自动化测试类型：

1.UI自动化 2.APP UI自动化 3.接口自动化 4.性能测试自动化

自动化驱动方式：

1.顺序驱动测试 2.操作驱动测试 3.关键字驱动测试 4.数据驱动测试

RF介绍：

是基于python语言开发的一个，用关键字驱动的自动化测试框架；主要用于UI自动化测试。

RF安装：

1.在电脑上面安装3.0以上版本的python，这样在安装过程中python中的pip包会自动安装便于后期的安装操作。

2.在python安装好后进行环境变量设置；然后使用进入CMD后的操作界面使用pip install steuptools命令安装python的增强工具来优化后期的安装操作。

3.在CMD操作界面使用pip install RobotFarmerwork命令来安装robotfarmerwork自动化框架主体

4.在CMD操作界面使用pip install wxpython命令来安装python的gui库，完成后续的robotfarmerwork图形界面库安装前期操作。

5.在CMD操作界面使用 pip install RobotFarmerwork-Ride命令来安装robotfarmerwork的图形界面库。在此操作完成后一般ride会直接在桌面上创建一个快捷方式。这样就完成了RF的全部操作

问题解决：RIDE未能创建快捷方式：<https://www.jianshu.com/p/76cc5d8afe30>

RF特点：

1.可以通过关键词来规定用例的操作流程及步骤，进行人工操作的模拟及重复

2.可以通过关键词来进行web页面元素内容的获取并使用断言语句来完成结果明确的测试检测工作

3.用例的编写方式灵活，可以通过分层方式来自由组合实现操作及检查点的设置

4.可以对用例中的检查及输入变量进行参数化，提高用例的通用性

5.在测试用例完成后会生成一个HTML文本的详细执行log

6.在用例的执行失败时有明确的错误提示，方便用例的调试

7.用例之间相对独立化；某条测试用例执行的失败不会影响其他测试用例的执行

8.有很好的扩展性，可以与多种集成工具进行集成，实现持续集成的工作的自动化

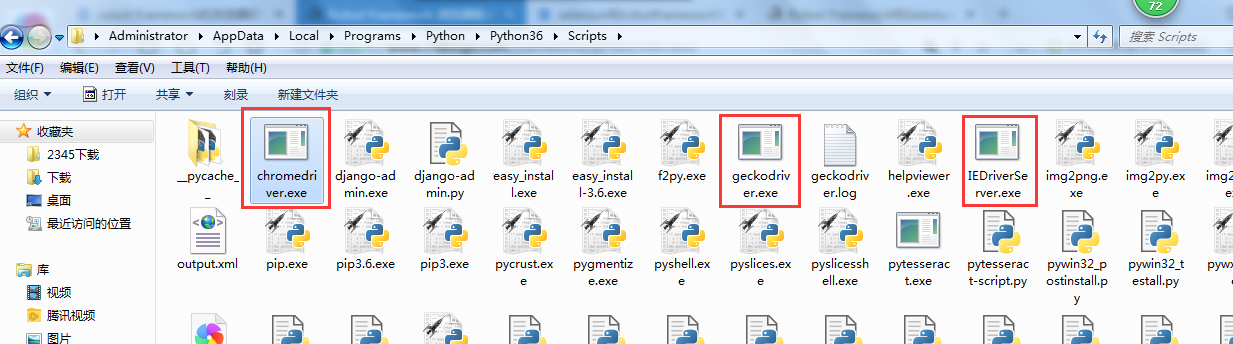
Selenium与RF区别：

1.适用性：RF不光适用于web程序的测试还支持其他类型的，selenium只支持web

2.测试功能：RF已经把selenium的功能很好的集成为了一个应用库运用于RF，同时RF还支持大量的外部库的应用，测试功能更强大

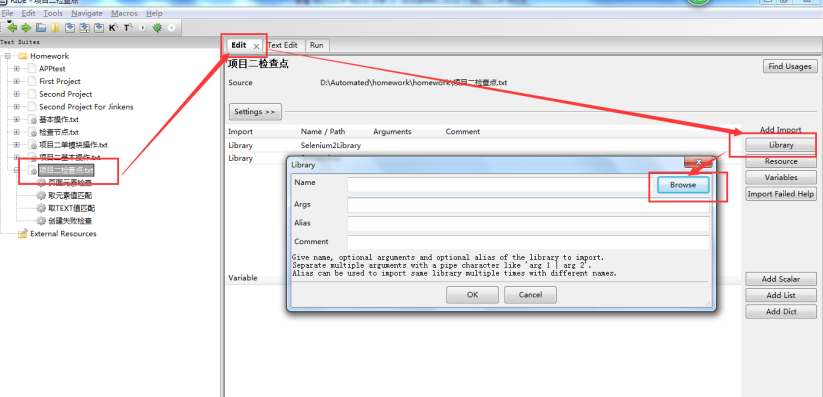
各浏览器驱动名称：

IE: IEDriverServer FF：geckodriver GC：chromedriver



Rf中如何加载和安装selenium

由于在前期的robotfarmerwork安装中已经安装了pip，可以直接在cmd界面使用 pip install seleniumlibrary来进行selenium库的安装，并在安装好了以后直接在集合的edit内进行库的调用



RF分层：

进行基本操作的参数化完成操作层封装（可以不需要进行非常详细的封装）

把多个参数化的操作整合在一起完成元素层的封装

把多个参数化的元素进行合理排列组合完成流程层的封装

把多个流程进行组合排列并把参数化的变量根据测试需要实参完成测试用例的组装完成数据层的封装

断言目的：

通过断言语句可以对测试用例制定位的页面位置的元素或内容进行取值比较；确定检查对象时候存在/不存在与页面；检查对象跟给定的预期结果进行匹配/不匹配比较；检查对象被预期结果包含/不包含。通过这些检测可以很好的判断产品是否满足我们对此处结果的控制要求（进行测试找出BUG）

