

selenium原理及脚本

讲师：芳姐

金斧子

| 5 金斧子五周年
ANNIVERSARY

科技提升投资品质

1、selenium3原理

1.1 webdriver

在Selenium 2.0 以后主推的是WebDriver,Selenium又名Selenium Webdriver

Selenium2以后将浏览器原生的API封装成WebDriver API，可以直接操作浏览器页面里的元素，甚至操作浏览器本身（截屏，窗口大小，启动，关闭，安装插件，配置证书之类的），所以就像真正的用户在操作一样。

Webdriver的一个简单的架构图，如下图所示：



- 1.2 webdriver工作原理
- webdriver是按照server-client的经典设计模式设计的:
- server端就是remote server，可以是任意的浏览器：我们的脚本启动浏览器后，该浏览器就是remote server，它的职责就是等待client发送请求并做出相应；
- client端简单说来就是我们的测试代码：我们测试代码中的一些行为，比如打开浏览器，转跳到特定的url等操作是以http请求的方式发送给被server端（也就是被测浏览器）server接受请求，并执行相应操作，并在response中返回执行状态、返回值等信息；
- 简单介绍一下webdriver的工作原理：
- （1）启动浏览器后，selenium-webdriver会将目标浏览器绑定到特定的端口，启动后的浏览器则作为webdriver的remote server。

- (2) 客户端(也就是测试脚本) , 借助ComandExecutor发送HTTP请求给sever端 (通信协议 : The WebDriver Wire Protocol , 在HTTP request的body中 , 会以WebDriver Wire协议规定的JSON格式的字符串来告诉Selenium我们希望浏览器接下来做什么事情) 。
- Sever端需要依赖原生的浏览器组件 , 转化Web Service的命令为浏览器native的调用来完成操作。
- 注 :
- the WebDriver Wire Protocol是Selenium自己设计定义的协议 , 这套协议非常之强大 , 几乎可以操作浏览器做任何事情 , 包括打开、关闭、最大化、最小化、元素定位、元素点击、上传文件等。
- WebDriver Wire协议是通用的 , 也就是说不管FirefoxDriver还是ChromeDriver , 启动之后都会在某一个端口启动基于这套协议的Web Service。

• 1.3 配置WebDriver

• 具体步骤如下：

- (1) 安装jdk，配置好Java环境变量
- (2) 在idea工具中，新建maven工程
- (3) 配置好maven环境变量
- (4) 配置pom.xml文件

```
<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
</properties>

<dependencies>
  <!-- 配置selenium依赖 -->
  <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>3.7.0</version>
  </dependency>

  <!-- 配置testng依赖 -->
  <dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>RELEASE</version>
  </dependency>

  <dependency>
    <groupId>com.relevantcodes</groupId>
    <artifactId>extentreports</artifactId>
    <version>2.41.2</version>
  </dependency>
  <dependency>
    <groupId>org.apache.commons</groupId>
    <artifactId>commons-collections4</artifactId>
    <version>4.1</version>
  </dependency>
</dependencies>
```

• 1.3 第一个脚本代码

- `public class FirstTestngDemo {`
- `static WebDriver driver;`
- `static String baseUrl = "http://10.1.2.211:8080/login.jsp"; //访问登录的网址`
- `@BeforeMethod //注解的方法将每个测试方法之前运行`
- `public void setUp() throws Exception {`
- `// 设定连接chrome浏览器驱动程序所在的磁盘位置`
- `System.setProperty("webdriver.chrome.driver", "D:\\Selenium_Automated\\driver\\chromedriver.exe");`
- `driver = new ChromeDriver();`
- `//设置隐性等待时间`
- `driver.manage().timeouts().implicitlyWait(8, TimeUnit.SECONDS);`
- `driver.get(baseUrl + "");`
- `}`

- @Test //标记一个类或方法作为测试的一部分
- public void FpLogin() throws Exception {
-
- driver.manage().window().maximize();
- // 文本框内输入用户名
- driver.findElement(By.name("username")).sendKeys("defang1");
- // 文本框内输入密码
- driver.findElement(By.name("password")).sendKeys("123456");
- // 点击登录
- driver.findElement(By.className("submit_wrap")).click();
- String title = driver.getTitle();
- System.out.print("当前的标题是：" + driver.getTitle());
- Assert.assertEquals("深圳市金斧子网络科技有限公司-ERP",title);
- }

- @AfterMethod //被注释的方法将被运行后，每个测试方法
- public void tearDown() throws Exception {
- driver.quit(); //关闭浏览器
- }
- }

查看源码的方法

```
est > java > cn > TestScripts > FirstTestngDemo  
Selenium_Automated x FirstTestngDemo.java x  
6  
7 @BeforeMethod //注解的方法将每个测试方法之前运行  
8 public void setUp() throws Exception {  
9  
10 // 设定连接chrome浏览器驱动程序所在的磁盘位置  
11 System.setProperty("webdriver.chrome.driver", "D:\\Selenium_Automated\\driver\\chromedriver.exe");  
12 driver = new ChromeDriver();  
13 WebDriver  
14 public abstract void get(String s)  
15 driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);  
16 driver.get(baseUrl + "");  
17 }  
18  
19 @Test //标记一个类或方法作为测试的一部分  
20 public void FpLogin() throws Exception {  
21  
22  
23 driver.manage().window().maximize();  
24 // 文本框内输入用户名  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
FirstTestngDemo
```

```
selenium > WebDriver >  
m_Automated × FirstTestngDemo.java × WebDriver.class ×  
d.class file, bytecode version: 52.0 (Java 8)  
+ /... /  
  
package org.openqa.selenium;  
  
+ import ...  
  
public interface WebDriver extends SearchContext {  
    void get(String var1);  
  
    String getCurrentUrl();  
  
    String getTitle();  
  
    List<WebElement> findElements(By var1);  
  
    WebElement findElement(By var1);  
  
    String getPageSource();  
WebDriver > get()
```

