

文档密级: 公司内部 A

RedwoodHQ 使用文档（底层代码开发）

陈晨

RedwoodHQ 使用文档（底层代码开发）	1
1. 背景.....	3
2. 安装.....	4
2.1 服务端.....	4
2.2 客户端（测试机器）	4
2.3 一些问题.....	5
3. 主界面功能介绍.....	6
3.1 选择项目 Choose Project 	6
3.2 设置 Settings 	6
3.2.1 用户管理 Users 	6
3.2.2 项目管理 Projects 	7
3.2.3 邮件配置管理 Email 	9
3.3 脚本开发 Scripts 	9

1. 背景

自动化测试从出现到发展到今天，一共经历了三个阶段：

1. 脚本录制/回放阶段。该阶段主要是以工具录制并记录操作的过程和数据形成脚本，通过回放来重复人工操作的过程。该方法的优点很明显：脚本生成简单，无需手动编写或者仅需少量修改即可投入使用；每一个用例都可以对应一个脚本，将脚本按照用例层次管理起来即可，非常方便。但是，该方案存在一个非常致命的缺点：界面或则操作的变化带来的维护成本太高。假如说界面自动化测试中，一个元素的查找关键（比如 ID 或者控件 Handle 什么的）发生了变化，那么与之相关的所有脚本都要进行修改，如果脚本量非常大，代价太大；另外，如果脚本不得不进行修改才能够使用，就得了解该工具的脚本编写方法，因此，对编写技术也是有一定的要求的。
2. 数据驱动 (data driven) 阶段。方法是：从数据文件读取输入数据，通过变量的参数化，将测试数据传入测试脚本。比如大家最了解的将测试数据样本按行存到 csv 或者 excel 当中，测试工具或者脚本读取这些文件，将数据分别传到待测产品中进行测试。该方法的优点：数据与脚本分离，可重用性提高，一个脚本或者工具就能搞定多组用例。缺点也很明显：仍然没能彻底解决变化带来的维护成本；脚本编写技术掌握需求仍然较高。
3. 关键字驱动 (keyword driven)：按照关键字/Action 进行分解，形成一系列的关键字对象用于测试。包含被操作对象、操作、参数和返回值四大类。优点：数据、脚本实现及对象、关键字三者全部分离，因此受待测对象变化影响最小（只需更新对应脚本即可，可能就几个，如果采用对象库维护，可能只需要改一个配置）。但它也有自己的缺点：依照上层越简单，底层越复杂的原则，底层脚本实现难度高，因此对底层开发人员要求非常高，难度超过了前两个阶段。

本文的说明对象 RedwoodHQ 是基于关键字驱动阶段的测试工具，它与比较著名的测试框架 RobotFramework 比较接近，且各有各的优点和缺点。在这里就不做横向比较了。

因为 RedwoodHQ 系统较为复杂，小功能点很多，本文档会尽可能的将经常会用的功能点介

绍出来。遗漏也在所难免，欢迎大家指出，也欢迎大家花点时间自行探索一下该系统。

2. 安装

2.1 服务端

目前电信平台组的服务端已经部署好了，地址是 <http://192.168.59.51:3000>。所以电信平台组的各位同事可以跳过此步骤，只需部署客户端就行了。

从官方网站 <http://redwoodhq.com> 下载服务端安装包，直接安装即可。安装过程中可以选择是否安装 VNC（用来远程查看测试机器，服务端一般都选择装），同时也可以调整服务端的访问端口等等选项，请按需配置。

如需要在安装后调整配置，可以修改安装目录下的 `properties.conf`。

安装完成后，通过 `http://服务端 IP:服务端端口`（默认是 3000）来访问。默认管理员用户名密码：`admin/admin`

安装完成后，任务管理器里会出现两个进程：`node.exe` 和 `mongod.exe`。如安装了客户端则会有两个 `node.exe`。

2.2 客户端（测试机器）

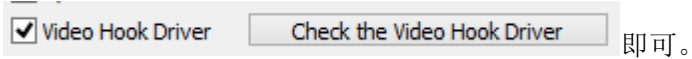
因为客户端安装包比较大，建议在服务端安装好以后，从服务端左上角的这个链接

[Download Agent](#)

