

21世纪高等教育计算机规划教材

COMPUTER

Python 程序设计

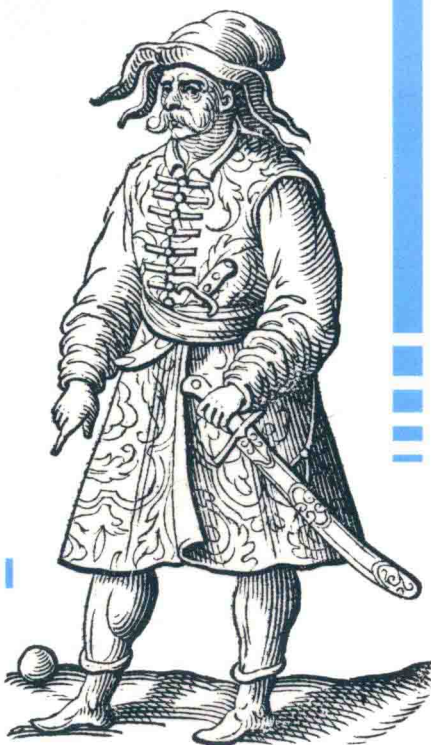
Python Programming

王学军 胡畅霞 韩艳峰 主编

理论知识全面系统，讲解简明扼要

案例丰富多样，涵盖多种开发场景

提供程序源代码和多媒体课件



中国工信出版集团



人民邮电出版社
POSTS & TELECOM PRESS

21世纪高等教育

COMPUTER

Python 程序设计

Python Programming

■ 王学军 胡畅霞 韩艳峰 主编



人民邮电出版社

北京

图书在版编目 (C I P) 数据

Python程序设计 / 王学军, 胡畅霞, 韩艳峰主编

— 北京 : 人民邮电出版社, 2018.1

21世纪高等教育计算机规划教材

ISBN 978-7-115-46930-4

I. ①P… II. ①王… ②胡… ③韩… 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(2017)第229847号

内 容 提 要

本书以 Python 3.4 版本为背景, 较为全面地介绍了 Python 高级语言程序设计的基本方法。全书共分 10 章, 主要包括: Python 概述、Python 语言基础、Python 函数及模块、面向对象编程、Python GUI 编程、Python 数据库及文件系统、Python 网络编程、Python 网站开发、Python 数字图像处理及 Python 科学计算。

本书可以作为高等院校程序设计课程教材, 也可供从事计算机应用开发的各类人员使用。



◆ 主 编 王学军 胡畅霞 韩艳峰

责任编辑 张 斌

责任印制 陈 犇

◆ 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>

北京市艺辉印刷有限公司印刷

◆ 开本: 787×1092 1/16

印张: 14.75

2018 年 1 月第 1 版

字数: 394 千字

2018 年 1 月北京第 1 次印刷

定价: 42.00 元

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

反盗版热线: (010)81055315

广告经营许可证: 京东工商广登字 20170147 号

计算机技术的发展,促进了程序设计语言的发展,特别是面向对象的程序设计语言的出现,极大地改进了传统的程序设计方法。在众多的程序设计语言中,由于 Python 语言具有简洁、易读、高效和可扩展性的特点,因而被越来越多的高校列为程序设计首选语言。2017 年 5 月 TIOBE 公布了编程语言排行榜,得益于人工智能方面的发展,Python 语言跃居第四。目前 Python 被越来越多地应用于信息处理、图像处理、Web 网站开发、人工智能等方面。

本书以培养学生编程思想和编程能力为目的,共分 10 章,第 1 章主要包括 Python 概述、特点、应用及开发环境;第 2 章主要包括 Python 数据类型、表达式、Python 数据结构及程序控制结构;第 3 章主要包括函数的定义、调用、参数的传递、模块的定义及引用;第 4 章主要包括面向对象的基本概念、类的定义和使用、类的属性和方法、类的继承及重载;第 5 章主要包括 PyQt GUI 编程;第 6 章主要包括数据库编程及文件系统;第 7 章主要包括 Python 网络编程;第 8 章主要包括 Python 网站的开发;第 9 章主要包括 Python 数字图像处理;第 10 章主要包括 Python 科学计算等。

参加本书编写的作者具有多年的计算机语言教学经验和丰富的心得和体会。全书内容广泛、重点突出,在编排上采用循序渐进、逐步扩展提高的方法,同时本书还精心设计了大量的示例和习题,以加深学生对内容的理解,提高学生分析问题、解决问题的能力。本书同时配有多媒体课件及例题源代码提供下载。

本书由王学军、胡畅霞、韩艳峰担任主编,龙海侠担任副主编。具体编写分工如下:第 1、2 章由尹佳利、李光曜编写,第 3、4 章由胡畅霞、张岩、李虎程编写,第 5、6 章由王学军、连丹丹编写,第 7、8 章由韩艳峰、龙海侠编写,第 9、10 章由王学军、赵天编写。本书的编写得到了各级领导的关心和支持,在此一并表示感谢。