断言方法：

使用${a} get text ${path} 取到指定的${path}位置对应的标签对中间的内容并赋值给${a}

使用${a} get element attribute ${path} ${b} 取到指定的${path}位置对应标签中的${b}标签元素内容

然后再在settings中设置带入的变量${c};使用断言判断语句对${a}和${c}进行断言判断

断言语句：http://blog.sina.com.cn/s/blog\_71bc9d680102wzma.html

隐形等待：

Wait Until Element Is Enabled 等待元素被启用

格式： Wait Until Element Is Enabled — 元素（必填）— 最长等待时间（选填）— 超时报错内容（选填）

Wait Until Element Is Visible 等待元素可见

格式：Wait Until Element Is Visible —元素（必填）— 最长等待时间（选填）— 超时报错内容（选填）

Wait Until Page Contains Element 等待，直到页面包含元素

格式：Wait Until Page Contains Element —元素（必填）— 最长等待时间（选填）— 超时报错内容（选填）

这3中隐形等待使用最频繁还有其他的可以再RF按F5查询

数据类型转换：

<https://www.cnblogs.com/xiaodouzhou-123/p/10251847.html>

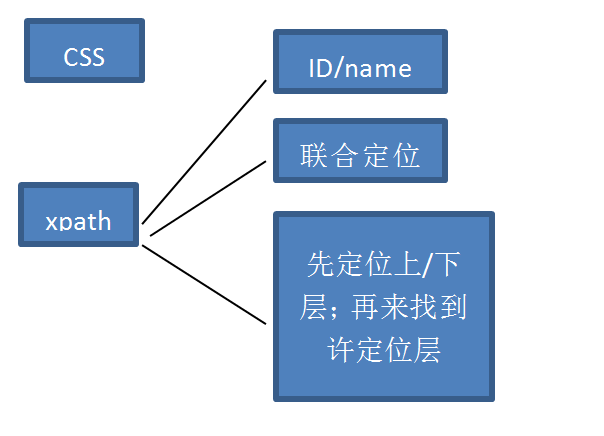
用例执行顺序编排：

https://www.cnblogs.com/colos/p/11083759.html

用例分布式及并发式实现：

https://www.jianshu.com/p/eecb9b2a22de

元素定位：



降低脚本维护成本：

1.对测试用例进行合理分层来使在脚本调试过程中进行问题定位明晰

2.在分层的最上层（数据层）进行变量实参化，提高底层脚本的复用率较少脚本编写/修改难度

3.做好脚本具体操作目的及检查目的的注释工作，加强脚本的易理解性

中文乱码的解决：

<https://www.jianshu.com/p/9c91c2f63156>

