****

**软件测试技术第二次实验报告**



**学 院 智能与计算学部**

**专 业 软件工程**

**年 级 2016级**

**姓 名 许致远**

**2019年 3 月 30 日**

# 软件测试技术第二次实验报告

1. 需求分析：

1.用Eclipse安装Selenium。

2.安装Firefox和SeleniumIDE插件。

3.尝试使用SeleniumIDE记录和导出脚本。

4.请使用Selenium完成以下任务：

“软件测试名单.xlsx”包含有关学生的信息，并且http://121.193.130.195:8800可以在登录后查看某人的信息（学生ID为用户名，学生ID的最后6位为密码）。请检查excel中的每条记录，以确保每个学生的信息与网站上的信息一致。

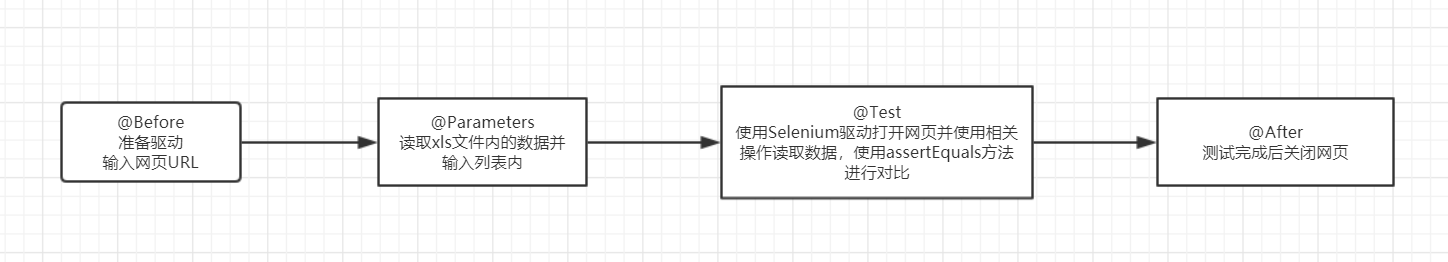
1. 概要设计（简单描述设计思路，配合UML图）

用Eclipse可以直接使用导入路径导入Selenium的geckodriver，从而直接启动Firefox来进行远程控制。

FireFox的SeleniumIDE可以直接在FireFox的插件商店里找到并安装。

使用SeleniumIDE记录脚本，即用SeleniumIDE记录访问百度的过程，可以回放以验证是否成功导入。

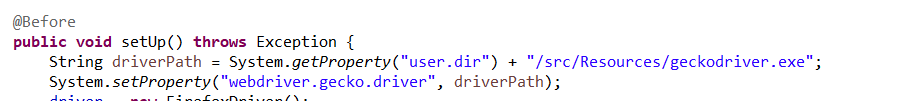
对于使用Selenium完成对学生信息的验证，我使用了Junit的集成化测试来测试每一条信息是否符合网页上的元素。在@Parameters里先读入XLS文件中的数据，输入四个字符串数组，再将数组的数据输入列表。在@Test内打开网页并提取网页元素与之前保存的列表元素，使用assertEquals方法进行一一对比。

