

作者多年从事网络爬虫领域的教学及研究工作，有着丰富的实践经验。
面向初学者全面介绍 Python 网络爬虫的实战宝典，涵盖网络爬虫的核心概念、算法和
技术实现，内容系统，案例丰富。



Python 网络爬虫 技术与实战

赵国生 王健 ○ 编著

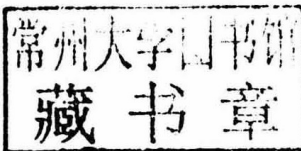
WEB CRAWLER TECHNIQUE AND ACTUAL
APPLICATION BASED ON PYTHON



机械工业出版社
China Machine Press

Python 网络爬虫 技术与实战

赵国生 王健 © 编著



WEB CRAWLER TECHNIQUE
AND ACTUAL APPLICATION
BASED ON PYTHON



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

Python 网络爬虫技术与实战 / 赵国生, 王健编著. —北京: 机械工业出版社, 2021.1

ISBN 978-7-111-67411-5

I. P… II. ①赵… ②王… III. 软件工具—程序设计 IV. TP311.56

中国版本图书馆 CIP 数据核字 (2021) 第 009282 号

Python 网络爬虫技术与实战

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 栾传龙

责任校对: 殷虹

印刷: 北京市荣盛彩色印刷有限公司

版次: 2021 年 1 月第 1 版第 1 次印刷

开本: 186mm×240mm 1/16

印张: 29.5

书号: ISBN 978-7-111-67411-5

定价: 89.00 元

客服电话: (010) 88361066 88379833 68326294

投稿热线: (010) 88379604

华章网站: www.hzbook.com

读者信箱: hzit@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

为什么写作本书

大数据时代已经到来，网络爬虫技术已成为这个时代不可或缺的一项技术，企业需要数据来分析用户行为、产品的不足之处以及竞争对手的信息等，而这一切的首要条件就是数据的采集。在互联网社会中，数据是无价之宝，一切皆为数据，谁拥有了大量有用的数据，谁就拥有了决策的主动权。如何有效地采集并利用这些信息成了一个巨大的挑战，而网络爬虫是自动采集数据的有效手段。网络爬虫是一种按照一定的规则，自动抓取互联网海量信息的程序或脚本。网络爬虫的应用领域很广泛，如搜索引擎、数据采集、广告过滤、大数据分析等。

笔者多年来一直从事网络爬虫相关课程的讲授及科学研究工作，有着丰富的教学和实践经验。在内容编排上，本书采用梯度层次化结构，由浅入深地介绍爬虫的知识点、原理及应用，并结合大量实例讲解操作步骤，使读者能够快速地理解网络爬虫的核心技术。

内容介绍

全书共 14 章，具体内容如下：

第 1 章主要介绍 Python 的安装、配置和基础语法，以及 Python 的字符串、数据结构、控制语句和函数等；

第 2 章主要介绍爬虫的类型、爬虫的抓取策略以及深入学习爬虫所需的网络基础等相关知识；

第 3 章主要对爬虫技术中经常使用到的 urllib、request、lxml 和 BeautifulSoup 库等进行详细介绍，最后展示了 4 个利用 Python 爬取数据的实例；

第 4 章主要对 Python 中正则表达式的语法、匹配规则和 re 模块常用函数进行详细阐述，并给出了实例；

第 5 章主要对 3 种主流库（PIL 库、Tesseract 库和 TensorFlow 库）的语法、类型、识别方法和案例进行介绍；

第 6 章详细介绍 Fiddler 的安装与配置、捕获会话、QuickExec 命令行的使用和 Fiddler 的

断点功能等；

第 7 章主要介绍数据存储在文件中和存储在数据库中这两种存储方式；

第 8 章重点介绍 Scrapy 框架的 Selector 用法，以及 Beautiful Soup 库和 CrawlSpider 的使用，然后介绍了 Scrapy Shell 和 Scrapyrt 的使用；

