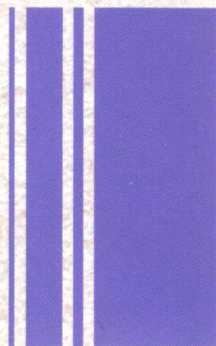
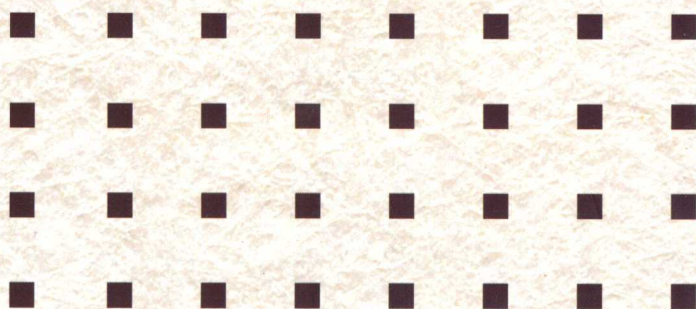


中国电子教育学会高教分会推荐·大数据系列教材
高等学校新工科人才培养“十三五”规划教材



Python大数据 基础与实战

范 晖 于长青 张文胜 编著



51

中国电子教育学会高教分会推荐·大数据系列教材

高等学校新工科人才培养“十三五”规划教材

Python 大数据基础与实战

范 晖 于长青 张文胜 编著



西安电子科技大学出版社

内 容 简 介

本书以 Python 3.7.2 为基础,系统地介绍了 Python 程序设计的基础知识。全书分为三篇,共 13 章。第一篇“Python 基础知识”,包含第 1~6 章;第二篇“Python 高级特性”,包含第 7~9 章;第三篇“Python 数据分析与处理”,包含第 10~13 章。其中每个知识点提供了大量的示例代码,并且重点讲述了数据可视化库 Matplotlib 和 Seaborn、科学计算库 Numpy、数据分析和处理库 Pandas 等的使用以及网络爬虫系统的设计。书中还提供了一些数据分析和处理的案例,可帮助读者对数据分析和处理方法以及 Python 的编程实现有一个初步的认识和掌握。

本书内容翔实,实例丰富,语言深入浅出,既适合作为高等学校计算机、大数据、人工智能等相关专业 Python 编程课程的教材,也可以作为 Python 开发人员的参考书。

图书在版编目(CIP)数据

Python 大数据基础与实战 / 范晖, 于长青, 张文胜编著. —西安: 西安电子科技大学出版社, 2019.7
ISBN 978-7-5606-5380-8

I. ① P… II. ① 范… ② 于… ③ 张… III. ① 软件工具—程序设计
IV. ① TP311.561

中国版本图书馆 CIP 数据核字(2019)第 131843 号

策 划 李惠萍

责任编辑 姚智颖 雷鸿俊

出版发行 西安电子科技大学出版社(西安市太白南路 2 号)

电 话 (029)88242885 88201467 邮 编 710071

网 址 www.xduph.com 电子邮箱 xdupfb001@163.com

经 销 新华书店

印刷单位 陕西天意印务有限责任公司

版 次 2019 年 7 月第 1 版 2019 年 7 月第 1 次印刷

开 本 787 毫米×1092 毫米 1/16 印 张 16

字 数 377 千字

印 数 1~3000 册

定 价 39.00 元

ISBN 978-7-5606-5380-8 / TP

XDUP 5682001-1

如有印装问题可调换

前 言

Python 由荷兰阿姆斯特丹的 Guido van Rossum(吉多·范罗苏姆)于 1989 年正式提出, 1991 年发布 Python 的第一个正式版本 V0.9。Python 秉承开放自由的思想, 是一种解释型、面向对象、动态数据类型的高级程序设计语言。它支持命令式编程、函数式编程, 完全支持面向对象程序设计, 拥有大量扩展库, 遍及各个领域。Python 不仅可以用于通用编程, 其免费开源的语言和环境使得它在数据分析与处理领域也具有巨大的潜力。

近年来, 随着大数据和人工智能需求的上升, Python 受到了越来越多的关注, 被誉为人工智能和大数据的专用语言。因此, 学好 Python 对于未来适应新一代信息技术产业的发展具有重要的意义。

本书分为三篇。

第一篇“Python 基础知识”, 包括第 1~6 章。第 1 章从使用者的角度介绍了 Python 的特点和应用、Python 的安装与版本选择、扩展库的安装、编码规范等内容。第 2 章从程序的编写者的角度介绍了变量、数据类型、运算符、表达式以及常用内置函数、基本输入输出等知识。第 3 章和第 4 章介绍了列表、元组、字典、集合等序列结构的使用和字符串的基本操作等内容。第 5 章介绍了选择和循环结构的基本形式。第 6 章介绍了过程式编程中的基本构件——函数, 包括函数的定义、形式参数、返回值、lambda 表达式、生成器等内容。

第二篇“Python 高级特性”, 包含第 7~9 章。第 7 章介绍了面向对象编程的基本单位——类。第 8 章从程序运行的角度介绍了异常处理结构。第 9 章从数据存储角度介绍了文件的常见操作。