限于编者水平,加之时间仓促,不当之处敬请广大读者批评指正,以使本书能不断完善。

编者

2017 年 8 月

第1章 Python 概述 1

1.1 Python 语言 1

1.1.1 Python 的发展 1

1.1.2 Python 的特点 2

1.1.3 Python 的应用领域 2

1.1.4 Python 的版本及安装 3

1.2 Python 开发环境及工具 5

1.2.1 开发工具 IDLE 5

1.2.2 开发工具 PyCharm 7

1.2.3 编写简单的小程序 8

1.2.4 PyCharm 配置 10

1.3 习题 11

第2章 Python 语言基础 12

2.1 Python 语言基础概述 12

2.1.1 Python 文件类型 12

2.1.2 Python 编码规范 12

2.1.3 输入与输出 14

2.2 Python 数据类型 15

2.2.1 Number (数字) 15

2.2.2 String (字符串) 15

2.2.3 变量及其赋值 17

2.3 运算符和表达式 18

2.3.1 算术运算符和表达式 18

2.3.2 赋值运算符和表达式 19

2.3.3 逻辑运算符和表达式 20

2.3.4 关系运算符和表达式 21

2.3.5 字符串运算符和表达式 21

2.3.6 位运算符和表达式 24

2.3.7 运算符的优先级 24

2.3.8 Python 常用的函数 25

2.4 Python 数据结构 27

2.4.1 列表 28

2.4.2 元组 31

2.4.3 字典 32

2.4.4 集合 33

2.5 程序控制结构 35

2.5.1 选择结构 35

2.5.2 循环结构 37

2.6 编程实践 40

2.7 习题 43

第3章 Python 函数及模块... 46

3.1 案例引入及分析 46

3.2 函数 47

3.2.1 函数的定义 48

3.2.2 函数的调用 48

3.2.3 函数的参数 49

3.2.4 函数的嵌套 52

3.2.5 函数的递归调用 52

3.3 变量的作用域 53

3.4 模块 54

3.4.1 导入和创建模块 54

3.4.2 模块包 56

3.5 编程实践 56

3.6 习题 60

第4章 面向对象编程 62

4.1 面向对象基础 62

4.1.1 面向过程与面向对象 62

4.1.2 面向对象基本概念 63

4.2 类的定义和使用 64

4.2.1 类的定义 64

4.2.2 类的使用 65

4.3 类的属性和方法 66

4.3.1 类的属性 66

4.3.2 类的方法 68

4.3.3 访问控制 70

4.3.4 构造函数和析构函数 71

4.4 类的继承 72

4.4.1 类的简单继承 72

4.4.2 类的多重继承 75

4.5 类的重载 76

4.5.1 方法重载 76

4.5.2 运算符重载 77

4.6 编程实践 79

4.7 习题 84

第 5 章 Python GUI 编程 86

5.1 PyQt GUI 工具包概述 86

5.1.1 GUI 简介 86

5.1.2 PyQt 工具包 87

5.1.3 编程测试 89

5.2 PyQt GUI 编程 93

5.2.1 信号和槽 93

5.2.2 主窗口 QMainWindow 96

5.2.3 对话框 QDialog 97

5.2.4 PyQt 输入控件 100

5.2.5 按钮 100

5.2.6 显示控件 102

5.2.7 表格控件 102

5.2.8 布局控件 104

5.3 编程实践 105

5.4 习题 108

第 6 章 Python 数据库及
文件系统 112

6.1 数据库技术基础 112

6.1.1 数据库基本概念 112

6.1.2 关系数据库 113

6.1.3 数据库应用系统的开发步骤 114

6.2 MySQL 数据库 114

6.2.1 数据库安装 115

6.2.2 创建数据库 119

6.2.3 删除数据库 119

6.2.4 MySQL 数据类型 120

6.2.5 创建表 122

6.2.6 编辑查看表 123

6.2.7 删除表 124

6.2.8 插入数据 124

6.2.9 修改数据 124

6.2.10 删除数据 125

6.2.11 使用 SELECT 查询数据 125

6.3 Python 中访问 MySQL 语句 126

6.4 Python 文件系统 128

6.4.1 文件的基础知识 128

6.4.2 文件的基本操作 129

6.4.3 文件的读写操作 130

6.4.4 文件与目录操作函数和语句 131

6.5 编程实践 133

6.6 习题 138

第 7 章 Python 网络编程 141

7.1 网络模型介绍 141

7.1.1 OSI 简介 141

7.1.2 TCP/IP 简介 142

7.2 Socket 编程 145

7.2.1 Socket 简介 145

7.2.2 Socket 编程 145

7.2.3 用 Socket 建立服务器端程序 146

7.2.4 用 Socket 建立基于 UDP 的
服务器与客户端程序 147

7.2.5 用 SocketSever 建立服务器 148

7.3 urllib 包与 httplib 包使用 149

7.3.1 urllib 包 149

7.3.2 使用 httplib 包访问网站 151

7.4 使用 ftplib 访问 FTP 服务 152

7.4.1 ftplib 包 152

7.4.2 使用 ftplib 包访问 FTP
服务器 153

7.5 电子邮件 154

7.5.1 SMTP 和 POP3.....	154	第 9 章 Python 数字图像处理.....	196
7.5.2 发送邮件.....	154	9.1 基本图像操作和处理.....	196
7.5.3 接收邮件.....	156	9.1.1 图像和像素.....	196
7.6 编程实践.....	158	9.1.2 颜色空间.....	197
7.7 习题.....	163	9.1.3 像素的位深.....	198
第 8 章 Python 网站开发....	165	9.2 Python 图像处理类库 PIL.....	198
8.1 常见的 Web 开发框架.....	165	9.2.1 PIL 模块基本介绍.....	199
