

Python 与软件测试

本章首先介绍 Python 的相关知识,包括 Python 语言的特点、应用场合等;其次介绍 Python 解释器,以及 Python 编辑器的安装和配置;最后,简要介绍 Python 的相关测试框架,如 unittest、Pytest、Selenium、Appium 等。

3.1 Python 简介

3.1.1 Python 的历史

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,被列入 LAMP (Linux、Apache、Mysql 以及 Python/Perl/PHP)。Python 由 Guido van Rossum 于 1989 年底发明,第一个公开发行人版发行于 1991 年。像 Perl 语言一样,Python 源代码同样遵循 GNU 通用公共授权(General Public License,GPL)协议。

Python 2.0 于 2000 年 10 月 16 日发布,实现垃圾回收,并支持 Unicode。Python 3.0 版本于 2008 年 12 月 3 日发布,常被称为 Python 3000,或简称 Py3k。它相对于 Python 的早期版本作了较大的升级,未考虑向下相容,导致早期 Python 程式无法在 Python 3.0 上正常执行。为此,Python 2.6 和 2.7 作为一个过渡版本,基本使用了 Python 2.x 的语法和库,同时考虑了向 Python 3.0 的迁移,允许使用部分 Python 3.0 的语法与函数。

3.1.2 Python 的特点

Python 是一种简单易学、功能强大的编程语言,它有高效率的高层数据结构,简单而有效地实现面向对象编程。Python 具有如下一些特点。

1. 简单易学

Python 作为代表简单主义思想的语言,语法简捷而清晰,结构简单。用户可以快速上手学习。在学习 Python 过程中,不用计较程序语言在形式上的诸多细节和规则,用户可以专注重程序本身的逻辑和算法,探究程序执行的过程。

2. 免费开源

Python 是 FLOSS(自由/开放源码软件)之一,用户可以自由地发布这个软件的拷

贝,阅读它的源代码,对它做改动,并将它用于新的自由软件中。

3. 解释型语言

计算机并不能直接接收和执行用高级语言编写的源程序,源程序在输入计算机时,通过“翻译程序”翻译成机器语言形式的目标程序,计算机才能识别和执行。这种“翻译”通常有两种方式:一种是编译执行;另一种是解释执行。

编译执行是指源程序代码先由编译器编译成可执行的机器码,然后再执行;解释执行是指源代码程序被解释器直接读取执行。编译执行和解释执行各有优缺点。编译执行可一次性将高级语言源程序编译成二进制的可执行指令,通常执行效率高;而解释执行是由该语言(如 HTML)运行环境(如浏览器)读取一条该语言的源程序,然后转变成二进制指令,交给计算机执行,通常可以灵活地跨平台。C、C++ 等采用编译执行方式,Python 作为解释型语言,与 Java 语言类似,不需要编译成二进制代码,而是通过解释器把源代码转换成称为字节码的中间形式,由虚拟机负责在不同的计算机运行。因此,Python 程序便于移植,可在众多平台运行,如 Linux、Windows、Macintosh、OS/2 等。

4. 面向对象

Python 是完全面向对象的语言。函数、模块、数字、字符串都是对象,并且完全支持继承、重载、派生、多重继承。用 Python 语言编写程序无须考虑硬件和内存等底层细节。

5. 丰富的库

Python 称为胶水语言,能够轻松地与其他语言(特别是 C 或 C++)联结在一起。其具有丰富的 API 和标准库,支持图形处理、科学计算、Web 开发、爬虫、人工智能等。

3.1.3 Python 的应用场合

Python 功能强大,主要应用于以下场合。

1. GUI 软件开发

Python 具有 wxPython、PyQT 等工具,因此可以快速开发出 GUI,并且不做任何改变就可以运行在 Windows、Linux 等平台上。

2. 网络应用开发

Python 提供了标准 Internet 模块,可以广泛应用到各种网络任务中,无论在服务端还是在客户端。另外,网站编程第三方工具,如 HTMLGen、mod_python、Django、TurboGears、Zop 可以帮助 Python 快速构建功能完善和高质量的网站。

3. 游戏开发

Pygame 是建立在 SDL(Simple DirectMedia Layer)基础上的软件包,提供了简单的方式控制媒体信息(如图像、声音等),专为电子游戏设计使用。Pygamer 的下载网址为

www.pygame.org, 如图 3.1 所示。



图 3.1 Pygame 网址

4. 科学计算

Python 具有科学计算的三剑客, 即 numpy、scipy、matplotlib。其中, numpy 负责数值计算、矩阵操作等; scipy 负责常见的数学算法, 插值、拟合等; matplotlib 负责数据可视化。

5. Web 与移动设备应用开发

Web2py 是一种免费的开源的 Web 开发框架, 帮助开发者分别设计、实施和测试 MVC(模型 Model、视图 View、控制器 Controller)模型。web2py 的下载网址为 www.web2py.com, 如图 3.2 所示。

6. 数据库开发

Python 支持所有主流数据库, 如 Oracle、Sybase、MySQL、PostgreSQL、Informix 等, 并通过标准的数据库 API 接口将关系数据库映射到 Python 类, 实现面向对象数据库系统。

