Python 数据抓取技术与实战

数据抓取，**大数据**领域研究的**第一个**环节！

数据就是**生产力**！

★ 潘庆和　赵星驰/编著 ★

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Python 数据抓取技术与实战

数据抓取，**大数据**领域研究的**第一个**环节！

数据就是**生产力**！

★ 潘庆和　赵星驰 / 编著 ★

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

数据抓取是获取大数据的一种主要手段。本书主要介绍使用 Python 语言及其相关工具进行数据抓取的方法，通过实例演示在数据抓取过程中常见问题的解决方法。通过本书的学习，读者可以根据需求快速地编写出符合要求的抓取程序。

本书技术性强，注重应用和实战，可供从事数据获取的工程技术人员、理工科院校相关专业的本科生及大数据从业人员使用。

图书在版编目（CIP）数据

Python 数据抓取技术与实战/潘庆和，赵星驰编著．—北京：电子工业出版社，2016.8
ISBN 978-7-121-29884-4

Ⅰ.①P… Ⅱ.①潘… ②赵… Ⅲ.①软件工具-程序设计 Ⅳ.①TP311.561

中国版本图书馆 CIP 数据核字（2016）第 217952 号

责任编辑：富　军　　特约编辑：刘汉斌
印　　刷：三河市华成印务有限公司
装　　订：三河市华成印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×1092　1/16　印张：16　字数：410 千字
版　　次：2016 年 8 月第 1 版
印　　次：2016 年 8 月第 1 次印刷
印　　数：3 000 册　　定价：49.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：(010)88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：(010)88254456。

前　言

大数据技术是当前工程和科学技术领域研究的热点。数据科学研究通常包括四个主要环节，即数据获取、数据存储、数据分析及数据可视化。本书主要聚焦数据获取环节。这是其他环节的基础。及时准确地获得丰富详实的数据，可为后续工作奠定坚实的基础，并提高分析结论的可信性和可靠性。

互联网的开放性为数据的获取带来了极大的便利。本书基于 Python 语言的数据抓取技术，主要介绍如何快速准确地从网络上获得所需的数据，构建满足要求的数据集或大数据集。Python 语言是一种通用编程语言，可以应用于各种编程领域，在数据科学领域也是一种十分热门的语言。本书使用 Python 作为数据抓取技术的实现语言，利用 Python 丰富的模块支持和语言特性，解决绝大部分数据抓取中经常会遇到的问题。为了使不了解 Python 语言的读者快速上手，在第 1 章中介绍了阅读本书所需的 Python 语言基础知识。

本书介绍了数据抓取涉及的各类技术问题和解决方法，并按章节进行组织，每章内容基本独立，可使读者在遇到问题时能够快速地进行问题定位。书中的内容侧重于将已有的成熟理论原理和流行框架应用于数据抓取实际问题的解决中。在编写过程中，只侧重介绍应用于数据抓取时的应用方式，并未对某些原理和框架进行详细的描述，感兴趣的读者可以进一步查找相关文献和资料来加深对概念和理论的理解。阅读时，读者可通过运行书中的实例代码，看到现象后再回头去分析，可有助于更好地理解相关的概念和原理，为进一步的研究打下基础。

本书主要面向初学者，读者可基于书中的运行实例进行改造，设计出符合自己要求的数据抓取程序。本书可以迅速用于实战，可供相关专业工程技术人员和高校本科生阅读参考。

感谢首席策划编辑富军老师的辛勤工作！感谢赵星驰老师在外文技术资料方面提供的帮助与协作！

如果读者阅读中发现问题，请及时与我们联系，希望大家多多批评指正。

编著者

目　　录

1 第1章 Python 基础

Python 是一种动态类型脚本语言，具有简洁的语法，强大的表达能力，是当前数据科学领域主流的编程语言。其丰富的包和模块的支持可以使 Python 几乎能应用于各个领域。本书使用 Python 作为数据抓取的编程实施语言。

本章主要介绍 Python 编程语言的基础知识及 Python 语言的安装和基本使用方法，通过两个抓取实例讲述 Python 的基本语法及后面各章中将会使用到的语法知识。

