全程87期appium第一次

### 内容1： appium 灵魂三问：

**什么是appium？**

appium 手机自动化测试的工具（本质 selenium 扩展版本 webdriver）

ui\_automator (google android Apple ios)官方地位

工具缺点 ： api代码 繁琐 简化 ===》 appium

**appium= ui\_automator （官方）+selenium**

**其他robotium 重签名**

**==》被测对象 app（web）**

**为什么appium？**

场景1： apk 文件--android安装包安装卸载100次（疲劳性测试）？

答 ： 效率低 （python脚本 让自动化工具自己做~）

场景2： UI自动化

答： appium 手机页面比 web页面小 更好 UI脚本

。。。。。

目标： 写个小测试框架？？（提升UI自动化 --多用例测试 测试断言 测试报告）

**如何掌握appium？**

**1疲劳性 测试做**

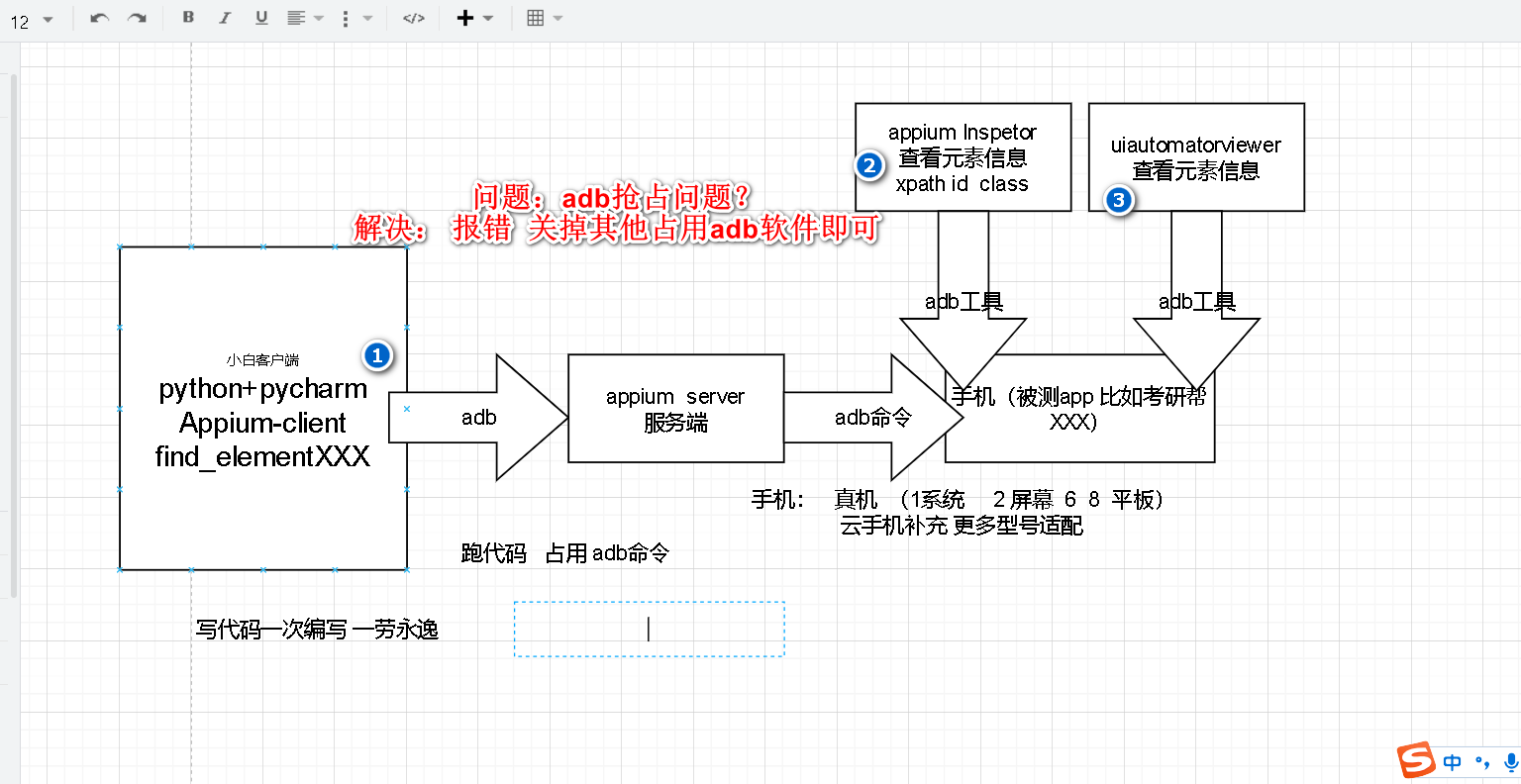
**2 UI自动化**

**3 测试框架**

**实际工作 难点：（环境）**

**手机~~**

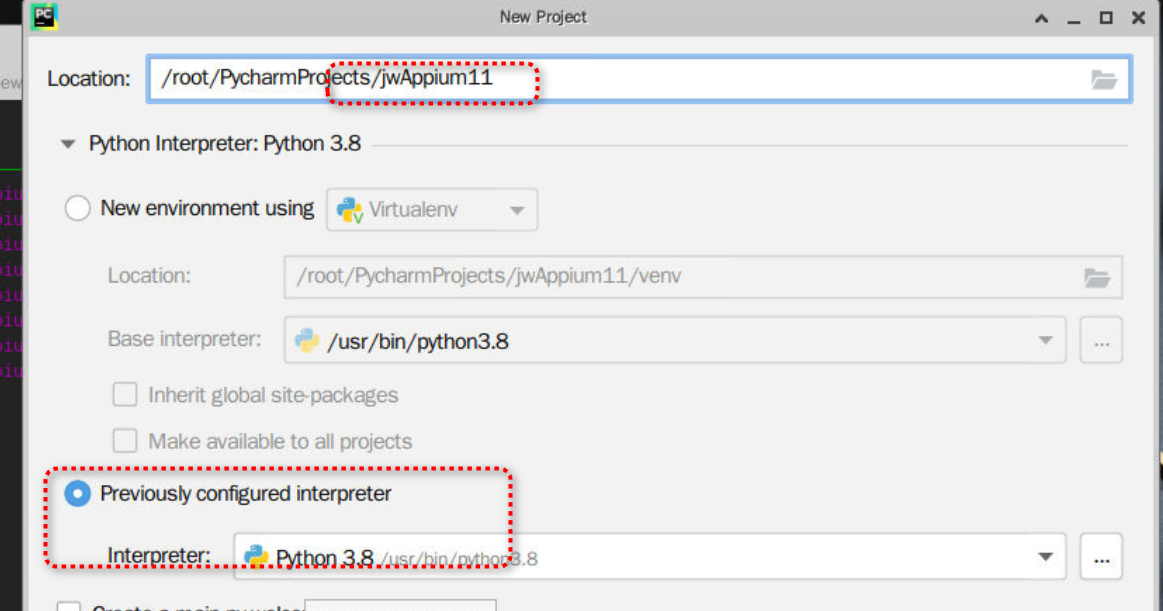
**环境架构：**

****

### 内容2：appium 快速入门 疲劳性测试

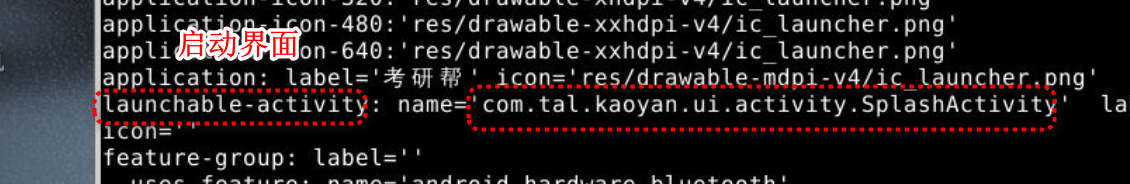
例子 python 自动化安装卸载 被测app （考研帮）3次

**准备： 手机打开 appium server pycharm**

****

**包名 aapt dump badging apk文件**

****

****

|  |
| --- |
