关键字驱动框架源代码详解

行为驱动框架源代码详解

数据驱动框架源代码详解

78个最佳实战自动化测试脚本实例

基于Grid组件的分布式并发自动化测试框架源代码详解

吴晓华 编著

Selenium WebDriver 实战宝典

中国工信出版集团

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Selenium WebDriver
实战宝典

吴晓华　编著

電子工業出版社
Publishing House of Electronics Industry
北京 · BEIJING

内 容 简 介

本书是一本从入门到精通模式的 Selenium WebDriver 实战经验分享书籍。全书共分为四个部分：第一部分为基础篇，主要讲解自动化测试相关的基础理论、WebDriver 环境安装、单元测试工具的使用方法及 WebDrvier 的入门使用实例；第二部分为实战应用篇，基于丰富的实战案例讲解页面元素的定位方法及 WebDriver 的最常用 API 使用方法；第三部分为自动化测试框架搭建篇，深入讲解了页面对象的设计模式，以及分布式并发执行测试框架、数据驱动测试框架、行为驱动测试框架和关键字驱动测试框架的实例源码；第四部分为常见问题和解决方法，讲解了 WebDriver 使用过程中的常见疑难问题和解决方法。

本书既适合 WebDriver 的初学者，也适合尝试编写自动化测试框架的中、高级自动化测试工程师参考。

图书在版编目（CIP）数据

Selenium WebDriver 实战宝典 / 吴晓华编著. —北京：电子工业出版社，2015.10

ISBN 978-7-121-27118-2

Ⅰ. ①S… Ⅱ. ①吴… Ⅲ. ①软件－测试 Ⅳ.①TP311.5

中国版本图书馆 CIP 数据核字（2015）第 215095 号

责任编辑：董亚峰
特约编辑：彭　瑛　罗树利
印　　刷：北京京科印刷有限公司
装　　订：北京京科印刷有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编 100036
开　　本：787×1092　1/16　　印张：23.75　　字数：614.4 千字
版　　次：2015 年 10 月第 1 版
印　　次：2015 年 10 月第 1 次印刷
印　　数：3000 册　　定价：59.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前言

随着互联网的高速发展，中国的互联网繁荣程度达到了一个空前的发达水平，数亿量级用户的产品登上了中国的互联网发展舞台，阿里巴巴、腾讯、百度等多个互联网巨头也开始在世界的互联网舞台崭露头角，互联网行业的从业人员也达到了上百万人的规模，中国的互联网产品已经深入到网民生活的方方面面。

随着互联网行业在中国的迅猛发展，对于中国的软件开发和测试行业也提出了更高的技术要求与质量要求，软件测试从业者的技术水平也被提升到空前且不绝后的高要求阶段。以往我们看到测试人员的招聘重点都是对于测试用例设计和业务的理解，现今我们看到更多的测试职位对测试人员提出了更高的技术能力要求，例如，精通一门编程语言，熟悉 MySQL 或者 Oracle 数据库，精通自动化测试和性能测试等。为了更好地适应互联网社会的发展潮流，软件测试从业者必须在技术能力上不断地提升自己，才能真正站在职业发展的巅峰。

自动化测试技术对于测试人员来说是一个必要的高级技能要求，越来越多的测试从业者并不甘于仅仅使用手工方式进行测试，他们非常希望使用自动化的方式来减少枯燥无味且不断重复的手工测试劳动。目前，主流的 Web 自动化测试开源工具 Selenium WebDriver 成为众多软件测试从业者学习的热点，但是市面上针对 Selenium 自动化测试方面的书籍很少，基于实践方式来讲解 Selenium 应用技术的书籍更是凤毛麟角。因此，笔者个人非常希望能够写出一本基于实践操作的 Selenium 教学书籍，来解决软件测试人员的自动化测试学习问题。

本书采用图文并茂的方式分步骤讲解 Selenium 的各种实用技巧，并且提供被测试对象的实现代码或者被测试对象的访问网址，方便读者在本地搭建自己的测试环境或者访问互联网上的网站，从而进行自动化测试技术的实践。

这是我第一次写一本技术专著的书籍，当我真正开始提笔的时候，我深深感觉到用简洁的语言讲明白工具的使用方法和测试程序代码绝非易事，为此我投入了大量的精力来不断组织、优化书中的文字和图片，希望能够让读者通过本书深入掌握 Selenium 的使用技巧，助大家在自动化测试方向上的一臂之力。