第 9 章主要介绍多线程和 Threading 模块的基本概念；

第 10 章主要介绍如何对动态网页进行信息爬取，首先介绍了浏览器开发工具的使用，然后介绍了异步加载技术、AJAX 技术和 Selenium 模拟浏览器；

第 11 章主要介绍分布式爬虫的原理及实现过程，然后介绍了 Scrapy-redis 分布式组件的工作机制和安装配置；

第 12 章主要介绍如何利用 Selenium 抓取并用 pyquery 解析电商网站的商品信息，然后将其保存到 MongoDB；

第 13 章主要介绍静态网页和动态网页的爬取方法，并对请求 - 响应关系进行了介绍，然后介绍了请求头和请求体；

第 14 章主要讲解如何通过 urllib 模块和 Scrapy 框架实现图片爬虫项目，以及利用 TensorFlow、KNN 和 CNN 等机器学习框架进行训练的方法与过程。

主要特点

本书针对网络爬虫学习的特点，结合作者多年使用网络爬虫的教学和实践经验，由浅入深、从简到繁、图文并茂地介绍了 Python 基础语法、爬虫原理、爬虫常用库模块、正则表达式、验证码识别、抓包工具 Fiddler、数据存储、Scrapy 爬虫框架、多线程爬虫、动态网页爬虫和分布式爬虫等方面的内容。本书内容条理清晰、针对性强，语言通俗易懂，在讲解的过程中配合大量的实例操作，符合读者的学习习惯。每章都是从基础知识开始介绍，然后是实例分析，最后附以练习题巩固学习效果，将理论与实践紧密结合。

具体来讲，本书具有以下鲜明的特点：

- 内容系统，由浅入深；
- 案例讲解，通俗易懂；
- 综合实战，注重实践。

读者对象

本书适合网络爬虫初学者，以及具有一定网络爬虫基础，但希望更深入地了解、掌握爬虫原理与应用的中级读者阅读。

本书可以作为本科或者大专院校网络安全、电子信息、数据科学、网络工程等相关专业的教材，也可作为从事网络爬虫相关工作的科研或者工程技术人员的参考书。

致谢

本书由哈尔滨师范大学的赵国生和哈尔滨理工大学的王健编写。其中，赵国生主要负责第1~11章的编写，王健负责第12~14章的编写。参与本书大量辅助性工作的研究生有邹伊凡、刘冬梅、张婧婷、廖玉婷、晁绵星、谢宝文等，在此表示感谢。

特别感谢以下项目对本书的支持：国家自然科学基金项目“可生存系统的自主认知模式研究”（61202458）、国家自然科学基金项目“基于认知循环的任务关键系统可生存性自主增长模型与方法”（61403109）、高等学校博士点基金项目（20112303120007）、哈尔滨市科技创新人才研究专项（2016RAQXJ036）和黑龙江省自然科学基金（F2017021）。

感谢您选择本书，虽然笔者在编写过程中力求叙述准确、完善，但由于水平有限，书中仍可能存在欠妥之处，希望您可以把对本书的意见和建议告诉我们。

最后，再次希望本书能够对您的工作和学习有所帮助！

目 录 Contents

前言

第 1 章 Python 环境搭建及基础学习 ··· 1

1.1 Python 3.6 的安装与配置 ····· 1
1.1.1 Windows 下的安装 ····· 1
1.1.2 Linux 下的安装 ····· 5
1.1.3 macOS 下的安装 ····· 6
1.2 IDE 工具: PyCharm 的安装 ····· 7
1.3 基础语法 ····· 11
1.3.1 第一个 Python 程序 ····· 11
1.3.2 Python 命名规范 ····· 13
1.3.3 行和缩进 ····· 15
1.3.4 注释和续行 ····· 15
1.3.5 Python 输出 ····· 16
1.4 字符串 ····· 18
1.4.1 字符串运算符 ····· 18
1.4.2 字符串内置函数 ····· 19
1.5 数据结构 ····· 22
1.5.1 列表 ····· 22
1.5.2 元组 ····· 25
1.5.3 集合 ····· 27
1.5.4 字典 ····· 29

1.6 控制语句 ····· 31
1.6.1 条件表达式 ····· 31
1.6.2 选择结构 ····· 32
1.6.3 循环结构 ····· 33
1.7 函数、模块和包 ····· 36
1.7.1 函数 ····· 36
1.7.2 模块 ····· 40
1.7.3 包 ····· 42
1.8 文件的读写操作 ····· 45
1.8.1 文件读写步骤与打开模式 ····· 46
1.8.2 文件的基本操作 ····· 48
1.8.3 文件写入操作 ····· 52
1.9 面向对象 ····· 53
1.9.1 类和对象 ····· 54
1.9.2 封装性 ····· 58
1.9.3 继承性 ····· 59
1.9.4 多态性 ····· 60
1.10 本章小结 ····· 61
练习题 ····· 61

