目录

[准备工作 2](#_Toc536112517)

[ATF初始设置 4](#_Toc536112518)

[ATF—ATF 处理Windows 网页的测试 6](#_Toc536112519)

[ATF—AIR 处理Android 手机的测试 8](#_Toc536112520)

[ATF—DB 处理 SQL Server 数据库的测试 10](#_Toc536112521)

[Test 文件讲解 11](#_Toc536112522)

[Xpath 12](#_Toc536112523)

# 准备工作

1. 安装python 3.6+，

方法一（推荐）

* 1. 下载安装环境管理工具

https://repo.anaconda.com/miniconda/Miniconda3-latest-Windows-x86\_64.exe

1.2. 通过miniconda创建环境

a. Cmd输入命令conda create -n testenv python=3.6.6

b. 输入y

c. 激活环境 activate testenv

（**\*\***如果使用cmd方式运行脚本，每次新开cmd窗口，都需要重复此步骤）

d. 查看安装路径： conda info

e. 将环境的地址加到环境变量的path中以;间隔

方法二

1. 官网下载python安装包（\*\*\*重要\*\*\*python版本需要为3.6.6）

<https://www.python.org/downloads/>

1. 为python添加环境变量，

将python 和python/scripts的地址分别加到path中以;间隔

打开cmd窗口，输入python检查是否安装成功

1. 下载Android studio 并将地址配置到环境变量中
2. 安装手机USB驱动
3. 安装Airtest IDE

http://airtest.netease.com

1. 下载关联包（安装后不要升级依赖包避免出现不兼容）



1. 打开cmd窗口
2. cd到requirement.txt 的文件夹
3. 输入命令 pip install -r requirement.txt（如果安装了miniconda，需要激活环境后再输入此命令）

Backup：如遇上述方法失败，请使用Backup方法

1. 安装selenium

pip install selenium==3.13

1. 安装 airtest

pip install Airtest==1.0.22

1. 安装pymssql

pip install pymssql

1. 安装pytest

pip install pytest==3.6.2

pip install pytest-html

1. 安装xlrd

pip install xlrd

1. 下载ATFPackage

并将ATFPackage放置在使用的python环境中

1. 以chrome浏览器为例，下载Chromedriver

<http://npm.taobao.org/mirrors/chromedriver>

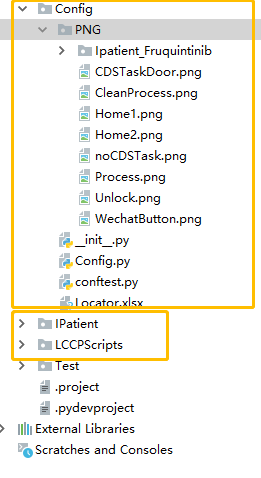
* 1. 根据chrome版本下载不同版本的chromedriver
  2. 将下载好的chromedriver 放在chrome文件夹中

i.e. C:\Program Files (x86)\Google\Chrome\Application

1. 安装编写工具（Pycharm--community，Eclipse）
   1. 配置pycharm的python环境 （适用于安装python 方法一）
2. 打开项目
3. 打开File🡪Setting🡪Project(当前项目)🡪project Interpreter🡪show all🡪点击+🡪xuanz Existing environment🡪点击… 🡪输入python环境地址（参见第一步）🡪选择python.exe
4. 选择使用的python环境

# ATF初始设置

1. 下载ATF框架，并将ATFPackage放置在python Lib目录下
2. 打开cmd
3. 输入python
4. 输入import sys
5. 输入sys.path后即可查看python lib的路径
6. 新建项目，在项目下 Package的**同级目录**下添加“Config”package------用存放相关配置及定位符



1. 在“Config”package中添加“PNG”package------用于存放ATF---AIR使用的.png文件
2. 在“Config”package中添加Locator.xlsx------用于存放ATF---ATF使用的页面元素定位符，详情参考附件