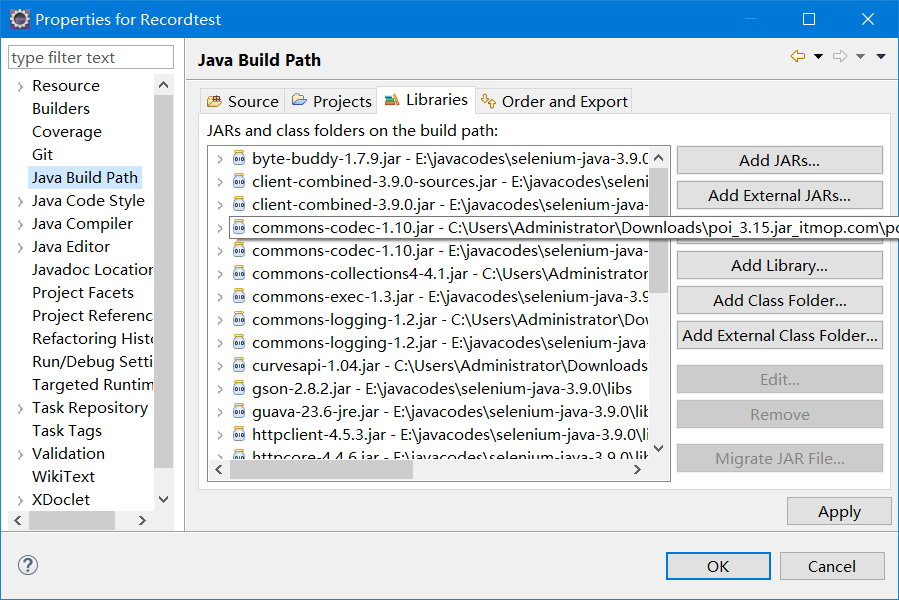


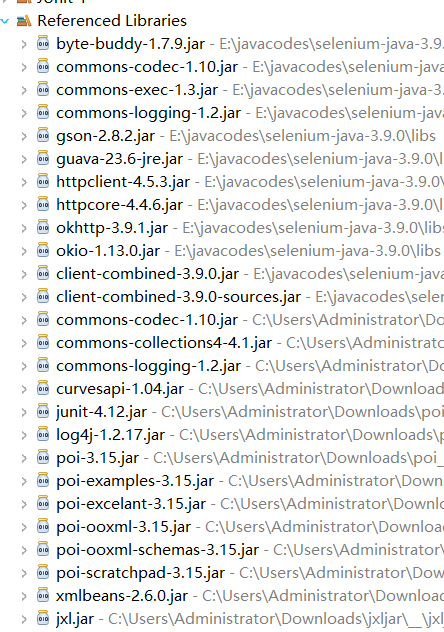
本次任务一共对比了142条不同数据。

1. 详细设计

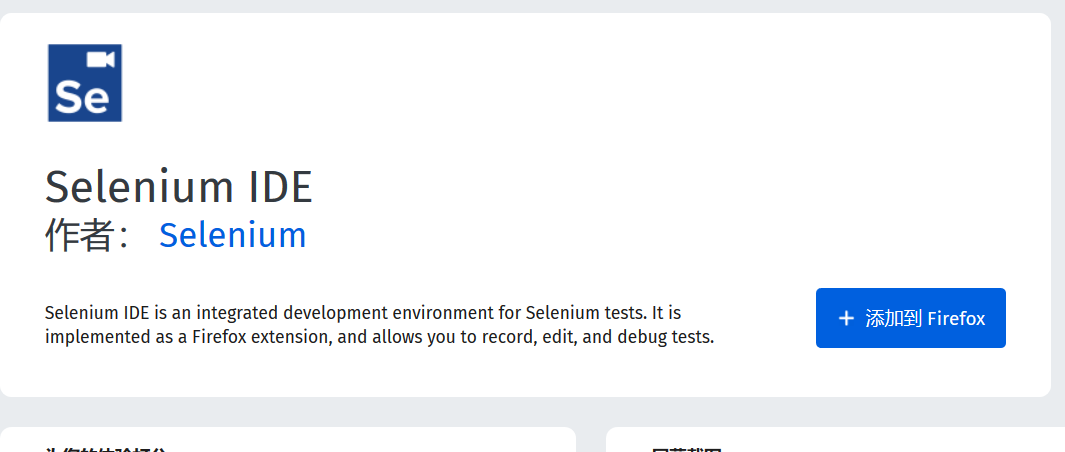
导入Selenium的驱动插件geckodriver.exe与各种需要的jar包