| #1声明好 被测app 考研帮  # app 在手机唯一标识（包名package） com.tal.kaoyan  # 活动（launchable activity） com.tal.kaoyan.ui.activity.SplashActivity  # 声明好 连接 appium server  from appium import webdriver  # noReset 首次打开app 非首次app 页面流程不一样  # 指定app是否首次打开  # noReset: False 确保app首次打开～  i=1  while i<=3:  print(f"准备第{i}次")  jw\_capabilities={  "platformName": "Android",  "platformVersion": "9",  "deviceName": "127.0.0.1:10027",  "app": "/home/appium/apps/kaoyan3.1.0.apk",  "appPackage": "com.tal.kaoyan",  "appActivity": "com.tal.kaoyan.ui.activity.SplashActivity",  "noReset":False  }  # 参数1： appium server地址  # 参数2： 字典 capabilities 包含 被测试手机 包名活动 apk 文件地址  jwdriver=webdriver.Remote("http://localhost:4723/wd/hub",jw\_capabilities)  #2 安装app +卸载app  # 判断 api is\_app\_installed（包名） 判断手机包  # remove\_app 卸载app  if jwdriver.is\_app\_installed("com.tal.kaoyan") == True:  print("安装成功")  else:  print("安装失败")  print("准备卸载app")  jwdriver.remove\_app("com.tal.kaoyan")  print("卸载完毕")  i=i+1  #3 循环 重复执行 |