第三篇“Python 数据分析与处理”, 包含第 10~13 章, 从数据科学家的角度介绍了 Python 科学计算扩展库的使用。第 10 章介绍了数据可视化技术 Matplotlib 和 Seaborn, 第 11 章介绍了矩阵和向量处理模块 Numpy, 第 12 章介绍了数据分析和处理模块 Pandas, 第 13 章介绍了网页数据的爬取。

本书遵循“**体系完整, 实用性强, 案例丰富, 让教和学更轻松**”的原则组织编写, 内容上以实用、易理解为主, 兼顾深度与广度。在讲述知识点时, 提供了大量的案例, 以加深读者对知识点的理解。同时本书为了学习方便, 还提供了授课 PPT、实验指导书和课后习题。读者在学习的过程中, 要坚持“**抓概念、抓思想、抓应用**”的基本思路, 在实践中理解知识点, 同时能将知识点应用到实际当中。

Python 的扩展库非常多, 由于篇幅所限, 本书不能全部涉及, 读者可以根据自己所从

事工作的需要，利用所介绍的基础知识，有针对性地选择相关扩展库来学习。

本书由多位老师分工完成：于长青负责第1章、第5章内容的编写，张文胜负责第4章、第7~9章内容的编写，范晖负责第2~3章、第6章、第10~13章内容的编写。全书由范晖负责策划、审校和定稿。

本书编写过程中得到了很多人的帮助。北京天融信教育科技有限公司的企业专家对本书结构和内容提出了很多有益的建议，并通过“教育部-天融信产学研合作协同育人”项目在课程建设方面提供了大力支持。西安电子科技大学出版社李惠萍编辑对本书的出版提供了很多意见和建议，在此表示衷心的感谢。

本书参考了很多Python语言方面的网络资源、书籍资料，在此向这些作者一并致谢。由于时间仓促，加之作者的水平和能力有限，书中的疏漏与不妥之处在所难免，衷心希望各位同行和读者批评指正。

作 者

2019年5月于西安

目 录

第一篇 Python 基础知识

第 1 章 初识 Python.....2	2.4.6 赋值运算符.....27
1.1 Python 的发展历史.....2	2.5 表达式.....28
1.2 Python 的特点和应用.....3	2.6 常用函数.....28
1.3 Python 的安装.....4	2.6.1 内置函数.....28
1.3.1 Windows 下安装 Python.....4	2.6.2 模块函数.....30
1.3.2 Linux 下安装 Python.....7	2.7 Python 程序基本结构.....31
1.3.3 Mac OS X 下安装 Python.....8	2.7.1 物理行和逻辑行.....32
1.3.4 环境变量的配置.....8	2.7.2 语句分隔.....32
1.4 Python 程序的运行方式.....8	2.7.3 缩进.....32
1.5 Python 库的使用.....10	2.7.4 注释.....33
1.5.1 扩展库的管理.....10	2.8 基本输入输出.....33
1.5.2 模块的导入与使用.....10	2.8.1 input 函数.....34
1.6 程序组成和编码规范.....12	2.8.2 print 函数.....34
1.7 案例实战.....13	2.9 案例实战.....35
本章小结.....15	本章小结.....36
课后习题.....16	课后习题.....36
第 2 章 Python 语言基础.....17	第 3 章 序列结构.....37
2.1 标识符与关键字.....17	3.1 序列概述.....37
2.1.1 标识符.....17	3.2 列表.....38
2.1.2 关键字.....18	3.2.1 列表的创建和删除.....38
2.2 变量.....18	3.2.2 列表的赋值和拷贝.....38
2.2.1 对象和类型.....18	3.2.3 列表的常用操作.....40
2.2.2 变量的创建.....19	3.3 元组.....48
2.2.3 变量的删除.....20	3.3.1 元组的创建和删除.....48
2.3 数据类型.....21	3.3.2 元组的基本操作.....49
2.4 运算符.....23	3.3.3 生成器推导式.....50
2.4.1 算术运算符.....23	3.4 字典.....51
2.4.2 逻辑运算符.....25	3.4.1 字典的创建和删除.....51
2.4.3 关系运算符.....25	3.4.2 字典的赋值和拷贝.....52
2.4.4 位运算符.....26	3.4.3 字典的基本操作.....52
2.4.5 矩阵相乘运算符.....27	3.5 集合.....54

3.5.1 集合的创建和删除	54	5.2.2 双分支选择结构	72
3.5.2 集合的赋值和拷贝	55	5.2.3 多分支选择结构	73
3.5.3 集合的基本操作	55	5.2.4 选择结构的嵌套	74
3.6 元组的封装与序列的拆封	56	5.3 循环结构	75
3.7 案例实战	57	5.4 break 和 continue 语句	77
本章小结	58	5.5 案例实战	77
课后习题	58	本章小结	79
第4章 字符串	59	课后习题	79
4.1 字符串的编码方式	59	第6章 自定义函数	81
4.2 字符串的表示形式	60	6.1 函数的定义	81
4.3 字符串的基本操作	61	6.2 函数的调用	82
4.3.1 字符串的访问方式	61	6.3 函数的参数	83
4.3.2 字符串的转义	62	6.3.1 位置参数	84
4.3.3 基本操作符	63	6.3.2 默认值参数	84
4.4 字符串的方法	63	6.3.3 关键字参数	85
4.5 字符串常量	67	6.3.4 可变长度参数	85
4.6 字符串的格式化	67	6.4 函数的返回值	86
4.6.1 格式化表达式	67	6.5 lambda 表达式	86
4.6.2 format()方法	68	6.6 生成器	88
4.7 案例实战	69	6.7 装饰器	88
本章小结	70	6.8 变量的作用域	89
课后习题	70	6.9 函数的递归	90
第5章 流程控制	71	6.10 案例实战	92
5.1 条件表达式	71	本章小结	93
5.2 选择结构	71	课后习题	93
5.2.1 单分支选择结构	71		