第 2 章 爬虫原理和网络基础 ····· 62

2.1 爬虫是什么 ····· 62

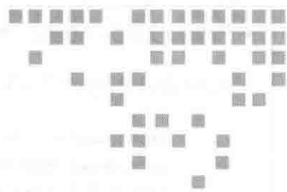
2.2	爬虫的意义	62	3.2.3	URL 编码和 URL 解码	96
2.3	爬虫的原理	64	3.2.4	urlparse() 和 urlsplit() 函数用法	97
2.4	爬虫技术的类型	66	3.3	request 库	99
2.4.1	聚焦爬虫技术	66	3.3.1	request 库的基本使用	99
2.4.2	通用爬虫技术	67	3.3.2	request 库的高级用法	109
2.4.3	增量爬虫技术	69	3.4	lxml 库	113
2.4.4	深层网络爬虫技术	70	3.4.1	lxml 库的安装和使用	113
2.5	爬虫抓取策略	71	3.4.2	XPath 介绍	114
2.5.1	深度优先遍历策略	71	3.4.3	XPath 语法	116
2.5.2	广度优先遍历策略	71	3.4.4	lxml 和 XPath 的结合使用	119
2.5.3	Partial PageRank 策略	72	3.5	Beautiful Soup 库	122
2.5.4	大站优先策略	72	3.5.1	Beautiful Soup 库的安装和使用	123
2.5.5	反向链接数策略	73	3.5.2	提取数据	125
2.5.6	OPIC 策略	73	3.5.3	CSS 选择器	131
2.6	反爬虫和反反爬虫	73	3.6	实战案例	134
2.6.1	反爬虫	73	3.6.1	使用 Beautiful Soup 解析网页	134
2.6.2	反反爬虫	77	3.6.2	微信公众号爬虫	135
2.7	网络基础	79	3.6.3	爬取豆瓣读书 TOP500	136
2.7.1	网络体系结构	79	3.6.4	使用 urllib 库爬取百度贴吧	137
2.7.2	网络协议	79	3.7	本章小结	139
2.7.3	Socket 编程	86		练习题	139
2.8	本章小结	88			
	练习题	88			
第 3 章	Python 常用库	89	第 4 章	正则表达式	140
3.1	Python 库的介绍	89	4.1	概念介绍	140
3.1.1	常用标准库	89	4.2	正则表达式语法	141
3.1.2	安装使用第三方库	91	4.2.1	正则模式的字符	141
3.2	urllib 库	92	4.2.2	运算符优先级	142
3.2.1	urlopen() 函数用法	93	4.3	匹配规则	143
3.2.2	urlretrieve() 函数用法	95	4.3.1	单字符匹配规则	143
			4.3.2	多字符匹配规则	144
			4.3.3	边界匹配	146