7. 系统编程

Python 对操作系统服务设置的内置接口, 使其成为编写可移植的维护操作系统的管理工具和部件。Python 程序可以搜索文件和目录树, 运行其他程序, 用进程或线程进行并行处理等。



图 3.2 web2py 网址

3.2 Python 解释器

3.2.1 在 Ubuntu 下安装 Python

Ubuntu(乌班图)是一个以桌面应用为主的 Linux 操作系统,它基于 Debian 发行版和 GNOME 桌面环境。与 Debian 不同的是,它每 6 个月会发布一个新版本。Ubuntu 的目标在于为用户提供最新的、同时又相当稳定的自由软件构建的操作系统。

在 Ubuntu 中内置 Python,如图 3.3 所示。

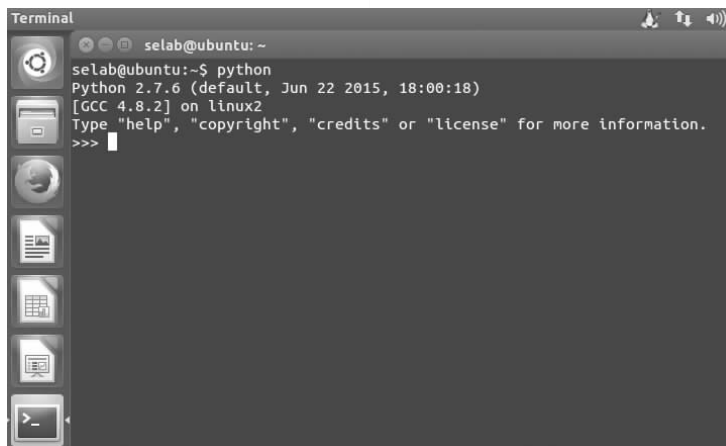


图 3.3 Ubuntu 下内置 Python

3.2.2 在 Windows 下安装 Python

在 Windows 下安装 Python,一般具有如下步骤:

步骤 1: 在浏览器中输入 <http://www.Python.org> 进入 Python 官网,在下载页选择 Python3 版本的安装包进行安装,如图 3.4 所示。

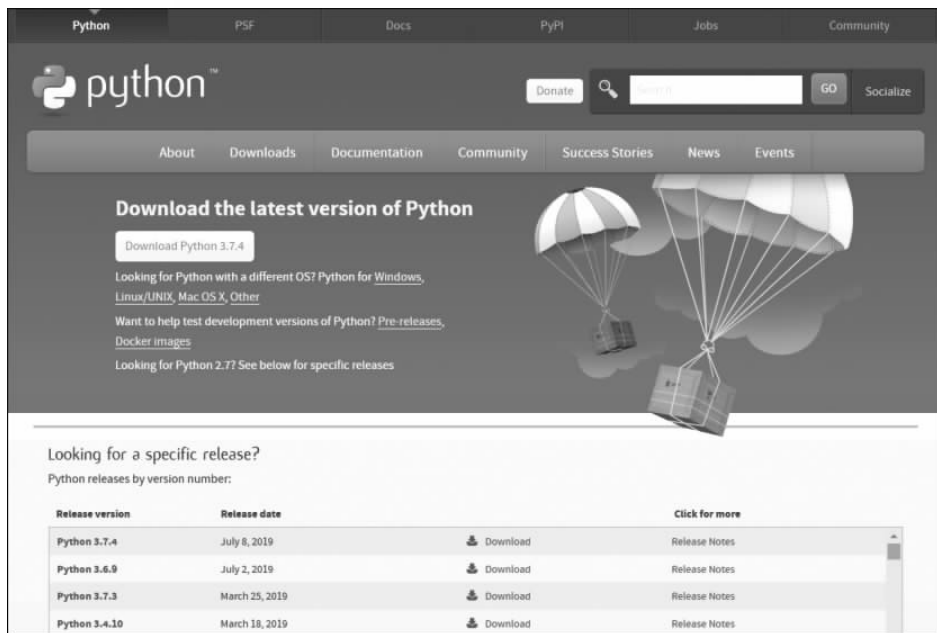


图 3.4 下载 Python3

步骤 2: 在 Windows 环境变量中添加 Python,将 Python 的安装目录添加到 Windows 下的 path 变量中,如图 3.5 所示。



图 3.5 设置环境变量

步骤 3: 测试 Python 安装是否成功

在 Windows 下使用 cmd 打开命令行,输入 Python 命令,图 3.6 表示安装成功。

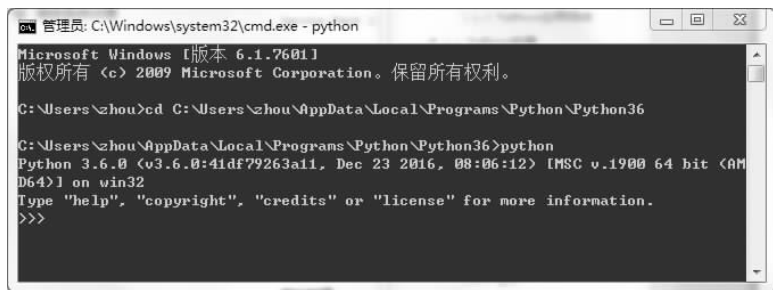


图 3.6 测试 Python 安装是否成功

3.3 Python 编辑器

Python 编辑器众多,除了 Python 自带的 IDLE 外,还有 notepad++、Sublime、Eclipse+PyDev、Ulipad 以及 Vim 和 emacs 等。其中,Linux 下的 Eclipse+PyDev 和 Windows 的 PyCharm 功能较强大,Anaconda 的应用较为广泛。下面依次介绍。

