

12 | 从0到1：你的第一个GUI自动化测试

2018-07-25 茹炳晟

软件测试52讲

[进入课程 >](#)



讲述：茹炳晟

时长 11:00 大小 5.04M



在前面的测试基础知识系列文章中，我分享了测试相关的基础知识，从测试用例的设计，到测试覆盖率，再到测试计划的制定，这些都是我认为测试人要掌握的一些基本知识。

那么，接下来我将要带你走入 GUI 自动化测试的世界，和你一起聊聊 GUI 自动化测试的技术、原理和行业最佳实践。

作为该系列的第一篇文章，我直接以一个最简单的 GUI 自动化用例的开发为例，带你从 0 开始构建一个 Selenium 的 GUI 自动化测试用例。

先让你对 GUI 自动化测试有一个感性认识，然后以此为基础，我再来解释 Selenium 自动化测试实现的核心原理与机制，希望可以帮你由点到面建立起 GUI 测试的基础知识体系。

构建一个 Selenium 自动化测试用例示例

测试需求非常简单：访问百度主页，搜索某个关键词，并验证搜索结果页面的标题是“被搜索的关键词” + “_ 百度搜索”。

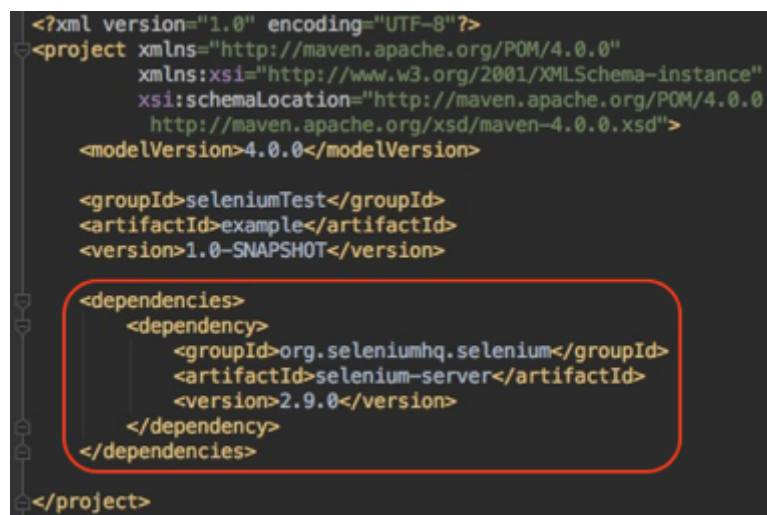
如果搜索的关键词是“极客时间”，那么搜索结果页面的标题就应该是“极客时间 _ 百度搜索”。

明白了测试需求后，我强烈建议你先用手工方式执行一遍测试，具体步骤是：

1. 打开 Chrome 浏览器，输入百度的网址 “www.baidu.com”；
2. 在搜索输入框中输入关键词“极客时间”并按下回车键；
3. 验证搜索结果页面的标题是否是“极客时间 _ 百度搜索”。

明确了 GUI 测试的具体步骤后，我们就可以用 Java 代码，基于 Selenium 实现这个测试用例了。

这里，我要用到 Chrome 浏览器，所以需要先下载 Chrome Driver 并将其放入环境变量。接下来，你可以用自己熟悉的方式建立一个空的 Maven 项目，然后在 POM 文件中加入 Selenium 2.0 的依赖，如图 1 所示。



```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>seleniumTest</groupId>
  <artifactId>example</artifactId>
  <version>1.0-SNAPSHOT</version>

  <dependencies>
    <dependency>
      <groupId>org.seleniumhq.selenium</groupId>
      <artifactId>selenium-server</artifactId>
      <version>2.9.0</version>
    </dependency>
  </dependencies>

</project>
```

