天津大学

软件测试技术第一次实验报告

学　　院 智能与计算学部

专　　业 软件工程

年　　级 2016

姓　　名 张环禹

2019 年 3 月 26 日

软件测试技术第二次实验报告

一、 需求分析

1. 在 JavaIDE 中安装 Selenium
2. 在 FirFox 浏览器中安装 Selenium 插件
3. 使用 Selenium 记录并导出脚本
4. 使用 Selenium 完成 lab2 的任务，即：

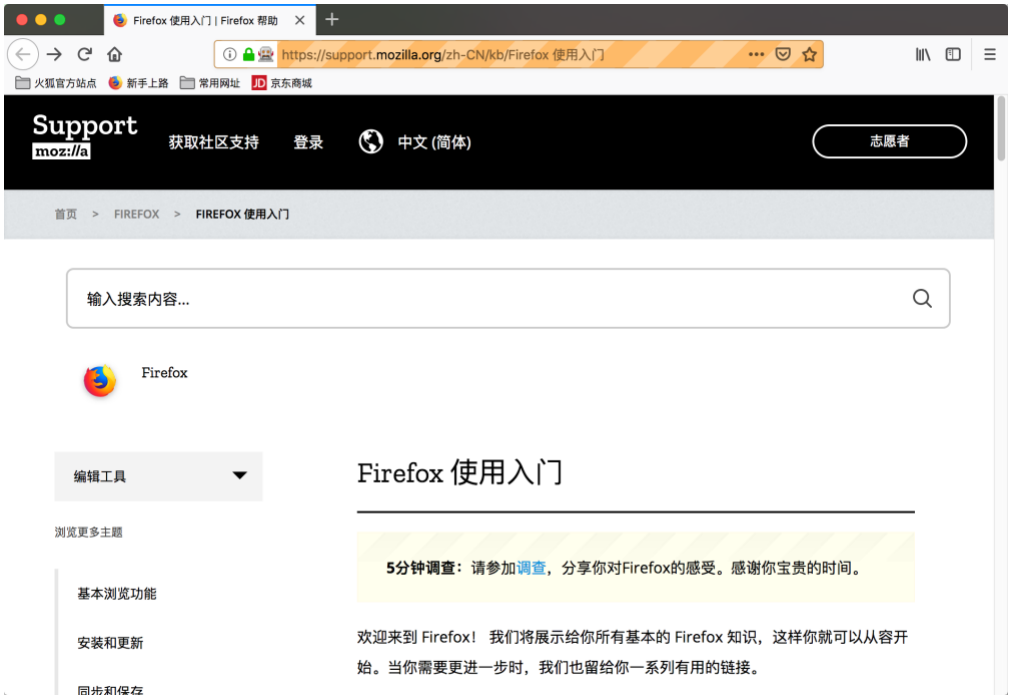
   文件软件测试名单.xlsx 包含学生信息，请在 http://121.193.130.195:8800 中查看某人信息，确保每一个学生信息和 excel 文件内部信息相同。

