14 回溯UI自动化测试工具的前世与今生

更新时间：2019-09-20 18:07:13

> 没有智慧的头脑，就像没有蜡烛的灯笼。——托尔斯泰

前面我们一起深刻认识了一下**自动化测试**，相信大家一定已经对自动化不再陌生。对于自动化测试的世界来说，UI 自动化测试是很重要的一个组成，可以说很多人听到自动化测试第一时间联想到的都是 UI 自动化测试；对于我自己而言，UI 自动化测试也是我最早接触的测试开发技术。所以，于公于私，今天都想透过我自己的一些经历跟大家分享一下 UI 自动化测试和 UI 自动化测试工具。

什么是自动化测试工具

也许面对这个问题，很多同学要说，自动化测试工具很简单啊，可以让我们完成自动化测试的工具就是自动化测试工具呗。我觉得对，也不对。诚然，目前行业上，并没有对所谓的自动化测试

但是，我呢，有一点小小的贪心，希望能在自动化测试工具的前边加上一个 “好” 字。所以，什么可以称之为一个好的自动化测试工具?

我觉得一个好的自动化测试工具，应该具备这么几个特征：

- 支持脚本化语言 (Scripting Language)
- 对程序界面中对象的识别能力
- 支持函数的可重用
- 支持外部函数库
- 抽象层 — 将程序界面中的对象实体映射成逻辑对象
- 支持数据驱动测试
- 错误处理
- 调试器
- 源代码管理
- 支持脚本的命令行 (Command Line) 方式

支持脚本化语言可以让测试人员更好的学习上手；

具备识别能力能够大大提升自动化测试定位的准确性；

可重用、外部函数支持、面向对象编程让我们的自动化有了代码属性，从 **“一锤子买卖”** 的脚本升级为可以复用的框架结构，也给了自动化测试未来以无限的可能性；

支持数据驱动能够让我们更好的扩展测试案例；

而错误处理、debug、源码管理和命令行执行则是便于我们调试和自动执行。

匆匆的制定了一套标准，那么，就让我们按照这样的规则去看看历史上的好自动化测试工具们。

第一代 UI 自动化测试工具–录制回放类

我开始工作的时候，正好赶上录制回放类自动化测试工具成为主流的时候，其中最为有代表性的就是 QTP。在那个年代，基本上自动化就等于 QTP，当然，在它之前还有 WinRunner 这样的庞然大物，不过也没有能够像 QTP 一样统治一个测试时代。

不知道还有没有人记得这个熟悉的界面，我们一起来重新认识一下它。

QTP 全称 Quick Test Professional，是一种基于录制回放的自动化测试工具。QTP 的老东家可不是我们现在看到的惠普公司，而是 Mercury Interactive，它的第一个版本发布于 1998 年 5 月，那时候的我还只是个不懂计算机的学生。之前一直声明不显，直到 2006 年被惠普收购又同时发布了 QTP 最经典的版本之一 —— QTP9.0，才正式进入世人（至少是我国测试人）的眼球。这是一个商业化、插件化的工具，一如惠普的其他工具。

它的特点可以简单概括成如下：

1. **录制和回放。** 录制和回放是当时自动化测试工具的标配，通过录制回放，可以非常轻松的录制出一套自动化测试脚本，稍加修改就可以使用。
2. **对象库存储。** 这是 QTP 最重要的特征，将元素对象与自动化脚本分离，同时基于内置的 SPY 工具和 Mandatory 定位方式快速获得对象并将对象识别存储，也可以进行复用。
3. **Action。** 这是 QTP 组织测试用例的具体形式，直到现在我在写自动化测试框架的时候还习惯把一些孤立的动作行为称之为 action，可见这个名词对我留下的印象之深。QTP 中的 Action 最重大的意义在于通过 Action 实现了代码的复用，一旦 action 设置为 share 类型，就可以被其他的 Test 使用，大大增加了脚本编写和维护的效率。
4. **通过 DataTale 实现参数化。** 可以基于 excel 维护，这也提升了自动化测试数据维护的容易度，比起当初其他工具通过 xml 维护数据更容易更便捷。
5. **可以引用外部的 VBS 库。** 让各种 VBS 库可以为所有的 Action 和 Test 共享

按照我们前边的标准，QTP 在当时就是自动化测试的标杆，几乎所有的后来者都要以 QTP 作为模板，当年各种商业化工具的介绍语大多都是 “XX 方面可以媲美 QTP”，可见 QTP 在那个时代的统治力。我在写这篇文章的时候，以 QTP 作为关键词简单搜索了一下，发现近几年更多的问题变成了：**QTP 还有人在用么?**

为什么呢?