图 1 在 POM 文件中加入 Selenium 2.0 的依赖

接着用 Java 创建一个 main 方法，并把如图 2 所示的代码复制到你的 main 方法中。

```

1  import org.junit.Assert;
2  import org.openqa.selenium.By;
3  import org.openqa.selenium.WebDriver;
4  import org.openqa.selenium.WebElement;
5  import org.openqa.selenium.chrome.ChromeDriver;
6
7  public class seleniumBaiduExample{
8      public static void main(String[] args) throws InterruptedException{
9
10         //创建Chrome driver的实例
11         WebDriver driver = new ChromeDriver();
12
13         //打开百度首页"www.baidu.com"
14         driver.navigate().to("http://www.baidu.com");
15         //driver.get("http://www.baidu.com");
16
17         //通过name属性找到搜索输入框
18         WebElement search_input = driver.findElement(By.name("wd"));
19
20         //在搜索输入框中输入搜索关键字"极客时间"
21         search_input.sendKeys("极客时间");
22
23         //递交搜索请求
24         search_input.submit();
25
26         //等待固定时间3秒
27         Thread.sleep(3000);
28
29         //验证搜索结果页面的标题
30         Assert.assertEquals("极客时间_百度搜索", driver.getTitle());
31
32         //关闭浏览器窗口
33         driver.quit();
34     }
35 }

```

图 2 基于 Selenium 的自动化测试用例的样本代码

现在，你可以尝试运行这个 main 方法，看看会执行哪些操作。

1. 这段代码会自动在你的电脑上打开 Chrome 浏览器；
2. 在 URL 栏自动输入 "www.baidu.com" ；
3. 百度主页打开后，在输入框自动输入“极客时间”并执行搜索；
4. 返回搜索结果页面；
5. Chrome 浏览器自动退出。

以上这些步骤都是由自动化测试代码自动完成的。

如果你已经接触过 GUI 自动化测试，你可能习以为常了，感觉没什么神奇的。但如果你是第一次接触 GUI 自动化测试，是不是觉得还蛮有意思的。

现在，我来快速解读一下这些代码，你可以看看这些自动化步骤是怎么实现的，更具体的原理和内部机制我会在后面文章中详细展开。

第 11 行，`WebDriver driver = new ChromeDriver()`，先创建一个 Chrome Driver 的实例，也就是打开了 Chrome 浏览器，但实际上没这么简单，后台还做了些额外的 Web Service 绑定工作，具体后面会解释；

第 14 行，`driver.navigate().to(s: "http://www.baidu.com")` 用刚才已经打开的 Chrome 浏览器访问百度主页；

第 18 行，`WebElement search_input = driver.findElement(By.name("wd"))`，使用 driver 的 `findElement` 方法，并通过 `name` 属性定位到了搜索输入框，并将该搜索输入框命名为 `search_input`；

第 21 行，`search_input.sendKeys(...charSequences: "极客时间")`，通过 `WebElement` 的 `sendKeys` 方法向搜索输入框 `search_input` 输入了字符串“极客时间”；

第 24 行，`search_input.submit()`，递交了搜索请求；

第 27 行，`Thread.sleep(millis:3000)`，强行等待了固定的 3 秒时间；

第 30 行，`Assert.assertEquals(expected: "极客时间_百度搜索", driver.getTitle())`，通过 junit 的 `assertEquals` 比较了浏览器的标题与预计结果，其中页面标题通过 driver 的 `getTitle` 方法得到，如果标题与预计结果一致，测试通过，否则测试失败；

第 33 行，`driver.quit()`，显式关闭了 Chrome 浏览器。

现在，你对 `main` 方法中的代码，已经比较清楚了。但是，你知道 Selenium 内部是如何实现 Web 自动化操作的吗？这就要从 Selenium 的历史版本和基本原理开始讲起了。

Selenium 的实现原理

首先，你要明确刚才建立的测试用例是基于 Selenium 2.0，也就是 Selenium + WebDriver 的方案。

其次，你需要知道，对 Selenium 而言，V1.0 和 V2.0 版本的技术方案是截然不同的，V1.0 的核心是 Selenium RC，而 V2.0 的核心是 WebDriver，可以说这完全是两个东西。

最后，Selenium 3.0 也已经发布一段时间了，V3.0 相比 V2.0 并没有本质上的变化，主要是增加了对 MacOS 的 Safari 和 Windows 的 Edge 的支持，并彻底删除了对 Selenium RC 的支持。