4.3.4 分组匹配	147	5.7 本章小结	199
4.4 re 模块常用函数	150	练习题	199
4.4.1 re.match 函数	150	第 6 章 抓包利器 Fiddler	200
4.4.2 re.search 函数	152	6.1 Fiddler 简介	200
4.4.3 re.compile 函数	153	6.2 Fiddler 的安装和配置	200
4.4.4 re.sub 函数	155	6.2.1 Fiddler 的安装	201
4.4.5 re.findall 函数	156	6.2.2 Fiddler 的配置	202
4.4.6 re.finditer 函数	157	6.3 Fiddler 捕获会话	205
4.4.7 re.split 函数	157	6.4 QuickExec 命令行的使用	207
4.5 本章小结	158	6.5 Fiddler 断点功能	209
练习题	158	6.6 Fiddler 的实用工具	210
第 5 章 验证码	159	6.7 实战案例	212
5.1 PIL 库	159	6.7.1 使用 Fiddler 抓取数据并分析	212
5.1.1 PIL 库的安装	159	6.7.2 使用 Fiddler 抓取 HTTPS 流量	214
5.1.2 PIL 库的常用函数	160	6.7.3 使用 Fiddler 抓取手机应用	215
5.1.3 PIL 库的应用	163	6.8 本章小结	219
5.1.4 应用 PIL 到实际开发	169	练习题	219
5.2 Tesseract 库	172	第 7 章 数据存储	220
5.2.1 Tesseract 库的安装	172	7.1 数据的基本存储	220
5.2.2 Tesseract 库的使用	174	7.1.1 数据存储至 TXT	220
5.2.3 Tesseract 库的识别训练	174	7.1.2 数据存储至 CSV	222
5.3 TensorFlow 库	180	7.1.3 数据存储至 JSON	223
5.3.1 TensorFlow 库的安装	180	7.2 数据存储至 MySQL 数据库	227
5.3.2 TensorFlow 基本操作	184	7.2.1 配置 MySQL 服务	227
5.3.3 TensorFlow 基础架构	186	7.2.2 安装 PyMySQL	228
5.3.4 TensorFlow 创建线性回归模型	189	7.2.3 创建示例项目	230
5.3.5 TensorFlow 识别知乎验证码	190	7.2.4 PyMySQL 基本操作	231
5.4 4 种验证码的解决思路	191	7.3 数据存储至 MongoDB 数据库	231
5.5 OCR 处理验证码	194	7.4 数据存储至 XML	234
5.6 实战案例	195		

7.5 常见数据存储方式的比较	235	8.8 Scrapy 对接 Selenium	262
7.6 本章小结	237	8.9 实战案例	264
练习题	237	8.9.1 Scrapy 知乎信息爬取	264
第 8 章 Scrapy 爬虫框架	238	8.9.2 Scrapy 微博信息爬取	268
8.1 Scrapy 框架介绍	238	8.9.3 Scrapy 机票信息爬取	272
8.2 Scrapy 框架详解	239	8.10 本章小结	274
8.2.1 框架内组件及作用	239	练习题	274
8.2.2 Scrapy 运行流程	240	第 9 章 多线程爬虫	275
8.2.3 数据流向	240	9.1 多线程和 Threading 模块	275
8.3 Scrapy 框架中的 Selector	240	9.1.1 多线程定义和特点	275
8.4 BeautifulSoup 库的使用	246	9.1.2 Threading 模块	276
8.4.1 简单示例	246	9.2 使用 Thread 类创建实例	277
8.4.2 四大对象种类	247	9.2.1 可传递函数的 Thread 类实例	277
8.4.3 遍历文档树	249	9.2.2 可调用的 Thread 类实例	278
8.4.4 搜索文档树	249	9.2.3 派生 Thread 子类	279
8.4.5 CSS 选择器	253	9.3 多线程方法的使用	280
8.5 CrawlSpider 的使用	254	9.3.1 多线程创建	280
8.5.1 Spider 的简单用法	254	9.3.2 多线程冲突及解决	283
8.5.2 CrawlSpider 概述	254	9.3.3 使用 Semaphore 调度线程	285
8.5.3 使用 CrawlSpider 获取 rules	256	9.3.4 生产者 - 消费者模式	286
8.5.4 使用 CrawlSpider 进行模拟 登录	257	9.3.5 共享全局变量及锁机制	288
8.6 Scrapy Shell 的使用	257	9.4 Queue 线程安全队列	289
8.6.1 启动 Scrapy Shell	258	9.5 实战案例	291
8.6.2 功能函数	258	9.5.1 多线程爬取糗事百科	292
8.6.3 Scrapy 对象	258	9.5.2 多线程爬取网站图片	296
8.6.4 Scrapy Shell 示例	258	9.6 本章小结	298
8.7 Scrapyrt 的使用	259	练习题	298
8.7.1 GET 请求	259	第 10 章 动态网页爬虫	299
8.7.2 POST 请求	261	10.1 浏览器开发者工具	299