第二篇 Python 高级特性

第7章 面向对象编程	96	7.5.2 多继承	103
7.1 类和对象	96	7.6 多态	104
7.2 属性和方法	97	7.7 案例实战	105
7.2.1 属性	97	本章小结	108
7.2.2 方法	98	课后习题	108
7.3 构造方法和析构方法	99	第8章 异常处理	110
7.3.1 构造方法	99	8.1 错误与异常	110
7.3.2 析构方法	100	8.2 异常类	111
7.4 封装	101	8.3 异常处理	111
7.5 继承	102	8.3.1 捕获指定异常	111
7.5.1 单继承	102	8.3.2 捕获多个异常	112

8.3.3 未捕获到异常	113	9.1.2 文件的关闭	120
8.3.4 try...except...finally 语句	114	9.2 文本文件的读写	121
8.4 自定义异常和抛出异常	114	9.2.1 写文件	121
8.5 断言	115	9.2.2 读文件	122
8.6 案例实战	116	9.3 二进制文件的读写	124
本章小结	117	9.4 文件的操作	125
课后习题	117	9.5 目录的操作	127
第 9 章 文件操作	119	9.6 案例实战	128
9.1 文件的打开和关闭	119	本章小结	129
9.1.1 文件的打开	119	课后习题	129

第三篇 Python 数据分析与处理

第 10 章 数据可视化技术	132	11.1.3 数组的组合和分割	164
10.1 pyplot 基本绘图流程	132	11.1.4 自动生成数组	166
10.2 基于函数的可视化操作	132	11.1.5 随机数函数	167
10.2.1 常用绘图函数	133	11.1.6 数组索引和切片	168
10.2.2 绘制多个子图	135	11.2 数组运算	172
10.3 基于对象的可视化操作	136	11.2.1 创建 Numpy 矩阵	172
10.4 配置文件	137	11.2.2 矩阵运算	173
10.5 中文显示	138	11.2.3 通用函数	174
10.6 分类图	139	11.2.4 统计函数	175
10.6.1 对数坐标图	139	11.2.5 线性代数	176
10.6.2 极坐标图	140	11.3 数组的存取	176
10.6.3 直方图	141	11.4 案例实战	177
10.6.4 柱状图	141	本章小结	178
10.6.5 饼状图	143	课后习题	178
10.6.6 散点图	144	第 12 章 Pandas 基础与实战	180
10.6.7 箱线图	144	12.1 Pandas 数据结构	180
10.6.8 三维绘图	145	12.1.1 Series	180
10.7 Seaborn 可视化	147	12.1.2 DataFrame	181
10.7.1 Seaborn 样式	147	12.2 Pandas 索引操作	182
10.7.2 分类图	148	12.2.1 重新索引	182
10.8 案例实战	157	12.2.2 更换索引	184
本章小结	160	12.3 数据选择	186
课后习题	161	12.3.1 索引与切片	186
第 11 章 Numpy 基础与实战	162	12.3.2 操作行与列	190
11.1 多维数组对象 ndarray	162	12.4 数据运算	191
11.1.1 创建 ndarray 对象	162	12.4.1 算术运算	191
11.1.2 变换数组的形状	163	12.4.2 函数应用与映射	193

12.4.3 排序	194	13.3.1 HTTP 请求	217
12.4.4 统计信息	195	13.3.2 HTTP 响应	219
12.4.5 唯一值与值计数	196	13.3.3 requests 库	219
12.5 数据清洗	197	13.4 网页解析基础	221
12.5.1 处理缺失值	197	13.4.1 HTML 简介	221
12.5.2 处理重复值	199	13.4.2 XPath 简介	223
12.5.3 替换值	200	13.4.3 正则表达式	225
12.6 数据分组	201	13.5 BeautifulSoup 库的使用	227
12.7 聚合运算	204	13.5.1 快速开始	228
12.7.1 聚合运算方法	204	13.5.2 对象类型	228
12.7.2 多函数应用	206	13.5.3 遍历文档树	229
12.8 数据的读取与存储	207	13.5.4 搜索文档树	230
12.8.1 文本数据的读取与存储	207	13.5.5 爬虫实例	232
12.8.2 Excel 数据的读取与存储	210	13.6 lxml 库的使用	235
12.9 案例实战	210	13.6.1 基本用法	235
本章小结	214	13.6.2 高级用法	236
课后习题	214	13.6.3 lxml 爬虫实例	237
第 13 章 网络爬虫基础与实战	216	13.7 Scrapy 爬虫框架	239
13.1 爬虫系统的架构	216	13.7.1 Scrapy 的安装	240
13.2 常用的爬虫技术	216	13.7.2 Scrapy 爬虫步骤	240
13.2.1 实现 HTTP 请求	216	13.7.3 Scrapy 爬虫实现	242
13.2.2 实现网页解析	217	13.8 案例实战	245
13.2.3 爬虫框架	217	本章小结	247
13.3 爬虫基础	217	课后习题	247
参考文献	248		