3.3.1 IDLE

IDLE 作为 Python 内置的集成开发工具,具有能够利用颜色突出显示语法的编辑器、调试工具,Python Shell 以及完整的 Python 3 在线文档集。Python 的 IDLE 具有命令行和图形用户界面两种方式,采用命令行交互式执行 Python 语句方便快捷,但必须逐条输入语句,不能重复执行,适合测试少量的 Python 代码,不适合复杂的程序设计。

在 Windows 下安装的 Python 文件如图 3.7 所示。

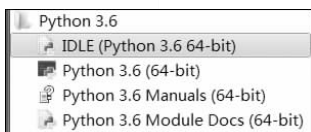


图 3.7 Python 3.6

3.3.2 PyCharm

PyCharm 具有一整套可以帮助用户使用 Python 语言开发时提高效率的工具,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,PyCharm 提供了一些高级功能,用于支持 Django 框架下的专业 Web 开发。下载 PyCharm 双击安装,如图 3.8 所示。

单击 Next 按钮,弹出界面如图 3.9 所示。

安装结束,运行 PyCharm,如图 3.10 所示。



图 3.8 安装 PyCharm 步骤 1



图 3.9 安装 PyCharm 步骤 2

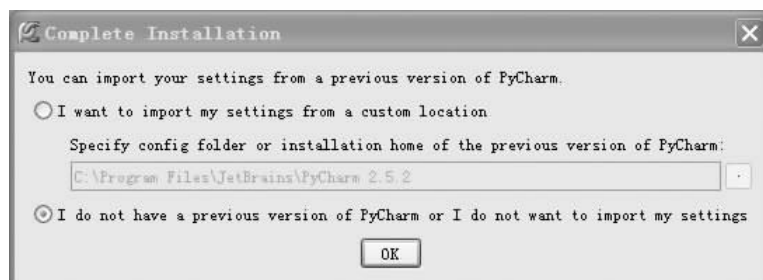


图 3.10 运行 PyCharm

在 PyCharm 主界面中单击 Create New Project, 输入项目名、路径, 选择 Python 解释器。如果没有出现 Python 解释器, 如图 3.11 所示, 选择 Python 解释器。