下载，比从官网下载快（因为是服务端自带的）。

安装时同样可以选择是否安装 VNC。但由于安装的 UltraVNC 是绿色的，没有带显示器驱动，所以在 Windows 下的表现不好，远控时很卡，如需要通过 RedwoodHQ 远程查看 Windows 测试机器的话，请到 UltraVNC 的官网下载一个显示器驱动安装到客户端上，能大大提高性能（地址：<http://www.uvnc.com/downloads/mirror-driver.html>, XP 以上请选择 Win7 驱动）。

安装后重启系统，重启后在系统托盘处右键 VNC 托盘图标，选择 Properties，勾上此项：

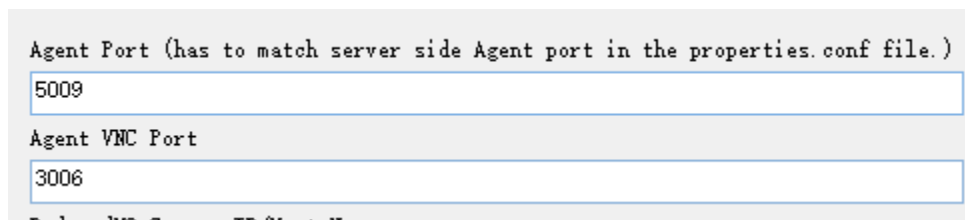


即可。

安装时也可以配置 IP 地址和端口等等一些参数，注意：



这两项需要和服务端的 IP 地址和端口号保持一致



这两项需要和服务端配置的端口一致。

安装完成后，任务管理器里会出现 `node.exe` 进程，同时在服务端的机器管理页面中会发现此机器的信息。

2.3 一些问题

1. Windows 服务端据说比 Linux 服务端表现稳定且性能好一些，所以建议在 Windows 下部署服务端。
2. 官方仅提供了 Ubuntu 版本的服务端和客户端，是否能在其他 Linux 发行版本下运行尚未测试。
3. Windows 下安装时如果使用了远程桌面，在断开或者最小化远程桌面的时候安装进程疑似会暂停，所以请保持远程桌面窗口，待安装完毕后再行断开或最小化（安装过程时间挺长）
4. 服务端或者客户端的 Windows 版本在多人登录这台机器时，会出现多个 `node.exe` 进程，应该是该工具将启动项放在了 `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run` 里有关，因为貌似不影响使用，可以暂时不管。
5. 在第 4 步出现的情况下，会出现使用工具提供的 "Stop RedwoodHQ" 和 "Stop Agent RedwoodHQ" 没用的情况（点击后发现 `node.exe` 或者 `mongod.exe` 进程都还在）。此时可