所以接下来，我会针对 V1.0 和 V2.0 来解释 Selenium 实现 Web 自动化的原理。

第一，Selenium 1.0 的工作原理

Selenium 1.0，又称 Selenium RC，其中 RC 是 Remote Control 的缩写。Selenium RC 利用的原理是：JavaScript 代码可以很方便地获取页面上的任何元素并执行各种操作。

但是因为“同源政策（Same-origin policy）”（只有来自相同域名、端口和协议的 JavaScript 代码才能被浏览器执行），所以要想在测试用例运行中的浏览器中，注入 JavaScript 代码从而实现自动化的 Web 操作，Selenium RC 就必须“欺骗”被测站点，让它误以为被注入的代码是同源的。

那如何实现“欺骗”呢？这其实就是引入 Selenium RC Server 的根本原因，其中的 Http Proxy 模块就是用来“欺骗”浏览器的。

除了 Selenium RC Server，Selenium RC 方案的另一大部分就是，Client Libraries。它们的具体关系如图 3 所示。

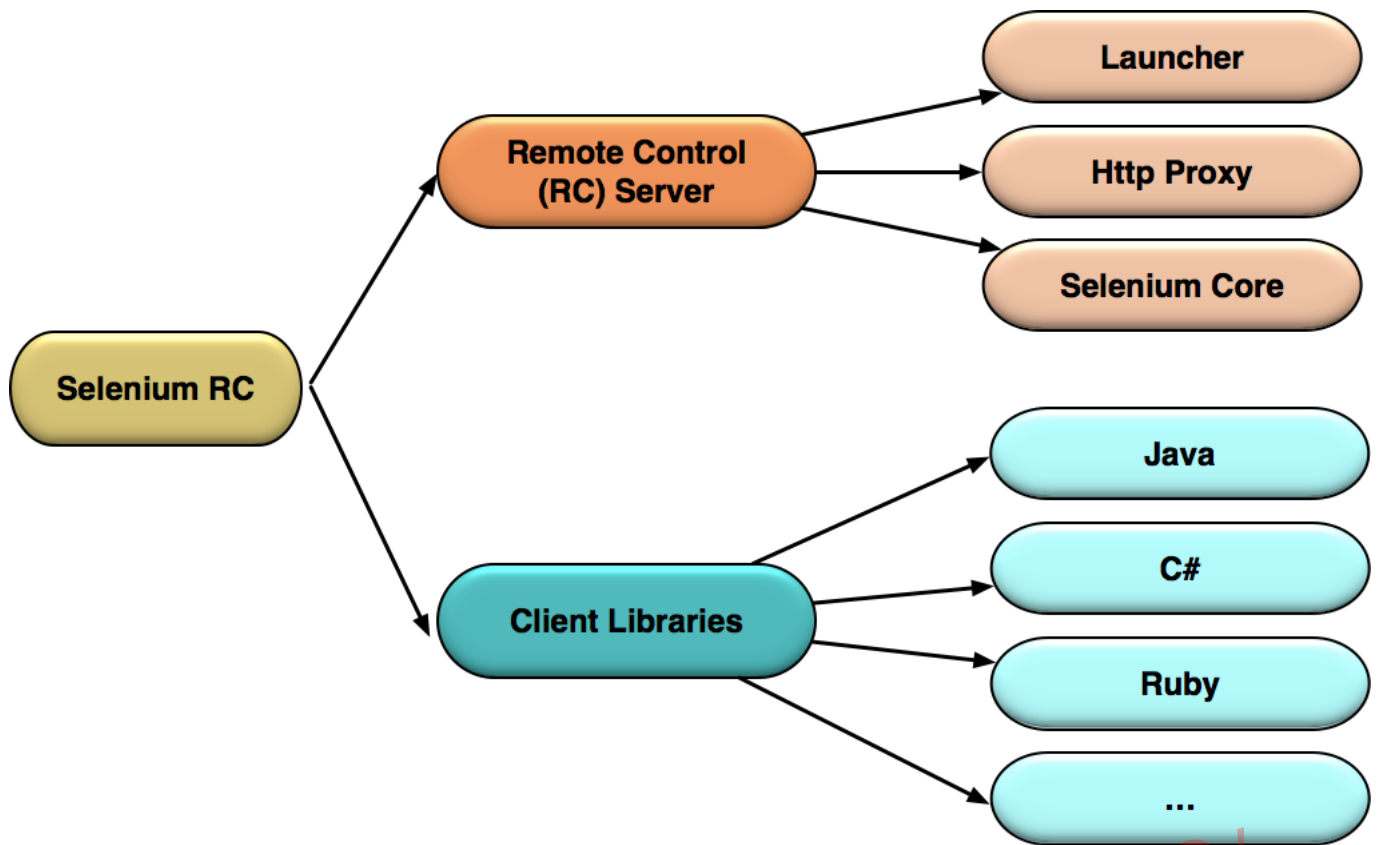


图 3 Selenium RC 的基本模块

Selenium RC Server，主要包括 **Selenium Core**，**Http Proxy** 和 **Launcher** 三部分：

Selenium Core，是被注入到浏览器页面中的 JavaScript 函数集合，用来实现界面元素的识别和操作；

Http Proxy，作为代理服务器修改 JavaScript 的源，以达到“欺骗”被测站点的目的；

Launcher，用来在启动测试浏览器时完成 **Selenium Core** 的注入和浏览器代理的设置。

Client Libraries，是测试用例代码向 **Selenium RC Server** 发送 **Http** 请求的接口，支持多种语言，包括 **Java**、**C#** 和 **Ruby** 等。

