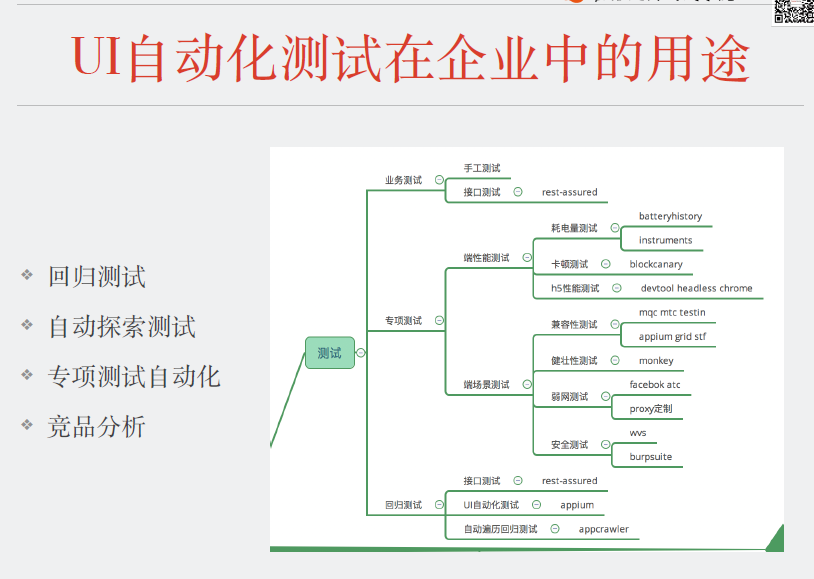
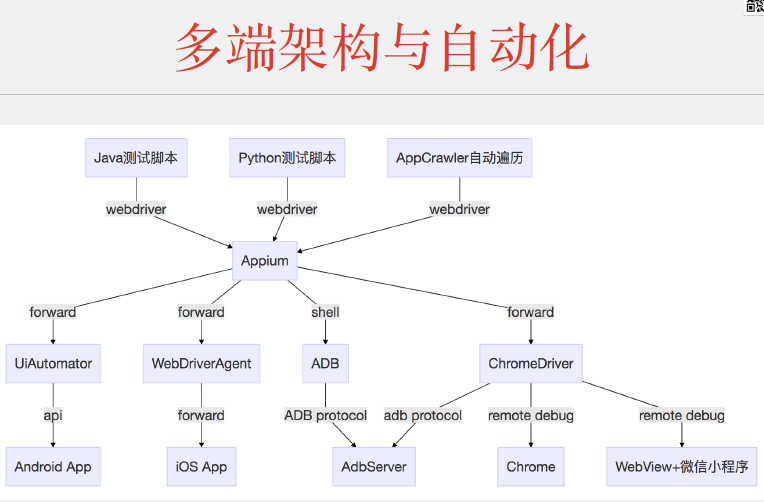
**课程表**

|  |  |
| --- | --- |
| 移动测试自动化Appium  10月13日  10月20日  10月27日  11月3日  11月10日  11月17日 | 1、Appium环境安装与架构介绍 2、Appium用例录制与用例流程介绍  3、元素定位与常见自动化动作 4、高级定位与断言机制  5、Android自动化用例编写 6、Android WebView测试与Toast识别  7、Android自动化测试实战 8、iOS测试用例编写  9、iOS真机测试与WebView测试 10、iOS测试实战  11、Appium错误排查与日志分析 12、PageObject设计模式  13、参数化与数据驱动实战 14、微信公众号与小程序测试实战  15、App自动化测试实战 16、自动遍历测试方法与AppCrawler基本使用  17、自动遍历测试进阶与实战演练 18、Appium Grid模式  19、多设备管理平台STF 20、兼容性测试实战 |
| 接口测试实战  11月24日  12月1日  12月8日  12月15日 | 1、接口测试体系与常见协议解析 2、代理工具抓包和篡改结果  3、接口测试框架选型与入门 4、Restful接口测试入门  5、Restful接口测试进阶与断言 6、通用协议加解密与schema断言  7、参数化与数据驱动 8、接口测试与持续集成结合  9、Web hook机制与可视化报告 10、接口测试实战  11、dubbo测试入门 12、dubbo测试进阶与应用  13、接口管理工具swagger入门 14、接口管理工具swagger实战  15、接口测试答疑课 |
| 移动专项测试  12月22日  12月29日 | 1、App性能测试 2、App流量分析 3、弱网测试  4、内存泄漏检测 5、卡顿检测 6、崩溃检测  7、耗电量分析 8、App反编译分析 9、App行为篡改 |
| Web测试自动化Selenium  1月5日  1月12日  1月19日  1月26日 | 1、Selenium简介与环境搭建 2、Selenium IDE用例录制与常见API讲解  3、元素定位与常见操作命令 4、Selenium的断言机制与等待机制  5、表单切换与窗口处理（窗口切换、设置窗口大小、关闭窗口）  6、Selenium多浏览器处理（Chrome、Firfox、IE、Safari）  7、文件上传与弹框处理 8、Cookie机制  9、测试报告生成与定制 10、Selenium基本操作实战演练  11、参数化与数据驱动实战（读取csv、Excel）12、PageObject设计模式  13、Selenium grid浏览器集群管理 14、Selenium项目实战演练  15、Selenium整体答疑 |
| 持续集成Jenkins  2月16日  2月23日  3月2日 | 1、Jenkins安装与入门介绍 2、Jenkins Job机制  3、Web Hook机制与高级插件 4、Selenium自动化测试持续集成实战演练  5、Appium自动化测试持续集成实战演练 6、接口测试持续集成实战演练  7、Pipeline入门 8、Jenkins file语法  9、Blue Ocean使用 10、Jenkins实战演练 |
| 容器技术应用Docker | 1、docker介绍与入门 2、docker基础命令  3、使用docker搭建selenium、jenkins、sonar等  4、使用docker搭建android模拟器与appium  5、docker镜像制作入门 6、docker镜像制作应用 7、k8s简介 |
| |  | | --- | | 代码审计平台Sonar | | 1、Sonar平台安装与基础使用 2、Sonar在单元测试与覆盖率上的应用  3、Sonar api使用 4、Sonar自定义代码扫描规则 5、Sonar与jenkins集成 |
| 质量监控平台建设ELK | 1、ELK系统搭建与基础使用 2、LogStash数据导入 3、ElasticSearch基本使用 4、Kibana高级报表制作 5、质量数据收集与监控面板制作 |