10.1.1	调试工具的介绍	299	11.5	通过 scrapy_redis 实现分布式爬虫	369
10.1.2	调试工具的使用示例	306	11.6	实战案例	371
10.2	异步加载技术	309	11.7	本章小结	376
10.2.1	异步加载技术介绍	309	练习题		377
10.2.2	AJAX 数据爬取	310			
10.3	表单交互与模拟登录	314	第 12 章 电商网站商品信息爬虫		
10.3.1	表单交互	314	项目		378
10.3.2	模拟登录	315	12.1	商品信息爬虫功能分析	378
10.4	Selenium 模拟浏览器	316	12.1.1	商品信息爬虫接口分析	378
10.4.1	Selenium 操作浏览器	316	12.1.2	商品信息爬虫页面分析	380
10.4.2	Selenium 和 ChromeDriver 的配合使用	332	12.2	商品信息爬虫实现思路	380
10.5	实战案例	337	12.2.1	Selenium 环境配置	380
10.5.1	Selenium 职位信息爬取	338	12.2.2	pyquery 环境配置	381
10.5.2	Selenium 直播平台数据爬取	339	12.3	电商网站商品信息编写实战	381
10.6	本章小结	341	12.3.1	获取电商网站商品信息列表	381
练习题		341	12.3.2	电商网站商品信息列表解析	383
			12.3.3	保存爬取的商品信息	385
			12.3.4	电商网站商品信息的页码 遍历	386
第 11 章 分布式爬虫		342	12.4	pyquery 解析电商网站商品信息	388
11.1	分布式爬虫概述	342	12.4.1	pyquery 调用 CSS 选择器	389
11.1.1	主从分布式爬虫	343	12.4.2	pyquery 使用 parent() 获取 父节点	391
11.1.2	对等分布式爬虫	343	12.4.3	pyquery 遍历商品信息	393
11.2	Scrapy-redis 分布式组件	345	12.4.4	pyquery 获取商品信息 内部文本	395
11.2.1	Scrapy-redis 简介	346	12.4.5	CSS 选择器	398
11.2.2	Scrapy-redis 工作机制	348	12.5	运行代码	399
11.2.3	Scrapy-redis 安装配置	349	12.5.1	爬虫的 Chrome Headless 模式	400
11.2.4	Scrapy-redis 常用配置	356	12.5.2	爬虫对接 Firefox	400
11.2.5	Scrapy-redis 键名介绍	357			
11.2.6	Scrapy-redis 简单示例	357			
11.3	redis 数据库	358			
11.4	Scrapy-redis 源码分析	364			

12.5.3 爬虫对接 PhantomJS	400	13.5.6 完整代码及结果	419
12.6 本章小结	401	13.6 调试与运行	421
练习题	401	13.7 本章小结	422
第 13 章 生活娱乐点评类信息爬虫		练习题	422
项目	402	第 14 章 图片信息类爬虫项目	423
13.1 功能分析	402	14.1 功能分析	423
13.1.1 项目描述	402	14.2 实现思路	423
13.1.2 静态网页抓取	402	14.2.1 urllib 模块的使用	424
13.1.3 动态网页抓取	404	14.2.2 Scrapy 框架的使用	426
13.2 请求 - 响应关系	410	14.3 程序执行	429
13.2.1 请求对象	410	14.4 实战演练图片验证码	429
13.2.2 请求方法	410	14.4.1 开发环境与工具	429
13.3 请求头和请求体	411	14.4.2 Anaconda3 的安装	430
13.3.1 请求头	412	14.4.3 问题步骤	434
13.3.2 响应	412	14.4.4 解决步骤	434
13.3.3 保存响应结果	412	14.4.5 图片预处理代码	435
13.4 通过 Selenium 模拟浏览器抓取	413	14.4.6 图片切割	437
13.4.1 Selenium 的安装	413	14.4.7 KNN 训练	437
13.4.2 Selenium 的实践案例	414	14.4.8 CNN 加载数据	439
13.4.3 Selenium 获取文章的所有 评论	414	14.4.9 训练 CNN 模型	440
13.5 实战演练 Scrapy 框架实例	415	14.4.10 CNN 模型预测	442
13.5.1 编写 spider	416	14.5 调试运行	443
13.5.2 编写 item.py	417	14.6 本章小结	444
13.5.3 为 items 对象赋值	417	练习题	444
13.5.4 编写 pipelines.py	418	练习题答案	445
13.5.5 配置 setting.py	418		