由于我平时还要忙于其他工作，只能利用业余时间来编写此书，所以足足耗时 8 个月之久才完成此书，共写了三十多万字，希望能够获得读者的欢迎。非常感谢我的妻子和女儿，她们对我给予了巨大支持。我相信通过我的努力一定可以改变一些人的命运，所以和大家共勉，希望我们共同努力，改变中国测试行业技术含量低的现状。

各章内容介绍：

第一篇“基础篇”：包括第 1～8 章。

第 1 章介绍了 Selenium 的发展历史及组成 Selenium 的工具套件，列举了 Selenium 1 和 Selenium 2 支持的浏览器和平台，讲解了 Selenium RC 和 WebDriver 的实现原理，比较了 Selenium 1 和 Selenium 2 的各自特点。

第 2 章介绍了在日常测试工作中常见的自动化测试目标，讲解了如何获得公司管理层对于开展自动化测试的支持，介绍了如何衡量自动化测试工作的投入产出比及在敏捷开发中的自动化测试应用，讲解了自动化测试工作的分工及测试工具的选择和推广使用，分享了在实际项目中最佳实践经验，说明了学习 Selenium 工具的能力要求。

第 3 章主要讲解了与 Selenium 工具使用相关的辅助工具 FireBug 和 FirePath 插件，主要介绍了这两种插件的安装方法和基本使用技巧。

第 4 章主要讲解了 Selenium IDE 插件的安装、界面和基本的使用方法，并且讲解了使用实例和导出脚本的方法。

第 5 章主要讲解了如何搭建 Java 环境和 Eclipse 集成开发环境，并且介绍了 Eclipse 开发环境的最佳配置方法。

第 6 章主要讲解了 WebDriver 的安装和配置方法。

第 7 章主要讲解了单元测试的基本知识，并且结合 JUnit 和 TestNG 单元测试框架讲解了单元测试的实例。

第 8 章主要讲解了在自动化测试过程中使用的页面元素定位方法，包括 Id 定位方法、Name 定位方法、链接定位方法、Class 定位方法、XPath 定位方法、CSS 定位方法和 jQuery 定位方法，推荐使用 XPath 定位方法作为页面元素定位的主要方法。

第二篇“实战应用篇”：包括第 9～11 章。

第 9 章讲解了如何使用 WebDriver 工具分别使用 IE 浏览器、Firefox 浏览器和 Chrome 浏览器进行自动化测试，介绍了 TestNG 工具的并发兼容性测试实例。

第 10 章通过实例全面讲解了 WebDriver 的常用 API，共介绍了 41 个实例。

第 11 章讲解了 WebDriver 的 20 个高级应用实例。

第三篇“自动化测试框架搭建篇”：包括第 12～16 章。

第 12 章讲解了数据驱动的概念，以及如何基于 TestNG 工具使用 CSV 文件、Excel 文件和 MySQL 数据库分别进行数据驱动测试。

第 13 章介绍了页面对象（Page Object）设计模式，通过使用类函数封装方式实现自动化测试框架的设计模式。

第 14 章介绍了基于 Cucumber 工具的行为驱动测试实例，分别基于中文测试用例文件和英文测试用例文件进行了实例讲解。

第 15 章介绍了如何基于 Selenium Gird 组件实现并发执行测试用例，并通过实例进行了深入讲解。

第 16 章深入讲解了如何从零搭建一个数据驱动测试框架和关键字驱动测试框架，提供了完整的框架实例代码。此章节为本书的最重要章节，建议读者在阅读前面所有章节后再阅读此章节。