我回想了一下，在那时候主流测试都是 CS 应用，更多的是 windows GUI 程序，纵然有一些是 web 应用，也非常简单。而现在呢？随着互联网的兴起，重量级的 QTP 已经很难胜任自动化的使命了。

录制和回放无法保证脚本的良好可读性，对象库中录制识别的东西也不同于手写，手写的那是亲儿子，自己一眼便能认识；对象库呢，是别人家的孩子，偶尔夸两句无妨，真要掏心掏肺，完全不认得它是谁；QTP 大型的架构让其运行的速度和性能极其低下，要是用 low 一点的服务器都不好意思动手做自动化。

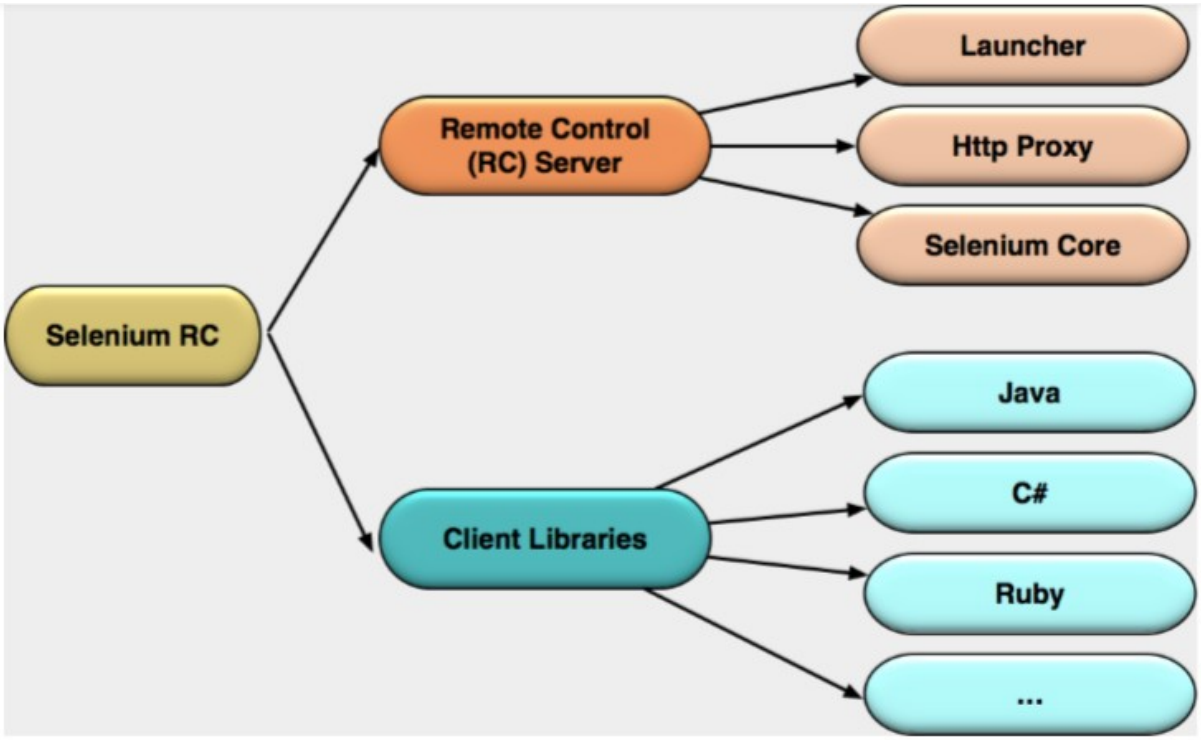
正是因为这些原因，在互联网大潮来临的日子里，在 2010 年第二个 “双十一” 到来的时候，我毅然决然的抛弃了 QTP 的大本营，加入了第二代 UI 自动化测试工具的大潮。