在语言版本上选择了 Python3.4 + 作为开发语言，使用 3 以上版本的 Python 在字符串处理上会更加方便。如果没有使用 Python3 以上引入的新特性，那么都可以在对字符串处理并稍加改动后使用 Python2.7 + 执行。随着 Python3 的发展，很多 Python2.7 可使用的库和模块，Python3 都可以使用了，因此建议没有使用过 Python 的读者，可以直接安装 3 以上的版本学习和开发。

在操作系统的选择上，本书的绝大部分实例是在 Ubuntu 14.04 trusty 64 位桌面系统下编写和运行的。如果 Python 版本相同的话，那么在 Windows 下可以直接运行书中的代码。使用 Windows 系统的读者可以利用 Vmware 虚拟机环境在虚拟机中搭建 Ubuntu 系统学习并使用书中的代码，可得到与书中一样的运行效果，也可以直接安装相应版本的 Ubuntu 操作系统。

1.1 Python 安装

Python 是一种跨平台编程语言，可以安装在各类操作系统平台上。在 Ubuntu 14.04 系统上已经安装了 Python，下面介绍在 Windows 环境下安装和使用 Python 的方法。

登录 Python 官网 https://www.python.org/，如图 1-1 所示。在首页单击“Downloads”进入下载页，如图 1-2 所示。

以“Python3.4.3”为例，单击下载。在新打开的下载页面中定位到底部，可以看到该版本的相关下载文件，如图 1-3 所示。

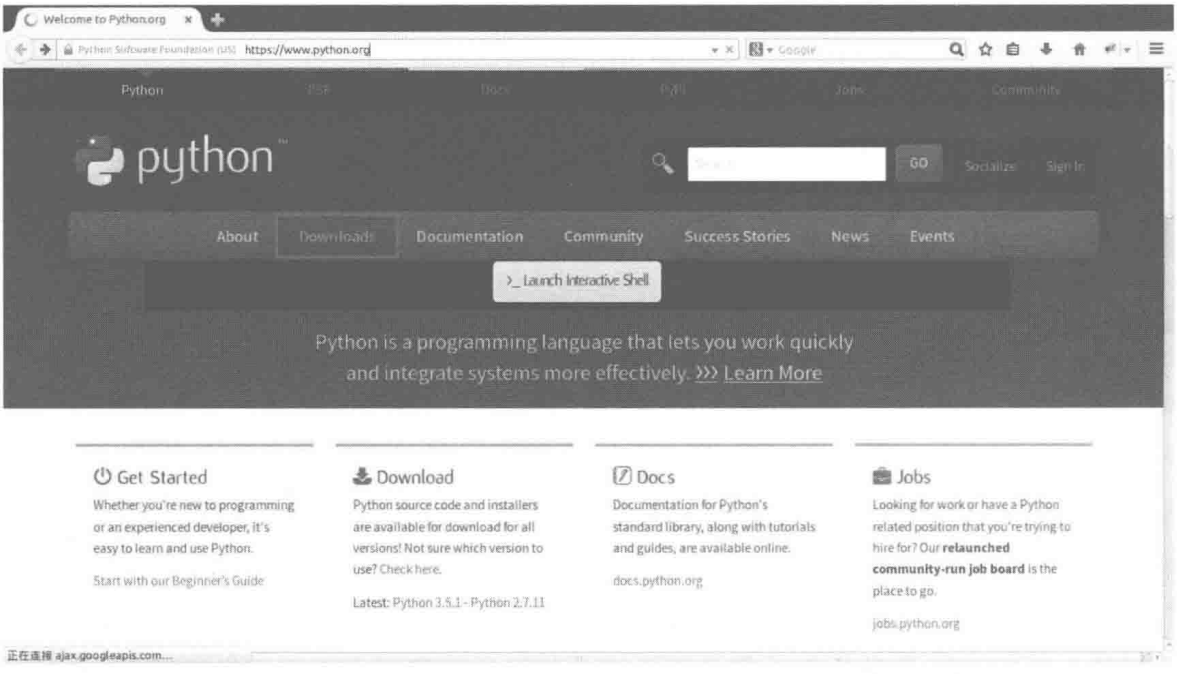

图 1-1　Python 首页

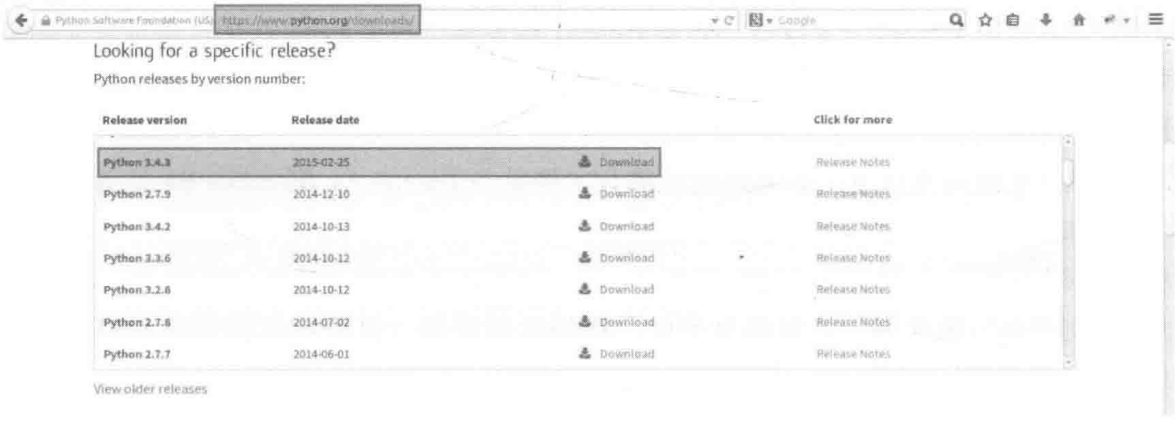

图 1-2　Python 下载页

以 Windows 7 为例，对于 64 位的系统可以选择在如图 1-3 所示中方框的文件下载。下载后双击安装即可，在安装的过程中要注意将 python. exe 添加到环境变量中，这样未来在打开 cmd 窗口时，就可以输入 python 进入 python 的交互环境了。图 1-4 给出了设置的位置，单击“Add python. exe to Path”前面的叉号，在弹出的下拉选项中选择“Will be installed on local hard drive”即可，操作后如图 1-5 所示。接下来就可以进行下一步的安装，直至安装完成。

安装完成后，启动 cmd，在命令行输入“python”，然后回车，即可启动 python 的命令交互环境。启动 cmd 的方式主要有三种。第一种是在左下角单击“开始”按钮，定位到