**小结： capabilities 中键 重点**

|  |
| --- |
| * 常用的配置项   platformName：手机操作系统   * + 获取方法：可在设备中点击‘设置’ - ‘关于平板电脑’中查看   platformVersion：手机操作系统版本   * + 获取方法：可在设备中点击‘设置’ - ‘关于平板电脑’中查看   deviceName：手机或模拟器类型   * + 获取方法：使用adb devices命令查看   app：安装包的绝对路径  appPackage：运行的android包名  appActivity：要启动的android activity   * + 获取方法1：使用 ‘aapt dump badging 被测试的app’ 命令查看package的name值   + 获取方法2：查看当前运行的app应用     - windows下使用命令：   adb shell dumpsys window | findstr mCurrentFocus   * + - linux下使用命令：   adb shell dumpsys window | grep mCurrentFocus  noReset：在当前session下不会重置应用的状态，默认为false   * 参考文档(官网)：<http://appium.io/docs/en/writing-running-appium/caps/> |

**小结： 代码 整理API （工具 对象 对接方法）**

**娶对象**

from XXX import 对象 基于对象 有对象使用对象~

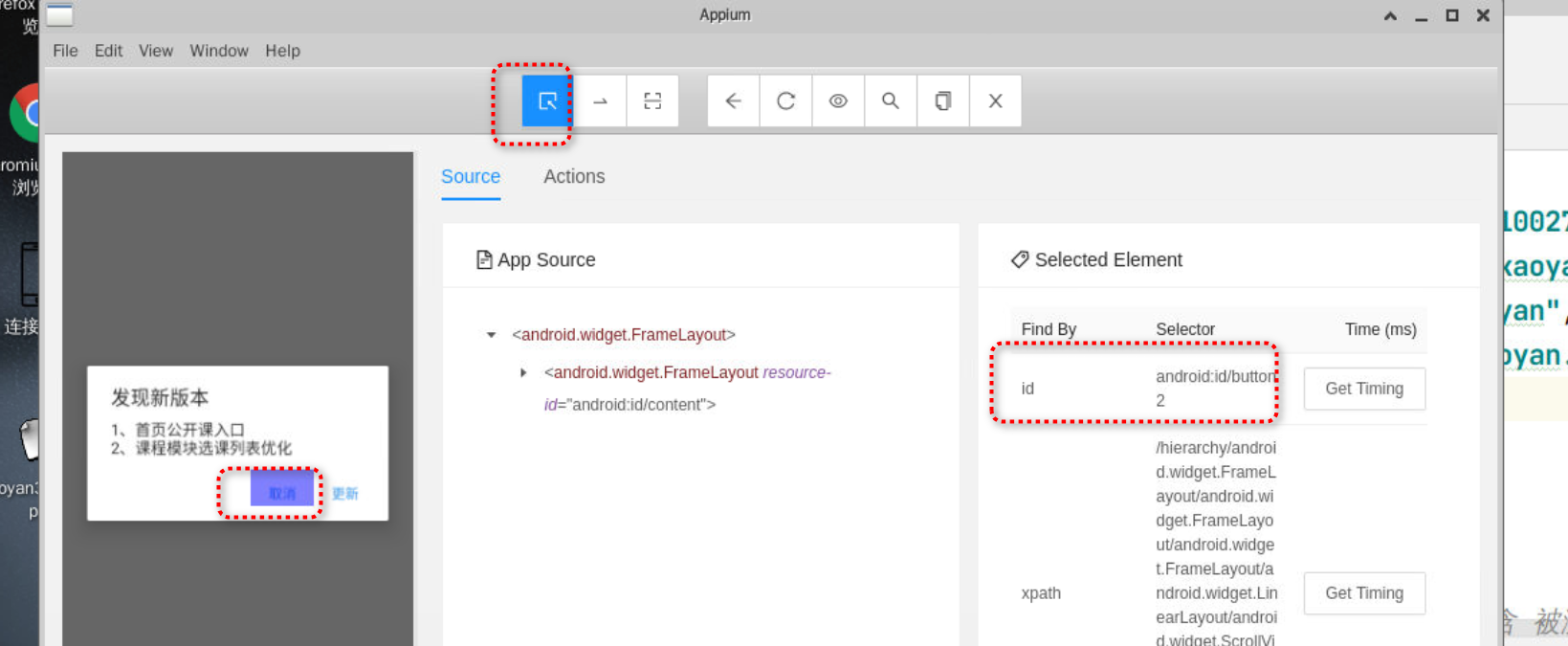
|  |  |
| --- | --- |
| **对象名字** | **对象功能方法** |
| **webdriver** | Remote("http://localhost:4723/wd/hub",capabilities)  # 参数1： appium server地址  # 参数2： 字典 capabilities 包含 被测试手机 包名活动 apk 文件地址  {  "platformName": "平台名称",  "platformVersion": "平台版本",  "deviceName": "设备名称",  "app": apk包路径",  "appPackage": "包名",  "appActivity": "活动名",  "noReset":是否全新方式打开  }  补充 包名  aapt dump badging 包名活动名  adb devices 设备名称 |
|  | is\_app\_installed("包名") 判断包存在  remove\_app("包名") 卸载包 |