1. 在“Config”package中添加Config.py------用于存放脚本使用的各类配置，

详情参考附件



1. 在“Config”package中添加conftest.py------测试初始化操作

详情参考附件

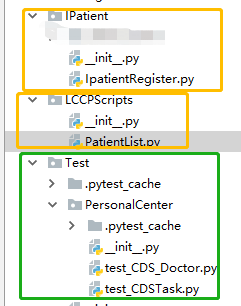


1. 在项目的中分别创建脚本文件夹和测试文件夹

如下：

Ipatient ，LCCPScripts为2个项目的测试用例抽离的基础方法文件

Test文件夹为执行测试用例的测试文件，为最终运行测试的文件夹



在test文件夹中放置 runner.py,即可执行Test文件夹中所有的test\_\*.py文件

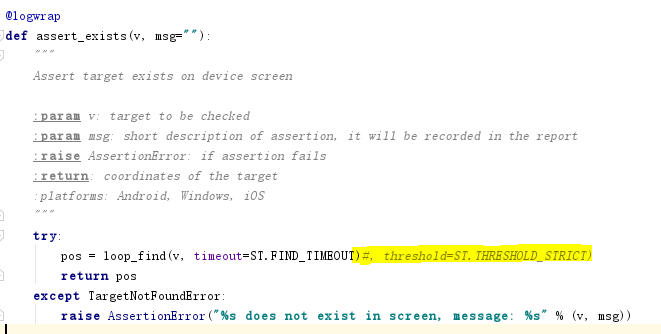
详情参考附件



1. 修改AIRTest 框架源码

修改 \*\*\*\Lib\site-packages\airtest\core 文件夹下api.py 文件

assert\_exists 方法



# ATF—ATF 处理Windows 网页的测试

1. 引入ATF包中ATF类

**from** ATFPackage.ATF **import** ATF

1. 继承ATF类

**class** LCCP(ATF):

1. 定义init方法

**def** \_\_init\_\_(self,browser):

ATF.\_\_init\_\_(self,Browser=browser,Filename=**"Locator.xlsx"**,Sheet\_name=**"Sheet1"**)

其中,Browser, Filename, Sheet\_name 为必须参数，

Filename为具体的locator的文件名

Sheet\_name为定位参数所在xlsx 中的sheet名

1. 定义测试用例方法，使用ATF中提供方法编写测试用例



1. 方法详解
2. ATF.GoToUrl(此处填写具体的url)
3. ATF.MaxWindows() 窗口最大化
4. ATF.ClickElement(此处填写locator.xlsx中的locator)
5. ATF.SendKeys(此处填写locator.xlsx中的locator, 此处为需要填写的内容)
6. ATF.CheckText(此处填写locator.xlsx中的locator, 此处为需要校验的内容)

此方法默认校验text 熟悉的值，

如果需要校验其他属性值，可以添加attr=’具体的属性名’

1. ATF.AssertExist(此处填写locator.xlsx中的locator), 校验某个元素是否存在
2. 实例详见下图

# ATF—AIR 处理Android 手机的测试

1. 引入ATF包中AIR类

**from** ATFPackage.ATF **import** AIR

1. 继承ATF类

**class** Register(AIR):

1. 定义init方法

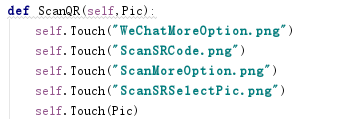
**def** \_\_init\_\_(self):

AIR.\_\_init\_\_(self,FilePath=**'D:\\Jingyuan.Gao\\workspace\\Claire\\Config\\PNG\\Ipatient\_Fruquintinib\\'**)