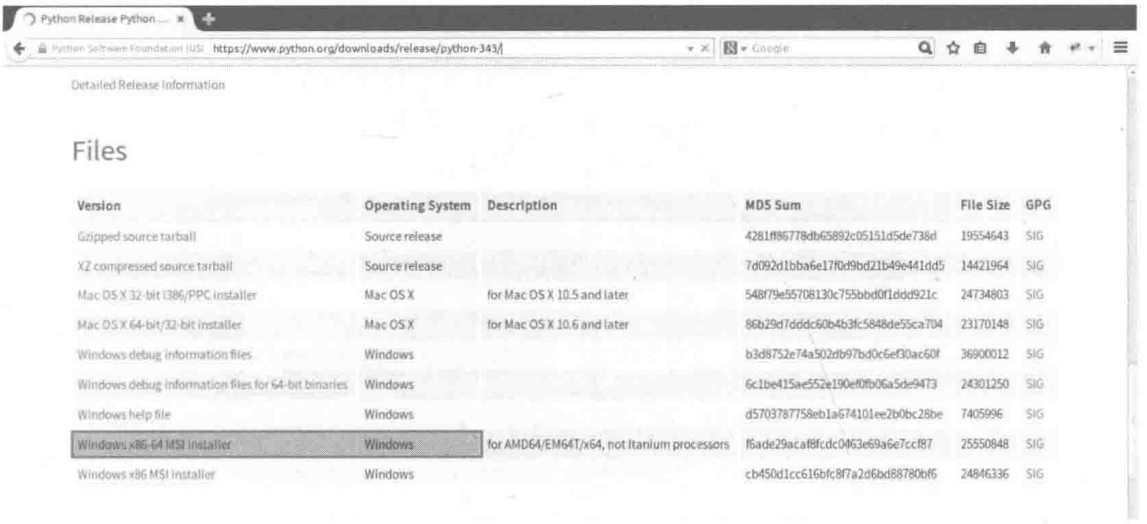

图 1-3　Python 的 Windows 下载文件

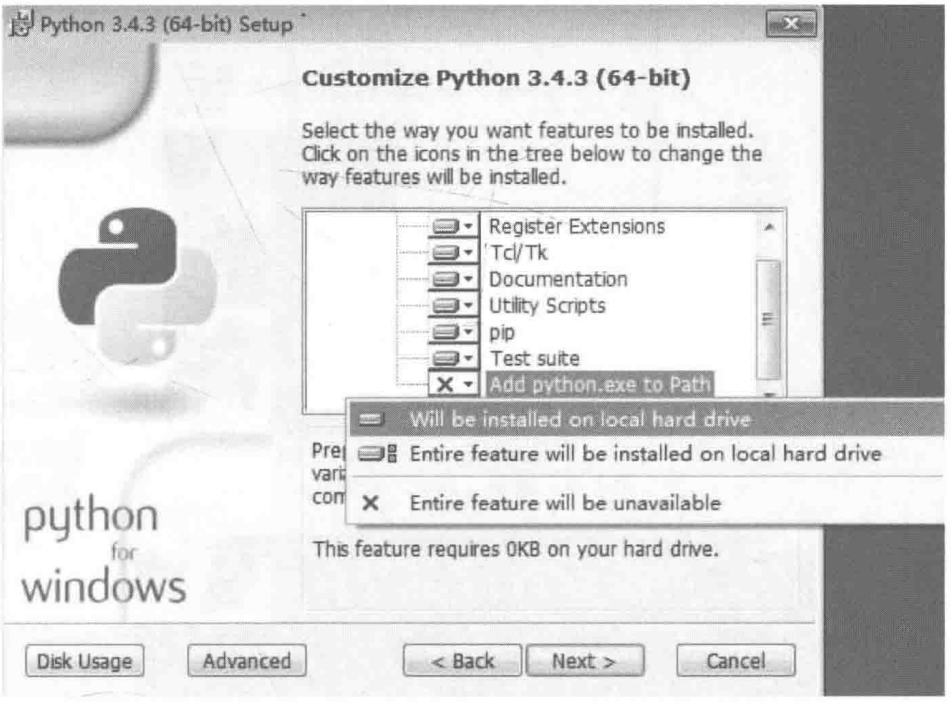

图 1-4　将 python. . exe 添加到系统路径的选项

“搜索程序和文件”的方框，如图 1-6 所示，输入“cmd”后回车，系统会自动搜索出 cmd. exe，如图 1-7 所示。

单击程序下的 cmd. exe，调出命令行，输入 python 执行即可，如图 1-8 所示。

第二种方式是在任何窗口按住 shift 键，单击鼠标右键会弹出菜单，如图 1-9 所示。单击方框内的“在此处打开命令窗口”即可调出命令窗口。

第三种是单击“开始”后，在“附件”中单击“命令提示符”启动命令行，如图 1-10 所示。

图 1-5　将 python. . exe 添加到系统路径

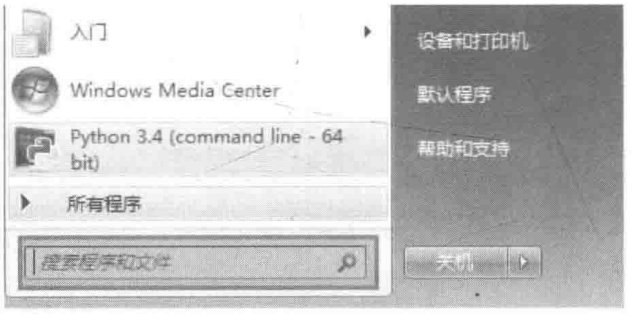

图 1-6　搜索“cmd”的位置

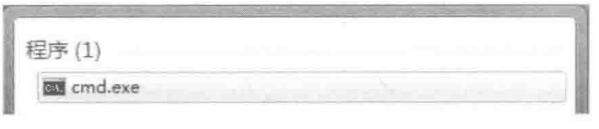

图 1-7　搜索出 cmd. exe

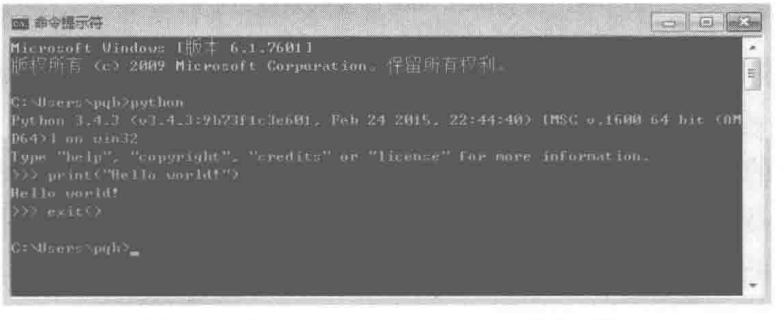

图 1-8　在 cmd. exe 中调出 python 执行环境

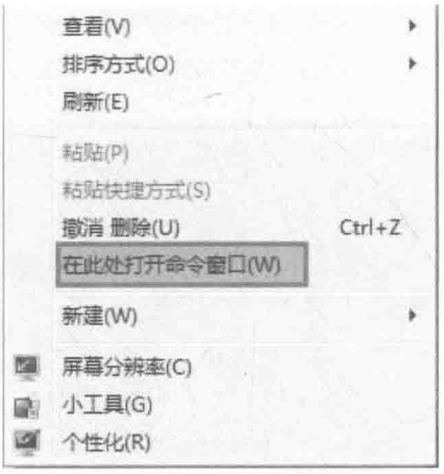

图 1-9　右键菜单打开 cmd. exe

图 1-10　在“附件”中打开 cmd. exe

在 Linux 平台，以 Ubuntu 14. 04 为例，在系统安装时就默认安装了 Python 环境，而且在同时默认安装了 Python2. 7 和 Python3. 4. 的情况下，在 shell 中输入 Python，启动的是 Python2. 7，如图 1-11 所示。

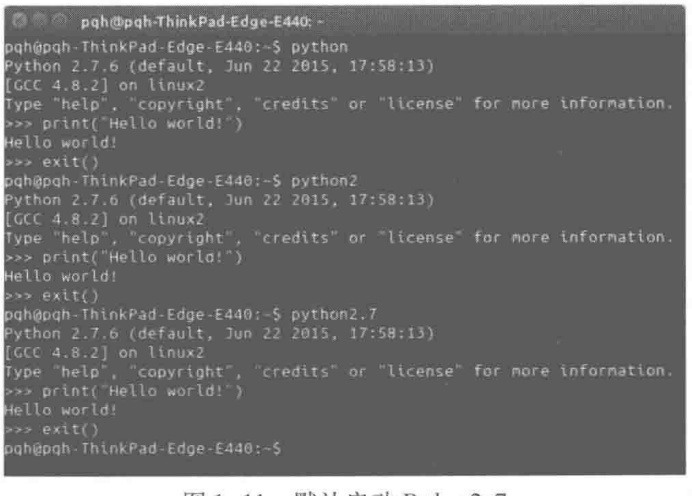

图 1-11　默认启动 Python2. 7

如果需要使用 Python3.4，则在 shell 命令行中输入 Python3.4 或 Python3 即可，如图 1-12 所示。

```
