**1、项目介绍**

本项目是在开源 [uiautomator2](https://github.com/openatx/uiautomator2) 和 [facebook-wda](https://github.com/openatx/facebook-wda) 项目的基础之上，进行了统一的封装。

项目采用业内主流的pytest+allure作为用例运行的基本脚手架和测试报告生成框架

脚本用例采用经典PO模式+业务流程封装+数据分离的思想进行组织和编写。

本项目编写的初衷是为了尽可能的降低UI自动化脚本的编写和使用门槛，着重在元素定位、异常和断言处理上进行了更进一步的策略优化和封装，使得测试人员可以聚焦在元素基本操作的编写和业务流程的逻辑组织上，而无需刻意关注随机的弹窗遮挡或点击按钮无法跳转下一页面的异常toast报错处理等操作，减少流程中过多的自定义断言处理，提高脚本稳定性和复用性(多平台复用、流程复用)，降低维护成本，加快脚本编写速度。

**2、功能介绍**

本项目主要实现Phone + Glass，单独UI自动化测试、互联UI自动化测试框架

实现了全流程的全自动化冒烟

**3、流程介绍**

**3.1、jenkins触发机制**

**功能：通过另外的一个jenkins任务（检测三个不同代码仓的每日固件编译是否完成），作为触发机制**

检测Phone固件每日版本是否构建完成

检测Glass固件每日版本是否构建完成

检测App固件每日版本是否构建完成

**均构建完成后触发我们的jenkins任务**

**3.2、jenkins脚本功能**

**检查当天固件地址文件数量（默认值保留15天，超过15个，即先删除一个再进行下载）**

下载Phone当天固件到特定服务器地址

下载Glass当天固件到特定服务器地址

下载App当天固件到特定服务器地址

**下载完成后调用项目入口程序（run.py）**

**4、框架自动化冒烟流程**

run.py程序会先拉取最新的项目代码

调用刷机流程代码

先获取所有连接的设备

通过serial号和model号判断是属于Phone还是Glass

先进行Phone刷机**（使用研发提供刷机教程：修改对应的shell脚本，提供参数输入：固件地址和刷机方式）**

刷机完成后，**默认跳过开机向导（setprop后重启），设备root**

安装互联App**（指定服务器地址固件进行安装）**

Glass刷机**（使用研发提供刷机教程：修改对应的shell脚本，提供参数输入：固件地址和刷机方式）**

刷机完成后，**默认跳过开机向导（研发提供prop指令），设备root**

打开相关日志

打开常规日志

打开WIFI日志：从设备拿出对应的WIFI配置文件后，更改对应字段**wifi\_verbose\_logging\_enabled为true，然后pull进设备**

data/misc/apexdata/com.android.wifi/WifiConfigStore.xml > WifiConfigStore.xml

打开Bluetooth日志：

os.system("adb -s %s shell setprop persist.bluetooth.btsnooplogmode full" % serial)

time.sleep(3)

os.system("adb -s %s shell svc bluetooth disable" % serial)

time.sleep(3)

os.system("adb -s %s shell svc bluetooth enable" % serial)

手机App跳过相关配置

**使用对应命令启动App（不登录账号） / 通过UI方式进行登录（需电话卡）**

登录进去后通过U2的watcher 监控对应字段进行跳过

调用互联代码进行互联操作

Glass触发互联

先判断眼睛是否处于亮屏状态

判断UI是否在触发互联界面，模拟键值触发单击指令（触发互联）

判断页面元素是否到了已发送互联指令，等待回连界面

Phone在App页面，判断手机端是否出现互联弹窗

实时监控弹窗，出现弹窗，UI识别勾选框，点击勾选，点击确认

判断Glass端是否出现连接中UI显示，等待连接成功

失败重连机制

Phone触发互联

Phone在App页面，UI判断点击互联按钮，点击后UI识别是否点击成功

Glass判断是否亮屏且蓝牙是否打开

Glass端判断是否出现互联弹窗，出现即确认

失败重连机制

**5、框架介绍**

glass\_ui:

common:

aircv：图像识别部分库

adb\_tool.py：adb部分接口继承

base\_page.py：封装了核心功能find\_element：包含Android和iOS部分

config\_parser.py：配置文件解析接口

myLogger.py：自己封装的日志接口

yaml\_parser.py：yaml配置文件解析接口

utils.py：工具类接口

install:

wda库

uiautomator2库

conftest.py：pytest框架的夹具函数和处理函数

config.ini: 总配置文件，main函数执行那个模块测试及异常日志检测包含哪些字段

main.py: 入口函数，通过run.py调用，解析配置文件，开始执行测试

pytest.ini: pytest配置文件，包含测试执行用例和失败重试机制以及测试次数；包含自己注册的用例标记

Smoke:

case：

用例层，用例主入口

data：

数据层，一些模块公用数据

element：

元素层，每个页面的对应元素

flow：

逻辑层：一些步骤的基本流程

image：

需要用到图像识别时的图像

page：

页面操作层，进行对应的页面操作

config.ini:

一些通用配置

指定设备号和需要执行的包名

一些配置的跳过弹窗

conftest.py：

pytest框架的夹具函数和处理函数

处理一些前置后置操作

处理一些报告结果收集和处理方式

@pytest.hookimpl(hookwrapper=True, tryfirst=True)

def pytest\_runtest\_makereport(item, call)

通过自定义结果收集函数，将冒烟测试结果输出到对应的txt文本中

**配置了allure，将输出结论通过allure服务的形式发送到邮件和对应飞书webhook接口，用户可点击邮件查看具体测试细节，那些用例执行失败，失败时候的相关日志和截图信息**