Python 环境搭建及基础学习

Python 是一种跨平台的计算机语言，也是一种解释型的、面向对象和动态数据类型的高级程序设计语言。Python 由吉多·范罗苏姆在 1989 年发明，第一个公开发行的版本出现于 1991 年。近年来，由于 Python 在人工智能和云计算领域的应用，其热度越来越高。自从 Facebook 开源了 PyTorch，Python 在 AI 时代登上“第一语言”的位置已成定局。Google 的 TensorFlow 大部分代码都是由 Python 语言编写的，目前最流行的云计算框架 OpenStack 也是基于 Python 开发的。Python 热度很高的另一个主要原因是 Python 拥有强大到无法想象的标准库和第三方库，无论想进行哪个方向的编程，几乎都能找到相应库的支持。本章将首先介绍 Python 的安装、配置和基础语法，然后介绍 Python 的字符串、数据结构、控制语句和函数等，最后介绍文件读写操作和 Python 面向对象的知识。

1.1 Python 3.6 的安装与配置

Python 官方同时发行和维护着 Python 2.x 和 Python 3.x 两个不同系列的版本，这两个系列的版本之间的很多用法是不兼容的，除了基本的输入、输出方式有所不同，很多内置函数和标准库的用法也有非常大的区别。Python 3.x 的设计理念更加合理、高效和人性化，代码开发和运行效率更高，2015 年底就已经出现 Python 3.x 全面普及和应用的趋势。

本节将分别介绍 Windows、Linux 和 macOS 下的 Python 3.6.x 的安装方法。

1.1.1 Windows 下的安装

1) 在网站 <https://www.python.org/downloads/windows/> 中找到需要的 Python 版本，本书选择的版本是 3.6.5，版本信息如图 1-1 所示。