pqh@pqh-ThinkPad-Edge-E440: ~
pqh@pqh-ThinkPad-Edge-E440:~$ python3.4
Python 3.4.0 (default, Apr 11 2014, 13:05:11)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello world!")
Hello world!
>>> exit()
pqh@pqh-ThinkPad-Edge-E440:~$ python3
Python 3.4.0 (default, Apr 11 2014, 13:05:11)
[GCC 4.8.2] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello world!")
Hello world!
>>> exit()
pqh@pqh-ThinkPad-Edge-E440:~$
```

图 1-12　启动 Python3.4

1.2　安装 pip

使用 Python 时，经常需要安装第三方的包文件，可以使用 pip 进行安装，在 Ubuntu14.04下虽然已经安装了 Python2.7 和 Python3.4，但并没有安装相应 Python 版本的 pip 包安装工具，如图 1-13 所示。

```
pqh@pqh-ThinkPad-Edge-E440: ~
pqh@pqh-ThinkPad-Edge-E440:~$ pip2
程序"pip2"尚未安装。 您可以使用以下命令安装：
sudo apt-get install python-pip
pqh@pqh-ThinkPad-Edge-E440:~$ pip3
程序"pip3"尚未安装。 您可以使用以下命令安装：
sudo apt-get install python3-pip
pqh@pqh-ThinkPad-Edge-E440:~$
```

图 1-13　Ubuntu 默认没有安装 pip 工具

系统已经给出了安装的方法，使用提示的命令分别安装即可，命令为

```
sudo apt - get install python - pip
sudo apt - get install python3 - pip
```