为了帮你更好地理解 Selenium RC 的基本原理，我从 Selenium 的官方网站截取了以下执行流程图，并把具体的 7 个步骤做了如下翻译。

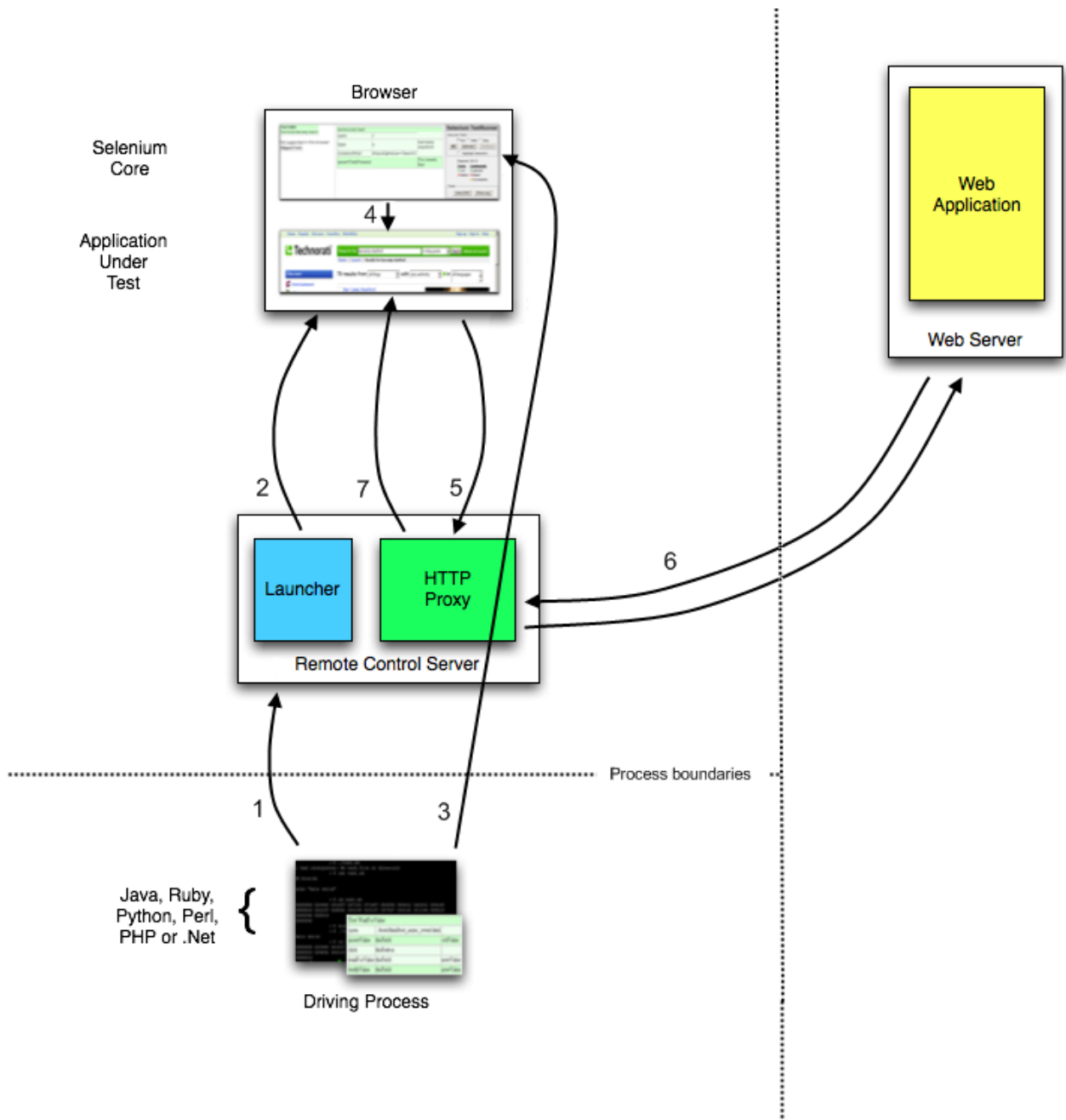


图 4 Selenium RC 的执行流程

1. 测试用例通过基于不同语言的 Client Libraries 向 Selenium RC Server 发送 Http 请求，要求与其建立连接。
2. 连接建立后，Selenium RC Server 的 Launcher 就会启动浏览器或者重用之前已经打开的浏览器，把 Selenium Core (JavaScript 函数的集合) 加载到浏览器页面当中，并同时把浏览器的代理设置为 Http Proxy。
3. 测试用例通过 Client Libraries 向 Selenium RC Server 发送 Http 请求，Selenium RC Server 解析请求，然后通过 Http Proxy 发送 JavaScript 命令通知 Selenium Core 执行浏览器上控件的具体操作。
4. Selenium Core 接收到指令后，执行操作。

5. 如果浏览器收到新的页面请求信息，则会发送 Http 请求来请求新的 Web 页面。由于 Launcher 在启动浏览器时把 Http Proxy 设置成为了浏览器的代理，所以 Selenium RC Server 会接收到所有由它启动的浏览器发送的请求。
6. Selenium RC Server 接收到浏览器发送的 Http 请求后，重组 Http 请求以规避“同源策略”，然后获取对应的 Web 页面。
7. Http Proxy 把接收的 Web 页面返回给浏览器，浏览器对接收的页面进行渲染。

第二，Selenium 2.0 的工作原理

接下来，我们回到上面那个百度搜索的测试用例，这个测试用例用的就是 Selenium 2.0。
Selenium 2.0，又称 Selenium WebDriver，它利用的原理是：使用浏览器原生的 WebDriver 实现页面操作。它的实现方式完全不同于 Selenium 1.0。

Selenium WebDriver 是典型的 Server-Client 模式，Server 端就是 Remote Server。以下是 Selenium 2.0 工作原理的解析。