其中, FilePath为不是必须参数，意义是AIR方法的png图片地址

默认地址为 Config下的PNG文件夹，如非默认文件夹，则需要填写FilePath参数

1. 定义测试用例方法，使用AIR中提供方法编写测试用例



1. 方法详解
2. AIR. Connect\_Device ("android:///") 连接Android 设备
3. AIR().Stop\_App(**"com.tencent.mm"**) 关闭微信,参数为软件的包名
4. AIR().Start\_App(**"com.tencent.mm"**)打开微信,参数为软件的包名
5. AIR().Wake() 唤醒屏幕
6. AIR().Touch(此处填写png图片文件名) 点击操作
7. AIR().Assert\_Exist(此处填写png图片文件名)判断图片是否存在
8. AIR().Text(此处填写png图片文件名, 此处为需要填写的内容)输入操作

以上所有方法中如果需要均可传入以下参数：

threshold=xxx, 默认值为**None**, 意义：定义图像识别匹配成功的标准，AIR框架有默认值，可根据实际情况选择是否定义特殊标准

target\_pos= xxx, 默认值为**None**, 意义：目标位置

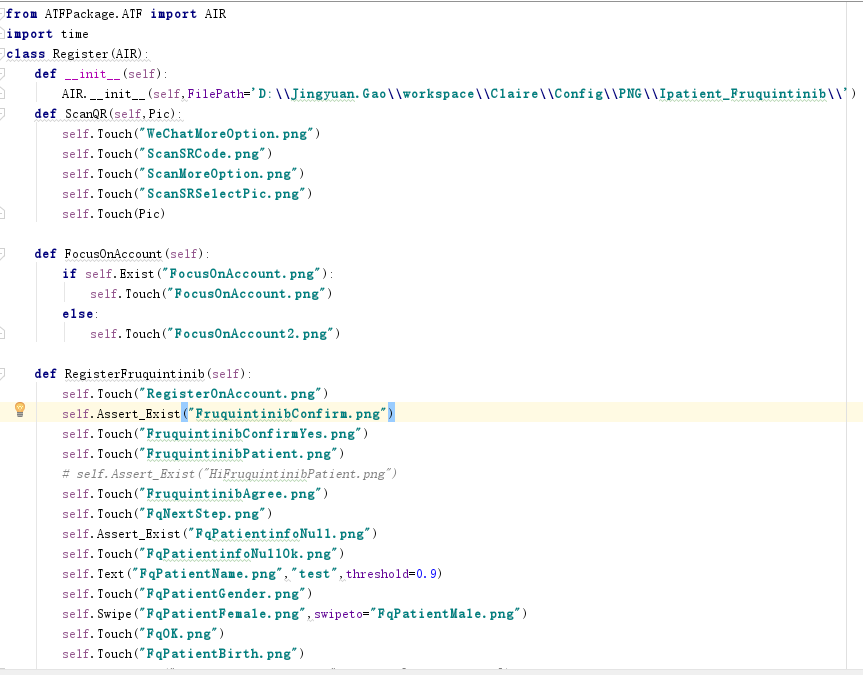
record\_pos= xxx, 默认值为**None**, 意义：录制时的相对坐标

resolution=(),默认为空，意义：分辨率

rgb= True/False, 默认值为**False,**

意义：是否去除背景色， False为保留背景色比较，True为去除背景色后比较

1. 实例详见下图



# ATF—DB 处理 SQL Server 数据库的测试

1. 引入ATF包中DB类

**from** ATFPackage.ATF **import** DB

1. 实例化DB类

DB\_Config={  
 **'host'**:**'192.168.\*\*\*.\*\*\*'**,  
 **'user'**:**'\*\*'**,  
 **'password'**:**'\*\*'** }

DBLCCP=DB(DB\_Config)

1. 方法详解
2. 打印查询结果

print(DBLCCP.SELECT(此处填写查询的SQl语句))

\*SQL语句必须要引号扩起

1. 查询结果比对， 校验**目标文字**是否在查询结果中存在

DBLCCP.CheckValue(此处填写查询的SQl语句,此处填写**目标文字**)

