## 项目概述

GT UI Test 是一个面向 Gumtree App (iOS 和 Android) 应用程序的 UI 自动化测试项目。项目集成了持续集成能力，可自动监控代码仓库变更并触发测试执行。

## 项目目录结构

GT\_ui\_test/

├── cases/ # 测试用例目录

├── pages/ # 页面对象模型

├── utils/ # 工具类

├── pics/ # 测试图片资源

├── data/ # 测试数据

├── report/ # 测试报告

├── main.py # 测试入口

└── airtest\_trigger\_server.py # CI 触发器

## 技术栈

### 核心框架

Python 3.x

Pytest - 测试框架

Airtest - UI 自动化框架

Allure - 测试报告框架

### 主要依赖

facebook-wda (1.4.9) - iOS 测试支持

opencv-contrib-python (4.6.0.66) - 图像识别

pocoui (1.0.94) - UI 自动化框架

tidevice (0.12.10) - iOS 设备管理

requests (2.32.3) - HTTP 请求

## 功能特性

### 1. 自动化测试框架

支持 iOS 和 Android 平台

页面对象模型设计

用例失败自动重试机制

详细测试报告生成

### 2. 持续集成 (CI)

GitHub 仓库监控

代码变更自动触发测试

多仓库并行监控

测试结果邮件通知

### 3. 测试报告

集成 Allure 报告框架

测试执行记录保存

失败用例截图

测试数据统计

## 实现步骤

### 1. 环境配置

请参见公司内部文档：

[58appUI自动化方案](https://docs.58corp.com/#/space/1821815927939854338)

### 2. 测试用例编写

@allure.feature('SRP')

class TestSRP:

def setup\_method(self):

assert\_record\_new\_line('SRP页面')

case\_setup()

sleep(10)

open\_Home()

sleep(2)

​

def teardown\_method(self):

case\_teardown()

​

@allure.story('搜索落地页检查')

def test\_click\_search(self):

assert\_record\_new\_line('点击搜索按钮')

find\_element\_and\_click\_by\_list([get\_pic\_element('tpl1750325083400.png')])

sleep(2)

text('iPhone')

sleep(5)

assert\_exist\_by\_text\_list(['Filter', 'Image'])

​

### 3. 执行测试

# 执行所有测试

python main.py

# 执行指定测试用例

pytest ./cases/test\_example.py -vs --alluredir=report

### 4. CI 服务配置

1.配置 GitHub Token

2.设置监控仓库信息

3.启动监控服务

**python airtest\_trigger\_server.py**

### 5. 项目配置

config.ini 示例:

android device\_id = 3AQ0225114003905 case\_path = ./cases/test\_Sell.py

### 6. CI 配置

import requests

import time

import subprocess

import os

from datetime import datetime

​

# 每分钟轮询一次

INTERVAL = 60

​

# 监听多个仓库配置

REPOS = [

{

"name": "gumtree-tech/ios-app",

"branch": "main",

"last\_commit\_file": "last\_commit\_ios.txt",

"script": "/Users/a58/PycharmProjectsHamony/GT\_ui\_test/main.py"

},

{

"name": "gumtree-tech/android-app",

"branch": "main",

"last\_commit\_file": "last\_commit\_android.txt",

"script": "/Users/a58/PycharmProjectsHamony/GT\_ui\_test/main.py"

}

]

​

# GitHub token：请替换成你自己的安全 token

GITHUB\_TOKEN = "ghp\_u8cC3GmhvCK2bnYNNdblzbrlxpCiZK2sn9Zg"

​

​

def log(msg):

print(f"[{datetime.now().strftime('%Y-%m-%d %H:%M:%S')}] {msg}")

​

​

def get\_latest\_commit\_sha(repo\_name, branch):

url = f"https://api.github.com/repos/{repo\_name}/commits/{branch}"

headers = {

"Accept": "application/vnd.github.v3+json",

"Authorization": f"token {GITHUB\_TOKEN}"

}

try:

r = requests.get(url, headers=headers)

r.raise\_for\_status()

return r.json()['sha']

except Exception as e:

log(f"❌ 获取 {repo\_name} 最新 commit 失败: {e}")

return None

​

​

def read\_last\_commit\_sha(path):

if os.path.exists(path):

with open(path, "r") as f:

return f.read().strip()

return None

​

​

def write\_last\_commit\_sha(path, sha):

with open(path, "w") as f:

f.write(sha)

​

​

def trigger\_airtest(script\_path, repo\_name):

log(f"🚀 检测到 {repo\_name} 有新提交，启动对应 Airtest 脚本...")

try:

subprocess.run(["python3", script\_path], check=True)

log(f"✅ {repo\_name} Airtest 脚本执行完成")

except Exception as e:

log(f"❗ {repo\_name} Airtest 执行失败: {e}")

​

​

def main():

log("🛰️ 启动多仓库监听服务...")

while True:

for repo in REPOS:

repo\_name = repo["name"]

branch = repo["branch"]

last\_sha\_path = repo["last\_commit\_file"]

script\_path = repo["script"]

​

latest\_sha = get\_latest\_commit\_sha(repo\_name, branch)

if latest\_sha is None:

continue

​

last\_sha = read\_last\_commit\_sha(last\_sha\_path)

​

if latest\_sha != last\_sha:

log(f"📦 {repo\_name} 检测到新提交: {latest\_sha}")

trigger\_airtest(script\_path, repo\_name)

write\_last\_commit\_sha(last\_sha\_path, latest\_sha)

else:

log(f"💤 {repo\_name} 无新提交，等待下次检测...")

​

time.sleep(INTERVAL)

​

​

if \_\_name\_\_ == "\_\_main\_\_":

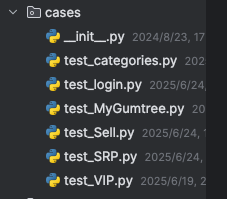
main()

​

## 案例

### 1. 测试用例组织

按功能模块分类



保持用例独立性

### 2. 页面对象设计

封装页面元素、提供业务方法、维护元素定位器

from utils.app\_utils import \*

​

​

def open\_MyGumtree():

wait\_for\_text('My Gumtree')

find\_element\_and\_click\_by\_text\_list(["My Gumtree"])

​

​

def open\_Messages():

wait\_for\_text('Messages')

find\_element\_and\_click\_by\_text\_list(["Messages"])

​

​

def open\_Sell():

wait\_for\_text('Sell')

find\_element\_and\_click\_by\_text\_list(["Sell"])

​

​

def open\_Saved():

wait\_for\_text('Saved')

find\_element\_and\_click\_by\_text\_list(["Saved"])

​

​

def open\_Home():

sleep(2)

find\_element\_and\_click\_by\_list([get\_pic\_element('tpl1750326185853.png')])

###### 常见问题

1. 设备连接问题：检查 USB 连接、验证设备授权状态

2. 用例稳定性：添加适当等待时间、使用显式等待、优化元素定位策略

3. CI 集成问题、验证 GitHub Token 权限、检查网络连接、确认脚本路径配置