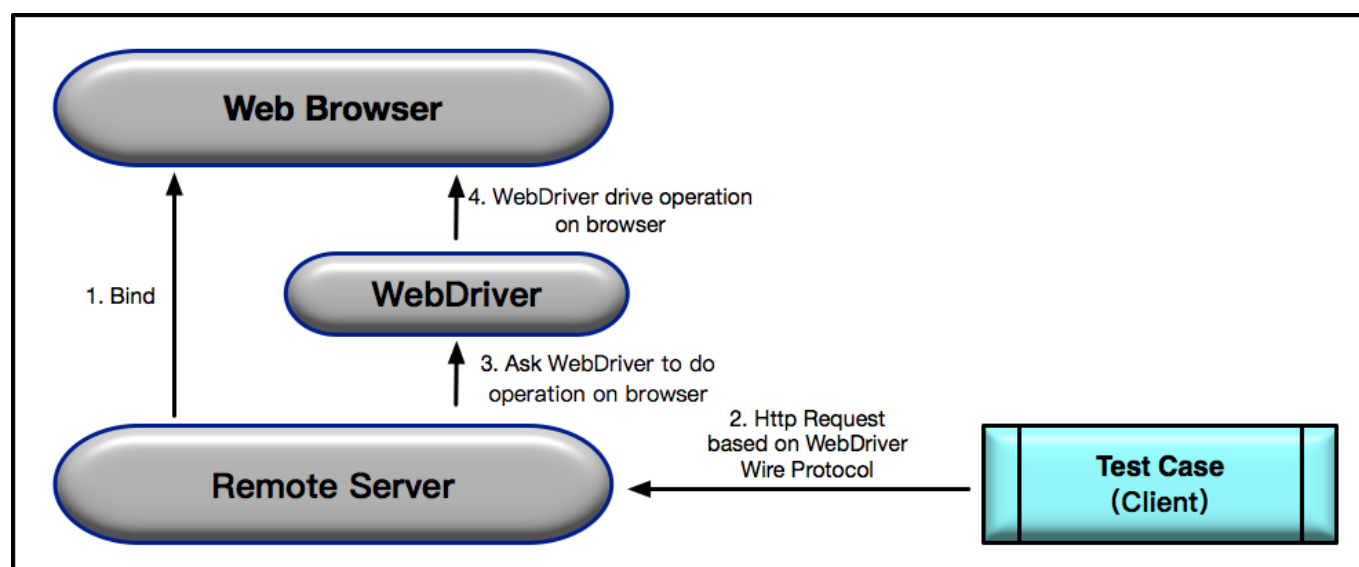


图 5 Selenium WebDriver 的执行流程

1. 当使用 Selenium2.0 启动浏览器 Web Browser 时，后台会同时启动基于 WebDriver Wire 协议的 Web Service 作为 Selenium 的 Remote Server，并将其与浏览器绑定。绑定完成后，Remote Server 就开始监听 Client 端的操作请求。
2. 执行测试时，测试用例会作为 Client 端，将需要执行的页面操作请求以 Http Request 的方式发送给 Remote Server。该 HTTP Request 的 body，是以 WebDriver Wire 协议规定的 JSON 格式来描述需要浏览器执行的具体操作。

3. Remote Server 接收到请求后，会对请求进行解析，并将解析结果发给 WebDriver，由 WebDriver 实际执行浏览器的操作。
4. WebDriver 可以看做是直接操作浏览器的原生组件（Native Component），所以搭建测试环境时，通常都需要先下载浏览器对应的 WebDriver。

总结

首先，我基于 Selenium 2.0，带你从 0 到 1 建立了一个最简单直接的 GUI 自动化测试用例。这个用例的实现很简单，但是只有真正理解了 Selenium 工具的原理，你才能真正用好它。

所以，我又分享了 Selenium 1.0 和 Selenium 2.0 的内部实现机制和原理：Selenium 1.0 的核心是，基于 JavaScript 代码注入；而 Selenium 2.0 的核心是，运用了浏览器原生支持的 WebDriver。

思考题

除了 Selenium，业内还有很多常用的 GUI 自动化测试框架，比如 UFT（以前的 QTP）、RFT、Nightwatch 等，你在平时的工作中接触过哪些 GUI 自动化测试框架？你知道它们的内部实现原理吗？

欢迎你给我留言。

软件测试52讲

从小工到专家的实战心法

茹炳晟

eBay中国研发中心
测试基础架构技术主管



新版升级：点击「 请朋友读」，10位好友免费读，邀请订阅更有**现金**奖励。

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 11 | 互联网产品的测试策略应该如何设计？

下一篇 13 | 效率为王：脚本与数据的解耦 + Page Object模型

精选留言 (27)

 写留言



Cynthia◆...

2018-07-28

 7

作者讲述的selenium工作原理，十分清晰有条理。

而其他教你学自动化测试学selenium的文章或者书籍，重点都在操作api上，或者就直接带你分析源码。而很少先把原理讲清楚，导致不少小伙伴学习并应用了很久，还是不够清楚背后的原理。所以也会看到类似的提问：为啥Chrome可以跑的case，Firefox跑不了，为啥webdriver还要一个浏览器装一个？ ...