第二代 UI 自动化测试工具–开源编程类

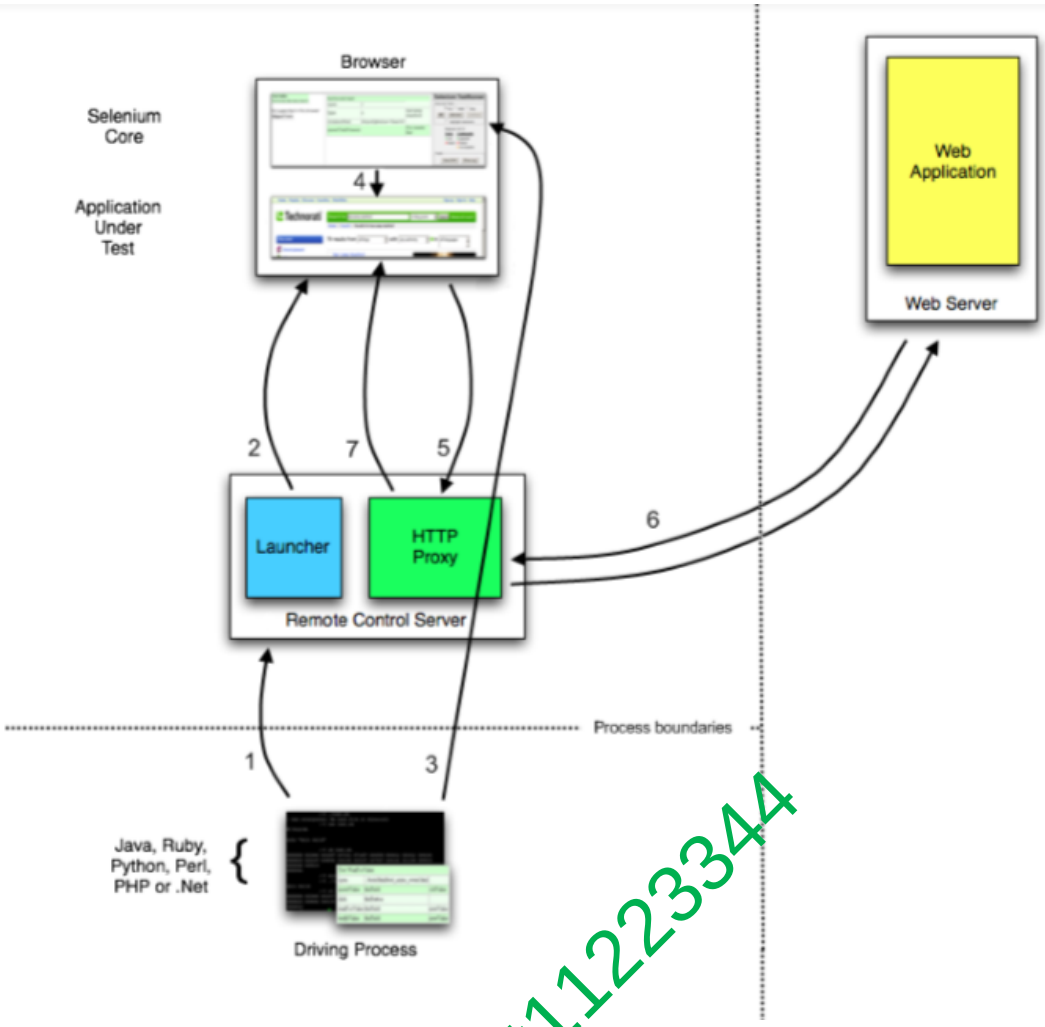
Selenium 比起 QTP 来，可以算是一个新生代了，它是由 ThoughtWorks 一个叫 Jason Huggins 的小伙儿和他的团队发起的，最初的目的仅仅是不想让自己的时间浪费在重复工作上，加上由于身处 ThoughtWorks 不好使用 QTP，所以决定换个思路通过 Javascript 来编写一种工具，这便是 Selenium 1.0 的核心。

说起来好笑的是，由于当时的大势，所有自动化工具都会或多或少的借鉴 QTP 的工作方式，所以 Selenium 提供了一个可以录制的 IDE 组件，更有意思的是 Selenium 的命名完全是由于当时 QTP 是主流公司，而又属于 Mercury 公司，在英文中 mercury 是汞的意思，所以他们选择了另外一种元素 “硒” 来与之对抗，所以命名为硒（Selenium）。

Selenium1.0 是由 Selenium IDE、Grid 和 Selenium RC 组成的，而其中 Selenium RC（Remote Control）是整个 Selenium1.0 的核心所在。我们先可以通过一个图来大概理解一下其内部模块结构：



可以看出来，RC 是由 RC Server 和 Client Libraries 构成的。在 RC Server 中 Core 作为 Javascript 的函数集合，实现界面的元素识别；Proxy 则是通过代理服务器的方式解耦浏览器的同源策略；Laucher 则是两者的融合剂，让我们能够在启动浏览器的瞬间，就完成 Core 的注入和针对代理浏览器的设置。如果要把其工作的方式串联起来，那么 Selenium 官方的这个图最有说服力：



已经描述的非常清晰，所以对于其工作方式，我在这里不再赘述，也由于 Selnium RC 的架构设计，也不可避免另外一个问题，由于无法完全绕过浏览器沙盒的限制，同时也无法调用操作系统级别的模拟用户输入方式。

