**自动化测试**

## 软件自动化测试

**名字里面有个“Auto”，给人的感觉神秘又高深。软件行业似乎对自动化测试也有神化的倾向。**实际上真正做过自动化测试的同学，我指的是对自动化测试有过一定思考的同学，对自动化的看法还是很现实的。我做自动化时间不长，大概有两年的时间，我很负责任的说，自动化其实也是一个坑。然而是火坑还是萝卜坑，与软件开发或是手工测试比是深还是浅？仁者见仁吧，我的看法：自动化不会解决所有问题，但是绝对值得拥有！

自动化测试其实就是通过自动化工具执行定制好的测试脚本，可以节省人力和时间成本,提高测试效率。但自动化测试不是并不能完全代替人工测试。自动化测试能解决很多问题，同时也带来很多问题。下面是我对自动化与手工测试的一个对比，算是一点心得。

**（1）覆盖率**

优势：在同等时间内，启动自动化测试能够覆盖更多的功能。

劣势：只适合回归测试，开发中的功能不划算。对于开发中功能，需求或者实现的更改，都会导致自动化脚本的变更，维护脚本的工作量和开发这个功能不相上下，实在是不划算。

**（2）测试效率**

优势：完成同等数目的测试，启动自动化速度更快。

劣势：脚本开发比用例开发耗时长，包括编写脚本、调试脚本、维护脚本，而手工测试也要对测试哦用例进行撰写、评审、修订。由于用例编写更多为自然语言，时间上肯定会少。

这里也引申一个另外的一种观点，直接用自动化脚本替代测试用例，也不乏是很好的做法。

**（3）执行可靠性**

优势：可靠的按脚本执行，后续定位、复现有明确的配置路径可循。

劣势：程序是死的，人是活的。目前而言，最智慧的还是人。可以说是成也萧何败也萧何，自动化的稳定来源于其死板，而人的智慧体现在思维的跳跃，跳跃的思维也会导致后期不易定位。

**（4）资源利用率**

优势：设备、仪表资源能够7\*24小时利用。

劣势：无。这点上，自动化完胜。

**（5）人力上限**

优势：可进行压力、负载、并发、重复等人力不易完成的任务。

劣势：无。这点，自动化完胜。

**（6）人员培养**

优势：提升测试人员能力，提高与开发沟通的效率。

劣势：培养一名自动化测试人员耗费资源更多，不但是功利的说自动化工程师更贵，同时在团队中推广自动化配套的培训、测试管理、产品开发环节都要跟上。增加的环节从某种意义上讲，就是浪费，因为如果不弄自动化，都可以省了。

## 二、自动化测试工具

**1 Web自动化测试工具：selenium、QTP。**

**（1） Selenium**

**@说明**

写过爬虫的朋友一定遇到过这样一个问题：网页的部分内容是由js(ajax)技术生成的，而这部分内容恰恰是我们想要的，并且这个 ajax请求的url或者生成的cookie信息我们很难拿到。普通的爬虫工具例如HttpClient只能模拟http发送请求，我们无法拿到url或者cookie，所以当然也无法拿到动态网页内容；htmlunit具有js渲染的功能，某些场景下使用htmlunit是一个比较合适的选择，但htmlunit对于某些复杂js却显得无能为力，关于htmlunit的使用，我们会在另外一篇博客中详细叙述。

Selenium，作为一个Web应用程序测试的工具，功能显然更加强大。Selenium的核心Selenium Core基于JsUnit，完全JavaScript编写，因此可运行于任何支持JavaScript的浏览器上。显然，Selenium非常适合解决上述我们提到的动态网页加载问题。

**@引入JAR包**

commons-codec-1.9.jar

commons-collections-3.2.1.jar

commons-exec-1.1.jar

commons-io-2.4.jar

commons-lang3-3.3.2.jar

commons-logging-1.1.3.jar

cssparser-0.9.14.jar

guava-15.0.jar

httpclient-4.3.4.jar

httpcore-4.3.2.jar

httpmime-4.3.3.jar

jna-3.4.0.jar

phantomjsdriver-1.2.0.jar

selenium-api-2.41.0.jar

selenium-chrome-driver-2.44.0.jar

selenium-firefox-driver-2.44.0.jar

selenium-htmlunit-driver-2.44.0.jar

selenium-ie-driver-2.44.0.jar

selenium-java-2.44.0.jar

selenium-remote-driver-2.41.0.jar

selenium-safari-driver-2.44.0.jar

selenium-support-2.44.0.jar

**@常见代码**

1、使用Selenium新建一个浏览器(这里以火狐浏览器为例,我这里是mac操作系统):

String browserPath = "/Applications/Firefox.app/Contents/MacOS/firefox";

System.setProperty("webdriver.firefox.bin", browserPath);

WebDriver driver = new FirefoxDriver();