第四篇“常见问题和解决方法”：包含第 17 章。

第 17 章讲解了 WebDriver 使用过程中的常见疑难问题和解决方法，读者可以在使用 WebDrvier 遇到问题时进行查阅。

特别致谢：

感谢我的好朋友李雄刚和部凯文帮我校对本书中的所有实例和代码，从中发现了不少书写错误和代码问题，在此对他们表示真挚的感谢。

吴晓华

2015 年 8 月

目录

第一篇　基础篇

第 1 章　Selenium 简介 1

1.1　Selenium 的前世今生 1

1.2　Selenium 工具套件介绍 2

1.3　Selenium 1 和 Selenium 2 支持的浏览器和平台 2

1.3.1　Selenium IDE、Selenium 1 和 Selenium RC 支持的浏览器和平台 2

1.3.2　Selenium 2（WebDriver）支持的浏览器 3

1.4　Selenium RC 和 WebDriver 的实现原理 4

1.4.1　Selenium RC 的实现原理 4

1.4.2　WebDriver 的实现原理 6

1.4.3　Selenium 1 和 WebDriver 的特点 6

第 2 章　自动化测试的那点事儿 7

2.1　自动化测试目标 7

2.2　管理层的支持 10

2.3　投入产出比 10

2.4　敏捷开发中的自动化测试应用 11

2.5　自动化测试人员分工 13

2.6　自动化测试工具的选择和推广使用 13

2.6.1　自动化测试工具的选择 13

2.6.2　Selenium WebDriver 和 QTP 的工具特点比较 14

2.7　在项目中实施自动化的最佳实践 15

2.8　学习 Selenium 工具的能力要求 17

第 3 章　自动化测试辅助工具 18

3.1　Firefox 浏览器的安装 18

3.2　安装 Firebug 插件 18

3.3　Firebug 插件的使用 19

3.3.1　启动 Firebug 插件 20

3.3.2　Firebug 插件的常用功能 20

3.4　安装 FirePath 插件 22

3.5 FirePath 插件的使用 23
3.5.1 FirePath 插件中使用 XPath 定位方式 23
3.5.2 FirePath 插件中使用 CSS 定位方式 25
3.6 IE 浏览器自带的辅助开发工具 27
第 4 章 Selenium IDE 29
4.1 什么是 Selenium IDE 29
4.2 安装 Selenium IDE 29
4.2.1 从 Selenium 官网安装 29
4.2.2 使用下载的 XPI 安装文件安装 31
4.3 Selenium IDE 插件界面和功能介绍 31
4.3.1 主界面 31
4.3.2 常用工具栏 32
4.3.3 脚本编辑区域 32
4.4 常用菜单项 32
4.4.1 “文件”菜单 32
4.4.2 “编辑”菜单 33
4.4.3 Actions 菜单 33
4.4.4 Options 菜单 34
4.5 录制和回放的脚本实例 35
4.6 Selenium IDE 脚本介绍——Selenese 37
4.7 Selenium IDE 的基本命令使用实例 38
4.7.1 waitForText、verifyText 和 assertText 命令 38
4.7.2 storeTitle 和 echo 命令 41
4.7.3 openWindow 和 selectWindow 命令 41
4.8 从 Selenium IDE 导出脚本 42
4.8.1 导出脚本文件 42
4.8.2 将 Selenium IDE 插件中的某行脚本导出为 Java 脚本 44
第 5 章 搭建 Java 环境和 Eclipse 集成开发环境 45
5.1 安装 Java JDK，配置 Java 环境 45
5.1.1 下载 JDK 1.6 版本安装文件 45
5.1.2 安装 JDK 1.6 版本 47
5.1.3 配置 Java 环境变量 48
5.2 安装 Java IDE 开发工具 Eclipse 51
5.3 新建一个 Java 工程和一个类 53
5.4 Eclipse 集成开发环境的使用技巧 56

5.4.1 将程序代码和注释字体变大 56
5.4.2 自动补全功能 57
第 6 章 WebDriver 的安装配置 58
6.1 在 Eclipse 中配置 WebDriver 58
6.2 第一个 WebDriver 脚本 61
第 7 章 单元测试框架的基本使用介绍 63
7.1 什么是单元测试 63
7.2 JUnit 单元测试框架 63
7.2.1 什么是 JUnit 63
7.2.2 安装 JUnit 4 63
7.2.3 JUnit 的常见注解 65
7.2.4 创建 JUnit 4 Test Suite 70
7.2.5 使用 JUnit 编写的 WebDriver 脚本 72
7.3 TestNG 单元测试框架 73
7.3.1 什么是 TestNG 73
7.3.2 TestNG 的优点 74
7.3.3 编写 TestNG 测试用例的步骤 74
7.3.4 安装 TestNG 74
7.3.5 在 TestNG 中运行第一个 WebDriver 测试用例 77
7.3.6 TestNG 的常用注解 81
7.3.7 测试集合 85
7.3.8 测试用例的分组 87
7.3.9 依赖测试 90
7.3.10 特定顺序执行测试用例 91
7.3.11 跳过某个测试方法 92
7.3.12 测试报告中的自定义日志 93
7.3.13 断言 94
第 8 章 页面元素的定位方法 97
8.1 定位页面元素方法的汇总 97
8.2 使用 ID 定位 97
8.3 使用 name 定位 98
8.4 使用链接的全部文字定位 99
8.5 使用部分链接文字定位 99
8.6 使用标签名称定位 100
8.7 使用 Class 名称定位 101