安装好后，可以在命令行查看，如图 1-14 所示。

```
pqh@pqh-ThinkPad-Edge-E440: ~
pqh@pqh-ThinkPad-Edge-E440:~$ pip2 --version
pip 1.5.4 from /usr/lib/python2.7/dist-packages (python 2.7)
pqh@pqh-ThinkPad-Edge-E440:~$ pip3 --version
pip 1.5.4 from /usr/lib/python3/dist-packages (python 3.4)
pqh@pqh-ThinkPad-Edge-E440:~$
```

图 1-14　安装了 pip 工具

安装好 pip 后，未来就可以在需要时安装第三方的 Python 包了。

1.3　如何查看帮助

1. help 函数

Python 提供了内置函数 help，可以通过向 help 传递模块、函数、类、方法或 Python 关键字的字符串来查看帮助，如查看有关 os 模块的信息，可在交互环境下输入 help('os')，命令为

```
>>> help('os')
```

在交互环境下会出现相关信息，如图 1-15 所示。

可使用回车或上下箭头按键逐行查看，也可使用翻页键进行翻页查看，在末行“:”后，输入“q”回车，即可退回到交互界面。

当然，也可以在遇到问题时去网络查找，但命令的好处在于可以直接在交互环境中调出帮助文档，可以脱离网络查看，而且文档内容更具针对性。在对 Python 语言及环境有了一定的了解后，通常的查找目标就是函数的返回值和参数信息，帮助文档可以帮助我们快速了解所需信息，提高工作效率。在后面的章节中，当遇到具体的函数时，经常会使用 help 函数帮助我们了解函数的基本用法。