展开 ▾

作者回复：你说的非常对，1.0基本淘汰了，重点是2.0，3.0的核心原理是个2.0一样的👍





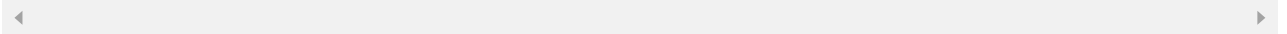
图·美克尔

2018-07-25

👍 6

希望能讲一些设计思想，比如我们构建web自动化测试时需要抽象出哪些类比如page oriented的设计，以及为什么这么设计，再就是想了解你们在搭建整个自动化框架的时候是具体使用到哪些模块，比如邮件通知模块以及报告结果模块或者日志模块等等，希望老师能给我们讲解一个业内目前的最佳实践。

作者回复: 你提的建议非常棒，下一篇文章就会讨论这些话题，第一篇只是给出一个最基本的gui用例，让刚入门的同学有个感性的认识



堂

2018-08-01

👍 3

近期也整理了一个GUI测试框架，pytest+selenium+allure+jenkins，目前只实现了抽离了公共方法、配置文件、测试数据、页面对象独立，使用上目前倒是没有什么问题，但总感觉还是不够灵活。最关键的是，现在脚本执行效率的问题。最近在研究如何实现脚本分布式运行的方法，但是还没有找到好的解决方案，好像jenkins通过节点实现脚本分割和测试报告归集（难点），不知道作者对脚本分布式运行有没有比较好的方案建议？万分感...
展开



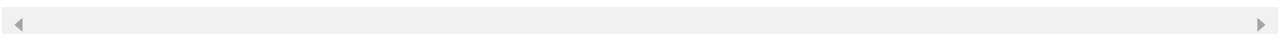
soul

2018-07-25

👍 3

值得注意得地方，预先装好浏览器的驱动器，不然会报错 "
java.lang.IllegalStateException: The path to the driver executable must be set by the webdriver.chrome.driver system property;大家如果再执行过程中提示如上错误，去这个地址下载即可
<https://sites.google.com/a/chromium.org/chromedriver/downloads>
展开

作者回复: 不同的浏览器需要下载对应的driver



C

2018-07-25

👍 3

一直纠结要不要做gui自动化，我遇到的问题有1.前端ui控件开发不统一，不是统一框架来做的ui设计，维护各模块脚本成本高。2.企业没有做单元测试和接口测试，没有信心做ui自

自动化测试3.ui界面动一动，ui感觉要维护成本较高。也听过别人说要针对ui框架设计UI自动化测试框架，不知道如何实施（目前还停留在ui脚本层）4.ui自动化能做到什么层度？冒烟是必须的，稳定模块的回归测试能否用ui代替，虽然成本高。

展开 ∨

作者回复: 总结的很到位，你说的这些问题我也都遇到过，要做gui自动化测试，这些问题都是不可回避的，后续的文章会介绍一些实践可以在一定程度上规避或者减轻这些问题。



红娟

2018-07-25

👍 2

配图很漂亮，颜色丰富的嫩芽。一看心情就很好

言归正传，我比较熟悉python，电脑里装的是selenium 3，待会儿试着写一下用例的case。重点是背后的实现原理。

我的问题是，如果需要了解背后的实现原理，是不是需要了解http网络协议？

作者回复: 不需要了解http网络协议，但是需要了解webdriver的实现原理以及web service的概念



sylan215

2019-02-13

👍 1

1.提到 Web UI 自动化目前用的最多的肯定还是 selenium 了；

2.之前也了解了 selenium 1.0 和 2.0 的区别，也对 WebDriver 有了基本的了解，但确实没有本次讲解的这么透彻，给茹老师点赞；

...

展开 ∨



Harry Po...