二、 概要设计

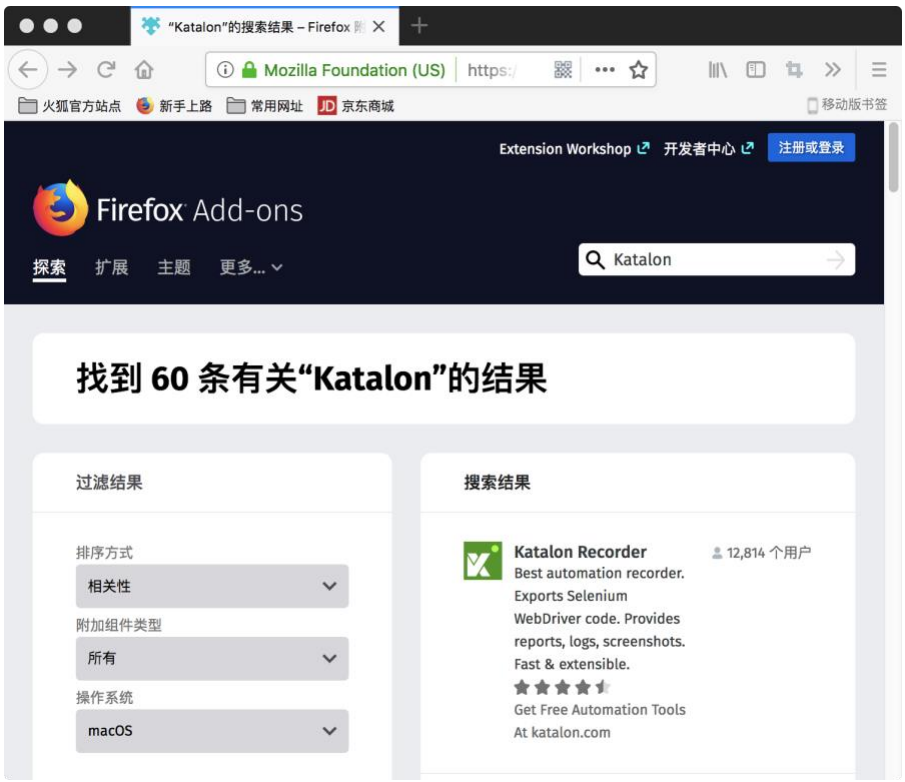
1. 由于新版 Selenium 无法导出 Java 代码文件，据资料介绍，使用了 Katalon 插件，其效果和 Selenium 一致，且能导出代码文件。
2. 在 Intellij Idea 内导入 Selenium 相关包、Excel 文件处理相关包。
3. 使用 Katalon 插件记录操作步骤并导出 Java/Junit 文件。
4. 修改导出文件，使得测试更加自动化。