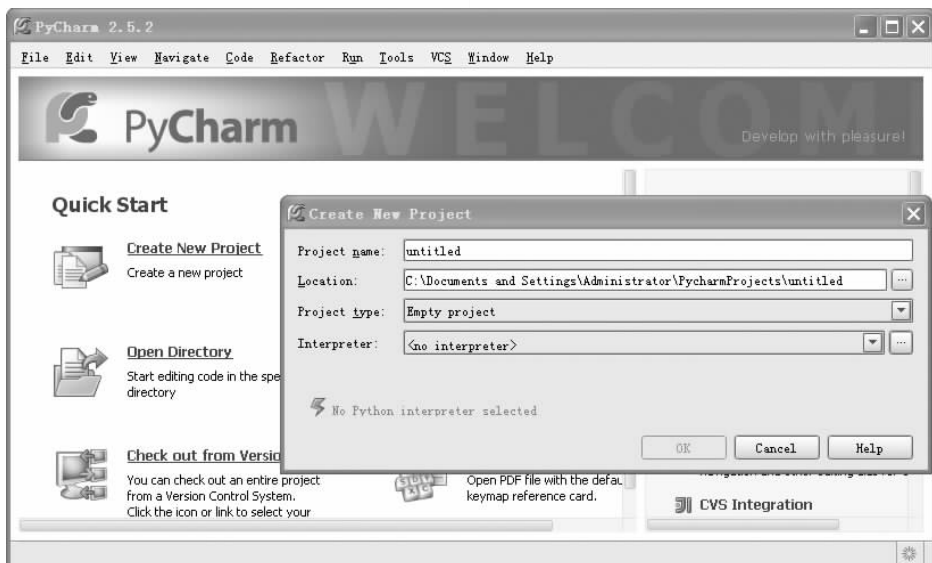


图 3.11 在 PyCharm 主界面中选择 Python 解释器

启动 PyCharm, 创建 Python 文件, 如图 3.12 所示。

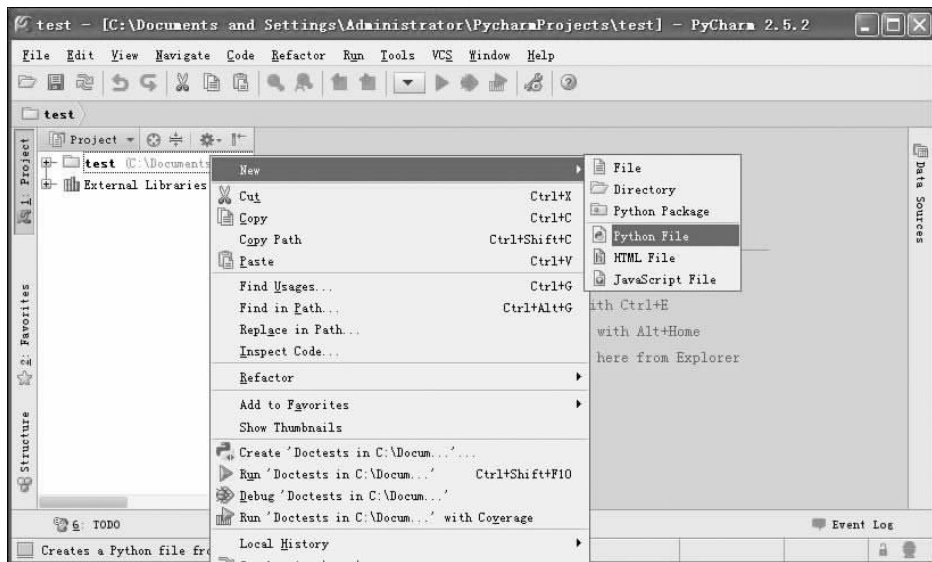


图 3.12 在 PyCharm 中创建 Python 文件

3.3.3 Anaconda

Anaconda 是一个开源的 Python 发行版本, 包含了 conda、Python 等 180 多个科学包

及其依赖项,涉及数据可视化、机器学习、深度学习等多方面。本书重点介绍 Anaconda,所有程序均在 Anaconda 下调试与运行。

第一,提供包管理。使用 conda 和 pip 安装、更新、卸载第三方工具包简单方便,不需要考虑版本等问题。

第二,关注于数据科学相关的工具包。Anaconda 集成了如 Numpy、Scipy、pandas 等数据分析的各类第三方包。

第三,提供虚拟环境管理。在 conda 中可以建立多个虚拟环境,为不同的 Python 版本本项目建立不同的运行环境,从而解决了 Python 多版本并存的问题。

Anaconda 的安装步骤如下:

打开 Anaconda 的官网(地址 <https://www.anaconda.com/download/>),如图 3.13 所示。

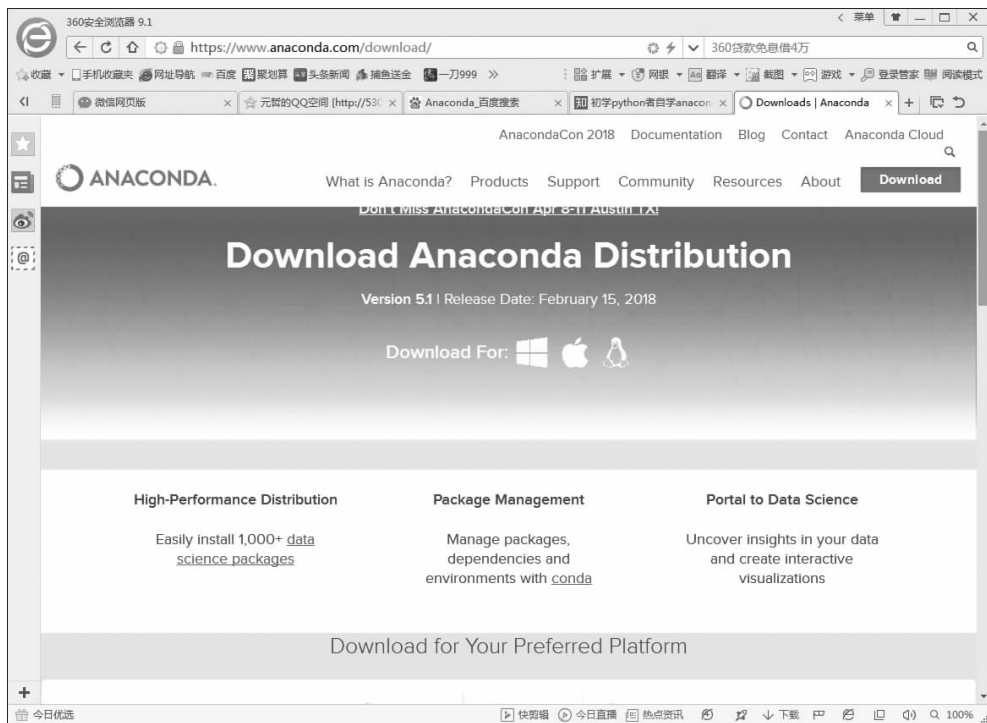


图 3.13 Anaconda 的网站

根据本机的操作系统是 32 位还是 64 位选择对应的下载版本,如图 3.14 所示。

下载 Python 3.6 version,选择本机保存目录,如图 3.15 所示。

下载 Anaconda3-5.1.0-Windows-x86_64.exe,文件大小约 500MB。

注意: 如果是 Windows 10 系统,注意在安装 Anaconda 软件的时候,右击安装软件,选择以管理员的身份运行。

选择安装路径,例如 C:\anaconda3,一直单击 Next 运行结果,完成安装,如图 3.16 所示。

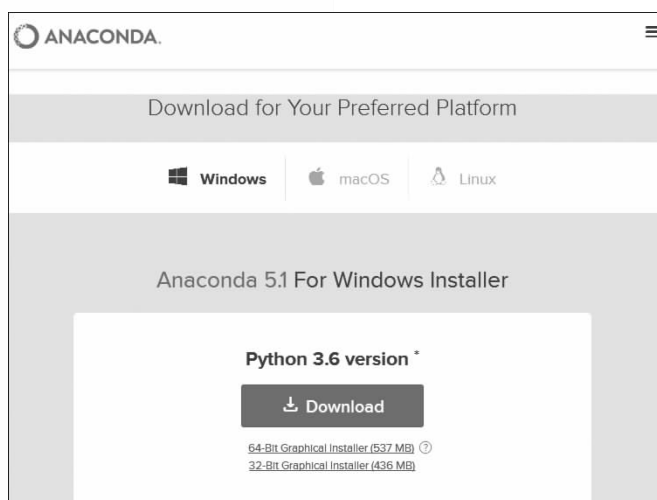


图 3.14 选择 Python 3.6 版本



图 3.15 下载 Anaconda 文件



图 3.16 程序运行结果

Anaconda 包含如下应用,如图 3.17 所示。



图 3.17 Anaconda 包含应用

(1) Anaconda Navigator: 用于管理工具包和环境的图形用户界面,后续涉及的众多管理命令也可以在 Navigator 中手工实现。

(2) Anaconda Prompt: Python 的交互式运行环境。

(3) Jupyter Notebook: 基于 Web 的交互式计算环境,可以编辑易于人们阅读的文档,展示数据分析的过程。

(4) Spyder: 一个使用 Python 语言、跨平台的、科学运算的集成开发环境。相对于 Pydev、PyCharm、PTVS 等 Python 编辑器, Spyder 对内存的需求小很多。

下面进行 Anaconda 的环境变量配置。在 Anaconda Prompt 中出现类似 cmd 的窗口。输入 `conda --version`,运行效果如图 3.18 所示。

在 Anaconda Prompt 中输入如下命令:

```
(base) C:\Users\Administrator>conda --version
conda 4.4.10
```

图 3.18 Anaconda 版本

```
conda create -n env_name package_names
```

其中,env_name 是设置环境的名称(-n 是指该命令后面的 env_name 是创建环境的名称),package_names 是安装在创建环境中的包名称。

```
conda create --name test_py3 python=3.6
#创建基于 Python3.6 的名为 test_py3 的环境
```

运行效果如图 3.19 所示。

```
(base) C:\Users\Administrator>conda create --name test_py3 python=3.6
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 4.4.10
  latest version: 4.5.2

Please update conda by running

    $ conda update -n base conda
```

图 3.19 创建基于 Python 3.6 的名为 test_py3 的环境

在 Anaconda Prompt 中使用 conda list 查看环境中默认安装的几个包,如图 3.20 所示。

```
(base) C:\Users\Administrator>conda list
# packages in environment at C:\ProgramData\Anaconda3:
#
# Name                                Version                                Build      Channel
_ipyw_jlab_nb_ext_conf                0.1.0                                py36he6757f0_0
alabaster                              0.7.10                               py36hcd07829_0
anaconda                              5.1.0                                py36_2
anaconda-client                        1.6.9                                py36_0
anaconda-navigator                     1.7.0                                py36_0
anaconda-project                       0.8.2                                py36hfad2e28_0
asn1crypto                             0.24.0                               py36_0
astroid                                1.6.1                                py36_0
astropy                                2.0.3                                py36hfa6e2cd_0
attrs                                  17.4.0                               py36_0
babel                                  2.5.3                                py36_0
backports                              1.0                                   py36h81696a8_1
backports.shutil_get_terminal_size    1.0.0                                py36h79ab834_2
beautifulsoup4                         4.6.0                                py36hd4cc5e8_1
bitarray                               0.8.1                                py36hfa6e2cd_1
bkcharts                               0.2                                   py36h7e685f7_0
blaze                                  0.11.3                               py36h8a29ca5_0
bleach                                 2.1.2                                py36_0
```

图 3.20 查看环境的默认包

在 Anaconda 下,Python 的编辑和执行有交互式编程、脚本式编程和 Spyder 三种运行方式。

1. 交互式编程

交互式编程是指编辑完一行代码,回车后会立即执行并显示运行结果。在 test_py3 环境中输入 Python 命令回车后,出现>>>,进入交互式编程模式,如图 3.21 所示。

```
(test_py3) C:\Users\Administrator>python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

图 3.21 进入交互式编程模式

在>>>之后输入 Python 语言的各种命令。例如,输入 print('Hello world!')命令,如图 3.22 所示。

```
>>> print('Hello world!')
Hello world!
```

图 3.22 print()输出

2. 脚本式编程

Python 和其他脚本语言,如 java、R、Perl 等一样,可以直接在命令行里运行脚本程序。首先,在 D:\目录下创建 Hello.py 文件,内容如图 3.23 所示。



```
print("Hello world!")
print("Hello Again")
```

图 3.23 Hello.py 文件内容

其次,进入 test_py3 环境后输入 Python d:\Hello.py 命令,运行结果如图 3.24 所示。