\*SQL语句必须要引号扩起，目标文字需要引号扩起

1. 查询数目比对， 校验**目标数目**是否在查询结果中存在

DBLCCP.CheckNumber (此处填写查询的SQl语句,此处填写**目标数目**)

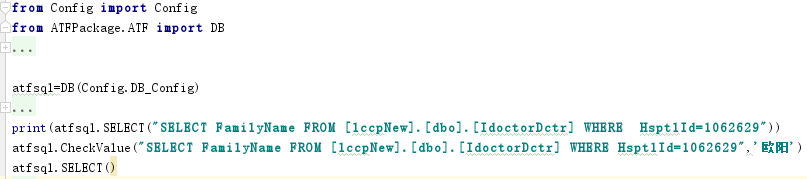
\*SQL语句必须要引号扩起，目标数目不需要加引号

1. 数据库update操作

DBLCCP. Modify(此处填写update SQl语句)

\*SQL语句必须要引号扩起

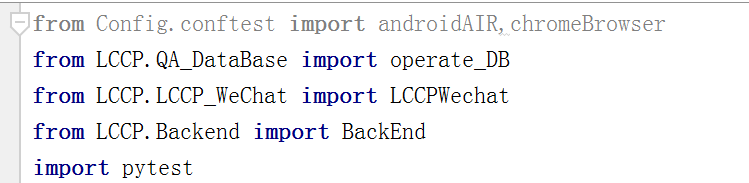
1. 实例详见下图



# Test 文件讲解

1. 再完成以上基础类之后，引入基础类

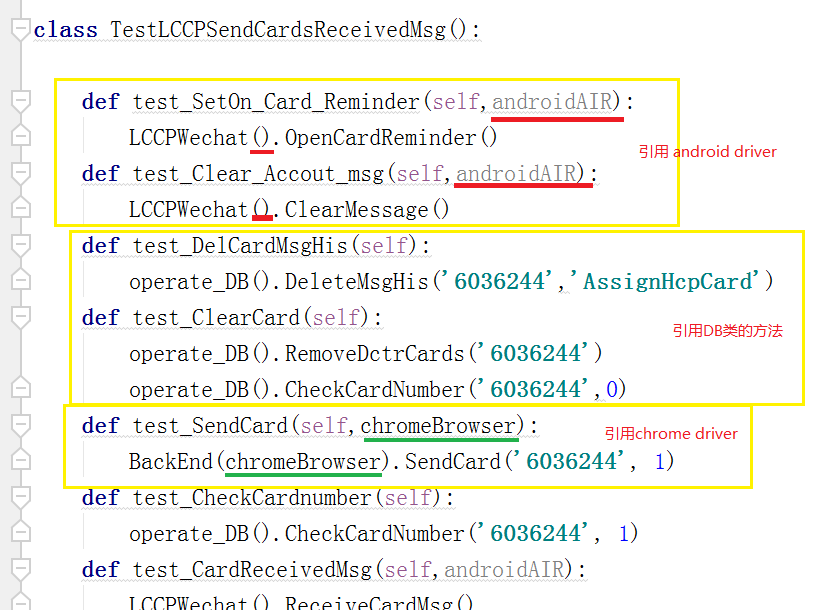
**from** ATFPackage.ATF **import** DB



1. 实例化基础类中的方法实现测试，代码如下

\*文件名需要test\_开头，class 名需要Test开头，方法名需要以test\_开头

详见下图实例，可以在同一个class下实现android，web以及DB的测试



# Xpath

1. 辅助工具

Chrome浏览器，F12 🡪 Console， 输入$x(“xpath”)

1. 通过html 节点 定位元素

//\*[@id="fmLogin"]/fieldset/label[1]/span/input

1. 通过text定位

.//\*[text()='登录']

.//\*[contains(text() , '登录')]

1. 通过元素属性定位

.//\*[@placeholder='用户名']

1. 当页面存在多个相同元素id或者name时

.//\*[@id='Login'][@name='Login']