2. dir 函数

dir 也是 Python 的一个内置函数，可以将字符串'dir'作为 help 的参数，查看 dir 函数的

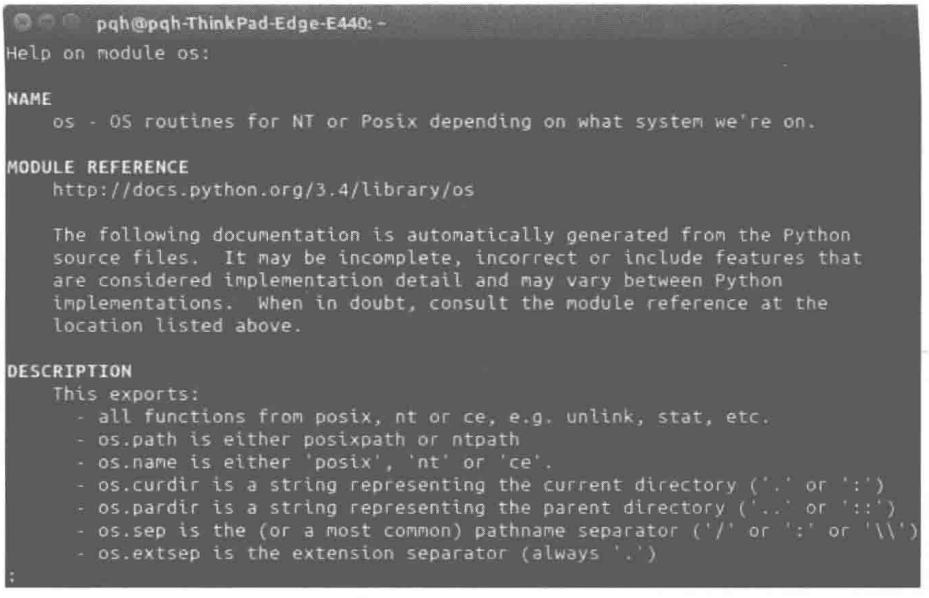

图 1-15　help('os')显示的内容

使用方法时可输入命令

```
>>> help('dir')
```

显示的内容如图 1-16 所示。

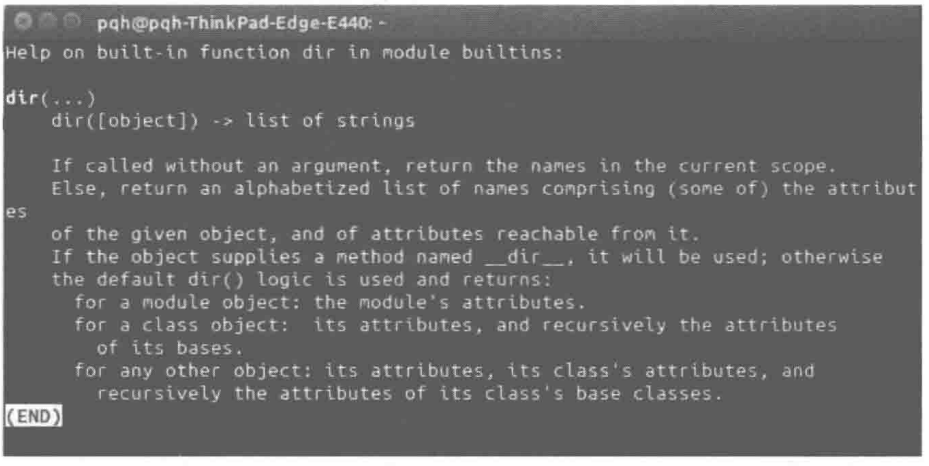

图 1-16　help('dir')显示的内容

根据上面的描述，可以分析出 dir 函数会返回一个字符串的列表。其中，dir([object])表示作为参数的 object 是可选的，即在使用时，既可以传入某个 object 作为参数，也可以不传入任何参数。object 表示对象，在 Python 中的数据类型、模块、函数等都可以视作 object。

如果没有向 dir 函数传递参数，则将返回当前范围的名字列表，如图 1-17 所示。

```
>>> dir()
['__builtins__', '__doc__', '__loader__', '__name__', '__package__', '__spec__']
>>>
```

图 1-17　dir( ) 显示的内容

如果将某个 object 作为参数传递给 dir，则 dir 会返回这个对象内所有属性和方法名称的列表。图 1-18 中将 os 模块作为参数传递给 dir，会列出很多 os 模块内属性和方法的名字。如果在使用时对某些 object 的属性和方法名称没有记清的话，则可以通过这种方式查看。注意上面给出的是 dir(os) 的结果。图 1-19 给出的是 dir('os') 显示的内容。

```
>>> import os
>>> dir(os)
['CLD_CONTINUED', 'CLD_DUMPED', 'CLD_EXITED', 'CLD_TRAPPED', 'EX_CANTCREAT', 'EX
_CONFIG', 'EX_DATAERR', 'EX_IOERR', 'EX_NOHOST', 'EX_NOINPUT', 'EX_NOPERM', 'EX_
NOUSER', 'EX_OK', 'EX_OSERR', 'EX_OSFILE', 'EX_PROTOCOL', 'EX_SOFTWARE', 'EX_TEM