第一篇 Python 基础知识

学习 Python 语言,重点要关注 Python 下基本数据类型和函数的使用。Python 提供了整数、小数、复数、布尔值等原子类型和列表、元组、字典、集合、字符串等容器类型。熟悉和掌握这些基本数据类型对 Python 下的编程非常重要。Python 既支持面向过程编程,也支持面向对象编程。函数是面向过程编程中很重要的一个概念,Python 提供了众多的内置函数,掌握这些内置函数的使用对编写 Python 风格的程序至关重要,另外还可以自定义函数。Python 下的函数在形参和返回值方面比其他语言要灵活得多。利用函数,可以编写功能强大的程序。

本篇共 6 章,重点对基本数据类型、选择、循环和函数进行详细介绍。其中:

第 1 章: 初识 Python

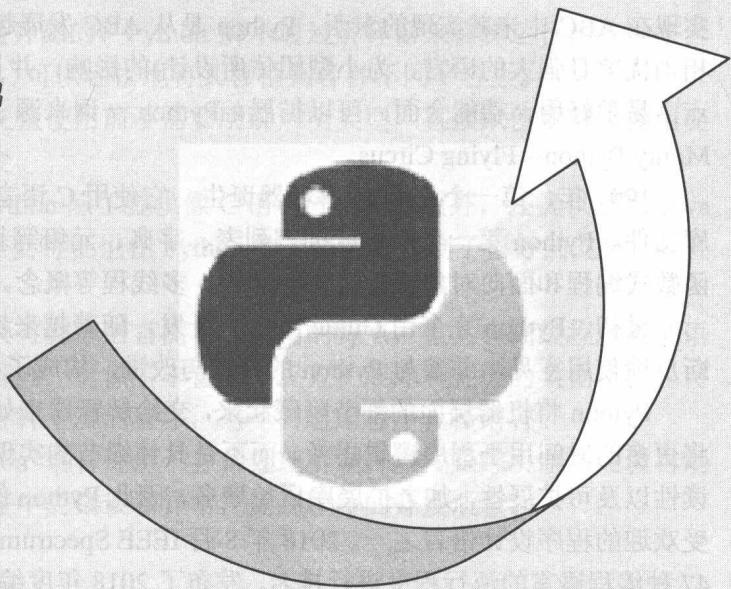
第 2 章: Python 语言基础

第 3 章: 序列结构

第 4 章: 字符串

第 5 章: 流程控制

第 6 章: 自定义函数



第 1 章 初识 Python

Python 是一种解释型、面向对象、动态数据类型的高级程序设计语言，由 Guido van Rossum 于 1989 年开始开发，并于 1991 年发布第一个公开版。Python 提供了非常完善的标准库，覆盖了网络、文件、GUI、数据库、科学计算等大量内容。除此之外，Python 还有大量的第三方库。Python 程序简单易懂，易于初学者学习。

本章介绍 Python 的特点、安装、程序的运行、扩展库的安装、Python 程序的组成和编码规范。通过本章学习，读者可以对 Python 语言有一个初步的认识。

1.1 Python 的发展历史

1989 年圣诞节期间，Python 的创始人 Guido van Rossum 在阿姆斯特丹为了打发圣诞节假期，决心开发一个新的脚本解释程序，作为 ABC 语言的一种继承。

ABC 是由 Guido 参与设计的一种专门为非专业程序员设计的教学语言。ABC 语言非常优美和强大，具备良好的可读性和易用性，但是没有成为流行语言。Guido 认为 ABC 未获得成功主要是其非开放性造成的。Guido 决心在 Python 中避免这一错误，同时，他还想实现在 ABC 中未曾实现的东西。Python 是从 ABC 发展起来的，主要受到了 Modula-3(一种相当优美且强大的语言，为小型团体所设计)的影响，并且结合了 Unix shell 和 C 语言的特点，易学好用，功能全面，可以扩展。Python 一词来源于 Guido 所挚爱的英国肥皂剧——Monty Python's Flying Circus。

1991 年，第一个 Python 解释器诞生，它使用 C 语言来实现，可以调用 C 语言编写的库文件。Python 第一个版本就拥有列表、字典、元组等基本数据类型，支持命令式编程、函数式编程和面向对象编程，支持异常、多线程等概念。

最初，Python 完全由 Guido 一个人开发，随着越来越多的同事使用这门语言，他们不断反馈使用意见，并参与 Python 的开发与改进，构成了 Python 的核心团队。