8.1.1 Zope 框架.....	165	9.2.2 Image 模块.....	199
8.1.2 TurboGears 框架.....	165	9.2.3 PIL 滤镜效果.....	202
8.1.3 Django 框架.....	166	9.3 Python 中使用 OpenCV.....	204
8.2 MVC 模式.....	167	9.3.1 OpenCV 安装.....	204
8.2.1 MVC 模式介绍.....	167	9.3.2 OpenCV 基本操作.....	205
8.2.2 MVC 模式的优缺点.....	168	9.3.3 处理视频序列.....	210
8.2.3 Django 框架中的 MVC.....	168	9.4 Matplotlib 绘图库.....	211
8.3 Django 开发环境的搭建.....	169	9.4.1 Matplotlib 安装.....	211
8.3.1 Django 框架的安装.....	169	9.4.2 Matplotlib 模块.....	211
8.3.2 Django 简单应用.....	171	9.4.3 Matplotlib 绘制简单图形.....	211
8.4 Django 框架的应用.....	173	9.5 编程实践.....	215
8.4.1 数据库的配置.....	173	9.6 习题.....	218
8.4.2 创建数据模型.....	175	第 10 章 Python 科学计算....	219
8.4.3 创建视图.....	177	10.1 NumPy 库.....	219
8.4.4 模板系统.....	179	10.1.1 ndarray 对象.....	219
8.4.5 URL 配置.....	182	10.1.2 ufunc 运算.....	222
8.4.6 发布 Django 项目.....	182	10.1.3 矩阵运算.....	223
8.5 Django 框架的高级应用.....	183	10.2 SciPy 数值计算库.....	224
8.5.1 管理界面.....	183	10.3 编程实践.....	226
8.5.2 编辑数据库.....	185	10.4 习题.....	227
8.5.3 Session 功能.....	186	参考文献.....	228
8.5.4 国际化.....	187		
8.6 编程实践.....	189		
8.7 习题.....	195		

本章重点

- Python 语言的发展及特点
- Python 的应用领域
- Python 的开发环境及工具

本章难点

- Python 安装及环境变量的配置
- PyCharm 的安装
- 使用 PyCharm 创建工程

Python（英式发音：/'paɪθən/；美式发音：/'paɪθɑːn/）是一种面向对象的解释性计算机程序语言。对于学习编程语言的初学者来说，Python 无疑是最好的选择。本章主要介绍 Python 语言的基本知识、Python 的开发环境和工具，以及简单的编程例子。

1.1 Python 语言

从程序设计语言的发展过程来分，计算机程序设计语言可分为：机器语言、汇编语言和高级语言。Python 是面向对象编程语言（Object-Oriented Programming），其语法优雅，具有高效率的数据结构。自从 20 世纪初 Python 诞生以来，它被越来越多地应用于信息处理、图像处理、Web 网站开发、人工智能等方面。

1.1.1 Python 的发展

Python 语言诞生于 20 世纪 90 年代初，由荷兰人吉多·范罗萨姆（Guido van Rossum）发明。Python 具有丰富和强大的库，又被称为胶水语言。它能够把其他语言制作的各種模块（尤其是 C/C++）轻松地结合在一起。由于 Python 语言简洁、易读、高效且具有可扩展性，许多的国内外高校将其列为程序设计课程。同时许多的软件包提供了 Python 的调用接口，以扩展 Python 的功能。

2017 年 5 月 TIOBE 公布了编程语言指数排行榜，得益于人工智能方面的发展，Python 首次超越 C# 跃居第四。作为人工智能的主要编程语言，从 2016 年开始，Python 的使用比例不断提升。

1.1.2 Python 的特点

Python 的设计秉承“优雅”“明确”“简单”的理念，具有以下特点。

1. 语法简单

Python 语言最大的特点就是简单，容易学习。Python 语法简单，容易上手，同时它也非常适用于非专业人员的入门学习。

2. 面向对象

Python 支持面向过程的编程和面向对象的编程，完全支持继承、重载、派生、多继承，有益于增强源代码的复用性。

3. 可移植

Python 具有源代码开放的特性，Python 程序都可以不加修改地运行在其他的平台上。例如 Linux、Windows、VMS、Solaris 等平台。

4. 解释性

Python 编写的程序不需要编译成二进制代码，可直接从源文件运行程序。Python 解释器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，这使得 Python 的使用更加简单，也更易于移植。

5. 可扩展性和可嵌入性

Python 提供了支持 C/C++ 接口，可以方便地使用 C/C++ 来扩展 Python。Python 提供了 API，通过使用 API 函数就可以编写 Python 扩展。

1.1.3 Python 的应用领域

随着 Python 语言的盛行，它使用的领域越来越广泛，例如：网站和游戏开发、机器人控制、航天飞机控制等。Python 主要有以下一些应用领域。

1. 系统编程

Python 提供应用程序编程接口（Application Programming Interface, API），能够进行系统的维护和开发。

2. 图形处理

Python 内置了 Tkinter（Python 默认的图形界面接口）和标准面向对象接口 Tk GUI API，可以进行程序 GUI 设计。同时有 PIL 等图形库支持，使其能够方便地进行图形处理。

3. 数据库编程

Python 语言提供了对目前主流的数据库系统的支持，例如 Microsoft SQL Server、Oracle、Sybase、DB2、MySQL、SQLite 等数据库。在编程的过程中，通过 Python DB-API（数据库应用程序编程接口）规范与数据库进行通信。另外，Python 自带有一个 Gadget 模块，提供了一个完整的 SQL 环境。

