****

**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 刘 洋**

**2019年 03 月 28 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：

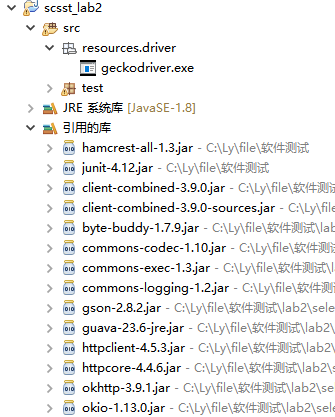
“软件测试名单.xlsx”包含有关学生的信息，并且http://121.193.130.195:8800可以在登录后查看某人的信息（学生ID为用户名，学生ID的最后6位为密码）。请检查excel中的每条记录，以确保每个学生的信息与网站上的信息一致。

1. 概要设计

对于前三项任务，根据相应规则下载完成浏览器以及插件的安装，第四项任务需求，先利用java提取Excel表格中的内容，再通过seleninm操作提取网页上的信息，通过Junit的assertEquals()函数来测试两者的数据是否相等，循环提取所有数据并进行判断

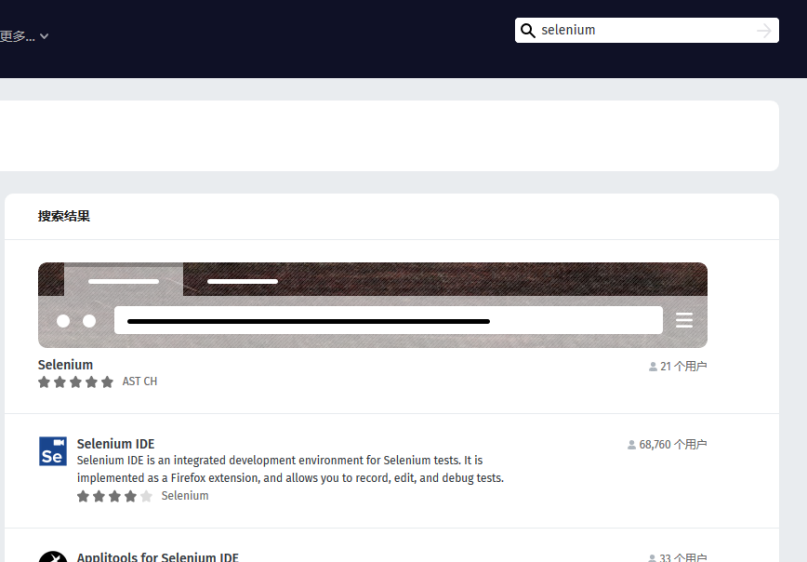
1. 详细设计
2. 用Eclipse安装Selenium

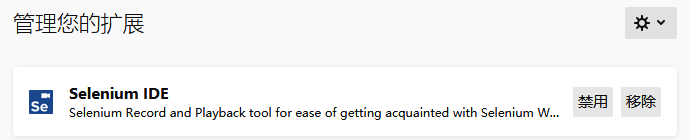
下载selenium-java-3.9.0.Zip包，将selenium导入到java项目中：



1. 安装Firefox和SeleniumIDE插件

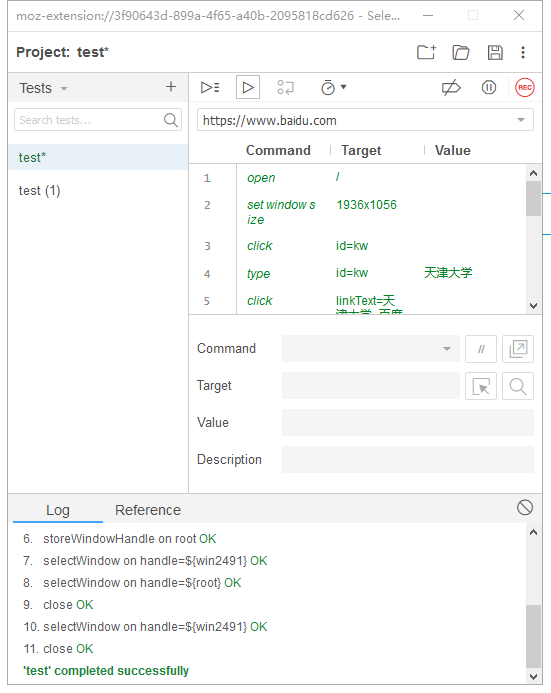
在官网下载并安装Firefox浏览器，在浏览器中右上角附加组件中添加SeleniumIDE组件





1. 尝试使用SeleniumIDE记录和导出脚本。

记录脚本如图：



保存脚本：



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。

Excel信息的导出：

下载poi，Apache POI是Apache软件基金会的开源项目，POI提供API给Java程序对Microsoft Office格式档案读和写的功能。将poi包导入项目，利用

Workbook wb = new XSSFWorkbook(fileName)

Sheet sheet = wb.getSheetAt(0)

来读取Excel文件，通过自带的函数getRow()、 getCell().getStringCellValue()获得单元格内容，其中Excel的学号列为数 字类型，因此需要通过setCellType(CellType.STRING)将单元格类型转化为 String才能读取。

编写Junit对多组数据进行参数化测试，代码如下：

@Test

**public** **void** testBaidu() **throws** Exception {

driver.get(baseUrl + "/");

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

driver.findElement(By.*name*("id")).sendKeys(stuId);

driver.findElement(By.*name*("password")).sendKeys(stuId.substring(4));