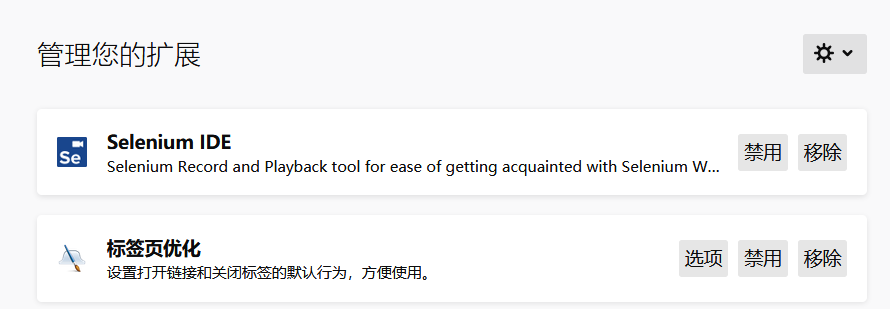




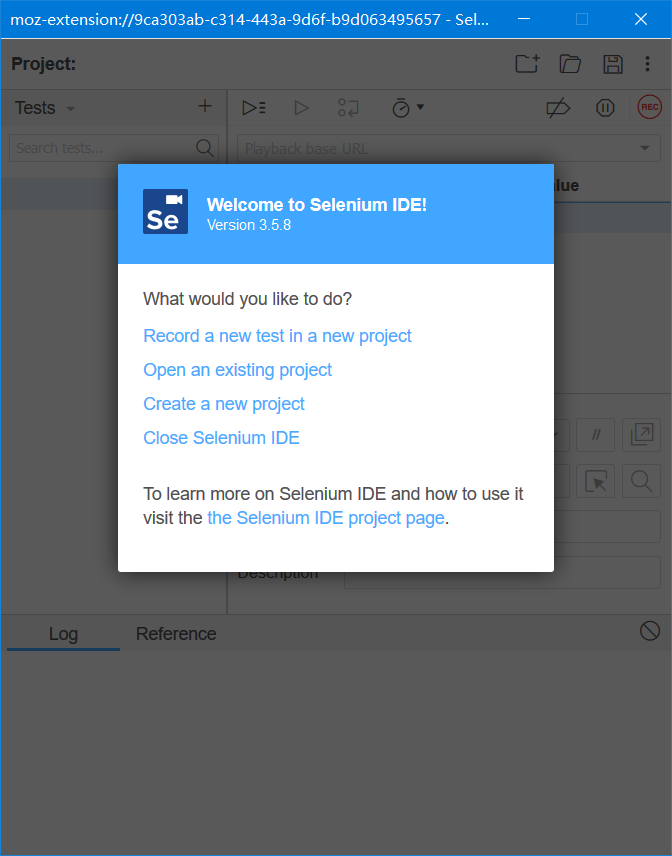


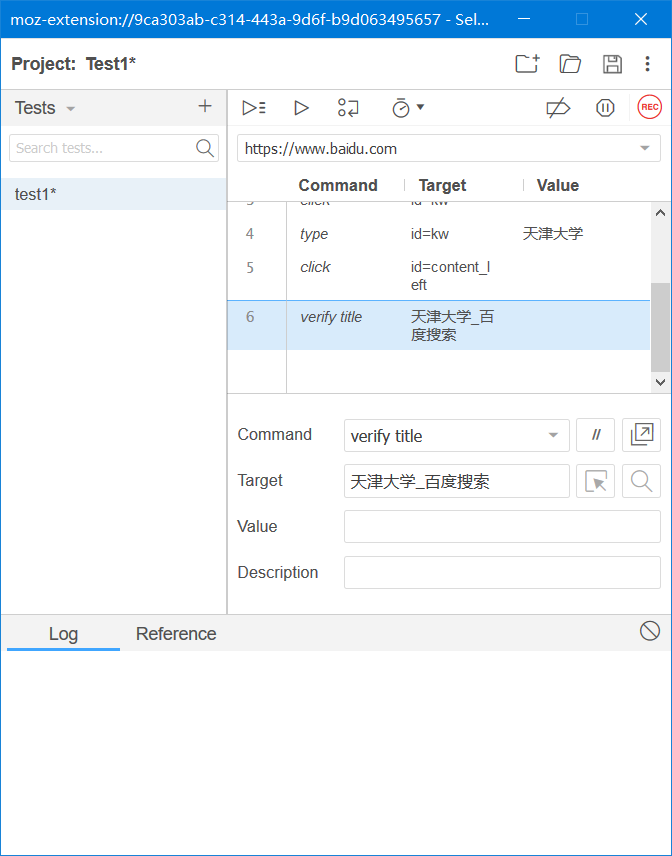
FireFox内SeleniumIDE的安装





使用Selenium录制脚本





按播放键可以重新播放脚本

Java程序：

package cn.tjucic.selenium;

import java.util.regex.Pattern;

import java.util.concurrent.TimeUnit;

import org.junit.\*;

import org.junit.runner.RunWith;

import org.junit.runners.Parameterized;

import org.junit.runners.Parameterized.Parameter;

import org.junit.runners.Parameterized.Parameters;

import static org.junit.Assert.\*;

import static org.hamcrest.CoreMatchers.\*;

import org.openqa.selenium.\*;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.support.ui.Select;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.FileReader;

import java.io.IOException;

import java.io.InputStream;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.Collection;

import java.util.List;

import jxl.Sheet;

import jxl.Workbook;

import jxl.read.biff.BiffException;

import jxl.Cell;