4. Internet 支持

Python 提供了标准的 Internet 模块，使 Python 能够轻松实现网络编程，例如，通过套接字（Socket）进行网络通信。此外，Python 支持许多 Web 开发工具包，使得 Python 能够快速构建功能完善和高质量的网站。

1.1.4 Python 的版本及安装

Python 是一门跨平台的语言，能够兼容不同的平台。考虑到 Windows 系统的使用者较多，因此，本书内容是基于 Windows 平台下的操作。

Python 的版本主要有 2.0 系列和 3.0 系列，不同的版本的语法结构有所不同。目前，2.0 系列的最新版本是 Python 2.7.14，3.0 系列的最新版本是 3.6.3。本书中使用的是 Python 3.4.1 版本。

1. Python 的安装

Python 是开源的、免费的。打开 Python 的官方网站选择相应的版本（python-3.4.1.msi）下载即可。安装过程如下。

(1) 双击运行下载后的程序，显示图 1-1 所示的对话框。单击“运行”按钮，显示图 1-2 所示的对话框。

(2) 在图 1-2 中使用默认设置，单击“Next”按钮，打开图 1-3 所示的对话框。

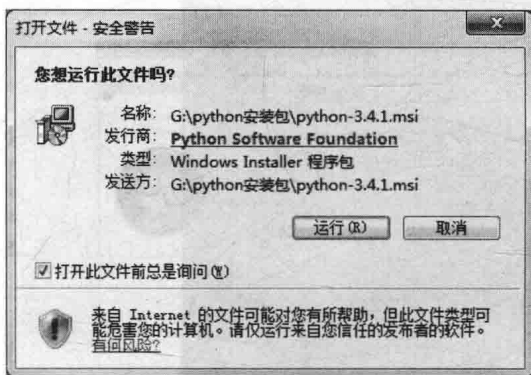


图 1-1 Python 安全警告



图 1-2 Python 安装向导

(3) 在图 1-3 中使用默认设置（可修改），单击“Next”按钮，显示图 1-4 所示的对话框。

(4) 在图 1-4 中单击“Next”按钮开始安装，安装进度如图 1-5 所示。

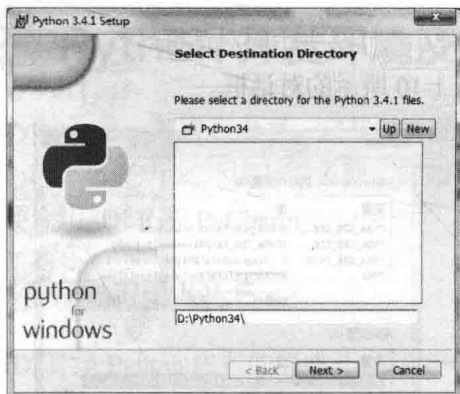


图 1-3 Python 安装路径

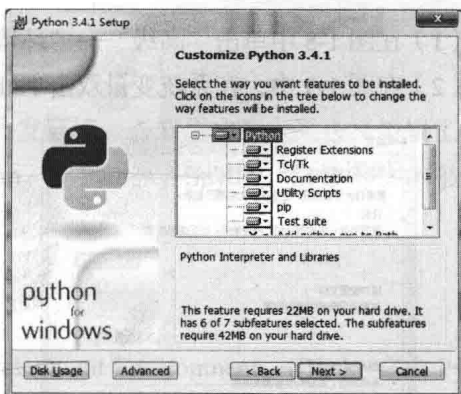


图 1-4 Python 安装功能

(5) 安装进度如图 1-5 所示，完成后如图 1-6 所示。

(6) 在图 1-6 中单击“Finish”按钮，完成安装。

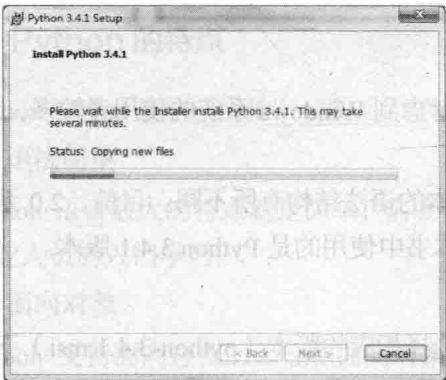


图 1-5 Python 安装进度对话框



图 1-6 Python 安装完成对话框

2. 环境变量的配置

打开系统属性窗体，显示图 1-7 所示的对话框。单击“高级系统设置”选项，显示图 1-8 所示的对话框。

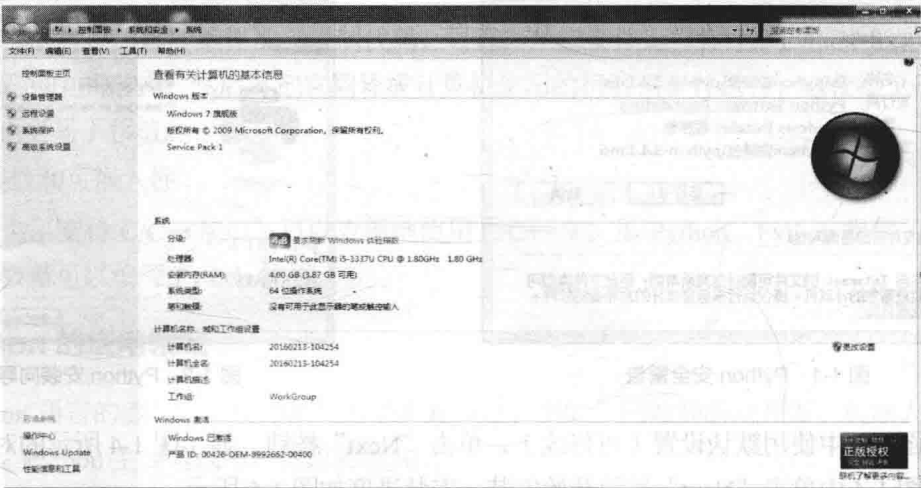


图 1-7 系统基本信息对话框

- (1) 在图 1-8 中单击“高级”→“环境变量”按钮，弹出图 1-9 所示的对话框。
- (2) 在图 1-9 中，在系统变量双击 Path 变量，弹出图 1-10 所示的对话框。

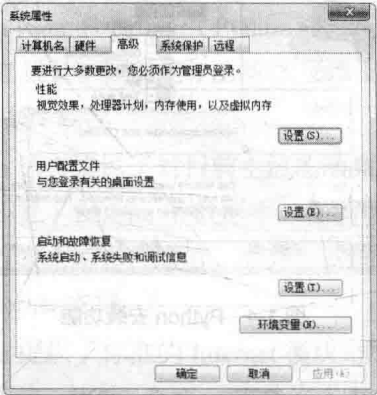