三、 详细设计

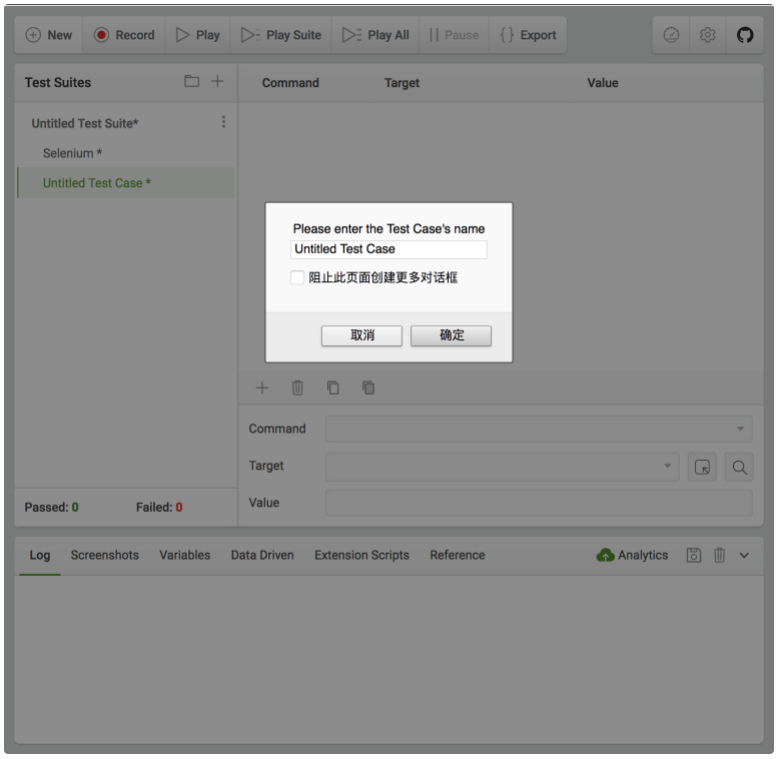
安装 Firefox 浏览器



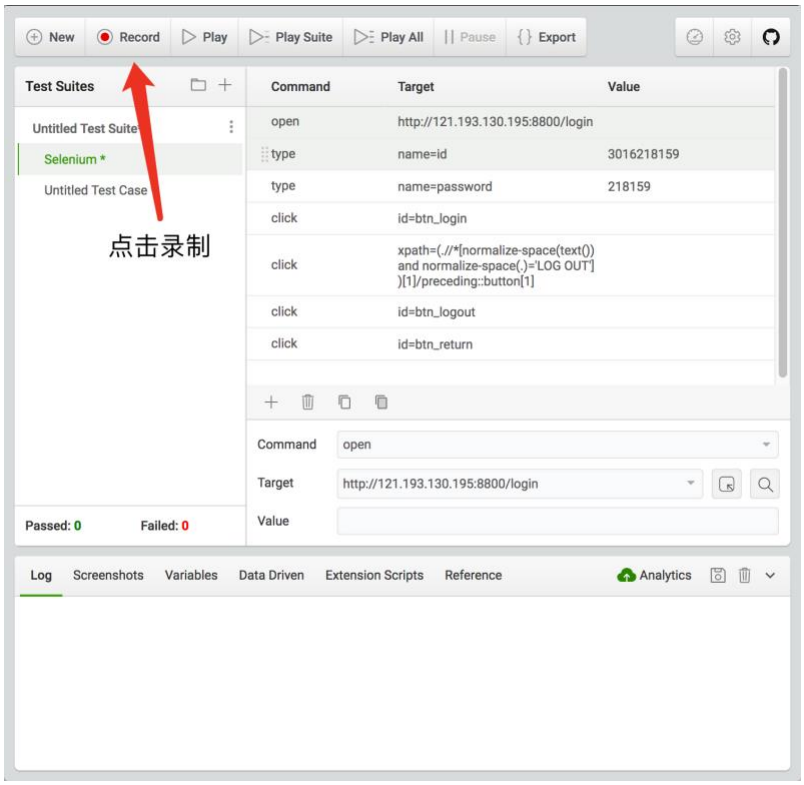
安装 Katalon IDE



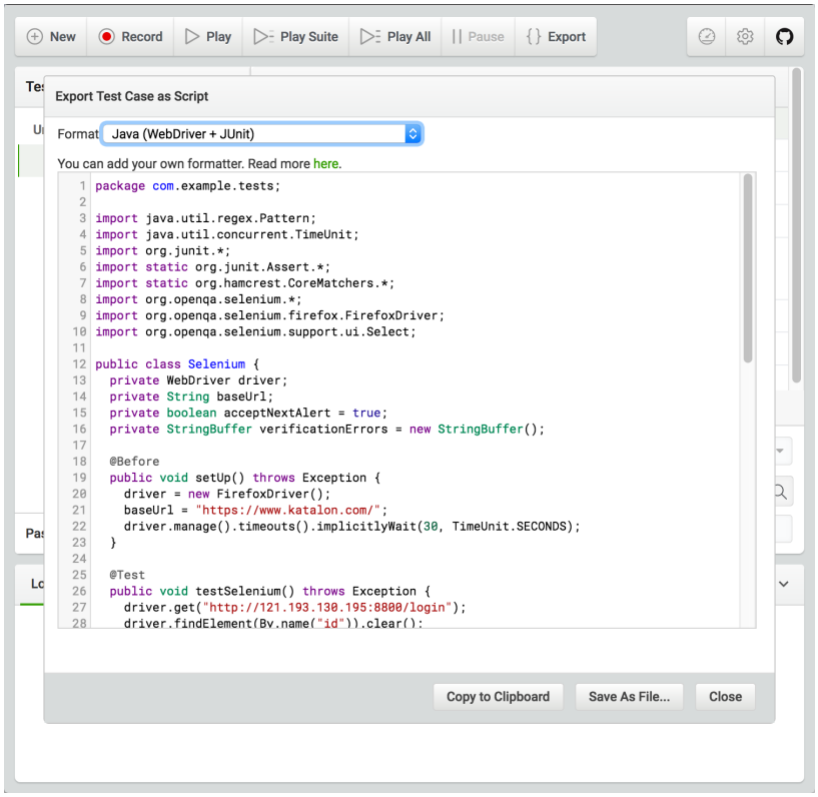
使用 Katalon 录制



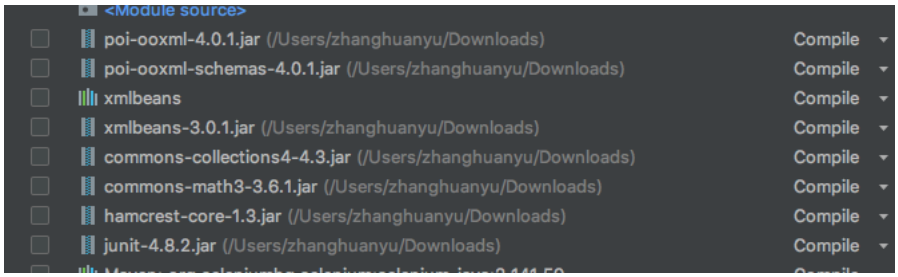
开始在想测试的网站上进行操作并录制



将录制的过程导出为 Java\Junit



导入相关 jar 包



构造 Student 类处理 Excel 表和数据

```
import org.apache.poi.xssf.usermodel.XSSFCell;

import org.apache.poi.xssf.usermodel.XSSFRow;

import org.apache.poi.xssf.usermodel.XSSFSheet;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.FileInputStream;

import java.io.InputStream;

import java.text.DecimalFormat;

import java.util.ArrayList;

import java.util.List;



public class Student {

    private int id;

    private String name;

    private String studentId;
```

```
private String git;

public int getId() {

	return this.id;

}


public String getName() {

	return this.name;

}


public String getStudentId() {

	return this.studentId;

}


public String getGit() {

	return this.git;

}


public void setId(int numericCellValue) {

	this.id=numericCellValue;

}
```

```
public void setName(String stringCellValue) {
        this.name=stringCellValue;
}

public void setStudentId(String stringCellValue) {
        this.studentId=stringCellValue;
}

public void setGit(String parse) {