```
(base) C:\Users\Administrator>python d:\Hello.py
Hello world!
Hello Again
```

图 3.24 运行 d:\Hello.py 文件

3. Spyder

单击 Anaconda 应用中的最后一个项目——Spyder。Spyder 是 Python 的集成开发环境,如图 3.25 所示。

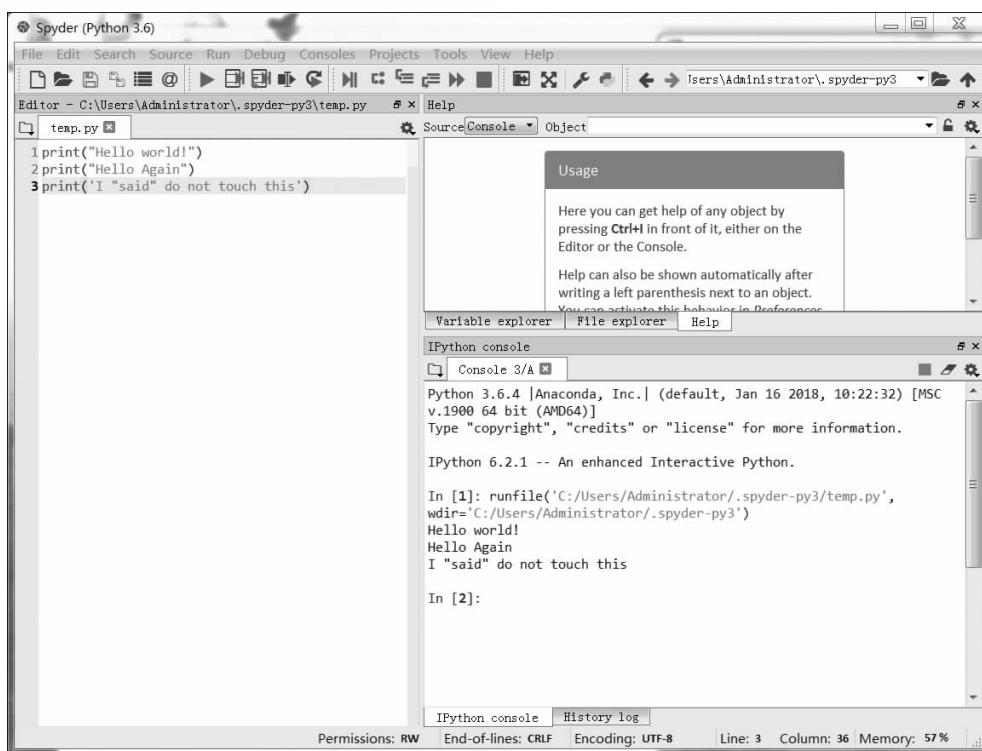


图 3.25 Spyder 编辑器

3.4 Python 测试框架

自动化测试工具需要脚本语言的支持,而 Python 作为“胶水语言”,其应用范围广泛。和 Python 相关的测试框架有 unittest、Pywinauto、Selenium、Pylot、Appium、pytest 等。

3.4.1 unittest

unittest 模块实现单元测试。单元测试是由开发人员(而不是测试人员)完成的测试,用于保证一个程序基本单元的正确性。单元测试框架代替开发人员完成了一些调用、IO 等与单元测试无直接关系的支撑代码,让开发人员可以专注于测试用例的编写,简化单元测试工作。

unittest 模块的 API 与 Java 的 JUnit、.net 的 NUnit、C++ 的 CppUnit 很相似,第 4 章进行详细介绍。

3.4.2 Pywinauto

Python 提供 Pywinauto 开源的框架进行图形用户界面(Graphical User Interface, GUI)测试,对开发环境可复用的构件进行操作的正误判断。

Pywinauto 与 QTP 测试工具的功能类似,用于测试 Windows 控件的一系列动作,如指定窗口、鼠标或键盘操作,获得控件属性等。Pywinauto 的官网是 <http://pypi.python.org/pypi/pywinauto/0.4.0>。

Pywinauto 的操作步骤如下所示:

步骤 1: 下载 Pywinauto,网址是 <https://sourceforge.net/projects/pywinauto/files/>,如图 3.26 所示。



图 3.26 Pywinauto 下载网页

步骤 2: 下载 pywinauto-0.4.0.zip 文件,解压缩到 C:\pywinauto-0.4.0,在 DOS 下进入 pywinauto-0.4.0 目录,执行如下安装命令: `python setup.py install`,如图 3.27 所示。

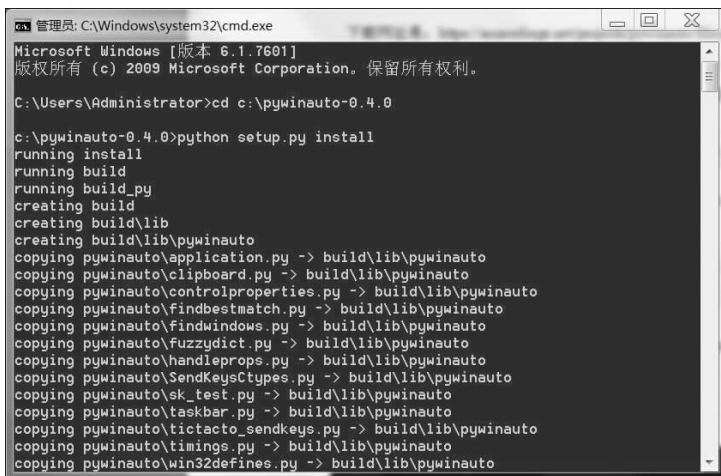


图 3.27 Pywinauto 安装界面

【例 3.1】 Pywinauto 举例。