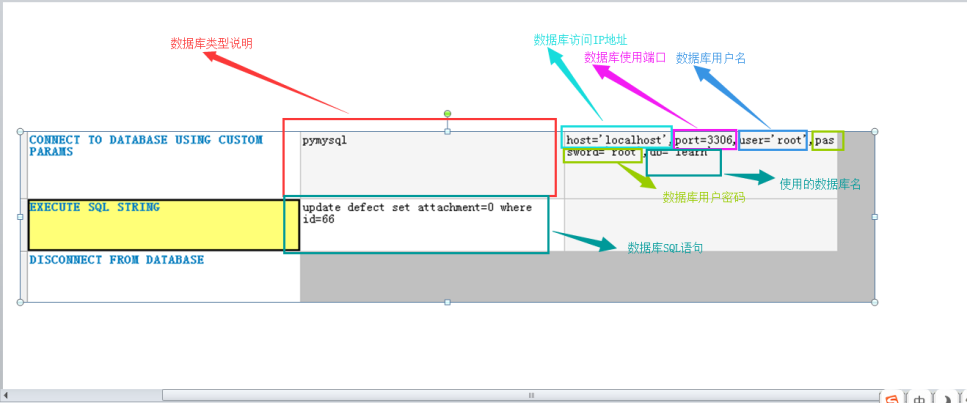
定位不准的报错：

Xpath=（编写的路径语句）下元素未被发现

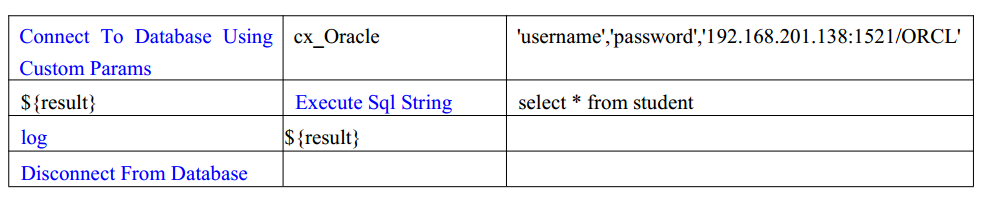
某某元素无法在视窗中找到

RF内部库（数据库使用，截图）：

DatabaseLibrary —— 数据库使用库



把SQL语句得到的结果赋给变量



Screenshot —— 截图库

TAKE SCREENSHOT 截取当前整个屏幕的内容

项目介绍共性问题：

介绍过程中停顿过多，准备不足——预先打草稿

没有突出在项目中自己在测试过程中工作内容及重心

在项目分享中很多人都提到了绝对路径这个绝对不能提

Bug准备：

在bug准备中不要过多的提功能bug，因为rf是在项目稳定后进行的测试所以不会有多少功能bug。同时不要提主流程及重要流程中的bug；多提使用频率不多的备选流的流程bug或者容易在测试中被遗留的bug

RF中的关键字类型有哪些

一般分为三种：当调用的为RF自带库时关键字就是原用关键字

当调用的为外部库时关键字就是外部关键字

当调用的为用户自定义库时关键字就是自定义关键字

RF中如何提升脚本的执行效率

1.较少脚本中不必要的sleep命令较少脚本运行的显性等待时间

2.在有可能需要等待的位置使用隐形等待关键词

3.简化脚本中的循环结构减少脚本的不必要循环

Python3.0与Python2.0之间的区别有哪些

1.3支持1/2直接输出0.5这个更合理的运算结果

2.3支持print(“XXXXXX” , END =” ”)这种打印格式定制语句

3.3中需要print的内容必须用()括起来

4.中对于循环范围的定义跟简洁 2-[0,1,2]==3-range(0,3)

5.对于整数2有 int跟long 3 精简到了只有int

6.3的多处语法进行了精简，排版更精简、美观

7.3加入的兼容库更多

Selenium1.0与selenium2.0之间的区别

<https://www.jianshu.com/p/4f0930c0b6a8>

RF中如何调用用户自定义关键字

同上面selenium库加载，但要注意自定义库加载过程一定要使用browse按键在自定义库路径下加载TXT文件。不能进行输入库名称进行加载；输入名称加载很容易失败

RF中什么情况下用JS定位