8.8 使用 XPath 定位 101
8.8.1 什么是 XPath 101
8.8.2 XPath 语法 102
8.9 CSS 定位 107
8.9.1 什么是 CSS 107
8.9.2 CSS 语法 107
8.9.3 XPath 定位和 CSS 定位的比较 111
8.10 jQuery 定位 112
8.10.1 什么是 jQuery 112
8.10.2 jQuery 的定位代码实例 112
8.11 表格的定位方法 114
8.11.1 遍历表格的全部单元格 114
8.11.2 定位表格中的某个单元格 116
8.11.3 定位表格中的子元素 117

第二篇 实战应用篇

第 9 章 WebDriver 的多浏览器测试 119
9.1 使用 IE 浏览器进行测试 119
9.2 使用 Firefox 浏览器进行测试 120
9.3 使用 Chrome 浏览器进行测试 121
9.4 使用 Mac 系统中的 Safari 浏览器进行测试 122
9.5 使用 TestNG 进行并发兼容性测试 123
第 10 章 WebDriver API 实例详解 127
10.1 访问某网页地址 127
10.2 返回上一个访问的网页（模拟单击浏览器的后退功能） 127
10.3 从上次访问网页前进到下一个网页（模拟单击浏览器的前进功能） 128
10.4 刷新当前网页 128
10.5 操作浏览器窗口 128
10.6 获取页面的 Title 属性 129
10.7 获取页面的源代码 129
10.8 获取当前页面的 URL 地址 130
10.9 在输入框中清除原有的文字内容 130
10.10 在输入框中输入指定内容 131
10.11 单击按钮 131
10.12 双击某个元素 132
10.13 操作单选下拉列表 132

10.14 检查单选列表的选项文字是否符合期望 133
10.15 操作多选的选择列表 134
10.16 操作单选框 135
10.17 操作复选框 136
10.18 杀掉 Windows 的浏览器进程 137
10.19 将当前浏览器的窗口截屏 137
10.20 检查页面元素的文本内容是否出现 138
10.21 执行 JavaScript 脚本 138
10.22 拖曳页面元素 139
10.23 模拟键盘的操作 139
10.24 模拟鼠标右键事件 140
10.25 在指定元素上方进行鼠标悬浮 140
10.26 在指定元素上进行鼠标单击左键和释放的操作 142
10.27 查看页面元素的属性 143
10.28 获取页面元素的 CSS 属性值 143
10.29 隐式等待 144
10.30 常用的显式等待 145
10.31 自定义的显式等待 146
10.32 判断页面元素是否存在 148
10.33 使用 Title 属性识别和操作新弹出的浏览器窗口 149
10.34 使用页面的文字内容识别和处理新弹出的浏览器窗口 150
10.35 操作 JavaScript 的 Alert 弹窗 151
10.36 操作 JavaScript 的 confirm 弹窗 152
10.37 操作 JavaScript 的 prompt 弹窗 153
10.38 操作 Frame 中的页面元素 154
10.39 使用 Frame 中的 HTML 源码内容来操作 Frame 156
10.40 操作 IFrame 中的页面元素 157
10.41 操作浏览器的 Cookie 159

第 11 章 WebDriver 的高级应用实例 160

11.1 使用 JavaScriptExecutor 单击元素 160
11.2 在 Ajax 方式产生的浮动框中，单击选择包含某个关键字的选项 161
11.3 设置一个页面对象的属性值 163
11.4 在日期选择器上进行日期选择 165
11.5 如何能够无人工接入地自动化下载某个文件 166
11.6 使用 sendKeys 方法上传一个文件附件 169
11.7 使用第三方工具 AutoIt 上传文件 170