其中browserPath是火狐浏览器的驱动路径。如果要新建一个PhantomJs浏览器，则代码如下:

String browserPath = "/opt/app/PhantomJS/phantomjs-2.1.1-linux-x86\_64/bin/phantomjs";

System.setProperty("phantomjs.binary.path", browserPath);

WebDriver driver = new PhantomJSDriver();

（2）为Selenium设置代理:

FireFox:

String proxyIp = "10.10.10.1";

int proxyPort = 80;

FirefoxProfile profile = new FirefoxProfile();

profile.setPreference("network.proxy.type", 1);

profile.setPreference("network.proxy.http", proxyIp);

profile.setPreference("network.proxy.http\_port", proxyPort);

profile.setPreference("network.proxy.ssl", proxyIp);

profile.setPreference("network.proxy.ssl\_port", proxyPort);

WebDriver driver = new FirefoxDriver(profile);

PhantomJs:

DesiredCapabilities desiredCapabilities = DesiredCapabilities.phantomjs();

Proxy proxy = new Proxy();

proxy.setProxyType(org.openqa.selenium.Proxy.ProxyType.MANUAL);

proxy.setAutodetect(false);

String proxyIp = "10.10.10.1";

int proxyPort = 80;

proxy.setHttpProxy(proxyIp + ":" + proxyPort);

desiredCapabilities.setCapability(CapabilityType.PROXY, proxy);

WebDriver driver = new PhantomJSDriver(desiredCapabilities);

（3）打开一个url并获取对应的网页源码:

String url = "http://www.baidu.com";

driver.get(url);

String pageSource = driver.getPageSource();

System.out.println(pageSource);

（4） 浏览器窗口最大化:

driver.manage().window().maximize();

（5）定位某个元素:

根据ID定位元素:

WebElement ele = driver.findElement(By.id("id"));

根据Class定位元素:

WebElement ele = driver.findElement(By.className("className"));

根据xpath定位元素:

WebElement ele = driver.findElement(By.xpath("xpath"));

（6）向表单填数据:

WebElement ele = driver.findElement(By.id("id"));

ele.clear();

ele.sendKeys("str");

（7）触发元素的点击事件:

WebElement ele = driver.findElement(By.id("id"));

ele.click();

**2 QTP**

**Quick Test Professional的简称，是一种自动测试工具。使用QTP的目的是想用它来执行重复的自动化测试，主要是用于回归测试和测试同一软件的新版本。**

1）QTP是一个侧重于功能的回归自动化测试工具；提供了很多插件，如：.NET的，Java的，SAP的，Terminal Emulator的等等，分别用于各自类型的产品测试。默认提供Web，ActiveX和VB。

2）QTP支持的脚本语言是VBScript，这对于测试人员来说，感觉要“舒服”得多（如相比SilkTest采用C#，Java,4Test语言）。VBScript毕竟是一种松散的、非严格的、普及面很广的语言。

3）QTP支持录制和回放的功能。录制产生的脚本，可以拿来作为自己编写脚本的template。

4）一个有用的工具：Object Spy，可以用来查看Run-time object和Test object属性和方法。

5）QTP通过三类属性来识别对象：a）Mandatory； b）Assistive； c）Ordinal identifiers。大部分情况下，通过对象的一些特定属性值就可以识别对象（类型a）。这些属性可以通过Tools->Object Identification 定义。

6）Object Repository（OR）是QTP存储对象的地方。测试脚本运行后，QTP根据测试脚本代码，从这个对象库中查找相应对象。每个Action可以对应有一个或者多个OR，也可以设置某个OR为 sharable的，这样可以供其他Action使用。注意，使用QTP录制功能时，默认将被测对象放在local OR中，可以通过 Resources->Object Respository，选择Local查看。

7）说到QTP的要点，不得不说Action。Action是QTP组织测试用例的具体形式，拥有自己的DataTable和Object Repository，支持Input和output参数。Action可以设置为share类型的，这样可以被其他test中的Action调用（注意：QTP是不支持在一个test中调用另外一个test的，只有通过sharable action来调用）。

8）如3）所述，一个test中，多个action的流程组织，只有通过Keyword视图查看和删除，在Expert视图中没有办法看到。

9）调用Action可以通过菜单Insert->Call to \*\*\* 来实现。QTP提供三种类型的调用方式：a）call to new Action，在当前test中创建一个新的Action；b）call to Copy of Action；c）call to existing action，调用一个re-usable action，如果这个re-usable action来自另外一个test，将以只读的方式插入到当前test中。