**UI自动化测试在企业中的用途**



**多端架构与自动化**



**夜神模拟器**

1. SDK的adb需与夜神的nox\_adb版本一致(如不一致则将SDK中的adb复制到模拟器bin目录下，将该名称改为nox\_adb.exe).
2. 启动模拟器，cmd至夜神bin目录，使用命令“nox\_adb.exe connect 127.0.0.1:62001”. 成功连接了模拟器之后，用"adb devices -l"命令查看设备
3. appium-desktop🡪 "deviceName": "127.0.0.1:62001"
4. 开发者模式: 设置->关于平板电脑->点击5次”版本号”

**获取app的信息**

app信息

* 获取当前界面元素：adb shell dumpsys activity top
* 获取任务列表：adb shell dumpsys activity activities

app入口

* adb (–s 192.168.59.102:5555) logcat |grep -i displayed
* aapt dump badging mobike.apk | grep launchable-activity
* apkanalyzer 最新版本的sdk中才有

启动应用

adb shell am start -W -n com.xueqiu.android/.view.WelcomeActivityAlias –S

**capabilities设置**

* app apk地址
* appPackage 包名
* appActivity Activity名字
* automationName 默认使⽤uiautomator
* noReset fullReset 是否在测试前后重置相关环境
* unicodeKeyBoard resetKeyBoard 是否输⼊非英文,测试完成后重置输⼊法
* udid 手机唯一标识

**测试步骤三要素：定位、交互、断⾔**

定位

* id定位 对应accessibility id
* accessibility id定位 对应content-desc (盲人专用)
* xpath定位
* 绝对定位(全路径)  
  /hierarchy/android.widget.FrameLayout/android.widget.LinearLayout/android.widget.FrameLayout/android.view.ViewGroup/android.widget.FrameLayout/android.widget.LinearLayout/android.widget.RelativeLayout/android.widget.LinearLayout/android.widget.HorizontalScrollView/android.widget.LinearLayout/android.widget.RelativeLayout[2]/android.widget.RelativeLayout/android.widget.TextView
* 相对定位(半路径)

//android.widget.TextView[@instance=2] 用属性定位

//android.widget.TextView[@text='股票'][1]

用属性,位置定位(xpath索引从1开始,属性值为单引号)(text是股票的第1个元素)

(//android.widget.TextView[@text='股票'])[1] 全局,第1个是text是股票的TestView   
//\*[@text='股票' and @resource-id='com.xueqiu.android:id/text']

未知节点,并用多个属性定位  
//\*[contains(@resource-id, 'text') and @text='股票'] 用包含,and定位

//\*[contains(@text, ‘登录’) or contains(@label, ‘登录’)]] 用包含,or定位

//android.widget.ListView[@resource-id='com.xueqiu.android:id/listview']/

android.widget.LinearLayout[last()] 取最后一个LinearLayout元素

//\*[contains(@text, '看点')]/ancestor::\*//\*[contains(name(), ‘EditText’)] 取先辈节点

//\*[@clickable="true"]/android.widget.TextView[string-length(@text)>0 and

string-length(@text)<20] 用属性长度定位

* Android UIAutomator Selector

new UiSelector().text("自选") #查找text为””自选”的元素

new UiSelector().className("android.widget.TextView").instance(1)

