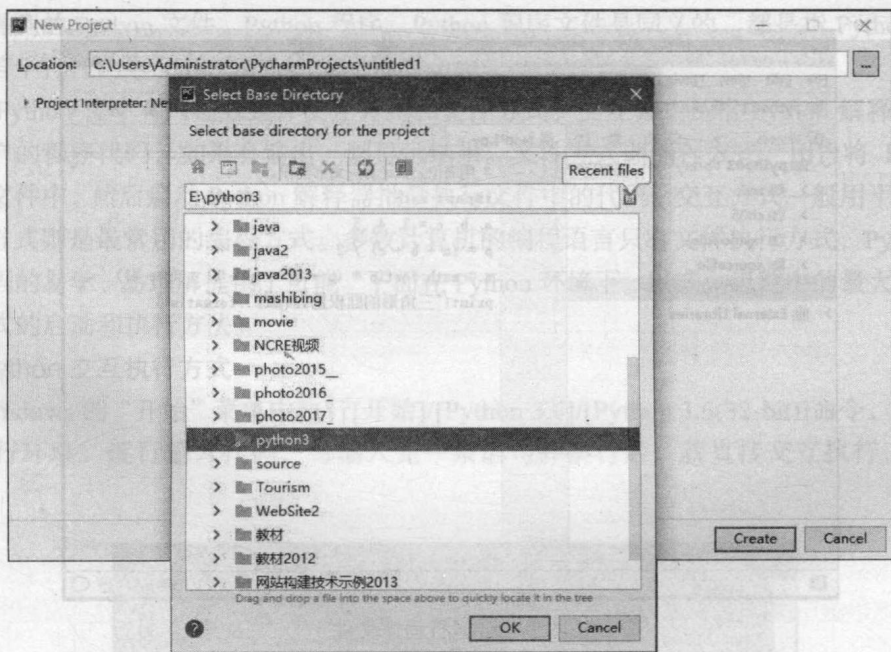


图 1-10 选择新建项目的存放路径

(2) 新建文件

项目创建完成后，如果要在项目中创建 Python 文件，可选中项目名称，单击鼠标右键，在弹出的快捷菜单中选择[New]/[Python File]命令来新建文件，如图 1-11 所示。

(3) 保存和运行文件

在程序编辑窗口输入代码后，可以保存文件，选择“Run”菜单中的命令可以运行程序。图 1-12 所示为一个完整的程序，使用“Run”菜单中的命令可以调试和运行程序。