所以在 2006 年的时候，Google 的工程师发起了 WebDriver 的项目，跳出了 Javascript 的沙箱束缚，提供了更快速、轻量级的原生浏览器模拟器 WebDriver。

2008 年，Selnenium 与 WebDriver 合并，成为了我们现在看到的更强大的自动化测试工具 Selenium 2.0。谈及合并原因，还是 WebDriver 创始人描述的比较好：

> 为何把两个项目合并？部分原因是 *WebDriver* 解决了 *Selenium* 存在的缺点（例如能够绕过 *JavaScript* 沙箱，我们有出色的 *API*），部分原因是 *Selenium* 解决了 *WebDriver* 存在的问题（例如支持广泛的浏览器），部分原因是因为 *Selenium* 的主要贡献者和我都觉得合并项目是为用户提供最优秀框架的最佳途径。

所以 Selenium 2.0 开始，前边提到的录制回放功能已经基本作废，相信很多熟悉或者使用过的同学甚至可能从来都不知道 Selenium 还有那么个 IDE，还可以录制和回放。所以到 Selenium2.0 开始，UI 自动化测试进入了新的脚本时代。

当然，到了 2016 年 Selenium 更新了 3.0 版本，除了修改了浏览器调用的核心以及完全舍弃了 1.0 时代的 RC 外，基本没有太大的变化了。Selenium 从 2.0 时代腾飞，到 3.0 已经趋于稳定，而现在，已经成为了最主流的 UI 自动化测试工具。

其实说起来风落自己，最早接触了解 Selenium 却是因为当初提供的简洁的 Firefox 插件 Selenium IDE 所给予的录制回放功能，为此而吸引开始研究。也在研究的过程中，逐步抛弃了录制回放，认识到了脚本编程的好处，认识到了自己能够完全掌控代码的优势，也开始了漫长的自动化测试探索之旅。

ps：第一次使用 Selenium2.0 却是因为 Selenium 的脚本抢马爸爸双十一的红包雨速度可比 QTP 快太多了……

2 Plus 时代 — Selenium 后时代

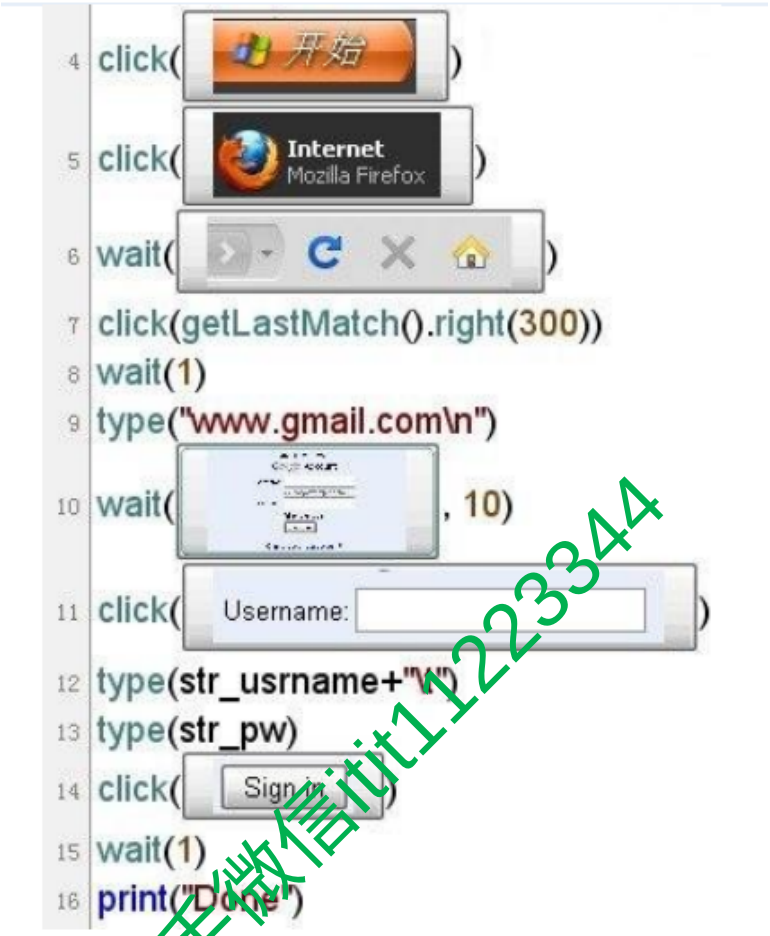
时至今日，Selenium 仍然是 Web 应用程序中，最受欢迎的开源测试框架，包括我在内的很多人都在上边进行封装、修改来设计属于自己的测试框架。所以，一部分非常优秀的自动化的测试工具，比如 Katalon Studio，Watir，Protractor 和 Robot Framework 都是以 Selenium 为底层核心框架的，国内也如是，我知道包括 Holmos、TestWriter 底层也都是基于 Selenium 的。所以，这个时代我们可以称之为 Selenium 后时代。