Python 将机器层面的细节隐藏起来，交给解释器来处理，程序员在使用 Python 时可以将更多的时间用于程序逻辑思考，而不是具体细节的实现。由于 Python 语言的简洁性、易读性以及可扩展性，加之扩展库日益增多，因此 Python 的使用率呈线性增长，已经成为最受欢迎的程序设计语言之一。2018 年 8 月 IEEE Spectrum 综合了 9 个来源的 11 个指标，对 47 种编程语言的流行程度进行排名，发布了 2018 年度编程语言排行榜，Python 雄踞第一，在综合指数、用户增速、就业优势和开源语言等单项中，全部霸占榜首。

随着物联网、云计算、大数据和人工智能的兴起,Python 在这些领域的应用与日俱增。国内外越来越多的研究机构使用 Python 语言来做科学计算,许多大学增加了 Python 程序设计课程。Python 不但标准库功能强大,而且众多开源的第三方库都提供了 Python 的调用接口,例如著名的计算机视觉库 OpenCV、三维可视化库 VTK、医学图像处理库 ITK,经典的科学计算扩展库 Numpy、Scipy、Matplotlib、Pandas 和 Scikit-learn。Python 现在已经成为编程新手和软件架构师们都偏爱的一门高级编程语言。

1.2 Python 的特点和应用

Python 是一门开源、跨平台、解释型的高级动态编程语言,具有 Shell 脚本的交互式操作和 C 语言的强大功能,语法精简,支持函数和类编程,拥有大量的扩展库,可以像“胶水”一样,将多种语言编写的程序融合到一起实现无缝拼接。

1. Python 的特点

Python 具有以下特点:

(1) 易于扩展。作为一门解释型语言,Python 脚本等同于可执行的代码,创建一个.py 文件并写入代码,就可以作为新的功能模块来使用。另外,Python 脚本支持 C 语言扩展,可以嵌入 C 语言开发的项目中。同时,也可以调用其他语言编写的代码,因此 Python 也被称为胶水语言。

(2) 语法简洁。Python 语言语法简洁、代码易读,抛弃了其他语言中的大括号、begin 和 end 等标记,不需要使用分号,代码使用空格或者水平制表符来分割,支持使用循环和条件语句进行数据结构的初始化。这些设计使得 Python 程序短小精悍,并且具有很高的可读性。

(3) 具有可移植性。Python 是开源的,Python 脚本通过解释器来解释运行,各种平台下都有非常完善的 Python 解释器,可以轻松地将移植到各种不同平台上。

(4) 属于动态语言。Python 变量使用前不需要明确声明类型,直接赋值就可以使用变量,而且变量的类型可以动态改变。

(5) 具有面向对象的特性。Python 除了提供像 C 语言一样的函数外,还提供了像 Java 语言一样的类。面向对象编程的主要特征也在 Python 的类模块中得到了很好的支持,而且变得极为简单,比其他语言容易。

(6) 功能强大。Python 语言具有脚本语言简单、易用的特点,也具有高级程序设计语言的强大功能。它支持自动内存管理,提供了丰富的第三方扩展库和强大的数据结构,可以应用到各种不同的领域,开发各种主流的程序,是一门通用程序设计语言。

(7) 具有较好的健壮性。Python 语言提供了异常处理机制、堆栈跟踪机制和垃圾自动回收机制,可帮助程序员定位程序中的错误和异常,提高程序的开发质量。

2. Python 的应用

作为一门优秀的高级程序设计语言,Python 被广泛应用于各种领域。常见的应用领域如下:

(1) 系统管理。Python 可以访问操作系统的 API，标准库集成了 POSIX(可移植操作系统接口)和其他常见操作系统工具，通过这些可以方便地编写系统维护和管理工具。

(2) GUI 编程。Python 的 GUI 工具集除了标准库 tkinter，还有功能强大的 wxPython、PyQt、PySide 等，可以非常简单、快捷地实现 GUI 程序的开发。

(3) Web 服务开发。Python 可以通过套接字进行网络通信，可以生成、解析和分析 XML 文件，可以处理 E-mail，可以通过 URL 获取网页内容，对网页进行解析。借助于 Django、Web2py 等框架，可以快速开发网站应用程序。

(4) 数据库编程。Python 提供了访问各种主流数据库的 API，包括 SQLite、Access、MySQL、SQL Server、Oracle 等。

(5) 数值运算和科学计算。Python 提供了数值计算库 Numpy、科学计算函数库 SciPy、数学符号运算扩展库 Sympy、数据分析和操作扩展库 Pandas 等。相比于著名的科学计算商业软件 MATLAB，Python 是完全免费的，而且是一门更容易学、更加严谨的程序设计语言。

(6) 多媒体设计。通过 Python 的 Pygame 扩展库可以进行图形和游戏的应用开发，而 PIL、PyOpenGL、Maya 等扩展库则提供了 3D 应用的开发接口。

(7) 人工智能。Python 被称为人工智能的专用语言，Python 下众多的开源框架对人工智能应用领域提供了强大的支持，如计算机视觉库 OpenCV、机器学习框架 TensorFlow 等。

(8) 网络爬虫。在爬虫领域，Python 几乎处于霸主地位，提供了 Scrapy、requests、BS 等工具库，可以对网页数据进行采集和处理。

1.3 Python 的安装

Python 支持众多的软件平台，例如 Windows、Linux/Unix 和 Mac OS X 等，Python 编写的程序虽然可以跨平台运行，但是在不同平台上 Python 的安装方法是不同的。