以采取杀对应进程的方式解决。

3. 主界面功能介绍

登录服务端网站后默认进入 Execution 主界面。主界面的主要功能大致划分为：

3.1 选择项目 Choose Project

Choose Project: Sample

在第一次使用 RedwoodHQ 的时候，仅有一个自带的 Sample 项目。后续使用的时候，需要对应产品创建不同的项目。

3.2 设置 Settings

Settings

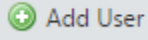
系统设置页面，通常由管理员管理。主要分为以下三个设置内容：

3.2.1 用户管理 Users

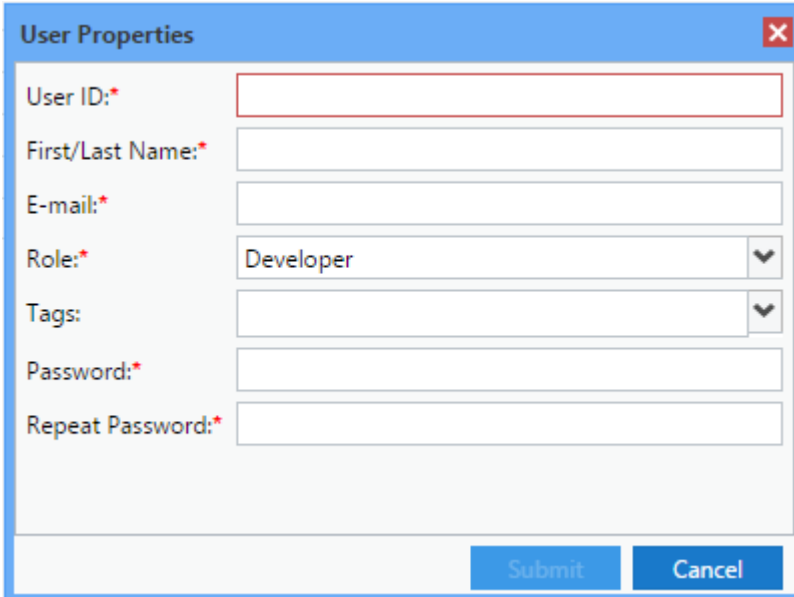
Users

可以在此处管理使用 RedwoodHQ 的用户（添加、配置、删除等）。目前该系统权限部分做的不够好，会发现各个用户都有创建用户和调整系统配置的权限。为了防止出现问题，目前建议该操作由 admin 账户进行。官方也提供了严格的权限管理补丁，目前尚未尝试：

<https://github.com/dmolchanenko/RedwoodHQ/pull/8>

使用  Add User

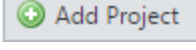
可以添加用户。弹出的页面如下所示：

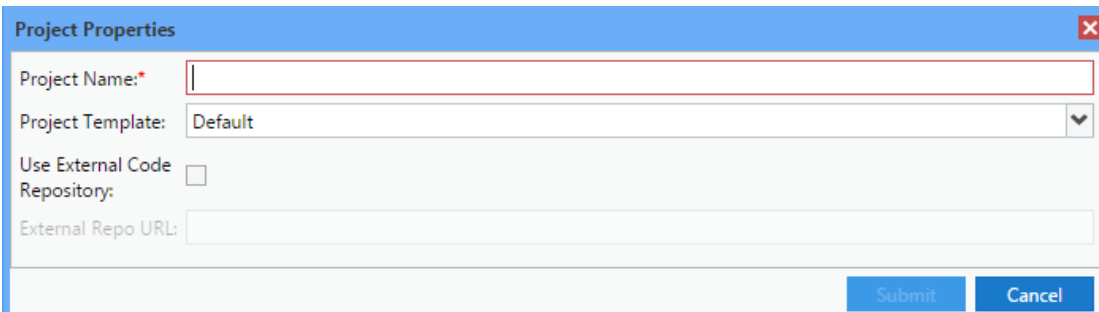
A dialog box titled "User Properties" with a close button (X) in the top right corner. It contains several input fields: "User ID:" (text box), "First/Last Name:" (text box), "E-mail:" (text box), "Role:" (dropdown menu with "Developer" selected), "Tags:" (dropdown menu), "Password:" (text box), and "Repeat Password:" (text box). At the bottom right, there are "Submit" and "Cancel" buttons.

注意：

1. Role 分为 Developer(底层开发人员,对整个系统都有访问和操作权限)以及 Test Designer (用例设计和执行人员。Script 部分仅有查看代码和 Git pull 更新代码的权限)。创建用户时建议严格按照角色权限进行选择。
2. Tags 用来创建备注,一般在搜索或者分类的时候用到。创建 tag 的方法是在 Tags 框里输入 tag 名称后回车,会自动创建一个 tag。已经创建的 tag 在下拉框里可以选择。

3.2.2 项目管理 Projects Projects

可以在此处管理项目（添加、配置、删除等）。使用  可以添加项目。弹出的页面如下所示：

A dialog box titled "Project Properties" with a close button (X) in the top right corner. It contains several input fields: "Project Name:" (text box), "Project Template:" (dropdown menu with "Default" selected), "Use External Code Repository:" (checkbox), and "External Repo URL:" (text box). At the bottom right, there are "Submit" and "Cancel" buttons.