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

*assertEquals*(stuId, driver.findElement(By.*id*("student-id")).getText());

*assertEquals*(stuName, driver.findElement(By.*id*("student-name")).getText());

*assertEquals*(gitUrl, driver.findElement(By.*id*("student-git")).getText());

}

通过@Parameterized.Parameters向测试代码传递参数：  
@Parameterized.Parameters

public static Collection<Object[]> getData(){

ArrayList<Object[]> ret = new ArrayList<>();

for(int i = 0; i < ids.size(); i++){

ArrayList<Object> temp = new ArrayList<>();

temp.add(ids.get(i));

temp.add(names.get(i));

temp.add(urls.get(i));

ret.add(temp.toArray());

}

return ret;

}

其中ids、names、urls是通过poi已导入的各位同学的数据。

具体代码如下：

@RunWith(Parameterized.class)

public class TestBaidu {

private WebDriver driver;

private String baseUrl;

private boolean acceptNextAlert = true;

private StringBuffer verificationErrors = new StringBuffer();

static List<String> ids;

static List<String> names;

static List<String> urls;

private String stuId = "";

private String stuName = "";

private String gitUrl = "";

public TestBaidu(String stuId,String stuName,String gitUrl){

this.stuId = stuId;

this.stuName = stuName;

this.gitUrl = gitUrl;

}

static {

ids = new ArrayList<>();

urls = new ArrayList<>();

names = new ArrayList<>();

Row row = null;

String stuId,stuName, gitUrl;

List<String[]> list = new ArrayList<String[]>();

try {

InputStream is = new FileInputStream("src/resources/软件测试名单.xlsx");

Workbook wb = new XSSFWorkbook(is);

Sheet sheet = wb.getSheetAt(0);

int rowNum = sheet.getPhysicalNumberOfRows();

for(int i = 2; i < rowNum; i++) {

row = sheet.getRow(i);

row.getCell(1).setCellType(CellType.STRING);

stuId = row.getCell(1).getStringCellValue();

stuName = row.getCell(2).getStringCellValue();

gitUrl = row.getCell(3).getStringCellValue();

ids.add(stuId);

names.add(stuName);

urls.add(gitUrl);

}

} catch (Exception e) {

e.printStackTrace();

}

}

@Parameterized.Parameters

public static Collection<Object[]> getData(){

ArrayList<Object[]> ret = new ArrayList<>();

for(int i = 0; i < ids.size(); i++){

ArrayList<Object> temp = new ArrayList<>();

temp.add(ids.get(i));

temp.add(names.get(i));

temp.add(urls.get(i));

ret.add(temp.toArray());

}

return ret;

}

@Before

public void setUp() throws Exception {

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

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

driver = new FirefoxDriver();

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

driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);

//driver.get(baseUrl + "/");

}

@Test

public void testBaidu() throws Exception {

driver.get(baseUrl + "/");

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

driver.findElement(By.*name*("id")).sendKeys(stuId);

driver.findElement(By.*name*("password")).sendKeys(stuId.substring(4));

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

*assertEquals*(stuId, driver.findElement(By.*id*("student-id")).getText());

*assertEquals*(stuName, driver.findElement(By.*id*("student-name")).getText());

*assertEquals*(gitUrl, driver.findElement(By.*id*("student-git")).getText());

}

@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

在对Excel文件进行提取时，利用row.getCell(1).getStringCellValue()提取学号信息并将其以String类型存储会导致报错。报错信息为：

Cannot get a STRING value from a NUMERIC cell

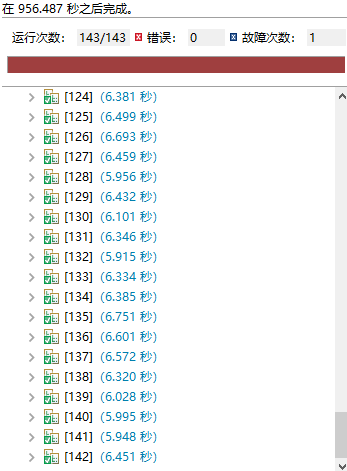
原因是对于学号信息在Excel文件中以数字类型存储，读取信息后，并不能直接将数字类型单元格直接以字符串形式存储，需要进行转换。利用的函数为row.getCell(1).setCellType(CellType.STRING)

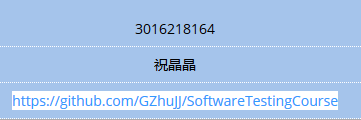
利用Selenium进行网页信息提取时，不能提取到ID框，查看网页源码，ID框使用的是name作为标识，而我用driver.findElement(By.id("id"))则不能查找到，将函数修改为driver.findElement(By.name("id"))，完成操作。

1. 测试结果（描述输入和输出）

输入：对利用poi提取的Excel中143位同学的学号、姓名、gitUrl信息与网页信息进行对比。

输出：



其中，第113号测试样本处发生故障，原因为学号为3016218164的同学的gitUrl在excel文件中与网页中信息不一致，在Excel中的git号比网页中的多了一个空格造成的，其他同学信息均保持一致未发生错误。

1. 总结

通过本次实验学习了利用Selenium进行网页代码的测试，selenium可以实现对网页的自动调用，利用selenium和Junit参数测试，可以省去大量的人工操作，自动化的进行网页测试，使用@RunWith 和 @parameter 注解来为单元测试传递参数，由此可以利用同一段代码对多组数据进行测试，避免了为大量数据测试编写大量的测试代码。同时学习了poi相关的基础操作来操作Excel文件提取文件各个单元格的信息。