```
from pywinauto import Application    #引入 Pywinauto 模块
import time
app=Application.start("notepad")     #调用 notepad
```

```

app.__setattr__("name", "notepad")
time.sleep(2)
app.Notepad.edit.TypeKeys('hello!')
i=0
while i<=10:
    app.Notepad.edit.TypeKeys('test')
    i+=1
time.sleep(2)
app.Dialog.Button1.Click()
time.sleep(1)
app.Notepad.Close()

```

程序运行结果如图 3.28 所示。

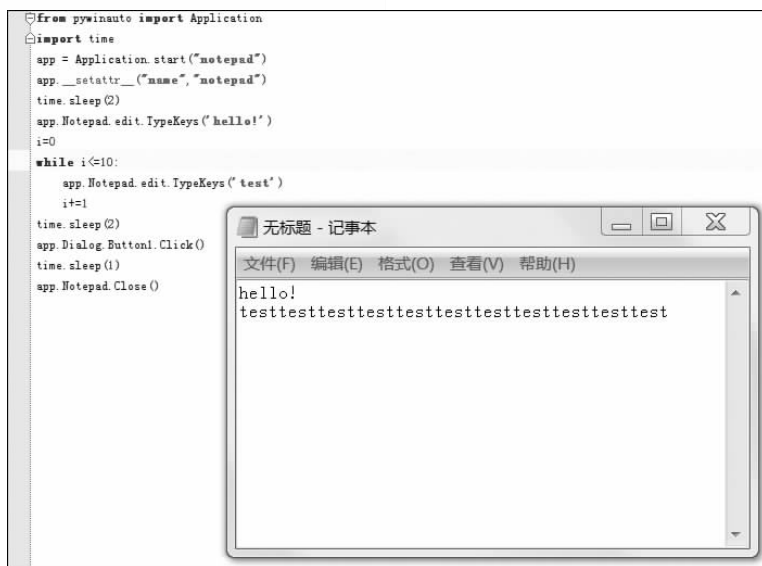


图 3.28 Pywinauto 的运行结果

3.4.3 Selenium

Python 进行 Web 自动化测试的工具很多,如 Selenium、RF 和 twill 等。其中较常用的是 Selenium。Web 负载测试解决方案几乎都是采用“录制-回放”的技术。通过捕获用户的每一步操作,如界面的像素坐标或对象(窗口、按钮、滚动条等)的位置,以及状态或属性的变化,可以用脚本语言记录。回放时,将脚本语言转换为屏幕操作,对比被测系统的输出记录与预先标准记录之间的关系。

Selenium 在第 5 章进行详细介绍。

3.4.4 Pylot

Pylot 工具实现的压力性能测试,用于测试被测应用程序能够承受的压力,即同时能够承受的用户访问量(容量),最多支持有多少用户同时访问某个功能。

Pylot 是 Python 编写的用以测试 Web 性能和扩展性的工具,进行 http 负载测试。Pylot 下载网址为 <http://www.pylot.org/>,如图 3.29 所示。



图 3.29 Pylot 下载网址

Pylot 具有以下优点:

(1) Pylot 是通过 Python 语言开发的第三方软件,继承了 Python 简洁优雅的代码风格,通俗易懂,十分容易上手。

(2) Pylot 体积小,十分适合个人网站以及中小型企业网站测试网站性能。它支持 http 和 https 请求,有着多线程负载生成器,可以自动处理 cookies,支持响应正则表达式,在安装了 wxpython 后还支持 GUI 模式,具有测试实时统计功能,有良好的跨平台性,支持 Windows 系统和 Linux 系统。

(3) Pylot 在进行压力测试时会模拟大量主机发起并发请求,在测试过程中检验服务器的响应,测试完成后会生成关于本次测试的结果报表。使用过程中,如果配合上 Numpy、Matplotlib,就可以在最后的测试报表中自动绘制图表来反映对被测试网站的测试过程。

【例 3.2】 Pylot 举例。

下载并解压 Pylot,如图 3.30 所示。

Pylot 测试步骤如下所示。

步骤 1: 配置 testcases.xml。

在 pylot_1.26 文件夹里修改 testcases.xml 文件,用记事本打开,将需要测试的网页地址添加进去。

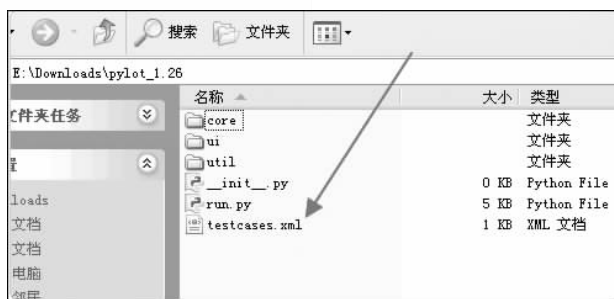


图 3.30 下载并解压 Pylot