其中：

1. 可以选择 Project Template 下拉框里的一个模板,来创建一个项目（默认安装只有 Java Based Selenium 一个类型）。例如如果想创建使用 Java 或者 Groovy 代码的网页类测试项

目，即可以在下拉框里选择 Java Based Selenium，这样创建好的项目里已经有现成的基于 Selenium 的各种 Action 了，可以直接拿来使用。

如果想创建自定义的模板，请参考服务端安装目录 \project_templates\java_project_selenium 目录里的结构，新建一个目录例如 **java_project_appium**，把代码等等按照此结构放进去，随后修改服务端的三处 js 文件即可，分别是：

- 1) cli\createProject.js, 追加红框里的内容。

```
var argv = require('optimist')
  .usage('Usage: %0 --name [name] --template [Default, Java Based Selenium, Java Based Appium]')
  .demand(['name', 'template'])
  .argv;
```

- 2) public\view\ProjectEdit.js, 追加红框里的内容。

```

xtype: 'combo',
fieldLabel: 'Project Template',
store: ["Default", "Java Based Selenium", "Java Based Appium"],
name: 'template',
value: "Default",
forceSelection: true,
editable: false,
allowBlank: false,
listeners: {
```

- 3) routes\scripts.js, 添加红框内的如下内容，也就是多添加几个判断。注意如果是其它语言，需要另起一个 language 判断，这里不再细说：

```
exports.CreateNewProject = function(projectName, language, template, callback){
  var templatePath = "";
  if(language == "Java/Groovy"){
    if (template == "Java Based Selenium") {
      template = "java_project_selenium";
    }
    else if (template == "Java Based Appium") {
      template = "java_project_appium";
    }
    else{
      template = "java_project";
    }
    //templatePath = path.resolve(__dirname, "../project_templates/"+multi_test");
    templatePath = path.resolve(__dirname, "../project_templates/"+template);
  }
}
```

- 4) 完成后重启服务端即可生效。

2. RedwoodHQ 默认是使用内置的 git 源来进行版本控制的，如果想使用外部的 git 源，可

以将 **Use External Code Repository:** ☒ 勾选，在 **External Repo URL:** 处填写外部 git 源的地址。

3.2.3 邮件配置管理 Email

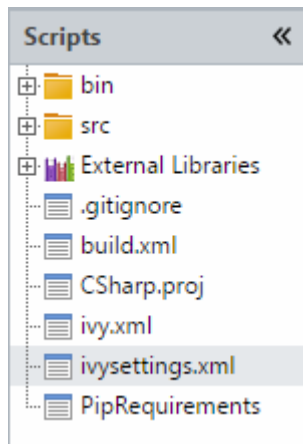
此处是配置发送邮件的一些参数，就不详细介绍了。其中 **Server Host:** 配置成服务端的 IP 地址就行了。

3.3 脚本开发 Scripts

供关键字驱动底层代码开发人员开发代码的地方。整个界面类似一个编辑器 IDE，略去大家都功能不说，仅挑一些重点地方来说。

1. 首先应该了解的，RedwoodHQ 在客户端机器运行测试的整个过程是：
 - 1) RedwoodHQ 对服务端编写好的 Java/Groovy/C#/Python 进行编译。其中：
 - Java/Groovy 的代码会使用 ivy.xml 和 build.xml 进行编译，C# 会使用 Csharp.proj 进行编译。编译完成后，产物会放到 build 目录下。基于 Java/Groovy 代码的脚本会编译成“项目名.jar”，基于 C# 的脚本会编译成"RedwoodHQAutomation.dll"。
 - 2) RedwoodHQ 通过客户端将这些产物、包括存放依赖库的 External Libraries 目录里的文件，都拷贝到客户端机器 Agent 安装目录\executionfiles 目录下一个随机的目录里面。对于 Java/Groovy/C#，编译产物以及之前 bin 目录里存放的东西都会放在 bin 目录下，External Libraries 目录里的文件会放到 lib 目录下。
 - 3)

2. 左侧的文件管理窗口结构大致如下图所示：



其中 bin 目录建议存放测试时调用的第三方 exe 或者 dll 的地方，测试运行时的工作目录也是在这里。