**对着笔记当做工作手册 写一边抄一边改**

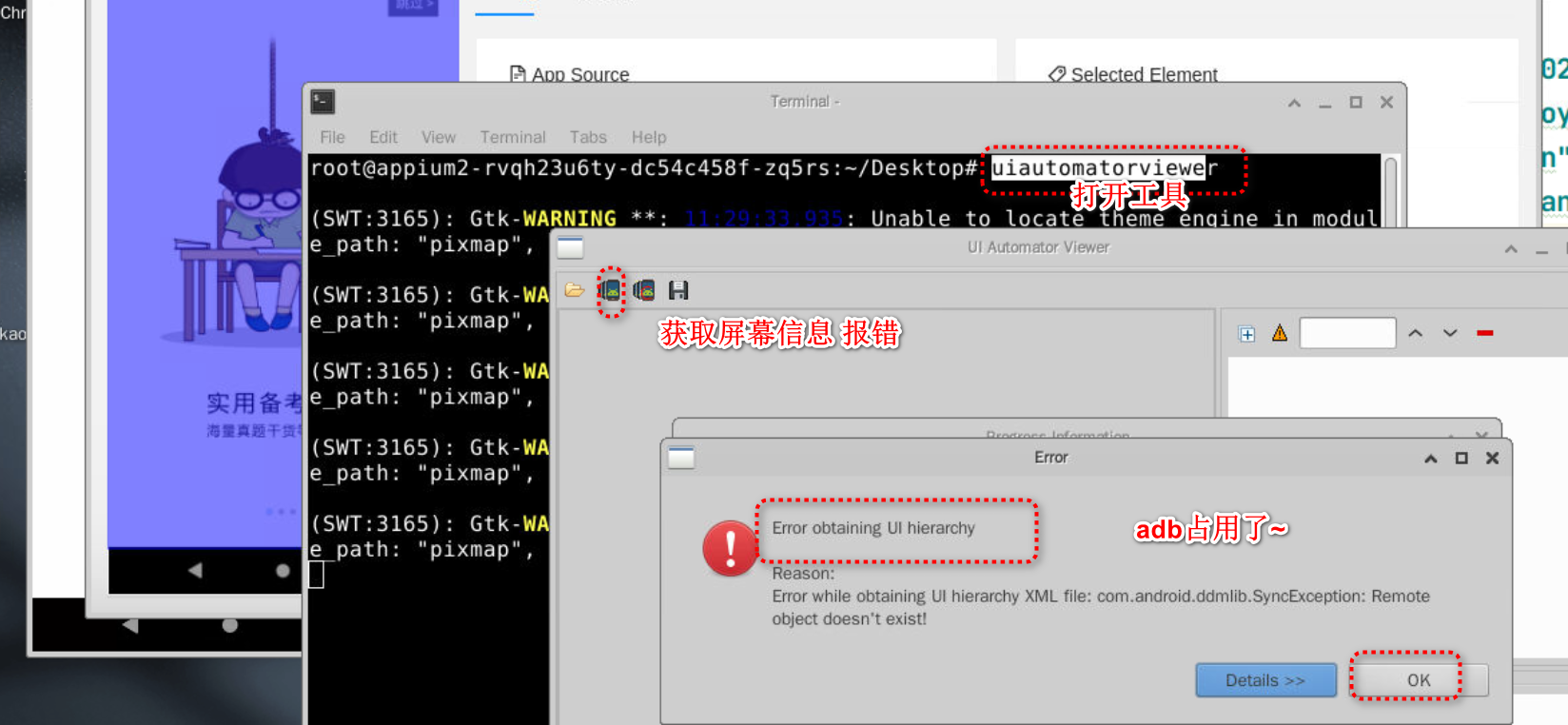
### 内容3： 查看元素定位信息

**需求 手机 不能浏览器 通过开发者工具检查？**

**解决 appium 自带工具 Inspetor工具（为主） ===》由外往内看~**