        this.git=parse;
}

public static List<Student> readXls() throws Exception {
        InputStream is = new FileInputStream("software_testing_list.xlsx");

        XSSFWorkbook excel = new XSSFWorkbook(is);
        Student stu = null;
        List<Student> list = new ArrayList<Student>();
```

```
// 循环工作表 Sheet
for (int numSheet = 0; numSheet < excel.getNumberOfSheets(); numSheet++) {
    XSSFSheet sheet = excel.getSheetAt(numSheet);
    if (sheet == null)
        continue;
    // 循环行 Row
    for (int rowNum = 1; rowNum < sheet.getLastRowNum(); rowNum++) {
        XSSFRow row = sheet.getRow(rowNum);
        if (row == null)
            continue;
        stu = new Student();
        XSSFCell cell0 = row.getCell(0);
        stu.setId((int)cell0.getNumericCellValue());
        XSSFCell cell1 = row.getCell(1);
        DecimalFormat df = new DecimalFormat("0");
       stu.setStudentId(String.valueOf(df.format(cell1.getNumericCellValue())));
        XSSFCell cell2 = row.getCell(2);
        stu.setName(cell2.getStringCellValue());
        XSSFCell cell3 = row.getCell(3);
        stu.setGit(cell3.getStringCellValue().replace(" ",""));
```

```
                    list.add(stu);
                }
            }
            return list;
        }
}
```