10）QTP提供excel 形式的数据表格DataTable，可以用来存放测试数据或参数。DataTable有两种类型：global 和local。QTP为DataTable提供了许多方法供存取数据，在对测试代码进行参数化的时候，这些方法非常有用。

selenium主要的特点是灵活轻巧，QTP主要的特点是功能强大。从Web测试的角度，两个工具都能满足正常的测试需求。当你用QTP/Selenium时，相信书和文档是你第一样会找的东西。在这样面，QTP使用在中国使用还是蛮广泛的，资料和书籍也相当丰富，而Selenium的资料就相对比较少一些，找资料最好的地方就是官方论坛。

selenium ：**开源，免费** 。应用领域WEB 。数据驱动要用编程实现。Selenium RC支持用多种语言编写测试案例,支持html，java,C#,ruby,python,perl,php自动化程度比较高，生成的perl代码基本不需要太多修改。录制回放成功率低。Selenium使用灵活，简单，写出的测试案例非常简洁，优美，也易于维护。

QTP:**商用需要付费且价格不菲**。应用领域WEB，WINDOWS GUI, .NET WINFORM,JAVA。它支持对操作Windows窗口，当你要从Web上下载一些东西，自然此时就会弹出一个“下载框”，由于那个框框是Windows窗口，Selenium是处理不了的，所以必须通过第三方的脚本处理。数据驱动支持灵活，可通过简单的设置完成数据驱动的自动化脚本。采用VBscript, 每个action的代码需要自己手动添加,需加校验。录制回放成功率高。如果你有一个测试平台，你会发现把QTP整合到平台是很麻烦的事情。

## 三、loadrunner、Selenium、QTP三者区别

1 Loadrunner是商业性能测试工具，收费，功能强大，适合做复杂场景的性能测试。

Selenium是开源的web自动测试工具，免费，主要做功能测试。

QTP是商业的功能测试工具，收费，支持web，桌面自动化测试。

**2 性能自动化测试工具：loadrunner、jmeter。**

1）、Jmeter的架构跟LoadRunner原理一样，都是通过中间代理，监控&收集并发客户端发现的指令，把他们生成脚本，再发送到应用服务器，再监控服务器反馈的结果的一个过程.

2）、分布式中间代理功能在Jmeter中也有，这个分布式分理是指可设置多台代理在不同PC中，通过远程进行控制，即通过使用多台机器运行所谓的Agent来分担LoadGenerator自身的压力，并借此来获取更大的并发用户数.LoadRunner也有些功能．

3）、Jmeter安装简单，只需要解压Jmeter文件包到C盘上就可以了，其实是没有安装．要是你想执行调试测试脚本，前提是：装上jdk和netbean插件.而LoadRunner安装包有1G多，在一台P3.0，1G内存的PC上安装要一个多小时.要是装过较旧的盗版还不能再装新版，解决办法倒是有，但麻烦且花时间.

4）、Jmeter没有IP欺骗功能，IP欺骗是指在一PC台上多个IP地址来分配给并发用户．这个功能对于模拟较真实的客户环境来说，是较有用.LoadRunner有此功能.

5）、Jmeter也提供了一个利用本地ProxyServer（代理服务器）来录制生成测试脚本的功能，但是这个功能并不好用，测试对象的个别参数却要手工增加上去，还得附带装个IE代理，如GoogleToolbarDownloader这些插件来捕捉参数.但是有一个工具bodboy，利用这个工具可以录制操作，然后选择将脚本保存为Jmeter脚本，然后利用Jmeter可以打开并修改脚本

6）、Jmeter的报表较少，对于要分析测试性能不足以作为依据.如要知道数据库服务器或应用程序服务的CPU，memory等参数，得在相关服务器上另外写脚本记录服务器的性能.

7）、Jmeter做性能测试，主要是通过增加线程组的数目，或者是设置循环次数来增加并发用户，而LoadRunner可以通过在场景中选择要设置什么样的场景，然后选择虚拟用户数。

8）、Jmeter可以通过逻辑控制器实现复杂的测试行为，相当于LoadRunner中的测试场景

9）、Jmeter可以做web程序的功能测试，利用Jmeter中的样本，可以做灰盒测试，LoadRunner主要用作性能测试

1. ）Jmeter是开源的，但是使用的人较少，网络上相关资料不全面，需要自己去揣摩，而LoadRunner是商业软件，如果是正版，有技术支持，同时，网络上的资料相当多。