PFAIL', 'EX_UNAVAILABLE', 'EX_USAGE', 'F_LOCK', 'F_OK', 'F_TEST', 'F_TLOCK', 'F_
ULOCK', 'MutableMapping', 'NGROUPS_MAX', 'O_ACCMODE', 'O_APPEND', 'O_ASYNC', 'O_
CLOEXEC', 'O_CREAT', 'O_DIRECT', 'O_DIRECTORY', 'O_DSYNC', 'O_EXCL', 'O_LARGEFIL
E', 'O_NDELAY', 'O_NOATIME', 'O_NOCTTY', 'O_NOFOLLOW', 'O_NONBLOCK', 'O_PATH', '
O_RDONLY', 'O_RDWR', 'O_RSYNC', 'O_SYNC', 'O_TMPFILE', 'O_TRUNC', 'O_WRONLY', 'P
OSIX_FADV_DONTNEED', 'POSIX_FADV_NOREUSE', 'POSIX_FADV_NORMAL', 'POSIX_FADV_RAND
OM', 'POSIX_FADV_SEQUENTIAL', 'POSIX_FADV_WILLNEED', 'PRIO_PGRP', 'PRIO_PROCESS'
, 'PRIO_USER', 'P_ALL', 'P_NOWAIT', 'P_NOWAITO', 'P_PGID', 'P_PID', 'P_WAIT', 'R
TLD_DEEPBIND', 'RTLD_GLOBAL', 'RTLD_LAZY', 'RTLD_LOCAL', 'RTLD_NODELETE', 'RTLD_
NOLOAD', 'RTLD_NOW', 'R_OK', 'SCHED_BATCH', 'SCHED_FIFO', 'SCHED_IDLE', 'SCHED_O
THER', 'SCHED_RESET_ON_FORK', 'SCHED_RR', 'SEEK_CUR', 'SEEK_DATA', 'SEEK_END', '
SEEK_HOLE', 'SEEK_SET', 'ST_APPEND', 'ST_MANDLOCK', 'ST_NOATIME', 'ST_NODEV', 'S
```