2018-09-13

👍 1

老师您好，请问cs架构项目的gui自动化测试除了QTP之外，还有其他比较好的开源工具吗？QTP是商用的，小一点的公司可能不愿意去花这个钱。



橄榄

2018-07-26

👍 1

没有做过GUI自动化测试，不过茹老师讲的也能听懂一丢丢

另外，能否专门讲个手机软件测试的专题，如何用monkey实现自动化，初学者请茹老师谅解

展开 ∨



塔矢亮的小...

2018-07-26

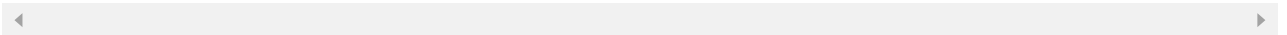
👍 1

刚入门自动化测试的小白想问一个很菜鸟的问|·ω·`)

我最近在看虫师的一本自动化测试的书来学习，用的是python+selenium，但是在元素定位这里一直受到了打击，要是定位按钮，链接都没问题，但是定位输入框比如说登录界面的输入框，用了各种办法xpath,css定位都说找不到这个元素，哦，我用的是chrome的插件xpath helper来自动生成xpath，按道理应该是没错的呀，一直非常困惑

展开 ∨

作者回复: 建议通过inspector观察需要定位的元素属性，然后再选择你的selector，如果单属性不好定位，也可以考虑使用组合属性



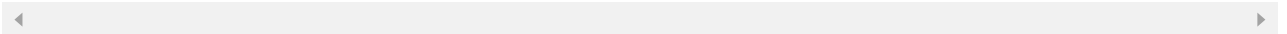
Jimmie.Zh...

2018-07-25

👍 1

前端修改越来越频繁，到大部分后端api修改较少，通常是新增，所以基于api的自动化测试效果刚加好。

作者回复: 说得很多，对于互联网产品的测试策略往往以api为主，但是前端还是会保留轻量级的自动化来保证最基本功能的回归，同时引入前端的探索式测试以发现更多的潜在问题



Geek_007

2019-05-02

👍

测试电脑桌面端软件 有什么好方案吗？

展开 ∨



bc

2019-03-26

👍

可以用来测试app吗？

展开 ▾



口水窝

2019-03-25



只有在深入了解原理的基础上，才能更好的去运用它实践操作，期待后面更多原理学习，实践操作。

作者后面说的这些自动化原理，都不了解，看来后面学习官方文档，很有必要。



涟漪852

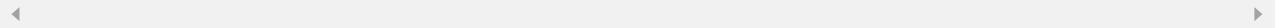
2018-11-23



Selenium的书籍，请老师推荐下(。·ω·。)/

展开 ▾

作者回复: 推荐官方网站的文档，没必要买书



王刚义

2018-10-25



老师，能讲一些传统软件的测试吗，比如，嵌入式软件根据需求写测试用例，单元测试，集成测试；感觉后面都是讲互联网软件方面的测试，谢谢了！



小老鼠

2018-10-24



学习Selenium 1.0原理现在还有意义吗？若有，意义在哪里？

展开 ▾



全都是泡沫

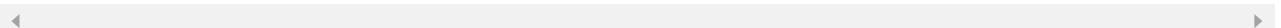
2018-09-29



老师，请问web service 技术现在有没有过时啊 还需要了解吗 感觉很难理解呢

展开 ▾

作者回复: web service目前依旧是主流，一定需要好好掌握





Geek_3a92c...

2018-08-21



做ui自动化也快1年了，让我说selenium webdriver的实现原理，还真不知道，值得反思



mm

2018-08-03



这篇文章讲的，我最近刚刚做过。我刚刚转行进入功能测试，没有代码基础，因此感觉这样没有浏览器插件录制脚本那么快捷。

我们做一个比较小的网站，只需要一些简单的回归测试，就是打开几个页面，点一点，验证一下页面上有没有出现应该有的数据。可能我们的需求还用不上webdriver吗？

而且不知为何，感觉webdriver打开网站比插件慢很多，还经常超时？和原理有关系吗， ...
展开 ∨

作者回复: 的确是，webdriver启动的浏览器过程中有很多额外的步骤。如果你的被测软件够简单，简单的录制回放是可以的，但是用例多了维护性的问题就会很严重！