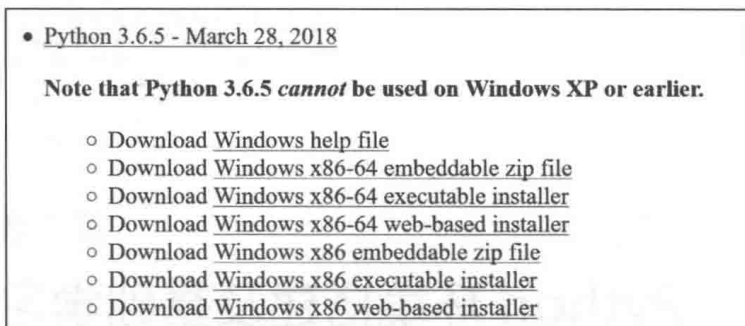


图 1-1 Python 3.6.5 版本信息

2) 运行下载的文件, 出现初始安装界面如图 1-2 所示, 在安装界面中选择 Customize installation 选项。

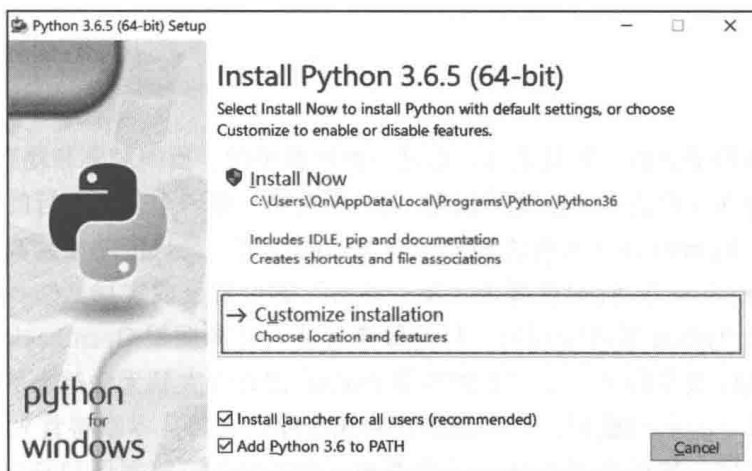


图 1-2 Python 安装界面

具体安装过程如图 1-3、图 1-4 和图 1-5 所示。

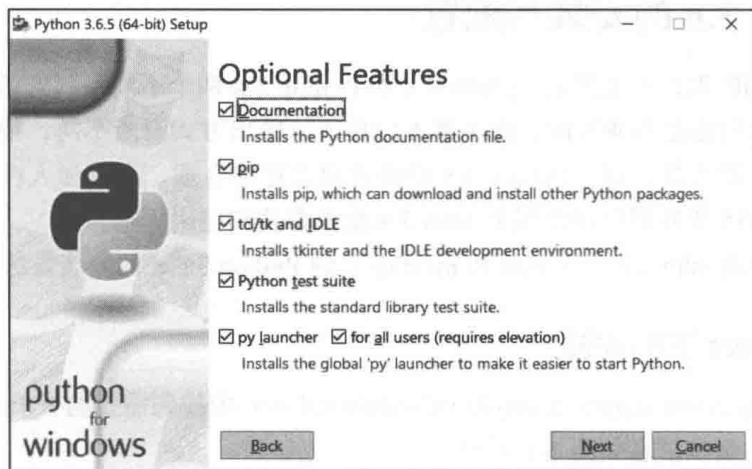


图 1-3 单击“Next”按钮

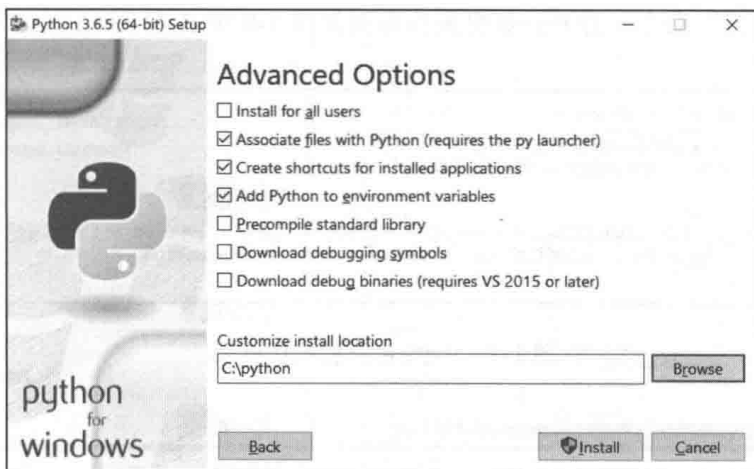


图 1-4 单击“Install”按钮

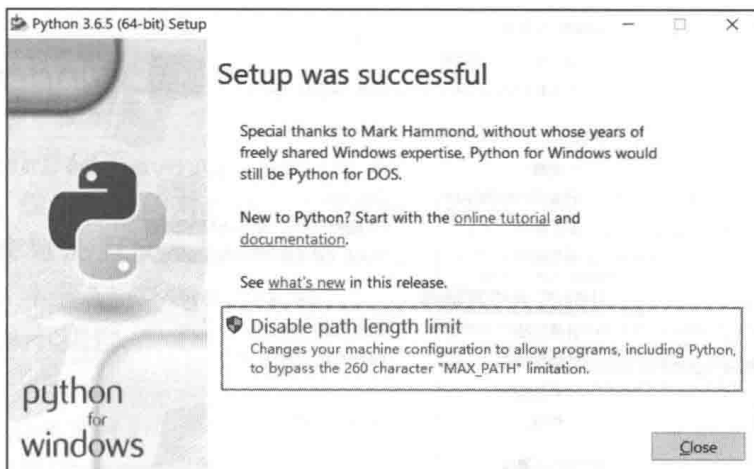


图 1-5 Python 安装完成

3) 运行软件。单击“开始”按钮，在弹出的输入框中输入“cmd”后按 Enter 键，如图 1-6 所示。

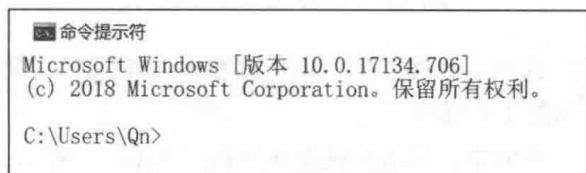
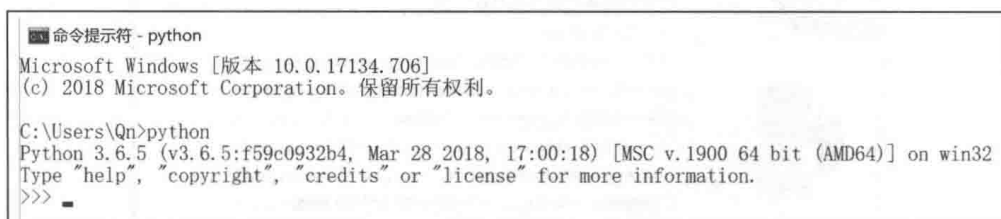