测试类进行适当修改后得到如下代码

```
import java.util.Arrays;
import java.util.Collection;
import java.util.List;
import java.util.concurrent.TimeUnit;
import org.junit.*;
import static org.junit.Assert.*;

import org.junit.runner.RunWith;
import org.junit.runners.Parameterized;
import org.openqa.selenium.*;
import org.openqa.selenium.chrome.ChromeDriver;
```

```
@RunWith(Parameterized.class)
public class KatalonTest {
    private static WebDriver driver;
    private static String baseUrl;
    private boolean acceptNextAlert = true;
    private StringBuffer verificationErrors = new StringBuffer();
    private Student student;

    public KatalonTest(Student s)
    {
        this.student=s;
    }

    @Parameterized.Parameters
    public static Collection studentData() throws Exception {

        List<Student> k=Student.readXls();
        Object [][]res=new Object[k.size()][1];
        int count=0;
        for (Student st:k)
```

```
            {
                res[count][0]=st;
                count++;
            }
            return Arrays.asList(res);
    }


@BeforeClass
public static void setUp() throws Exception {
        System.setProperty("webdriver.chrome.driver","chromedriver");
        driver = new ChromeDriver();
        baseUrl = "https://www.katalon.com/";
        driver.get("http://121.193.130.195:8800/login");
        driver.manage().timeouts().implicitlyWait(30, TimeUnit.SECONDS);
}

public boolean useKatalon(Student student) throws Exception {
        String stuId=student.getStudentId();
        driver.findElement(By.name("id")).sendKeys(new String[]{stuId});
```

```
        driver.findElement(By.name("password")).sendKeys(new String[]{stuId.substring(stuId.length()-6)});
        driver.findElement(By.id("btn_login")).click();
        String webStudentId=driver.findElement(By.id("student-id")).getText();
        String webStudentName=driver.findElement(By.id("student-name")).getText();
        String webStudentGit=driver.findElement(By.id("student-git")).getText();
        System.out.println(student.getGit());
        driver.findElement(By.id("btn_logout")).click();
        driver.findElement(By.id("btn_return")).click();
        System.out.println(webStudentName.equals(student.getName()));
        System.out.println(webStudentId.equals(student.getStudentId()));
        System.out.println(webStudentGit.equals(student.getGit()));
        return
webStudentName.equals(student.getName())&&webStudentId.equals(student.getStudentId())&&webStudent
Git.equals(student.getGit());
    }

    @Test
    public void testKatalon() throws Exception {
        assertTrue(useKatalon(this.student));
    }
```

```
@After
public void tearDown() throws Exception {
    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;
        }
```

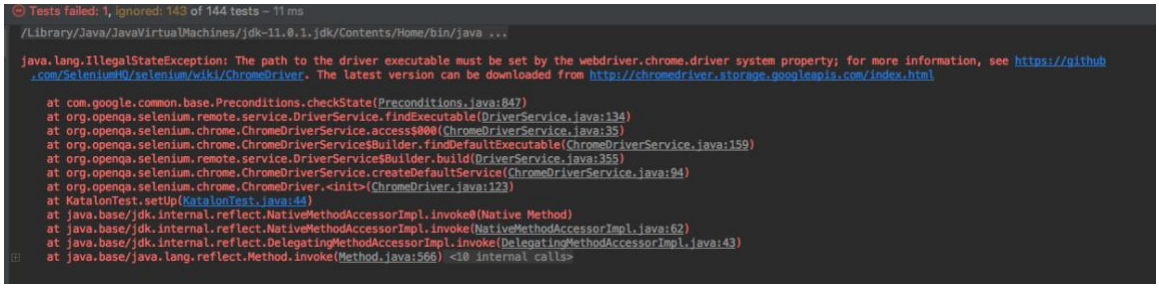
```
    }

}
```

修改代码根据和具体分析在第四部分详细介绍

四、 调试分析

导出 Katalon 脚本后运行得到如下错误



此处说明未存在 WebDriver 驱动来获得浏览器控制，我们只需将 ChromeDriver 下载好放在文件夹里使用即可。

代码修改分析：

1. 源代码里中使用 FireFox 作为浏览器进行测试，经过测试发现远慢于 Chrome 浏览器，故使用 Chrome 进行测试。
2. 使用 Student 类进行处理数据
3. 使用参数化方式进行自动化测试，即 Parameterized 注释
4. 每检查完一个学生信息后不再关闭浏览器，而是接着进行测试，大大减少耗时