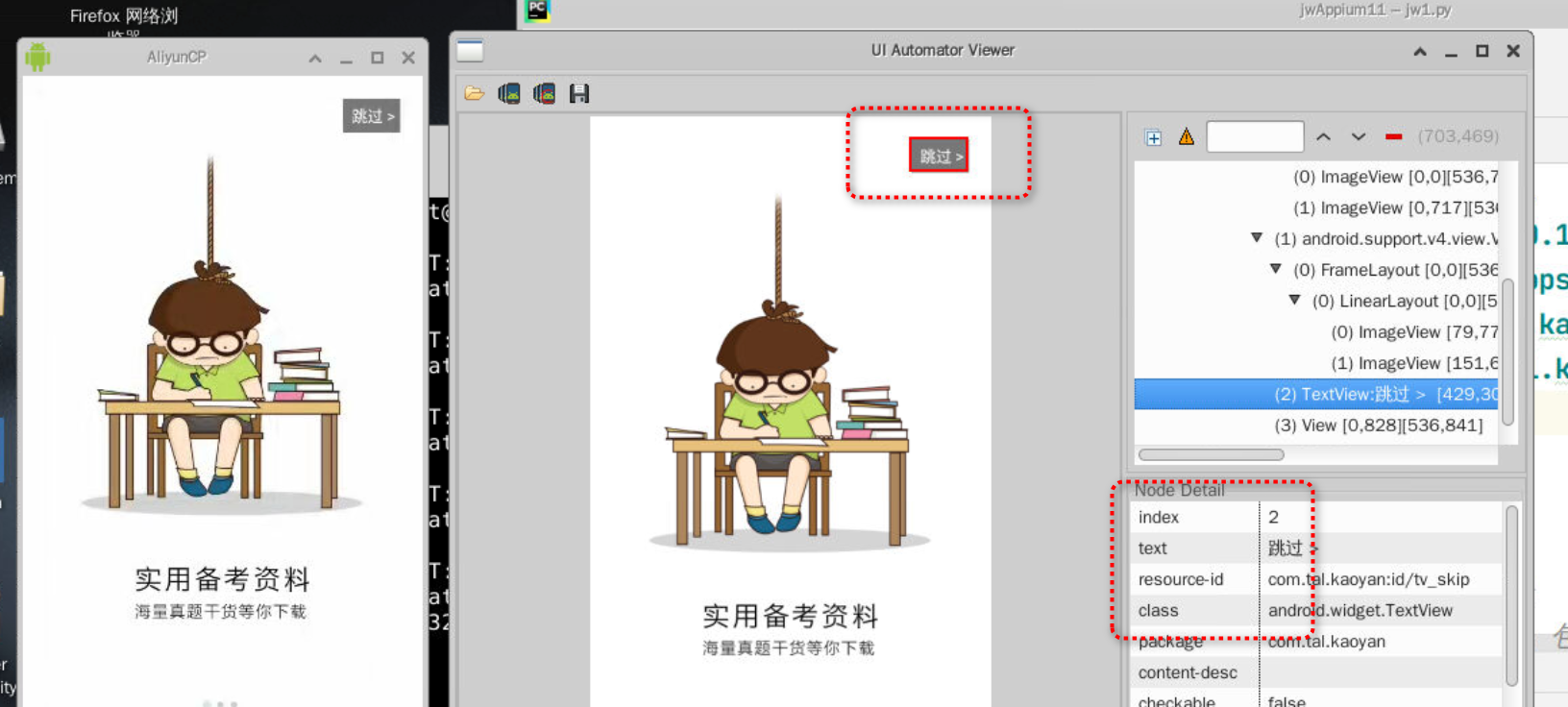
****

**uiautomatorviewer工具（为辅助）===》由内往外看~**

****

**解决：**

**appium Inspetor 停掉 +手机重开 一下~**

****

**下午UI自动化~**