## 四、接口自动化测试工具：SoapUI、postman。

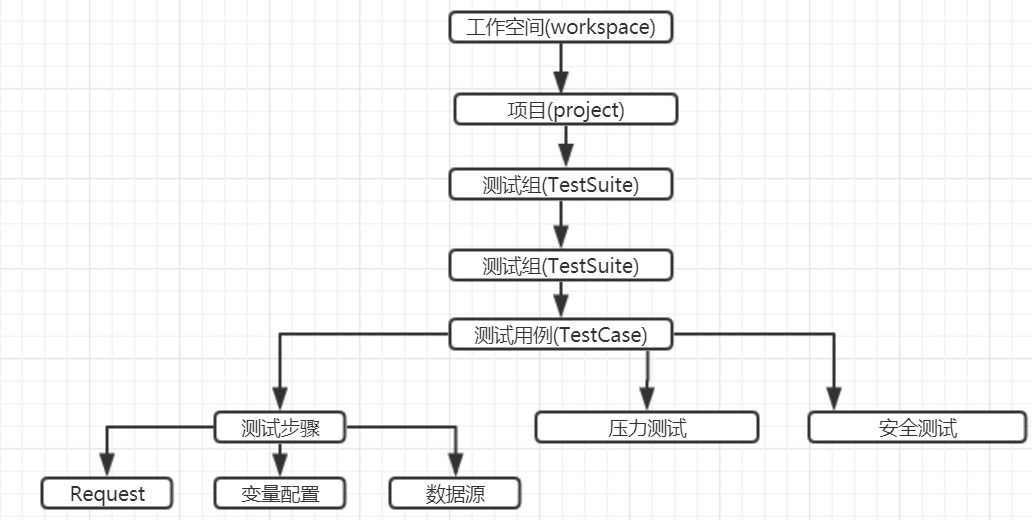
**1.从上面的实现来看,SoapUI自动化需要测试人员有一定的编码能力**，比Postman会对测试人员要求高些。

**2.从两种工具用例组织方式来看:**

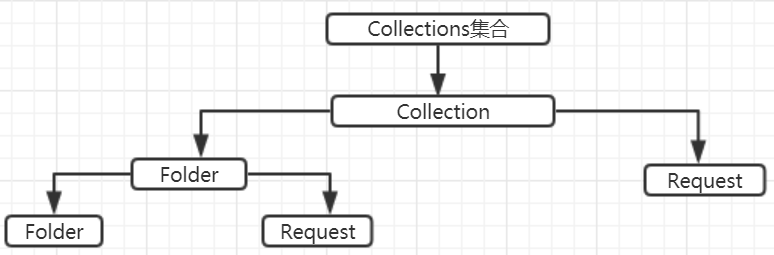
SoapUI的组织方式如下图，最上层是WorkSpace，所以每个WorkSpace中可以打开多个Project，一个Project也可以在不同的WorkSpace中。

Project对应我们的测试项目，其中可添加WSDL、WADL资源、TestSuite以及MockService。

TestSuite对应我们的测试模块，比如商户中心，其中可以添加TestCase，TestCase对应我们对某个模块的不同接口，比如订单管理接口。而一个接口可以能需要多个Step完成，变量、数据源、请求等都是一个Step。



Postman功能上更简单，组织方式也更轻量级，它主要针对的就是单个的HTTP请求。Collection就相当于是Project，而Collection中可以创建不定层级的Folders，可以自己组织TestSuite。每个Request可以当做是一个TestCase或者Step：



**3. 从长期的团队协作来看:**

SoapUI：本身一个project是一个xml文件，但是可以通过配置变成一系列文件夹，每个Case、每个Suite均是独立的文件，这样可通过svn/git/TFS进行团队协作。支持性较好。

Postman：有团队协作的功能，需要付费。

 因此从项目支持的接口类型，不同集成测试需求和后期维护来考虑，我们可以根据上面几点选择适合自己项目的接口自动化工具。

## 五、手机自动化测试工具：robotium、appium。

**1 Appium是基于UIAutomator框架实现的。**Appium测试进程与目标应用进程是分开的，所以Appium不能直接访问目标应用的各种element属性进行copy&paste，而只能模拟触发相应的事件对目标应用进行操作。这就好比触摸屏监控驱动和目标应用的关系：驱动监控到用户点击屏幕的事件后，驱动就会去判断点击的位置是否是一个文本框，如果是的话，就去打开系统键盘给用户进行输入。

**2 Robotium是基于Instrumentation框架的。**Robotium测试进程与目标应用是在同一个进程中作为两个不同的线程运行的。也就是说Robotium测试线程是有办法直接访问目标应用的各种element属性的，可以访问浮层，Intent之类的。所以它根本不需要触发任何事件，直接就可以在内部修改相应的数据，比如这个例子中的文本框输入文字。

**3 Robotium是直接Copy&Paste，**所以效率当然会更高，特别是在长字符串输入的时候，比如，输入一篇文章，如果在Appium上调用键盘，特别当你的安卓设别性能不高的情况下，你会明显感觉到卡卡的。Robotium因为不需要调出键盘，所以很容易就能输入各种不同语言；而Appium就不一样了，如果当前英文而你需要输入中文，那么你要去调用系统setting的应用先去把语言设置成中文，选择中文输入法等，然后再切换到目标应用进行输入操作。

**4 Appium更能模拟用户的行为操作；**而Robotium偏向测试金字塔更底层点的Unit Test（所以Robotium除了能支持测试目标应用的Activity外，还能测试Content Provider等）