我敢肯定的说没有哪一个高级测试开发没有在 Selenium 的基础上搭建过测试框架，甚至 6 成以上的测试架构都下手去拆过 Selenium 的源码，到这里，Selenium 如同当年的 QTP 一样，几乎完成了对一个领域的统治。当然，不可否认，UFT（QTP 的升级版）、RFT 等工具仍占据一部分份额，但已经不是主流了。

当然，随着 Selenium 的深入使用，大家开始越来越多的找到一些 Selenium 基于浏览器无法解决的问题，尽管它已经跳出了沙盒的限制，但仍然很难处理一些基于系统的、自定义的处理。比如大家现在随处可见的自定义上传框、系统域账户认证等在 Selenium 中就很难处理。所以，在 Selenium 后时代，我们逐渐也开始寻找到一些可以辅助 Selenium 进行自动化测试的小工具。

我常用的一些小工具包括：

2. **Sikuli**： 一种新型思路的自动化测试工具，是由 MIT 的研究团队发布的新型图形化编程技术。它以图像检索技术为基础，提供了一套基于 Jython（python 语言在 java 中的完整实现）的脚本语言以及集成开发环境。使用者可利用屏幕截图直接引用 GUI 元素进行编程，完成交互操作。第一次见到 Sikuli 的脚本是这样的：



初看这样的脚本，真的非常惊喜，但是实际使用中仅可作为辅助使用，毕竟对于图像的识别要求过高，在不同的系统、不同分辨率、不同浏览器等等情况下都很难兼容。当然，后边又有 Airtest 这样的可以算是 Sikuli 的后代的工具，是网易出品的一款基于图像识别和 poco 控件识别的一款 UI 自动化测试工具。Airtest 的框架是网易团队自己开发的一个图像识别框架。对于我来说，并没有感觉它比 Sikuli 更好用，仅仅是 IDE 上的功能相对更加强大便捷。

总结

谈过了 UI 自动化测试工具的前世今生，现在的我们可以说是站在巨人的肩膀上。“巨人的肩膀上” 既是机遇，也是莫大的挑战。对于知学善进、能踏实研究的你来说，在 Selenium 已经成熟的今天，你可以做的更好。你可以让你的脚本从出生（脚本开发完成）、第一声啼哭（脚本自测）到成长（框架实现）、生存（执行与维护）这每一个阶段都变得更加美好。

也许你还在尝试着自己的第一个脚本，也许你还在沉迷于录制回放的“常规操作”之中，也许你已经开始领略到 Selenium 的优势，也许你正在尝试编写自己的第一个框架，我希望有一天，你会非常兴奋的给我留言：风落，我刚刚开源了一套 UI 自动化测试框架源码，欢迎大家都去看看！

那么从今天开始，从自动化测试工具走向自动化测试框架的漫漫长路吧，可以略感欣慰的是，这条路上不只有你，还有陪着你的风落，还有和你一样努力的”他“。

← 13 你真的了解自动化测试么?

15 驱动方式PK：数据驱动 VS 关键字驱动 VS 行为驱动 →

精选留言 3

欢迎在这里发表留言，作者筛选后可公开显示

慕神7099299

老师，想这种网银登录界面，密码输入有安全控件的，应该怎么处理呢?

1 回复 2020-02-07

风落几番 回复 **慕神7099299**

这方面一般更多是第三方的，我们尽量在测试环境mock掉。如果一定要自动化测试的话，用selenium是搞不定的，可能要配合autoit之类的工具了~

回复 2020-02-11 09:42:35

SCXR

老师，我朋友电商公司购买的阿里的码栈底层是不是也用selenium的?

1 回复 2019-11-28

风落几番 回复 **SCXR**

阿里的码栈实际上比较落后了，很小众啊。。web自动化也是selenium的说

回复 举报 2019-12-02 14:29:17

乞丐大叔

老师， 你开源的 UI 自动化测试框架源码地址在哪?

1 回复 2019-10-09

风落几番 回复 **乞丐大叔**

偶木有开源吖，我的框架。。放在我的自动化课程里边讲咧~所以就不开源了~

回复 2019-10-11 16:06:28

千学不如一看，千看不如一练