在安装之前，我们先来了解一下 Python 的安装版本。目前 Python 有两个系列的版本，一个是 2.x 版，一个是 3.x 版，这两个版本是互不兼容的。在 3.x 版中，一些语法、内置函数和对象的方法有所调整，3.x 版越来越普及，因此本书将以最新的 Python 3.7.2 版本为基础进行讲解。

最简单的安装方法是从 Python 的官网(<https://www.python.org/>)上下载安装程序。

1.3.1 Windows 下安装 Python

由于本书是在 Windows 操作系统下进行编写的，因此我们以 Windows 下的 Python 安装为例进行讲解。

(1) 打开 <https://www.python.org/> 网址，在【Downloads】菜单下选择 Windows 平台下的安装包。本次我们下载的文件为 python-3.7.2.exe。具体参见图 1-1。

(2) 双击 python-3.7.2.exe 进入 Python 安装界面，如图 1-2 所示。注意安装界面底部的复选框，第一个默认自动勾选，第二个默认不自动勾选，需要手动勾选。【Add Python 3.7 to PATH】可以将 Python 的安装路径添加到环境变量 Path 中，勾选后可以免去安装完成后的

手工添加，建议在安装时勾选。

在图 1-2 中有两种安装模式。第一种是默认安装模式，所有的选项会按照默认值的方式进行自动设置。第二种是自定义安装模式，用户可以自定义安装路径等。这两种模式可以根据需要进行选择。本书以自定义安装为例进行讲解。

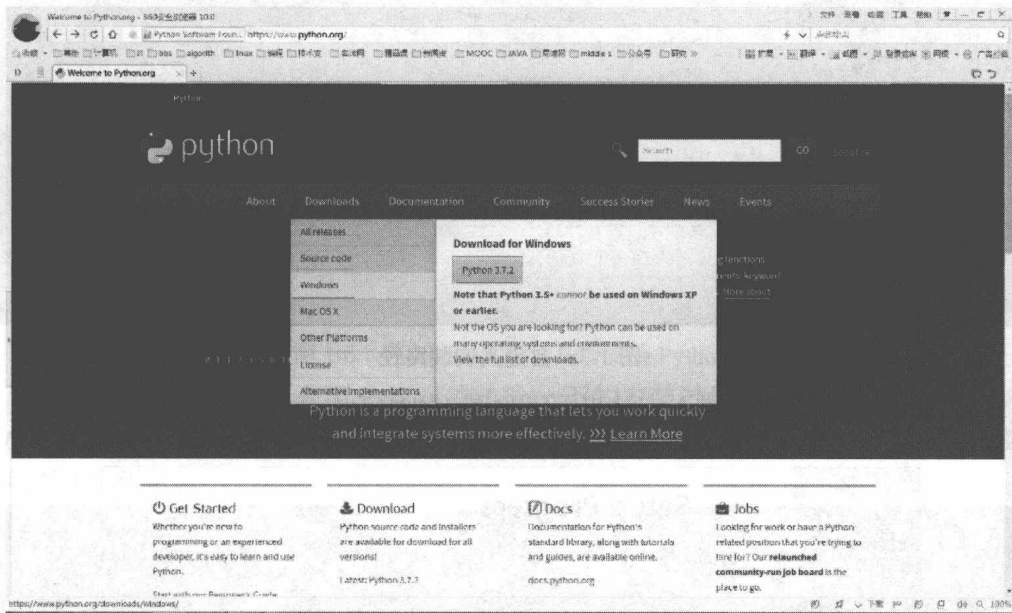


图 1-1 下载 Python 安装包

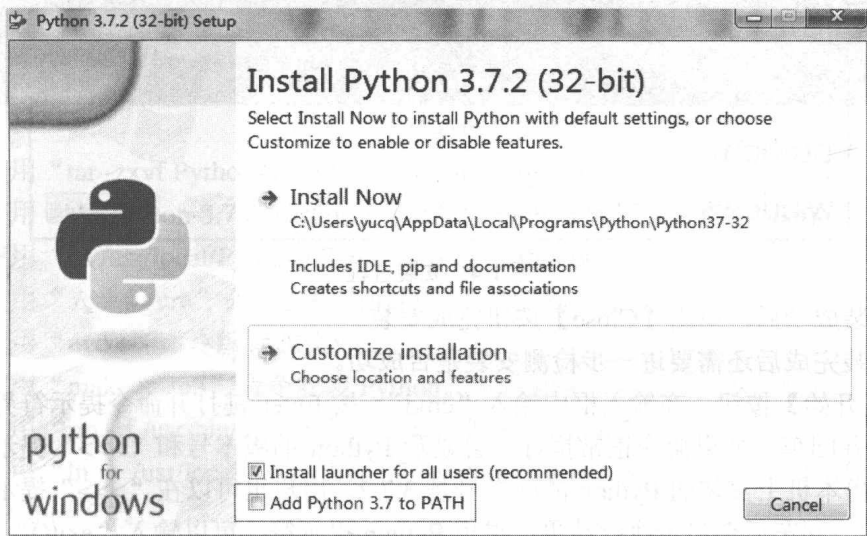


图 1-2 Python 安装界面

(3) 单击【Customize installation】选项，进入下一安装界面，单击【Next】按钮进入下一步。

(4) 在图 1-3 中，可以单击【Browse】按钮，指定安装路径。本书指定安装路径为 D:\Program Files (x86)\Python\Python37-32。