图 1-8 高级系统属性对话框



图 1-9 环境变量对话框

(3) 在图 1-10 中修改 Path 的变量值,即在变量值的字符串末尾加上一个分号,然后再输入 Python 的安装路径,如图 1-11 所示。

(4) 在图 1-11 中,单击“确定”按钮,完成环境变量的配置。

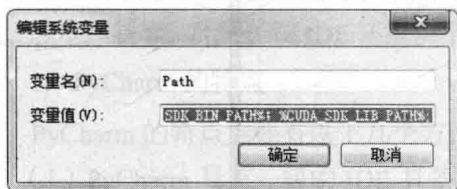


图 1-10 编辑系统变量对话框

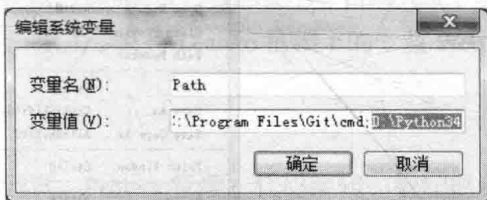


图 1-11 编辑完成后的环境变量值

3. 运行

环境变量配置完成后,打开命令行,输入“python”后回车,结果如图 1-12 所示,则配置成功。



图 1-12 测试 Python 安装及配置成功

1.2 Python 开发环境及工具

Python 是一种脚本语言,它并没有提供一个官方的开发环境,需要用户自主来选择编辑工具。目前,Python 的开发环境有很多种,例如 IDLE、PyCharm、DrPython、Spyder、SPE 等。本书中我们主要介绍 IDLE 和 PyCharm。

1.2.1 开发工具 IDLE

IDLE 是 Python 内置的集成开发环境 (Integrated Development Environment, IDE),它由 Python 安装包来提供,也就是 Python 自带的文本编辑器。在“开始”菜单的“所有程序”中,选择 Python3.4 下的 IDLE 菜单项,打开 IDLE 编辑器,如图 1-13 所示。

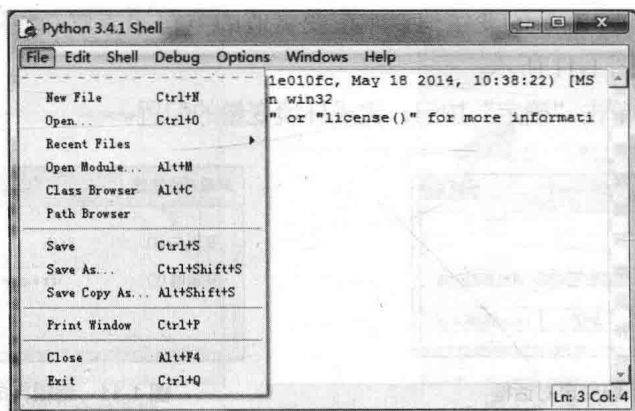


图 1-13 文本编辑器 IDLE

启动 IDLE 后首先看到的是 Python Shell, 通过它可以在 IDLE 内部执行 Python 命令。除此之外, IDLE 还带有一个编辑器, 用来编辑 Python 程序(或者脚本); 一个交互式解释器用来解释执行 Python 语句; 一个调试器来调试 Python 脚本。

IDLE 为开发人员提供了许多有用的功能, 如自动缩进、语法高亮显示、单词自动完成以及命令历史等, 在这些功能的帮助下, 用户能够有效地提高开发效率。下面通过一个实例来介绍 IDLE 的使用方法。

1. 新建文件

单击“File”→“New File”打开一个新的窗口, 输入如下程序:

```
1 print('欢迎使用 Python')
2 a=5
3 b=4
4 print(a+b)
```

2. 保存程序

在 IDLE 编写完程序之后, 在菜单里依次选择“File”→“Save”(或者用 Ctrl+S 组合键)来进行保存, 首次保存时会弹出文件对话框, 要求用户输入保存的文件名。此时文件会保存在 Python3.4.1 的安装目录下。