@RunWith(Parameterized.class)

public class TestBaidu {

private String studentId;

private String github;

private String passwd;

private String studentName;

private WebDriver driver;

private String baseUrl;

private boolean acceptNextAlert = true;

private StringBuffer verificationErrors = new StringBuffer();

public TestBaidu(String id,String passwd,String git,String names) {

this.studentId = id;

this.github = git;

this.passwd = passwd;

this.studentName = names;

}

@Before

public void setUp() throws Exception {

String driverPath = System.getProperty("user.dir") + "/src/Resources/geckodriver.exe";

System.setProperty("webdriver.gecko.driver", driverPath);

driver = new FirefoxDriver();

baseUrl = "http://121.193.130.195:8800";

//121.193.130.195:8800

driver.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);

}

@Parameters

public static Collection<Object[]> getData() throws BiffException, IOException{

File f=new File(System.getProperty("user.dir") + "/src/软件测试名单.xls");

String[] stuid = new String[145];

String[] stupw = new String[145];

String[] stugh = new String[145];

String[] stunm = new String[145];

Workbook book=Workbook.getWorkbook(f);//

Sheet sheet=book.getSheet(0); //获得第一个工作表对象

for(int i=2;i<sheet.getRows();i++){

Cell cell1=sheet.getCell(1, i);

Cell cell2=sheet.getCell(2, i);

Cell cell3=sheet.getCell(3, i);

stuid [i-2] = cell1.getContents();

stupw [i-2] = cell1.getContents();

stugh [i-2] = cell3.getContents();

stunm [i-2] = cell2.getContents();

}

for(int s = 0 ;s<142;s++)

{

stupw[s] = stupw[s].substring(4);

}

Object[][] objects = new Object[142][4];

for(int i = 0; i < objects.length; i++)

{

objects[i][0] = stuid[i];

objects[i][1] = stupw[i];

objects[i][2] = stugh[i];

objects[i][3] = stunm[i];

}

List<Object[]> strlist = Arrays.asList(objects);

return strlist;

}

@Test

public void testBaidu() throws Exception {

driver.get(baseUrl + "/");

WebElement we = driver.findElement(By.name("id"));

we.click();

// driver.findElement(By.id("kw")).click();

driver.findElement(By.name("id")).clear();

driver.findElement(By.name("id")).click();

driver.findElement(By.name("id")).sendKeys(this.studentId);

driver.findElement(By.name("password")).click();

driver.findElement(By.name("password")).sendKeys(this.passwd);

driver.findElement(By.id("btn\_login")).click();

Object gitAddress = driver.findElement(By.id("student-git")).getText();

Object stuname = driver.findElement(By.id("student-name")).getText();

Object stuid = driver.findElement(By.id("student-id")).getText();

//System.out.println(gitAddress.toString());

assertEquals(this.github, gitAddress);

assertEquals(this.studentName, stuname);

assertEquals(this.studentId, stuid);

}

@After

public void tearDown() throws Exception {

driver.quit();

String verificationErrorString = verificationErrors.toString();

if (!"".equals(verificationErrorString)) {

fail(verificationErrorString);

}

}

private boolean isElementPresent(By by) {

try {

driver.findElement(by);

return true;

} catch (NoSuchElementException e) {

return false;

}

}

private boolean isAlertPresent() {

try {

driver.switchTo().alert();

return true;

} catch (NoAlertPresentException e) {

return false;

}

}

private String closeAlertAndGetItsText() {

try {

Alert alert = driver.switchTo().alert();

String alertText = alert.getText();

if (acceptNextAlert) {

alert.accept();

} else {

alert.dismiss();

}

return alertText;

} finally {

acceptNextAlert = true;

