#### iOS自动化回归测试框架部署说明

Instrument uiautomation 是苹果官方提供的iPhone手机应用的自动化测试工具。控件元素的识别准确，属性获取，元素操作的API丰富。可以很方便的录制测试脚本、回放和查看运行结果。基于instrument，我们扩展了该框架，对其元素操作的稳定性进行了增强，统一了错误异常的处理及详细的运行日志，提供了方便的校验机制。结合具体项目，还实现了脚本的运行调度、日志的处理以及与Hudson集成的日常回归。

###### 一、框架结构

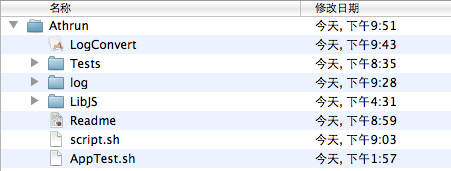


图1 框架目录结构

上图1为最后的框架结构，Athrun目录放置在系统的根目录。其中LibJS文件夹存放扩展的框架js文件，test目录存放具体的测试用例，log目录回归测试日志的存放。AppTest.sh 负责遍历test目录挨个运行其中的测试用例；script.sh 负责生成用例模板；LogConvert.app 负责合并运行的日志转换成Juint格式便于在Hudson上展示测试结果。

script.sh 使用方法如下：命令行进入Athrun目录，运行 ./script.sh <用例名称>

LibJS是扩展的框架文件，有什么共用的功能模块（如登录功能）或者修改、增强框架功能可以在这里修改框架文件。

Test目录下用于存放录制的脚本，这里建议根据用例的功能模块创建目录便于用例的管理，后续可以使用版本管理工具进行脚本的管理，AppTest.sh 会遍历test目录进行回归测试。

需要注意的是，因为每个具体的用例需要引入框架imports.js 才会在框架中运行，所以需要根据目录结构调整，如test目录下这样引入： #import “../LibJS/imports.js.

###### 二、具体用例制作过程：

1. 编译目标应用

终端命令行进入到项目主目录，执行如下命令：

xcodebuild –target <目标应用名称> -configuration debug –sdk iphonesimulator5.0 ARCH=i386

就会在项目的build目录相应debug目录生成应用的 app文件。

1. 开始录制脚本

有两种方式创建脚本：

一种方式为通过Athrun 下的script.sh 在命令行运行生成脚本，然后在instrument里面导入生成的脚本并开始录制用例步骤。

另一种方式直接在instrument里面创建脚本导出到相应的目录即可。

下面为具体instrument的使用步骤：

a.打开instrument 选择 Automation模板，上方target选择第一步生成的被测应用app，左下方点击add –Create 脚本或import已经存在的脚本。如下图2所示：

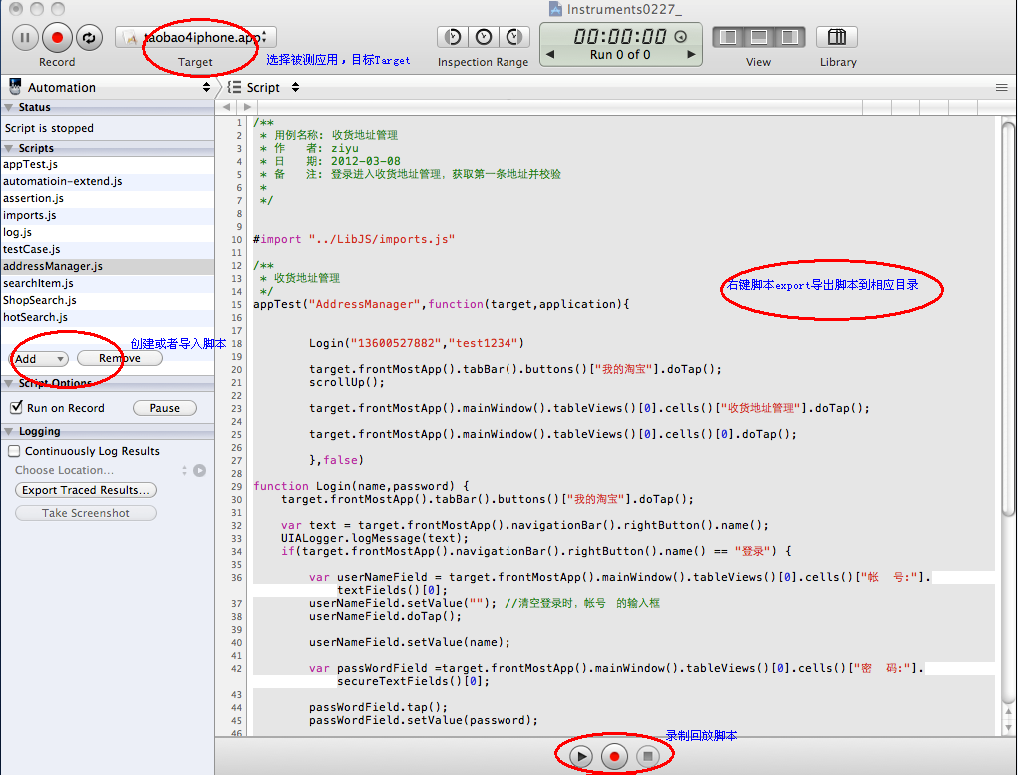


图2 instrument automation主要界面功能

b.右键脚本编辑区域，点击export导出脚本到框架结构介绍时创建的用例目录结构相应位置，导入的脚本则不需要此步骤。

c.创建好规范的格式，如用例的功能描述，引入框架js，用例名称及具体步骤。用例格式如下图3所示。将鼠标指针放在需要生产脚本的位置，然后开始录制脚本。

脚本录制完成后，修改操作为框架的API，保证脚本运行的稳定性。比如：Tap()替换为doTap()



