第4节 测试环境准备

> 在了解了Selenium这个工具之后，接下来的工作就是搭建Selenium环境了，本系列博客基于Python 3.x + Selenium 3.x展开。

1. Selenium安装

Selenium 设置与其他商业工具的设置完全不同。 要在自动化项目中使用 Selenium，您需要为您选择的语言安装语言绑定库。 此外，对于要自动运行并运行测试的浏览器，您将需要 WebDriver 二进制文件。

一、首先，你需要为自动化项目安装 **Selenium** 绑定库，库的安装过程取决于您选择使用的语言

基于Python的技术栈，可以使用 pip 安装 Selenium 库：`pip install selenium -i https://pypi.tuna.tsinghua.edu.cn/simple` 或者下载源码 **PyPI source archive** (selenium-x.x.x.tar.gz) 并使用 *setup.py* 进行安装：`python setup.py install` 。

二、其次，要执行项目并控制浏览器，需要安装特定于浏览器的 **WebDriver** 二进制文件

当前Selenium框架支持的浏览器及对应驱动如下：

浏览器	支持的操作系统	维护者	支持的版本	驱动下载
Chromium/Chrome	Windows/macOS/Linux	Google	所有版本	Downloads
Firefox	Windows/macOS/Linux	Mozilla	54及以上版本	Downloads
Edge	Windows 10	Microsoft	所有版本	Downloads
Internet Explorer	Windows	Selenium Project	6及以上版本	Downloads
Safari	macOS El Capitan and newer	Apple	10及以上版本	内置
Opera	Windows/macOS/Linux	Opera	10.5及以上版本	Downloads

三、最后，配置浏览器驱动位置

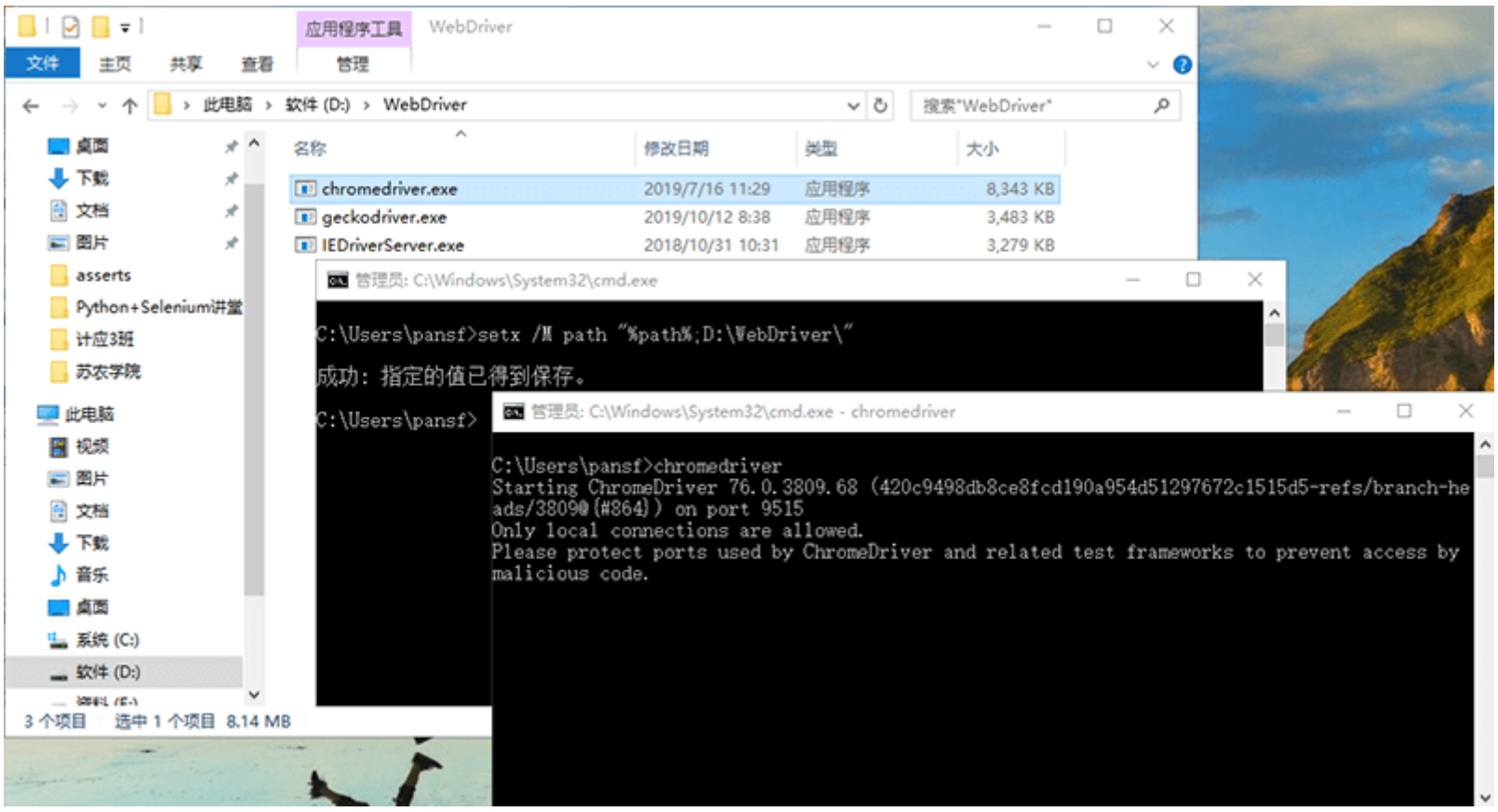
基于Selenium WebDriver编写的自动化脚本需要通过浏览器的WebDriver 来驱动WEB 浏览器的行为，所以必须让脚本知道对应的浏览器驱动的所在位置，你可以选择在你的测试脚本中指定可执行文件的存放位置，但这会使测试的可移植性降低，因为驱动程序必须位于每台计算机上的同一位置，或包含在测试代码存储库中。

我们推荐通过将包含WebDriver二进制文件的文件夹添加到系统路径，Selenium将能够找到所需的驱动程序，而无需通过测试代码来定位驱动程序的确切位置。

1. 创建一个目录来放置驱动程序，如：`D:\WebDriver`
2. 将该目录添加到你的系统变量`PATH`中
3. 在Windows上，以管理员身份打开命令提示符，然后运行以下命令将目录永久添加到计算机上所有用户的路径中：

```
setx /m path "%path%;D:\WebDriver\"
```

4. 现在你可以测试配置是否生效了。关闭所有打开的命令提示符，然后打开一个新的提示符。 输入您在上一步中创建的文件夹中的二进制文件之一的名称，例如：`chromedriver`
5. 如果您的PATH配置正确，你将看到一些有关驱动程序启动的输出：



2. 小试牛刀

场景描述：打开示例网站应用系统 (iWebSNS) 网址： http://iwebsns.bljt.top，等待3s后，关闭示例网站，退出浏览器。

```
# 从selenium库中导入 webdriver 模块
from selenium import webdriver
import time

browser = webdriver.Chrome()                # 创建基于Chrome浏览器的驱动对象，通过该驱动对象打开Chrome浏览器
browser.get("http://iwebsns.bljt.top/")     # 加载示例网站
time.sleep(3)
browser.close()             # 关闭示例网站
browser.quit()              # 释放驱动对象，结束执行
```

预知新作如何，请看下节分解。