}

}

}

1. 调试分析

测试过程中导入jxl包时发生了问题，在更换包以后才得以解决。

在写程序的时候忘了使用构造函数内的参数，导致程序一直在报wrong number of arguments的数组越界问题，在仔细检查程序后解决。

在联网测试时，使用有线TJU测试，结果无法连接测试网站。问题在连接网站时FireFox一直默认使用系统代理导致无法测试，并且无法在默认启动的时候修改相关参数。

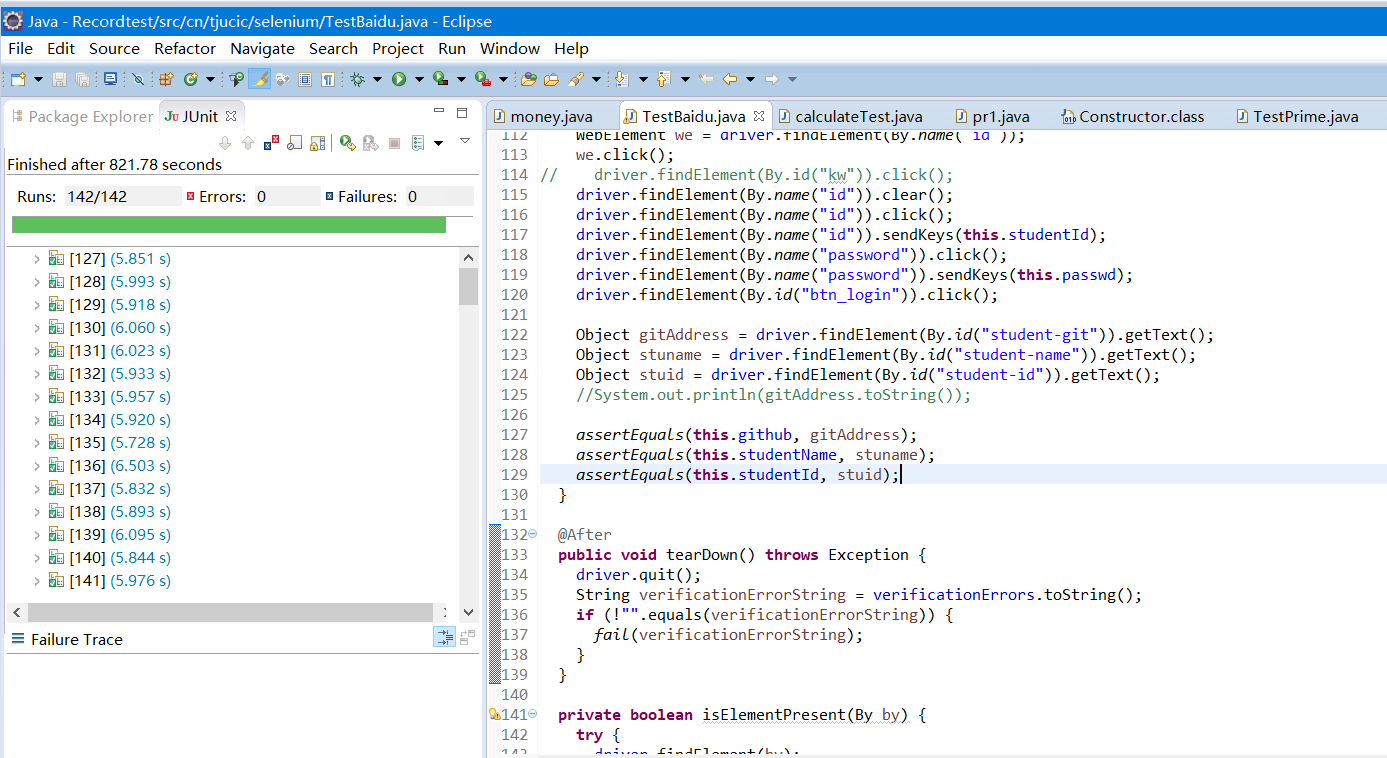
若是只连网线不登录TJU，结果是可以登入网站，但是由于网页的某些元素需要联网下载导致停在登录界面无法继续，导致报错并使测试失败。

在其他电脑上测试本程序时也会出现：

WebDriverException: java.net.SocketException: Read Timeout的超时问题。

最终解决方案是在机房连接无限TJU，最后问题解决。

1. 测试结果



如图，142个测试样例在821.78秒后全部测试完毕，测试结果为全部成功。

1. 总结

这次实验使我学会了如何使用Selenium进行测试脚本的录制，并且使用Junit和Selenium一起进行集成测试，学习到了其中的参数测试技术。由于测试需要的网络问题导致这次实验花费的时间相当长，很多时候是在与火狐浏览器的网络设置和TJU斗志斗勇。总结来看，软件测试不是一个简单的事，需要面对很多的问题与挑战，只有用耐心和细心才能将问题一一解决。