图 1-18　dir(os) 显示的内容

```
>>> dir('os')
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '_
_eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs
__', '__gt__', '__hash__', '__init__', '__iter__', '__le__', '__len__', '__lt__'
, '__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__r
epr__', '__rmod__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subcl
asshook__', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', '
expandtabs', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isd
ecimal', 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'issp
ace', 'istitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'pa
rtition', 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip
', 'split', 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate
', 'upper', 'zfill']
>>>
```

图 1-19　dir('os') 显示的内容

这时，dir 的参数是字符串，dir 返回的结果是字符串对象支持的属性和方法名称，与字符串是什么无关，当传入其他字符串时，给出的结果是不变的，都是字符串对象支持的属性和方法名称，如图 1-20 所示。其中的常用方法将在第 2 章专门介绍。

help 和 dir 是两个常用的内置函数，建议读者在遇到问题时综合使用，在后面的实例中，我们通常也会首先调用 help 函数来说明和解释一些概念和内容。

```
>>> dir('os123')
['__add__', '__class__', '__contains__', '__delattr__', '__dir__', '__doc__', '__
eq__', '__format__', '__ge__', '__getattribute__', '__getitem__', '__getnewargs__
', '__gt__', '__hash__', '__init__', '__iter__', '__le__', '__len__', '__lt__', '
__mod__', '__mul__', '__ne__', '__new__', '__reduce__', '__reduce_ex__', '__repr_
_', '__rmod__', '__rmul__', '__setattr__', '__sizeof__', '__str__', '__subclassho
ok__', 'capitalize', 'casefold', 'center', 'count', 'encode', 'endswith', 'expand
tabs', 'find', 'format', 'format_map', 'index', 'isalnum', 'isalpha', 'isdecimal'
, 'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprintable', 'isspace', 'i
stitle', 'isupper', 'join', 'ljust', 'lower', 'lstrip', 'maketrans', 'partition',
 'replace', 'rfind', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'split'
, 'splitlines', 'startswith', 'strip', 'swapcase', 'title', 'translate', 'upper',
 'zfill']
>>>
```

图 1-20　dir('os123')显示的内容

1.4　第一个实例

下面给出第一个抓取程序。该程序的主要功能是获得电子工业出版社 http://www.phei.com.cn 的首页内容。程序代码包含模块的引入、函数的定义和使用、缩进风格、字符串的使用及简单的分支判断结构等语言内容。下面是程序代码，保存在文件 first_get.py 中。该程序需要 requests 模块，可使用 sudo pip3 install requests 安装。

```
#引入 requests 模块
import requests

#定义 get_content 函数
def get_content(url):
    resp = requests.get(url)
    return resp.text

#定义 url,值为要抓取的目标网站网址
url = "http://www.phei.com.cn"

#调用函数返回值赋值给 content
content = get_content(url)

#打印输出 content 的前 50 个字符
print("前 50 个字符为:",content[0:50])

#得到 content 的长度
```

```
content_len = len(content)
print("内容的长度为：",content_len)

#判断内容长度是否大于 40KB
if content_len >= 40 * 1024:
    print("内容的长度大于等于 40KB.")
else:
    print("内容的长度小于等于 40KB.")
```

下面对程序进行分析和说明。

1. 注释

在代码中对代码的主要功能和意义均使用注释进行了说明，以“#”开头的行都是注释，如代码的第一行

```
#引入 requests 模块
```

用来说明下一句 import requests 所起的作用。Python 中的注释有两种类型：单行注释和多行注释。以“#”开头的注释属于单行注释，“#”不一定位于行首，也可位于代码行尾，写在代码的后面，如

```
import requests #引入 requests 模块
```

的形式也是可以的。

'''...'''和"""..."""可以表示多行注释。前者是用成对出现的三个单引号表示的，后者是用成对出现的三个双引号表示的，如

```
'''
    引入 requests 模块
    引入 requests 模块
    引入 requests 模块
'''
或
"""
    引入 requests 模块
    引入 requests 模块
    引入 requests 模块
"""
```