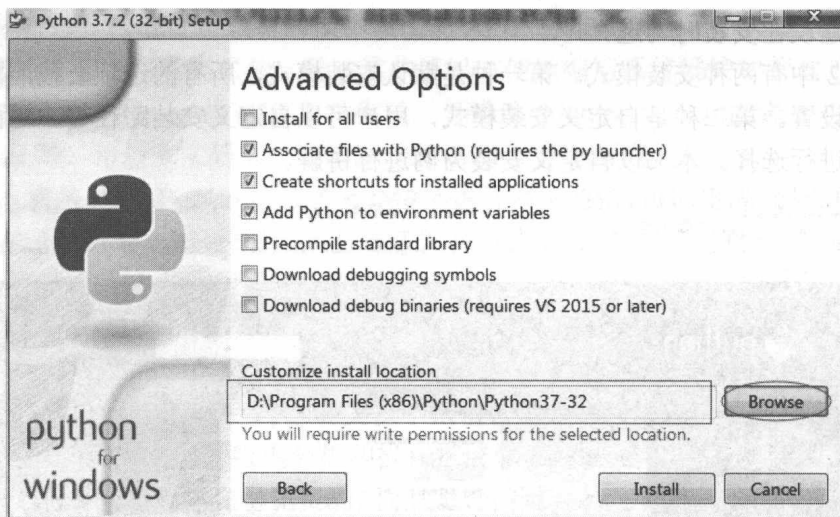


图 1-3 自定义安装路径

(5) 单击【Install】按钮进入安装界面，如图 1-4 所示。

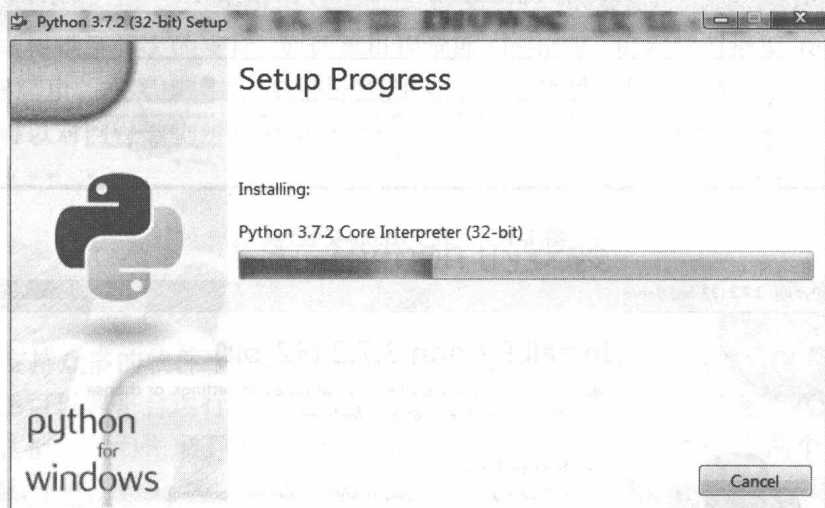


图 1-4 安装进程

(6) 安装成功后，单击【Close】按钮完成安装。

(7) 安装完成后还需要进一步检测安装是否成功。

单击【开始】按钮，在输入框中输入“cmd”，按 Enter 键打开命令提示符界面，输入“python”并回车。如果命令正常执行，会显示 Python 的版本号和“>>>”提示符。从图 1-5 可以看出本机上安装的 Python 的版本号是 3.7.2。用户也可以在“>>>”提示符下输入 Python 语句，回车后会显示执行结果。退出 Python 提示符，可以输入“exit()”。

```
C:\Users\yucq>python
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32 bit
<Intel>] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

图 1-5 Python 命令行

1.3.2 Linux 下安装 Python

目前绝大多数 Linux 操作系统默认已经安装了 Python，如图 1-6 所示，可以通过输入“python”命令进行验证。如果版本太低，还必须重新安装新版本。

```
CentOS Linux 7 (Core)
Kernel 3.10.0-862.el7.x86_64 on an x86_64

localhost login: root
Password:
Last login: Sun Dec 23 05:35:29 on tty1
[root@localhost ~]# python
Python 2.7.5 (default, Apr 11 2018, 07:36:10)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

图 1-6 在 Linux 下验证是否安装 Python

下面以 Centos 7 为例，介绍 Python 在 Linux 下的安装过程。

(1) 使用“`wget https://www.python.org/ftp/python/3.7.2/Python-3.7.2.tgz`”命令下载安装包，如图 1-7 所示。

```
[root@66c88192 ~]# wget https://www.python.org/ftp/python/3.7.2/Python-3.7.2.tgz
--2019-01-26 14:19:43-- https://www.python.org/ftp/python/3.7.2/Python-3.7.2.tgz
Resolving www.python.org (www.python.org)... 151.101.100.223, 2a04:4e42:11::223
Connecting to www.python.org (www.python.org|151.101.100.223|:443)... connected.
HTTP request sent, awaiting response... 200 OK
Length: 22897002 (22M) [application/octet-stream]
Saving to: 'Python-3.7.2.tgz'

69% [----->] 15,971,752 22.4KB/s eta 4m 30s
100%[=====>] 22,897,002 21.4KB/s in 16m 13s

2019-01-26 14:19:50 (21.0 KB/s) = 'Python-3.7.2.tgz' saved [22897002/22897002]

[root@66c88192 ~]#
```