11.8 操作 Web 页面的滚动条 175
11.9 启动带有用户配置信息的 Firefox 浏览器窗口 177
11.10 Robot 对象操作键盘 179
11.11 对象库（UI Map） 182
11.12 操作富文本框 185
11.13 精确比较网页截图图片 189
11.14 高亮显示正在被操作的页面元素 191
11.15 在测试中断言失败的步骤进行屏幕截图 193
11.16 使用 Log4j 在测试过程中打印执行日志 198
11.17 封装操作表格的公用类 202
11.18 控制 HTML5 语言实现的视频播放器 205
11.19 在 HTML5 的画布元素上进行绘画操作 207
11.20 操作 HTML5 的存储对象 209

第三篇 自动化测试框架搭建篇

第 12 章 数据驱动测试 211
12.1 什么是数据驱动 211
12.2 使用 TestNG 进行数据驱动 211
12.3 使用 TestNG 和 CSV 文件进行数据驱动 214
12.4 使用 TestNG、Apache POI 和 Excel 文件进行数据驱动测试 216
12.5 使用 MySQL 数据库实现数据驱动测试 219
第 13 章 页面对象（Page Object）模式 224
13.1 页面对象模式简介 224
13.2 使用 PageFactory 类 225
13.2.1 使用 PageFactory 类给测试类提供待操作的页面元素 225
13.2.2 使用 PageFactory 类封装页面元素的操作方法 226
13.3 使用 LoadableComponent 类 228
13.4 多个 PageObject 的自动化测试实例 230
第 14 章 行为驱动测试 237
14.1 行为驱动开发和 Cucumber 简介 237
14.2 Cucumber 在 Eclipse 中的环境搭建 238
14.3 在 Eclipse 中使用 JUnit 和英文语言进行行为驱动测试 240
14.4 在 Eclipse 中使用 JUnit 和中文语言进行行为驱动测试 247

第 15 章 Selenium Grid 的使用 251
15.1 Selenium Grid 简介 251
15.2 Selenium Grid 的使用方法 252
15.2.1 远程使用 FireFox 浏览器进行自动化测试 252
15.2.2 远程使用 IE 浏览器自动化测试 256
15.3 通过 TestNG 使用 Firefox、IE 和 Chrome 浏览器进行并发的远程自动化测试 258
15.3.1 使用静态类实现并发的远程自动化测试 259
15.3.2 通过 TestNG 的配置文件参数方法进行远程并发自动化测试 262
15.4 使用 Selenium Grid 时，在远程节点计算机进行 屏幕截屏 267
第 16 章 自动化测试框架的 Step By Step 搭建及测试实战 270
16.1 什么是自动化测试框架 270
16.2 数据驱动框架及实战 272
16.3 关键字框架搭建及实战 303

第四篇 常见问题和解决方法

第 17 章 自动化测试常见问题和解决方法 365
17.1 如何让 WebDriver 支持 IE 11 365
17.2 “Unexpected error launching Internet Explorer. Browser zoom level was set to 75%（或其他百分比）”的错误如何解决 366
17.3 如何消除 Chrome 浏览中的--ignore-certificate-errors 提示 367
17.4 为什么在某些 IE 浏览器中输入数字和英文特别慢 367
17.5 常见异常和解决方法 367

第一篇　基础篇

第1章　Selenium 简介

Selenium 工具诞生的时间已经超过了 10 年，目前已经在软件开发公司中得到大规模的应用，但是很少有人能够清晰地描述此工具的发展历史和特点，通过本章的介绍让读者和 Selenium 工具来一次亲密接触，了解它的前世今生及特点。

1.1 Selenium 的前世今生

2004 年，在 ThoughtWorks 公司，一个名为 Jason Huggins 的测试同行为了减少手工测试的工作量，自己实现了一套基于 JavaScript 语言的代码库，使用这套库可以进行页面的交互操作，并且可以重复地在不同浏览器上进行各种测试操作。通过不断地改进和优化，这个代码库逐步发展成 Selenium Core。Selenium Core 为 Selenium Remote Control（RC）和 Selenium IDE 提供了坚实的核心基础能力。