图 1-6 cmd 界面

在 cmd 命令界面中输入“python”后按 Enter 键，成功安装界面如图 1-7 所示。

4) 若没有在初始安装界面(图 1-2)中选择“Add Python 3.6 to Path”，那么程序在 cmd 命令框中就无法正常运行。此时可以手动配置环境变量。鼠标右键单击桌面“计算机”图标，

在菜单中选择“属性”命令，打开计算机属性界面如图 1-8 所示。



```

命令提示符 - python
Microsoft Windows [版本 10.0.17134.706]
(c) 2018 Microsoft Corporation。保留所有权利。

C:\Users\Qn>python
Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
  
```

图 1-7 Python 安装成功界面



图 1-8 计算机属性界面

单击“高级系统设置”选项，在弹出的“系统属性”界面选择“高级”选项卡，单击右下角的“环境变量”按钮，如图 1-9 所示。

在弹出的“环境变量”界面中，双击系统变量中的“Path”，如图 1-10 所示。

在弹出的“编辑环境变量”界面，单击“新建”按钮，在输入框中输入 Python 的安装路径（如 C:\Python\），如图 1-11 所示。单击“确定”按钮以保存添加的环境变量，之后重复第 3 步验证即可。至此，在 Windows 上安装 Python 的操作已完成。

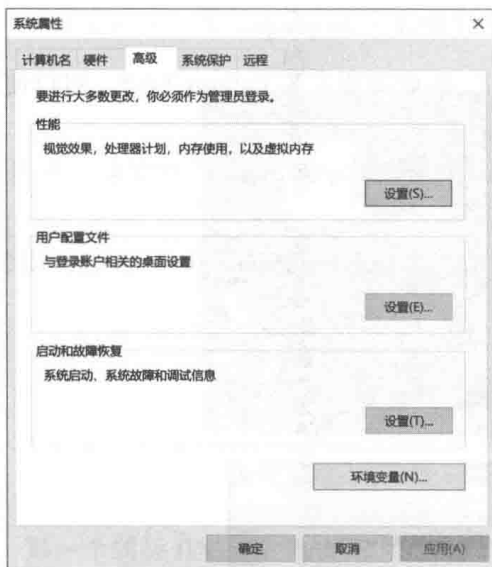


图 1-9 系统属性界面



图 1-10 环境变量界面

1.1.2 Linux 下的安装

1) 准备编译环境 GCC。

2) 去官网下载要安装的对应版本的 Python 的源代码。下载地址为 <https://www.python.org/downloads/source/>，本书选择 Python 3.6.5 版本。

3) 解压下载的代码包。

4) 配置。

① 查找 configure 文件。

```
find . -name configure
cd /usr/local/python-3.6.5/
```

② 进行配置。

```
./configure
```

5) 编译。

```
make
make install
```

6) 替换以前的 Python 默认版本（创建新的软链接）。

```
cd /usr/bin/
rm -rf python
ln -s /usr/local/Python-3.6.5/bin/python ./python
```

至此，在 Linux 下安装 Python 的操作已完成。

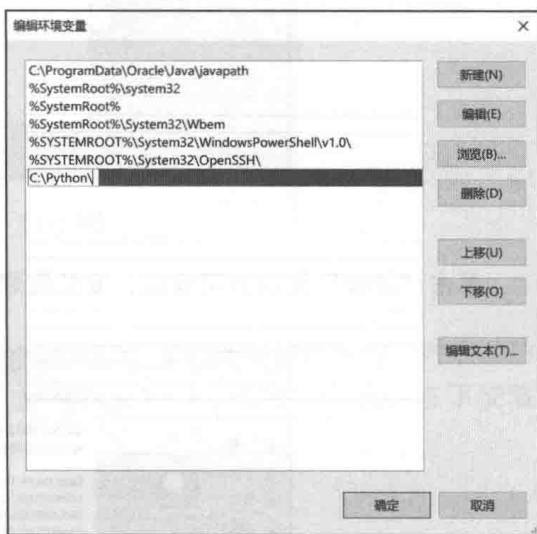


图 1-11 编辑环境变量界面