Python+Selenium笔记（三）：使用unittest

#网络很慢可能会运行报错

（一）   前言

Selenium WebDriver是一个浏览器自动化测试的API集合。它提供了很多与浏览器自动化交互的特性，并且这些API主要是用于测试Web程序。如果仅仅使用Selenium WebDriver，我们无法实现执行测试前置条件、测试后置条件，比对预期结果和实际结果，检查程序的状态，生成测试报告，创建数据驱动的测试等功能。

（二）   unittest单元测试框架

unittest（一般称为PyUnit）是从Java程序开发中广泛应用的JUnit启发而来的。unittest使我们具备创建测试用例、测试套件、测试夹具的能力。

（三）   xUnite测试框架

Arrange：是用来初始化测试的前置条件，包含初始化被测试的对象、相关配置和依赖。

Act：用来执行功能操作。

Assert：用来校验实际结果与预期结果是否一致。

（四）   TestCase类

通过继承TestCase类并且在测试类中为每一个测试添加测试方法来创建单个测试或者一组测试。

（五）   使用unittest

（1）       定义一个继承于TestCase 类的子类。

（2）       在类中定义setUp()方法。

（3）       编写测试方法

（4）       代码清理（在类中定义tearDown()方法）

（5）       运行测试

说明：

1、  一个测试用例是从setUp()方法开始执行的，我们可以用这个方法在每个测试开始前去执行一些初始化的任务。可以是这样的初始化准备：比如创建浏览器实例，访问URL，加载测试数据和打开日志文件等。此方法没有参数，而且不返回任何值。当定义了一个setUp()方法，测试执行器在每次执行测试方法之前优先执行该方法。

2、  有了setUp()方法，就可以写一些测试用来验证我们想要测试的程序的功能。测试方法命名以test开头，这种命名约定通知test runner哪个方法代表测试方法。

3、  类似于setUp()方法在每个测试方法之前被调用，TestCase类也会在测试执行完成之后调用tearDown()方法来清理所有的初始化值。一旦测试被执行，在setUp()方法中定义的值将不再需要，所以最好的做法是在测试执行完成的时候清理掉由setUp()方法初始化的数值。

（六）   新建一个目录 data ，创建一个文件category.txt，录入以下数据

编程语言,Java,C++,PHP,Delphi,Python,Ruby,C语言,Erlang,Go,Swift,Scala,R语言,Verilog,其它语言

（七）   测试博客园首页编程语言下的分类信息是否与预期结果一致

import unittest

from selenium import webdriver  
from selenium.webdriver.common.action\_chains import ActionChains  
import time  
class SearchTest (unittest.TestCase):  
    def setUp(self):  
        self.driver  
= webdriver.Firefox()  
        self.driver.implicitly\_wait(10)  
        self.driver.maximize\_window()  
        self.driver.get("https://www.cnblogs.com/")  
  
    def test\_search\_by\_category(self):  
        #读取category.txt文件，返回一个字典  
        with open('data/category.txt', encoding='UTF-8') as category\_file:  
            category\_dict = dict()  
            category\_data =  
category\_file.readline().strip().split(',')  
            the\_class =  
category\_data.pop(0)  
            category\_dict[the\_class] =  
category\_data  
        #定位首页网站分类中的编程语言  
        self.seach\_class = self.driver.find\_element\_by\_xpath('//li/a[@href="/cate/2/"]')  
        #光标悬停在“编程语言”上  
        ActionChains(self.driver).move\_to\_element(self.seach\_class).perform()  
        # 以列表形式返回编程语言下的所有小类  
        seach\_small = self.driver.find\_elements\_by\_xpath( '//div[@id="cate\_content\_block\_2"]/div[@class="cate\_content\_block"]/ul/li')  
        #休眠3秒  
        time.sleep(3)  
        small\_cate = []  
        for s in seach\_small:  
            #去掉小类最后面的3个字符（0），并添加到列表small\_cate中  
            small\_cate.append(str(s.text)[:-3])  
        #检查表达式是否为true（此处检查编程语言下的小类是否与预期结果一致）  
        self.assertTrue(small\_cate == category\_dict["编程语言"])  
        # self.assertEqual(small\_cate,category\_dict["编程语言"])  
  
    def tearDown(self):  
        self.driver.quit()  
#加上下面2句，可以通过命令行运行测试，不加的话不影响通过IDE运行测试  
if \_\_name\_\_ == '\_\_main\_\_':  
    #加verbosity=2参数，在命令行中显示具体的测试方法  
    unittest.main(verbosity=2)

（八）    
测试结果如下

通过命令行运行：