当时的自动化测试工具比较稀少，现有的工具也不够灵活地支持各种复杂的测试操作，大部分测试人员只能使用手工的方式完成 Web 产品的测试工作。开发人员不断地开发代码，测试人员不断地发现 bug，开发人员不断地修改 bug，测试人员不断地回归测试来确认 bug 已经被修正，并且确认程序没有引入新的 bug。这样的产品开发模式，导致测试人员必须经常性地手工回归测试系统的大部分功能，由此产生了大量的重复性手工操作。Jason Huggins 想改变这样的现状，所以他开发了基于 JavaScript 语言的代码库，希望帮助测试人员从日常的重复性工作中解脱出来。经过不懈地努力，Selenium 1 版本诞生了。

Web 自动化测试工具 Selenium 是划时代的，因为它允许测试工程师使用多种开发语言来控制不同类型的浏览器，从而实现不同的测试目标。Selenium 是开源工具软件，用户无须付费就可以使用它，甚至可以根据自己的使用需求来进行深入的定制化，改写其原有的一些功能。基于以上说明的优点，越来越多的测试人员开始使用此工具来进行 Web 系统的自动化测试工作。在短短几年时间内，全世界范围内都出现了 Selenium 工具的忠实拥护者，目前中国的几大互联网公司均使用 Selenium 作为 Web 自动化测试实施的主要工具。

但是随着互联网技术的不断发展及浏览器对于 JavaScript 语言的安全限制，Selenium 的发展也遇到了瓶颈。由于其自身实现的机制，Selenium 无法突破浏览器沙盒的限制，导致很多测试场景的测试需求难以被实现。

2006 年，Google 的工程师 Simon Stewart 开启了一个叫作 WebDriver 的项目，此项目

可以直接让测试工具调用浏览器和操作系统本身提供的内置方法，以此来绕过 JavaScript 环境的沙盒限制，WebDriver 项目的目标就是为了解决 Selenium 的痛处。2008 年，Selenium 和 WebDriver 这两个项目进行了合并，Selenium 2 出现了，也就是我们现在常常看到的 Selenium WebDriver（简称 WebDriver）。

Selenium 2 =Selenium 1 + WebDriver

Selenium 的官网地址是 www.seleniumhq.org，网站提供了 Selenium WebDriver 的安装文件和使用教程。Selenium 2 是 Selenium 1 的升级版本，它本身向下兼容 Selenium 1 的所有功能，同时又提供了更多的新 API 来完成自动化测试的各种复杂需求。现阶段，Selenium 1 已经退出了历史舞台，大部分 Web 自动化测试人员已经完全转向使用 Selenium 2（WebDriver）来搭建自己的自动化测试框架。为了迎合历史的发展潮流，本书全部的案例均基于 Selenium 2 的 WebDriver API 进行讲解。

1.2 Selenium 工具套件介绍

- Selenium 2（Selenium WebDriver）：提供了极佳的特性，例如，面向对象 API，提供 Selenium 1 的接口用于向下兼容。
- Selenium 1（Selenium RC 或 Remote Control（RC））：支持更多的浏览器，支持更多的编程语言（如 Java、JavaScript、Ruby、PHP、Python、Perl 和 C#）。
- Selenium IDE（集成开发环境）：Firefox 插件，提供图形界面来录制和回放脚本。此插件只是用来做原型的工具，并不希望测试工程师使用此工具来运行大批量的测试脚本。此插件需要使用第三方的 JavaScript 代码库才能支持循环和条件判断。
- Selenium -Grid 可以在多个测试环境以并发的方式执行测试脚本，实现测试脚本的并发执行，缩短大量测试脚本的执行时间。

1.3 Selenium 1 和 Selenium 2 支持的浏览器和平台

Selenium 1 的一大特点就是能够在多种操作系统上支持多种浏览器的自动化测试，通过下面的章节我们可以了解此工具能够支持的平台列表和浏览器类型列表。

1.3.1 Selenium IDE、Selenium 1 和 Selenium RC 支持的浏览器和平台

Selenium IDE、Selenium 1 和 Selenium RC 支持的浏览器和平台如表 1-1 所示。

表 1-1

浏览器	Selenium IDE	Selenium 1 （即 Selenium RC）	操作系统
Firefox 3.x	录制脚本和回放脚本	启动浏览器 运行测试脚本	Windows、Linux、Mac