1.4 从代码角度去解释启动firefox的过程

为了更好去描述和理解这个过程，我们举例，通过查找源码的方式去理解Selenium启动谷歌浏览器的过程

```
System.setProperty("webdriver.chrome.driver", "D:\\Selenium_Automated\\driver\\chromedriver.exe");
```

按下Ctrl+鼠标悬停在setProperty上方，点击鼠标左键，可以看到Java中setProperty的源码。自己去阅读下代码中关于setProperty的介绍。其实就是设置指定键对值的系统属性。上面webdriver.chrome.driver就是键，D:\\Selenium_Automated\\driver\\chromedriver.exe就是值。这样就把geckodriver设置成为系统的全局变量！这个时候driver就相当于一个静态变量，存放在内存里，直到driver关闭。

所谓的 system property，system 指的是 JRE (runtime)system，不是指 OS。设置指定键指示的系统属性，可以利用系统属性来加载多个驱动。所以，上面这行代码，就是通过键和值指定chrome的驱动位置。

```
WebDriver driver = ChromeDriver();
```

点击查看WebDriver发现是一个接口，它的备注这样写的：WebDriver是一个测试的主要接口，它展现了一个理想化的web浏览器，它主要包括三个目录。1) 控制浏览器本身 2) 查找和选择元素 3) 调试程序，比如异常处理。

driver这里是一个实例对象，学习了Java中类和对象，就应该不难理解。

new 是一个关键字，Java中通过new这个关键字，可以在内存中开辟一块空间，用来加载变量。

ChromeDriver(),是WebDriver这个接口在chrome上的一个实现具体类。ChromeDriver这个类里面,还包含一些firefox浏览器的一些选项设置。这行代码的意思用一句话来讲:初始化一个chrome类型的driver实例对象。这里除了chrome,还有IE, Safari,Firefox等对应的driver启动方法,你可以查看*\Selenium-Java-src\org\openqa\selenium,可以找到这些接口文件。

```
driver.manage().window().maximize();
```

这里driver,就是指上面我们初始化的firefox的实例对象,就是类似一个真实浏览器。manage是Options这个接口的一个方法,window().maximize(), window也是一个接口,这个接口下,有maximize这个方法,也就是最大化浏览器, window下也有全屏, 设置窗口大小的方法。

```
driver.manage().timeouts().implicitlyWait(8, TimeUnit.SECONDS);
```

manage上面提到是一个方法,直接来看timeouts, timeouts是接口Timeouts的一个实例对象,它的左右是针对webdriver实例管理超时的一个接口。implicitlyWait是一个隐式等待,当在一定时间内,如果还没有找到页面元素,就报超时。参数有两个,第一个是8,第二个是时间单位,这里选择秒,所以这里是8秒后超时。

```
driver.get(baseUrl);
```

这里的get方法的作用是，在当前浏览器窗口，加载一个新的web页面，是通过http get发生请求完成的。参数类型是String，一般是url。get方法就是打开一个网页的作用。

```
driver.quit();
```

退出有quit和close两种，这里quit表示退出当前浏览器，关闭这个浏览器有关联的所有窗口。

1.5 Navigation接口介绍

Navigation接口，主要包括平时浏览器的前进，后退，打开网址，刷新当前页操作。在Navigation接口下，我能找到下面四个方法：

`void back()`

`void forward()`

`void to(String url)`

`void to(URL url)`

`void refresh()`

先来解释下他们的作用

`back()`：浏览器上地址栏前面向左的箭头，即后退操作

`forward()`：浏览器上地址栏前面向右边的箭头，即前进操作，或者转到下一页。

`to(String url)`：在当前网页打开一个新的网页，这个和新的tab打开是有区别的
`to(URL url)`：也是一样，就是url被作为一个URL对象传入，这个不怎么使用，这里不介绍。

`refresh()`：浏览器刷新按钮操作，或者等同于按下F5。

下面用一个示例来演示上面几个方法的使用。

//to(String url)在当前页打开新的网页

```
driver.navigate().to("http://10.1.2.211:8080/erp/crm/buydetailFinal/get.ht?id=20000119546391");
```

//back () 后退到之前的购买记录页面
driver.navigate().back();

//forward()前进到新打开的网页
driver.navigate().forward();

Thread.sleep(2000);
//刷新当前页
driver.navigate().refresh();


```
from selenium import webdriver
```

```
driver = webdriver.Chrome()
```

```
driver.maximize_window() # 最大化浏览器
```

```
driver.implicitly_wait(8) # 设置隐式时间等待
```

```
driver.get("http://10.1.2.58:8080/login.jsp")
```

```
title = driver.title
```

```
#if (title == "深圳市金斧子网络科技有限公司-ERP"):
```

```
#    print("测试成功，结果和预期结果匹配！")
```

```
assert "深圳市金斧子网络科技有限公司-ERP" in driver.title
```

```
print("标题" + driver.title)
```

```
#driver.quit()
```

作业

- 1、完成登录的脚本（Java，Python）
- 2、熟悉怎么去查看源码
- 3、关键字驱动已经更新到GitHub:

http://gitlab.jfz.net/liudefang/Selenium_Automated/tree/master

Thanks!

科技提升投资品质