图 1-7 下载 Python 安装包

- (2) 使用“`tar -zxvf Python-3.7.2.tgz`”命令解压 tgz 包。
- (3) 使用“`mv Python-3.7.2 /usr/local`”命令把 Python 解压文件移到/usr/local 文件夹下。
- (4) 使用“`cd /usr/local/Python3.7.2/`”命令进入 Python 目录。
- (5) 使用“`./configure`”命令执行当前目录下的配置。
- (6) 使用“`make`”命令编译源文件。
- (7) 使用“`make install`”命令安装 Python。
- (8) 使用“`rm -rf /usr/bin/python`”命令删除原来 Python 2.7 的软链接。
- (9) 使用“`ln -s /usr/local/bin/python3.7 /usr/bin/python`”命令创建新的软链接，链接到新安装的 Python 3.7。
- (10) 使用“python”命令查看是否安装成功，如图 1-8 所示，表示已成功安装 Python 3.7.2。

```
[root@guog ~]# python
Python 3.7.2 (default, Jan 26 2019, 15:11:41)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-28)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

图 1-8 Python 3.7.2 安装成功

1.3.3 Mac OS X 下安装 Python

Mac OS X 系统默认安装了 Python，如果要安装最新版本，从 Python 官网下载 macOS 64-bit/32-bit installer，双击按照提示完成安装即可。

1.3.4 环境变量的配置

在安装 Python 过程中如果没有勾选相关选项，例如【Add Python 3.7 to PATH】选项(配置 Python 解释器的路径)，后续使用中会出现错误提示的情况，这时就需要重新配置 Python 的环境变量。

下面以 Windows 操作系统为例，讲解环境变量 Path 的手动添加。

在桌面选中【计算机】图标，单击鼠标右键，选择【属性】，在弹出的窗口中选择【高级系统设置】，在【系统设置】下选择【环境变量】。

进入【环境变量】设置窗口，选择【系统变量】中的【Path】项目，单击【编辑】按钮，在最后添加“D:\Program Files (x86)\Python\Python37-32\Scripts\;D:\Program Files(x86)\Python\Python37-32\”。

注意：读者一定要根据自己的 Python 安装目录来修改划线部分的内容。具体参见图 1-9。

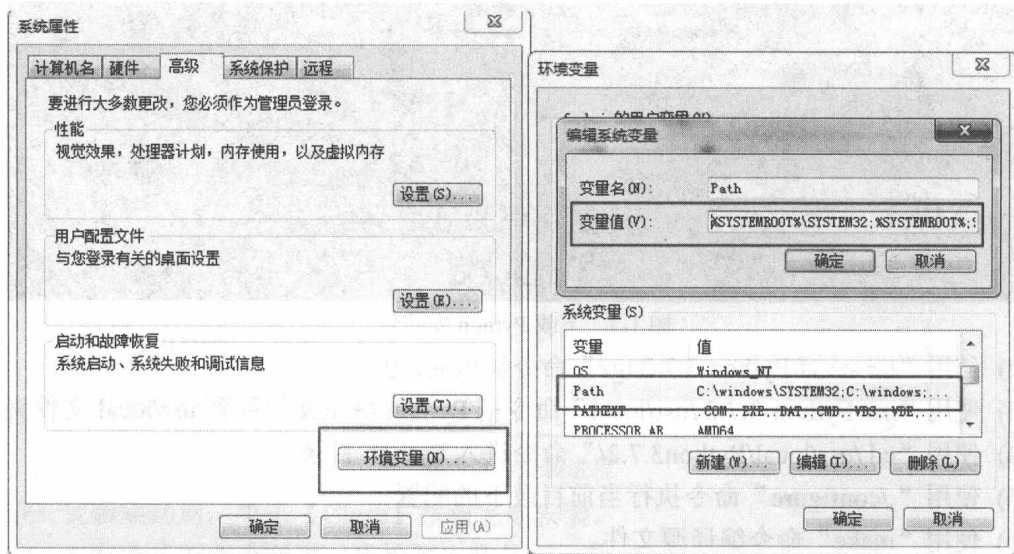


图 1-9 系统环境变量 Path 的配置

1.4 Python 程序的运行方式

从计算机的角度来看，Python 程序的运行过程分为两步：解释器解释和虚拟机运行。执行 Python 程序时，首先由 Python 解释器将.py 文件中的源代码翻译成字节码，再由 Python 虚拟机 PVM 逐条将字节码翻译成机器指令执行，Python 的这种机制和 Java、.NET 类似。

Python 还可以通过交互方式运行。交互模式下，直接输入 Python 语句就可以执行。