说明: Python 是以 .py 为后缀名进行保存文件的。

3. 运行程序

编辑完成后, 可以按 F5 键运行程序, 或单击“Run”→“Run Module”菜单项。

4. 调试程序

用户可以使用 IDLE 调试程序。在“Python Shell”窗口中单击“Debug”→“Debugger”菜单项启动 IDLE 交互式调试器。这时, IDLE 会打开“Debug Control”窗口, 并在“Python Shell”窗口中输出“[DEBUG ON]”并后跟一个“>>>”提示符。这样, 用户就能像平时那样使用这个“Python Shell”窗口了, 只不过现在输入的任何命令都是允许在调试器下。用户可以在“Debug Control”窗口查看局部变量和全局变量等有关内容。如果要退出调试器, 可以再次单击“Debug”→“Debugger”菜单项。

1.2.2 开发工具 PyCharm

PyCharm 是由 JetBrains 打造的一款 Python IDE，它带有一整套可以帮助用户使用 Python 语言开发时提高其效率的工具，比如调试、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。此外，该 IDE 还提供了一些高级功能，以用于支持 Django 框架下的专业 Web 开发。

1. PyCharm 的特点

PyCharm 的特点主要有以下几个方面。

(1) PyCharm 具有一般的 IDE 具备的功能，比如调试、Project 管理、代码跳转、智能提示、自动完成、单元测试、版本控制等。

(2) PyCharm 提供用于 Django 开发/工具，并且支持 Google App Engine 和 IronPython。

(3) Python 重构功能使用户能在项目范围内轻松进行重命名，提取方法/超类，导入域/变量/常量，移动和前推/后退重构。

(4) Python 支持 Google App 引擎，用户可选择使用 Python 运行环境为 Google APP 引擎进行应用程序的开发，并执行例行程序部署工作。

(5) Python 集成版本控制功能，将登入、录出、视图拆分与合并等功能都能在统一的 VCS 用户界面（可用于 Mercurial、Subversion、Git、Perforce 和其他的 SCM）中得到。

(6) Python 的可自定义&可扩展功能可以绑定 Textmate、NetBeans、Eclipse & Emacs 键盘主盘，以及 Vi/Vim 仿真插件。

2. PyCharm 安装以及配置

首先打开 PyCharm 官网下载相应的版本，PyCharm 在 Windows 环境下有两个不同的版本：专业版（Professional）和社区版（Community）。安装步骤如下。

(1) 双击运行下载后的程序，显示图 1-14 所示的对话框。单击“Next”按钮，显示图 1-14 所示的对话框。

(2) 选择 PyCharm 安装路径，单击“Next”按钮，显示图 1-15 所示的对话框。



图 1-14 PyCharm 安装对话框

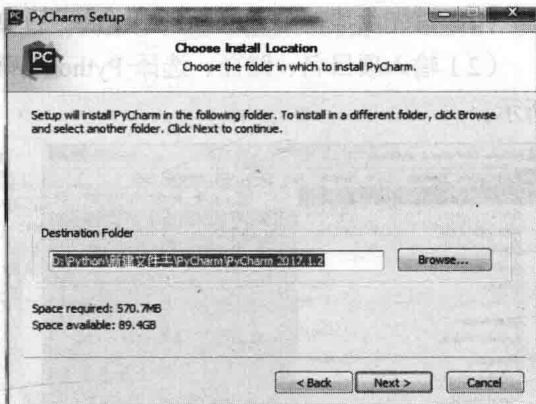


图 1-15 PyCharm 安装路径对话框

(3) 在图 1-16 中，选择相应的选项，单击“Next”按钮，出现图 1-17 所示的对话框。

在图 1-17 中使用默认设置，单击“Next”按钮，安装完成。

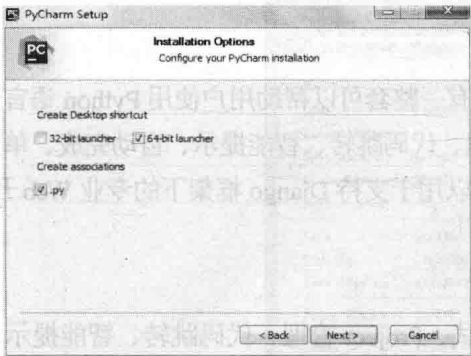


图 1-16 PyCharm 选项对话框



图 1-17 PyCharm 菜单文件对话框

(4) PyCharm 的使用需要激活，激活方式有三种：购买正版、选择试用、使用激活码，如图 1-18 所示。激活后弹出图 1-19 所示的对话框，进入 Pycharm 编辑环境，即可编程。



图 1-18 PyCharm 激活对话框

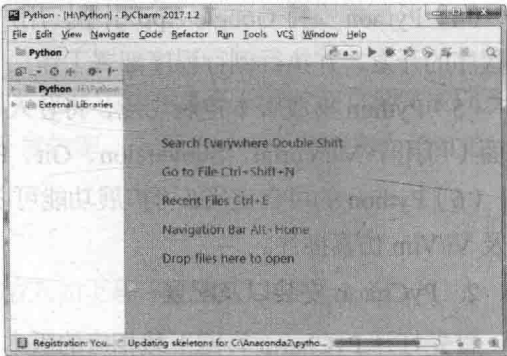


图 1-19 PyCharm 运行环境

1.2.3 编写简单的小程序

【例 1-1】编写一个小的程序（first.py），实现两个数的加法并输出。

- (1) 首先创建一个 Python 工程，如图 1-20 所示。单击“File”→“New Project”。
- (2) 输入项目名、路径、选择 Python 解释器。单击“Create”按钮进行下一步操作，如图 1-21 所示。

