Python+Selenium笔记（四）：unittest的Test Suite（测试套件）

（一） Test Suite测试套件

一个测试套件是多个测试或测试用例的集合，是针对被测程序的对应的功能和模块创建的一组测试，一个测试套件内的测试用例将一起执行。

应用unittest的TestSuites特性，可以将不同的测试组成一个逻辑组，然后设置统一的测试套件，并通过一个命令来执行测试。这都是通过TestSuites、TestLoader和TestRunn类来实现的。

（二） 类级别的setUp()方法和tearDown()方法

使用setUpClass()方法和tearDownClass()方法及@classmethod标识来实现各个测试方法共享初始化数据。具体看下面的代码。

（三） searchtest.py

import unittest

from selenium import webdriver

from selenium.webdriver.common.action\_chains import ActionChainsimport time

（六） 其他说明

#另外一个文件在上一篇中已经说明

要完全使用上面的代码，需要创建一个文件menu.txt，输入以下数据：

（七） 运行smoketests.py

class SearchTest(unittest.TestCase):  
    '''通过setUpClass()和@classmethod标识,实现  
     在类级别初始化数据，所有测试方法共享这些初始化数据.  
     不使用这个的话，每个测试方法都会单独创建一个实例'''  
    @classmethod  
    def setUpClass(cls):  
        cls.driver = webdriver.Firefox()  
        cls.driver.implicitly\_wait(10)  
        cls.driver.maximize\_window()  
        cls.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.search\_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.search\_class).perform()  
        # 以列表形式返回编程语言下的所有小类  
        self.search\_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 self.search\_small:  
            #去掉小类最后面的（0），并添加到列表small\_cate中  
            small = str(s.text).split('(')  
            small\_cate.append(small[0])  
        #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 test\_search\_by\_look(self):  
        self.seach\_class = self.driver.find\_element\_by\_xpath('//li/a[@href="/cate/2/"]')  
        #定位编程语言下的小类Python  
        self.seach\_small =self.driver.find\_element\_by\_xpath('//li/a[@href="/cate/python/"]')  
        ActionChains(self.driver).move\_to\_element(self.seach\_class).perform()  
        self.seach\_small.click()  
        #检查打开的网页标题是不是 Python - 网站分类 - 博客园  
        # assert "Python - 网站分类 - 博客园" in self.driver.title  
        self.assertEqual(self.driver.title,"Python - 网站分类 - 博客园" )  
    '''通过tearDownClass()和@classmethod标识,实现  
     在类级别初始化数据，所有测试方法共享这些初始化数据'''  
    @classmethod  
    def tearDownClass(cls):  
        cls.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)

（四） homepagetest.py

import  unittest  
from selenium import  webdriver  
from selenium.common.exceptions import NoSuchElementException  
from selenium.webdriver.common.by import By  
class HomePageTest(unittest.TestCase):  
    '''通过setUpClass()和@classmethod标识,实现  
     在类级别初始化数据，所有测试方法共享这些初始化数据.  
     不使用这个的话，每个测试方法都会单独创建一个实例'''  
    @classmethod  
    def setUpClass(cls):  
        cls.driver = webdriver.Firefox()  
        cls.driver.implicitly\_wait(10)  
        cls.driver.maximize\_window()  
        cls.driver.get("https://www.cnblogs.com/")  
    def test\_search\_field(self):  
        #通过by,检查博客园首页有没有搜索框,is\_element\_present()是自定义的方法  
        self.assertTrue(self.is\_element\_present(By.ID,"zzk\_q"))  
    def test\_search\_btn(self):  
        # 通过by,检查博客园首页有没有找找看按钮  
        self.assertTrue(self.is\_element\_present(By.CLASS\_NAME,"search\_btn"))  
    def test\_menu(self):  
        #该方法检查博客园首页菜单栏信息是否与预期一致  
        #读取menu.txt文件数据  
        with open('data/menu.txt',encoding='UTF-8') as menu\_file:  
            menu\_data = menu\_file.readline().strip().split(',')  
        #以列表形式返回博客园首页菜单栏信息  
        self.check\_menu = self.driver.find\_elements\_by\_xpath('//div[@id="nav\_menu"]/a')  
        the\_menu = []  
        for c in self.check\_menu:  
            #将博客园首页的菜单名称和URL添加到列表the\_menu  
            the\_menu.append(c.text + c.get\_attribute('href'))  
        #检查2个列表是否一致（检查博客园首页的菜单名称及URL是否和预期一致）  
        self.assertListEqual(the\_menu,menu\_data)  
    def is\_element\_present(self,how,what):  
        #找到元素，返回True,否则返回False  
        try:  
            self.driver.find\_element(by=how,value = what)  
        except NoSuchElementException as e:  
            return False  
        return True  
    @classmethod  
    def tearDownClass(cls):  
        cls.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)

（五） smoketests.py （测试套件）

import unittest  
from searchtest import SearchTest  
from homepagetest import HomePageTest  
#获取SearchTest类 和 HomePageTest类中的所有测试方法  
search\_test = unittest.TestLoader().loadTestsFromTestCase(SearchTest)  
home\_page\_test = unittest.TestLoader().loadTestsFromTestCase(HomePageTest)  
#创建一个包括SearchTest和HomePageTest的测试套件  
smoke\_tests = unittest.TestSuite([home\_page\_test,search\_test])  
#运行测试套件  
unittest.TextTestRunner(verbosity=2).run(smoke\_tests)

园子https://home.cnblogs.com/,新闻https://news.cnblogs.com/,博问https://q.cnblogs.com/,闪存https://ing.cnblogs.com/,小组https://group.cnblogs.com/,收藏https://wz.cnblogs.com/,招聘https://job.cnblogs.com/,班级https://edu.cnblogs.com/,找找看http://zzk.cnblogs.com/