五、 测试结果

本次实验测试用例为全体同学，选取用例如下：

序号	学号	姓名	git地址
1	3016218051	邓杰	https://github.com/popboykingdjko
2	3016218052	董小璇	https://github.com/xiaoxuaoj
3	3016218053	范立水	https://github.com/nbdfls
4	3016218054	黄逸群	https://github.com/hyqzz1
5	3016218055	霍欣芷	
6	3016218056	李陈	https://github.com/FBIbaby
7	3016218057	李今	https://github.com/riginAI/test
8	3016218058	李凯	https://github.com/likai3016218058
9	3016218059	李亚康	https://github.com/forAragaki/Software-Testing.git
10	3016218060	刘崇玮	https://github.com/liuchongwei1998
11	3016218061	刘岳森	https://github.com/liuyuesen
12	3016218062	楼林	https://github.com/loulin206
13	3016218063	彭昱蕃	https://github.com/acan777
14	3016218064	石善炜	https://github.com/Winbeeone/Hello-GitHub
15	3016218065	唐颂	https://github.com/tangsongTJU
16	3016218066	万子航	https://github.com/LITTLEEEEE/st.git
17	3016218067	王晨	https://github.com/WangChen0902
18	3016218068	王冠	https://github.com/skye0207
19	3016218069	王磊	https://github.com/wanglei201810/HHH.git
20	3016218070	文健玮	
21	3016218072	熊清鸥	https://github.com/Panda216
22	3016218073	许炳秋	https://github.com/AQITF/software-testing-course

```
@Parameterized.Parameters
public static Collection studentData() throws Exception {

    List<Student> k=Student.readXls();
    Object [][]res=new Object[k.size()][1];
    int count=0;
    for (Student st:k)
    {
        res[count][0]=st;
        count++;
    }
    return Arrays.asList(res);
}
```

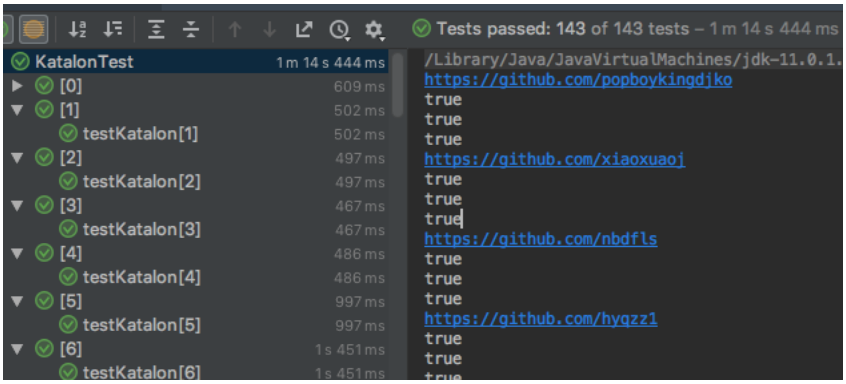
```
public static List<Student> readXls() throws Exception {
    InputStream is = new FileInputStream( name: "software_testing_list.xlsx");

    XSSFWorkbook excel = new XSSFWorkbook(is);
    Student stu = null;
    List<Student> list = new ArrayList<Student>();

    // 循环工作表Sheet
    for (int numSheet = 0; numSheet < excel.getNumberOfSheets(); numSheet++) {
        XSSFSheet sheet = excel.getSheetAt(numSheet);
        if (sheet == null)
            continue;
        // 循环行Row
        for (int rowNum = 1; rowNum < sheet.getLastRowNum(); rowNum++) {
            XSSFRow row = sheet.getRow(rowNum);
            if (row == null)
                continue;
            stu = new Student();
            XSSFCell cell0 = row.getCell( cellnum: 0);
            stu.setId((int)cell0.getNumericCellValue());
            XSSFCell cell1 = row.getCell( cellnum: 1);
            DecimalFormat df = new DecimalFormat( pattern: "0");
            stu.setStudentId(String.valueOf(df.format(cell1.getNumericCellValue())));
            XSSFCell cell2 = row.getCell( cellnum: 2);
            stu.setName(cell2.getStringCellValue());
            XSSFCell cell3 = row.getCell( cellnum: 3);
            stu.setGit(cell3.getStringCellValue().replace( target: " ", replacement: ""));
            list.add(stu);
        }
    }

    return list;
}
```

得到结果如下



六、 总结

本次试验遇到了许多问题，最主要的是版本不兼容的问题，无论是处理 Excel 的 poi 包还是 Firefox 里的 Selenium IDE ,大大影响了实验的完成效率。但好在这些问题能以其它替代品的使用而解决。

最大的收获在于对于 Selenium 的掌握 ,这不仅仅是在软件测试学科上的重要工具，也是爬虫等地方使用较多的工具，非常有技术意义。