续表

浏览器	Selenium IDE	Selenium 1 （即 Selenium RC）	操作系统
Firefox 3	录制脚本和回放脚本	启动浏览器 运行测试脚本	Windows、Linux、Mac
Firefox 2	录制脚本和回放脚本	启动浏览器 运行测试脚本	Windows、Linux、Mac
IE 8	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows
IE 7	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows
IE 6	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows
Safari 4	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows、Mac
Safari 3	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows、Mac
Safari 2	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows、Mac
Opera 10	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows、Linux、Mac
Opera 9	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows、Linux、Mac
Opera 8	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows、Linux、Mac
Google Chrome	仅能通过 Selenium 1（RC）来运行测试脚本	启动浏览器 运行测试脚本	Windows、Linux、Mac

1.3.2　Selenium 2（WebDriver）支持的浏览器

官网中并没有明确列出 WebDriver 支持浏览器的所有版本号，仅仅列出浏览器的名称。下面结合笔者个人的实际使用情况列出 WebDriver 支持的浏览器版本，请在测试实践中进行再次确认。

- Google Chrome。
- IE 6、7、8、9、10 和 11。
- Mac 操作系统的 Safari 默认版本均支持。
- Firefox 的大部分版本。
- Opera。
- HtmlUnit。
- Android 手机操作系统的默认浏览器。
- iOS 手机操作系统的默认浏览器。

1.4 Selenium RC 和 WebDriver 的实现原理

当执行 Selenium 自动化测试脚本时，测试人员可以看到浏览器中发生的神奇一幕：页面上会自动进行各种操作，例如，打开新窗口、在输入框中输入文字、选择下拉列表框等。测试人员为此不禁要多问一句：“它到底是怎么实现的？”为了了解 Selenium 工具的神奇之处，读者必须深入了解它的实现原理和机制。

1.4.1 Selenium RC 的实现原理

Selenium RC 的实现原理如图 1-1 所示。

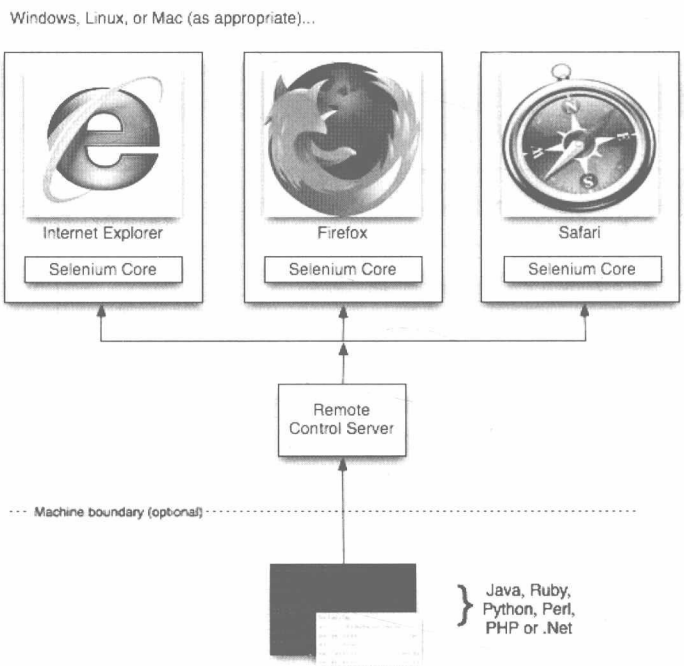

图 1-1

Selenium 1 的自动化测试执行步骤如下。

第一步：测试人员基于 Selenium 支持的编程语言编写好测试脚本程序。

第二步：测试人员执行测试程序。

第三步：测试脚本程序发送访问网站的 Http 请求给 Remote Control Sever（RC）。

第四步：Remote Control Sever（RC）收到请求后，访问被测试网站并获取网页数据内容，并在网页中插入 Selenium Core 的 JavaScript 代码库，然后返回给测试人员执行测试的浏览器。

第五步：测试脚本在浏览器内部再调用 Selenium Core 来执行测试代码逻辑，最后记录测试的结果，完成测试。