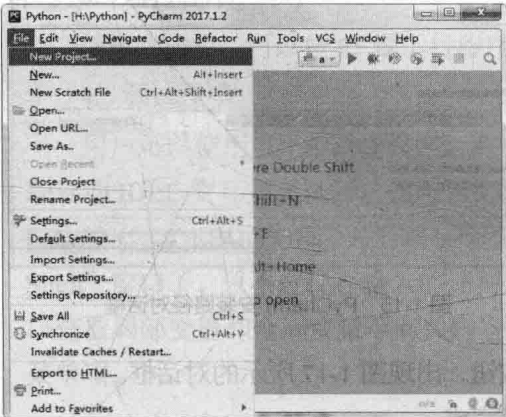


图 1-20 创建 Python 工程

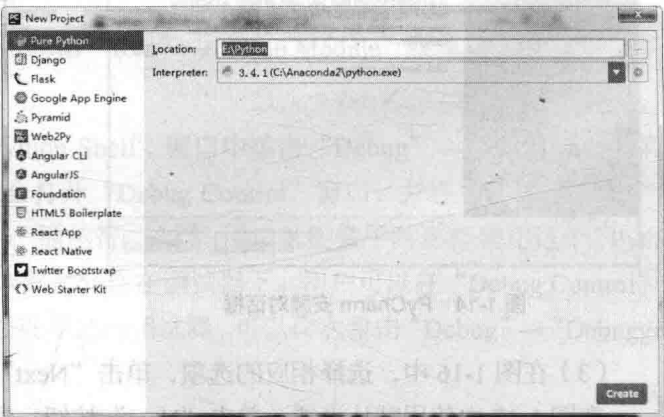


图 1-21 Python 工程路径及解释器

(3) 在图 1-22 中, 鼠标右键单击工程名→“New”→“Python File”, 创建一个 Python 文件。

(4) 输入文件名, 创建一个文件 Python。例如, first.py 的脚本文件。

(5) 编写程序, 如图 1-23 所示。

```
1  # -*- coding: UTF-8 -*-
2  num1 = input('输入第一个数字: ')
3  num2 = input('输入第二个数字: ')
4  # 求和
5  sum = float(num1) + float(num2)
6  # 显示计算结果
7  print('sum=%d'%sum)
```

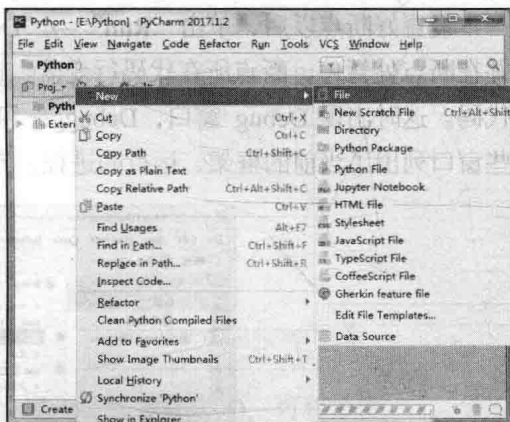


图 1-22 创建 Python 文件

(6) 鼠标右键单击“first.py”→“Run ‘first’”, 如图 1-24 所示。

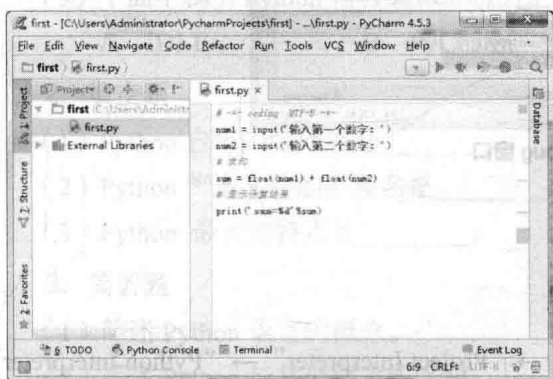


图 1-23 PyCharm 编译程序

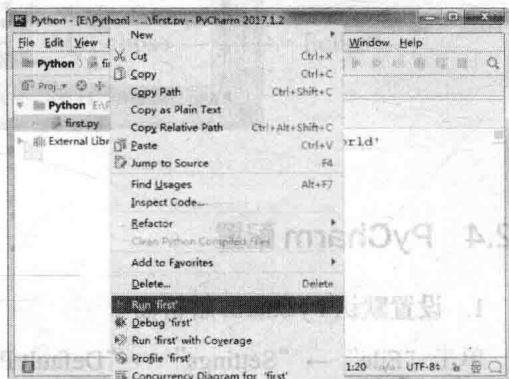


图 1-24 PyCharm 运行程序

(7) 显示运行结果如图 1-25 所示。

(8) 程序的调试。在 PyCharm 中可以方便地对程序进行调试, 准备工作就是在程序必要的地方加设断点 (Breakpoint) 标记了一个行的位置, 当程序运行到该行代码的时候, PyCharm 会将程序暂时挂起以方便用户对程序的运行状态进行分析。设置断点的方法是单击代码左侧的空白灰色槽, 断点会将对应的代码行标记为红色, 如图 1-26 所示。