图3 instrument automation主要界面功能

d.修改、调整、回放用例，检查运行结果直至满足预期。

###### 三、扩展框架提供的功能

为了简化用例步骤，方便各种校验。扩展提供了统一的校验方法和特殊元素属性的设置、获取校验等。

**基本的校验操作：**

**1、相等校验**

*/\*\**

*\* 这里校验文本，控件的 显示文本或者value值是否与期望的相等。*

*\* 不匹配则抛出异常，用例运行失败，匹配则写入日志信息并继续运行下一步骤*

*\* actual为实际值，expect 为期望值，message为附加log信息*

*\*/*

assertEquals(actual, expect l, message);

使用示例：

var cellText = target.frontMostApp().mainWindow().tableViews()["空列表"].cells()[0].name();

assertEquals(cellText ,”iOS 自动化测试”,”验证表格第一行显示为 iOS自动化测试”)

**2、正则表达式校验**

*/\*\**

*\* 正则表达式验证*

*\* 第一个参数 regexp 为自己设定的正则表达式，第二个参数 expression 为要验证的文本(比如控件的name或者value值)*

*\* 判断要验证的文本是否符合给定的正则表达式。*

*\* message为要写入的日志信息*

*\*/*

assertRegExptest(regExp, expression, message);

**3、包含文本校验**

*/\*\**

*\* 验证传入的控件对象是否包含期盼的文本，主要通过元素的name和label进行匹配判断*

*\* obj为传入的UIAElement，expectContain为期盼包含的字符，message为验证通过时添加的日志信息*

*\* 未作异常处理，参数不满足预期类型系统抛出异常被appTest捕获。*

*\* 注：这里可以扩大匹配范围，遍历传入obj中包含的element，逐个进行校验返回结果。暂时不支持*

*\*/*

assertContainText(obj, expectContain, message);

**4、弹出窗口title的校验**

*/\*\**

*\* 验证弹出的弹出框是否为期望的，通过popWindow的title进行判断*

*\*/*

assertPopWindowTitle(expectTitle, message);

**5、Switch控件状态的校验**

*/\*\**

*\* 验证Switch控件的当前开关状态。state =1 代表 on 状态，state =0 代表 off状态*

*\* obj 为要验证的Switch控件，state 为要验证的状态*

*\*/*

assertSwitchState(obj, state)

**输入操作：**

在需要输入的时候，但点击输入框，调出键盘后，为了简化键盘操作，请直接使用typeString(str)方法。Str为要输入的文本字符串。避免录制键盘操作生成的脚本不连续。输入完需要点击回车操作的时候，直接在str后面加上”\n”即可，如 typeString(“iPhone4s\n”)

中英文混输或英文数字混合输入的时候，请直接使用textView 的setValue(str) 操作，避免回放失败。

**滑动操作：**

instrument录制滑动的时候，不同位置点击滑动、滑动时间的长短，力度等不同，生成的脚本很不确定。为了更方便的处理，统一提供了上下左右的滑动操作：

scrollUp(): 模拟触摸向上滑动半个屏幕高度

scrollDown() :模拟触摸向下滑动半个屏幕高度

scrollLeft(): 向左滑动翻页

scrollRight():向右滑动翻页

使用时在需要滑动的地方，根据需要调用这些方法即可。

更多扩展详细功能请开打automatioin-extend.js、assertion.js查看。

###### 四、脚本运行的方式

创建好脚本之后，有两种方式运行：

一种是在instrument里面直接选中用例运行，适用于脚本的调试。

另外一种方式为通过执行 AppTest.sh 这个命令运行。进入到Athrun目录，运行如下命令：

./AppTest.sh ~/taobao4iphone/build/Distribution-iphonesimulator/taobao4iphone.app /Athrun/Tests

其中第一个参数为被测应用的app位置，第二个参数为用例的存放目录。

###### 五、回归集成

创建好用例并调试无误之后，我们就可以在搭建好的平台上自动运行这些脚本进行回归了。

在本机搭建Hudson环境，建立专门的项目并进行配置即可。instrument 提供了命令行的方式运行用例的js脚本。通过shell遍历用例目录逐个并逐个运行。运行完成之后启动日志转换工具对产生的日志进行转换，形成Hudson能解析的Junit形式的日志报告。构建完成之后即可在hudson上查测试的运行结果。

用到的两个文件：

1、AppTest.sh 负责遍历用例目录并逐个运行，并把运行日志放在/Athrun/Log目录下。

2、LogConvert.app 负责运行完脚本之后的日志转化，转换成Junit格式的xml文件，以便在Hudson上查看运行结果。

以下是一个示例，Hudson 的简单配置如下:

主要构建命令如下：

cd  /Athrun

./AppTest.sh ~/taobao4iphone/build/Distribution-iphonesimulator/taobao4iphone.app /Athrun/Tests

cd  /Athrun/LogConvert.app/Contents/MacOS/  
./LogConvert  
cd ~/.jenkins/jobs/iOS-test/workspace/xmlLog/

cp /Athrun/log/log.xml log.xml

注释：

~/.jenkins/jobs/iOS-test/workspace/xmlLog/ 为Hudson项目配置的工作区  
 cp /Athrun/log/log.xml log.xml 拷贝转化后的log.xml到工作区

结合Hudson，配合版本控制的管理，能够实现项目的自动编译，自动运行测试用例，实现了项目的自动构建自动回归测试。