```
<testcases>
  <!--SAMPLE TEST CASE -->
  <case>
    <url>http://www.example.com/</url>
  </case>

  <!--SAMPLE TEST CASE -->
  <!--
  <case>
    <url>http://search.yahooapis.com/WebSearchService/V1/webSearch</url>
    <method>POST</method>
    <body><![CDATA[appid=YahooDemo&query=pylot]]></body>
    <add_header>Content-type: application/x-www-form-urlencoded</add_
    header>
  </case>
  -->
</testcases>
```

在上面的代码中,把 `http://www.example.com/` 改为需要测试的网址,然后保存文件。

步骤 2: 执行测试命令。

在 Pylot 目录下执行 `run.py`。如并发 10 台主机,测试时间 3s,就可以执行以下命令:

```
python run.py -a 10 -d 3
```

其中 `a` 即 agent,代表并发的连接主机数,`d` 即 duration,表示测试时间,以显示测试进度。测试过程如图 3.31 所示。

在生成报表后,在 `pylot_1.26` 文件夹下的 `result` 文件夹中会有测试完成后所生成的数据图表,如图 3.32 所示。

3.4.5 Appium

Python 提供 Appium 测试框架进行移动测试(Mobile Testing)。移动测试是指对移动设备提供真机测试服务的平台,用于发现 App 中的各类隐患,如应用崩溃、各类兼容性

```

C:\pilot_1.26>python run.py -a 10 -d 3

-----
Test parameters:
  number of agents:      10
  test duration in seconds: 3
  rampup in seconds:     0
  interval in milliseconds: 0
  test case xml:         testcases.xml
  log messages:          False

Started agent 10

All agents running...

[#####100%#####] 3s/3s

Requests: 133
Errors: 0
Avg Response Time: 0.097
Avg Throughput: 43.06
Current Throughput: 02
Bytes Received: 8102271
-----

```

图 3.31 测试过程

Pylot - Performance Results			
report generated: 05/04/2016 22:09:14			
test start: 05/04/2016 22:09:05			
test finish: 05/04/2016 22:09:14			
Workload Model			
test duration (secs)	9		
agents	10		
rampup (secs)	0		
interval (millisecs)	0		
Results Summary			
requests	133		
errors	0		
data received (bytes)	13219754		
Response Time (secs)		Throughput (req/sec)	
avg	0.412	avg	16.625
stdev	1.025	stdev	23.585
min	0.084	min	1
50th %	0.136	50th %	4
80th %	0.173	80th %	54
90th %	0.806	90th %	55
95th %	2.213	95th %	55
99th %	6.991	99th %	55
max	7.025	max	55

图 3.32 所生成报表

问题、功能性问题、性能问题等。

Appium 测试框架在第 8 章进行详细介绍。

3.4.6 Pytest

Pytest 是 Python 最流行的单测框架之一。它与 Python 自带的 unittest 测试框架类似,提供测试用例的详细失败信息,使得开发者可以快速准确地改正问题,还兼容 unittest、doctest 和 nose 等测试工具。

Pytest 具有如下优点：

- 允许直接使用 `assert` 进行断言,而不需要使用 `self.assert *`。
- 可以自动寻找单测文件、类和函数。
- 非常容易上手,入门简单,文档丰富,文档中有很多实例可以参考。
- 能够支持简单的单元测试和复杂的功能测试。
- 支持运行由 `nose`、`unittest` 编写的测试 case。
- 具有很多第三方插件,并且可以自定义扩展。
- 方便地和持续集成工具集成。

在 Anaconda Prompt 下使用命令 `pip install -U pytest` 进行安装,如图 3.33 所示。

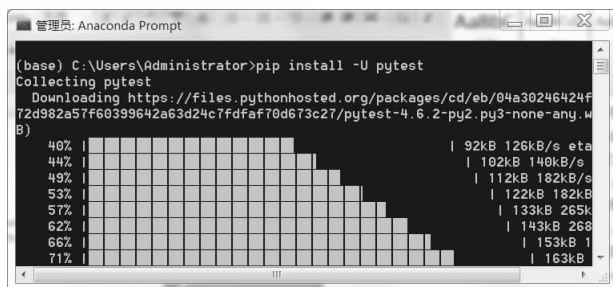


图 3.33 安装 Pytest

【例 3.3】 Pytest 举例。

实现如下功能：

- (1) 定义被测试函数 `inc`,将传递进来的参数加 1 后返回。
- (2) 定义测试函数 `test_answer` 对 `func` 进行测试。
- (3) 在 `test_answer` 中利用断言进行结果验证。
- (4) 保存代码,命名为 `test_sample.py`。

`test_sample.py` 代码如下所示：

```
def inc(x):
    return x+1

def test_answer():
    assert inc(3)==5
```

在命令行执行如下命令运行：

```
Pytest test_sample.py
```

运行完成后,可以得到如图 3.34 所示结果。

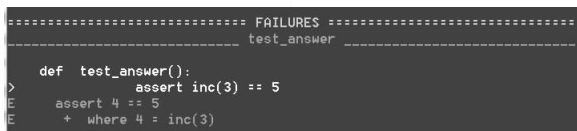


图 3.34 程序运行结果