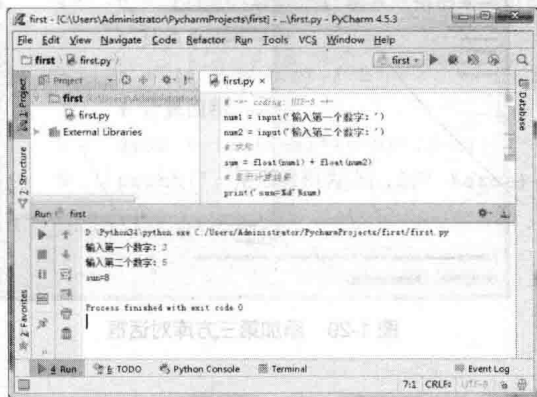


图 1-25 运行结果

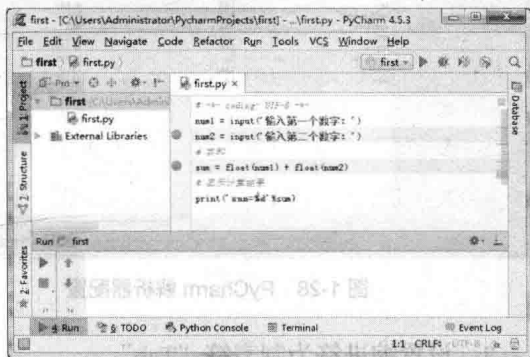


图 1-26 断点设置

设置好断点以后，单击“Run”→“Debug”菜单项或按下绿色的甲壳虫按钮，程序开始运行，并在断点处暂停，断点所在代码行变蓝，说明程序进程已经到达断点处，但尚未执行断点所标记的代码。这时出现 Debug 窗口，Debugger 窗口分为三个可见区域：Frames、Variables 和 Watches。这些窗口列出了当前的框架、运行的进程，方便用户查看程序空间中变量的状态等，如图 1-27 所示。

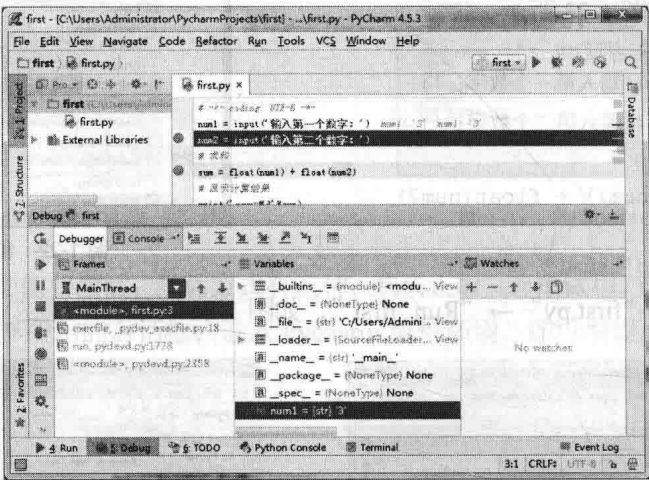


图 1-27 Debug 窗口

1.2.4 PyCharm 配置

1. 设置默认 PyCharm 解析器

单击“File”→“Settings”→“Default Project”→“Project Interpreter”→“Python Interpreter”选择系统安装的 Python，如图 1-28 所示。

2. 添加第三方库

在图 1-28 所示的界面中，单击“+”弹出图 1-29 所示的对话框。输入所需安装的库的名称，单击“Install Package”按钮安装。

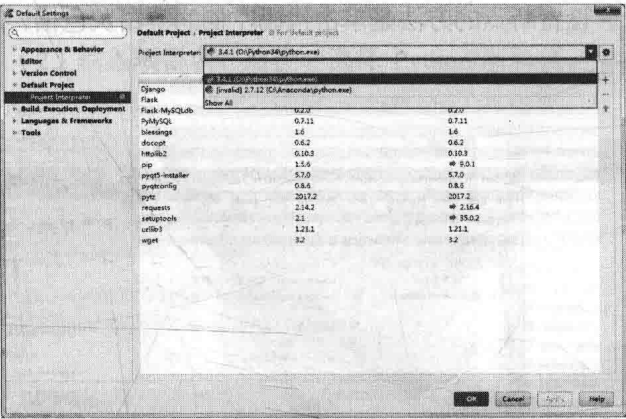


图 1-28 PyCharm 解析器配置

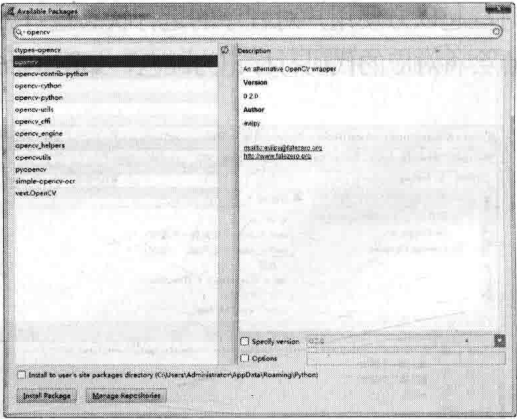


图 1-29 添加第三方库对话框

3. 设置缩进符为制表符“Tab”

单击“File”→“Default Settings”→“Editor”→“Code Style”→“Python”，勾选“Use tab character”。