#查找第一个TextView

new UiScrollable(new UiSelector().scrollable(true).instance(0)).getChildByText(new UiSelector().className("android.widget.TextView"), "食品饮料") #查找滚动元素

交互:

动作: click, sendKeys, sendKeyEvent

self.driver.find\_element\_by\_id("agree").click()

手势: press, release, moveTo, tap wait, longPress, cancel, perform

滑屏: TouchAction(driver).press(x=300, y=2400).move\_to(x=300, y=300).release(). perform()

其他常用API:

截图: driver.get\_screenshot\_as\_file("login.png")

隐式等待: driver.implicitly\_wait(5)

获得xml布局: print(driver.page\_source)

**安装卸载Appium Server**

安装: npm install -g appium

或 淘宝cnpm:

npm install -g cnpm --registry=[https://registry.npm.taobao.org](https://registry.npm.taobao.org/)

cnpm install -g appium (--verbose)

启动: appium –p 4723

卸载: npm remove –g appium

查询已发布版本: npm view appium versions --json

指定安装版本: npm install [appium@x.x.x](mailto:appium@x.x.x)

**调试分析方法**

* Appium Log分析

appium –g /tmp/appium.log

grep “Running” /tmp/appium.log | less

grep “Running” /tmp/appium.log | grep –o “192.168.59.101\\\:5555.\*” | awk ‘{print “adb” $0}’ | sed –e “s#’\$##g”

获取sessions信息

创建session，提供app信息

查找java版本

查找adb版本

查找设备及其他信息

查找aapt

adb shell am force-stop com.xueqiu.android  
adb shell input keyevent 3  
adb shell ps  
adb shell getprop ro.build.version.sdk  
adb shell getprop ro.build.version.release  
adb shell wm size  
adb shell wm density  
adb shell getprop ro.product.model  
adb shell getprop ro.product.manufacturer  
adb shell am force-stop io.appium.uiautomator2.server  
adb shell pm list packages io.appium.uiautomator2.server  
adb shell dumpsys package io.appium.uiautomator2.server  
adb shell pm list packages io.appium.uiautomator2.server.test  
adb shell dumpsys package io.appium.uiautomator2.server.test  
adb shell pm list instrumentation  
adb shell echo ping  
adb shell pm list packages io.appium.settings  
adb shell dumpsys package io.appium.settings  
adb shell am start -W -n io.appium.settings/.Settings -a android.intent.action.MAIN -c android.intent.category.LAUNCHER -f 0x10200000  
adb shell appops set io.appium.settings android\:mock\_location allow  
adb shell pm list packages io.appium.unlock  
adb shell dumpsys package io.appium.unlock  
adb shell rm -rf /data/local/tmp/strings.json  
adb shell pm path com.xueqiu.android  
adb shell pm list packages com.xueqiu.android  
adb shell am force-stop com.xueqiu.android  
adb shell pm clear com.xueqiu.android  
adb shell am force-stop io.appium.uiautomator2.server  
adb shell am instrument -w io.appium.uiautomator2.server.test/android.support.test.runner.AndroidJUnitRunner  
adb shell dumpsys window  
adb shell am start -W -n com.xueqiu.android/.view.WelcomeActivityAlias -S  
adb shell cat /proc/net/unix

* 得到PageSource

界⾯的完整dom结构. xml⽂件



* 调试脚本

driver.findElementsByXPath(“//\*") #取出所有元素,方便下一步过滤/获取信息

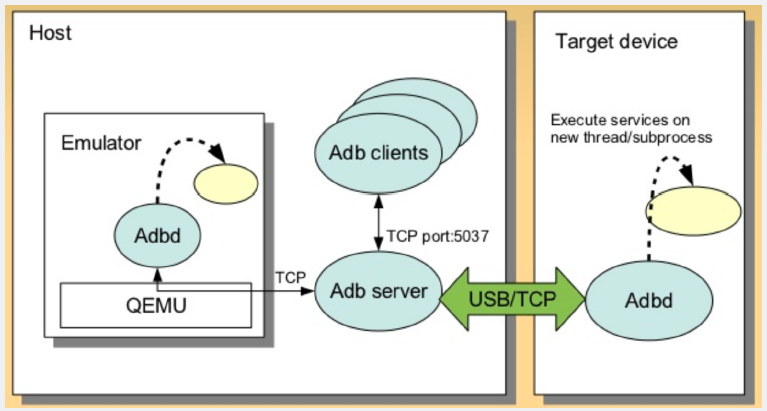
for x in self.driver.findElementsByXPath(“//\*")

print(“text=”x,text)

print(“tag\_name=”x,tag\_name)

print(“location=”.location)

**Adb知识与排错**



**脚本优化**

将”定位”与”行为”隔离

self.driver.find\_element\_by\_id(“agree”).click() 更新为

self.button\_agree=(MobileBy.ID,”agree”)

self.driver.find\_element(self.button\_agree).click()