# 概述

UI自动化自动化，当然也包括appium， 主要就是干三种事情：

1选择 界面 元素

2操作元素 （包括输入操作：点击、输入字符、拖拽等。 输出操作：获取页面元素的各种属性）

3根据Appium获取的数据进行分析和处理。

对于使用 appium进行的移动app 自动化来说，

前面两个我们要依靠Appium 提供的能力，这是我们前面没有学习过的。是appium课程的核心要学的东西

第3 个则是依赖于我们前面学过的python语言的知识。这本质上是看大家的编程基础能力怎样。也是确实我们整个自动化课程的的重点。

=============

我们先回顾一下上次的示例代码, 上次我们只是大概的过了一遍。 具体的每行代码的细节原理，没给大家讲。现在我们在和大家仔细的来分析一下。

大家注意注释讲解

# 代码解释

from appium import webdriver

import time,traceback  
  
  
*# 这里定义的 desired\_capabilities，作为下面 webdriver.Remote  
# 初始化一个webdriver的参数。*

*# 这些键值对告诉appium server 测试程序希望进行的是什么样的测试*

desired\_caps **=** {}

*#测试平台,不能写错*  
desired\_caps['platformName'] **=** 'Android'

*#平台版本,不能写错*desired\_caps['platformVersion'] **=** '7'

*#设备名称，android上随便填写一个值*desired\_caps['deviceName'] **=** 'test'

*#apk 文件路径名，如果设备还没有此应用，则会安装。*

desired\_caps['app'] **=** r'd:\apk\toutiao.apk'

*#app package名，一定要有，是开发者给app取的名字，可以唯一标识这个app。*desired\_caps['appPackage'] **=** 'io.manong.developerdaily'

*#app启动Activity，也是必须的参数。安卓应用一个Activity 就对应  
一个用户操作界面*desired\_caps['appActivity'] **=** 'io.toutiao.android.ui.activity.LaunchActivity'

*# 如果我们自动化，要输入的信息里面有中文字符，需要这个参数，否则可以不要  
#加上这个参数会新加一种unicode输入法*

desired\_caps['unicodeKeyboard'] **=** True

*# 表示测试完后，输入法还要还原成原来的，比如原来是搜狗输入法*

desired\_caps['resetKeyboard'] **=** True

*# 这个noReset参数，非常重要！！*

*# 保证了app 测试前不会清除数据，缺省是会清除数据的*desired\_caps['noReset'] **=** True

*# 我们的测试程序连接上appium server后，appium server 一直等待client 的命令。*

*如果超过一定的时间，没有命令，比如debug的时候，它就会认为客户端已经退出了。就会断开连接*

*这个参数就是指定多长时间，appium server可以认为连接已经断开了*desired\_caps['newCommandTimeout'] **=** 6000

*#启动Remote RPC，将 capability参数传入，*

*创建一个 webdriver对象， 和Selenium 类似。*

driver **=** webdriver.Remote('http://localhost:4723/wd/hub', desired\_caps)  
  
try**:** *# 和Selenium含义一样* driver.implicitly\_wait(10)

*# 下面的这些代码，就是根据id找到元素，并操作，比如点击、输入文本， 或者获取文本，判断是否和预期相同。*

driver.find\_element\_by\_id(

"io.manong.developerdaily:id/tab\_bar\_plus").click()  
 time.sleep(1)  
 driver.find\_element\_by\_id(

"io.manong.developerdaily:id/btn\_email").click()  
 time.sleep(1)  
  
 *# 输入用户名、密码* ele **=** driver.find\_element\_by\_id("io.manong.developerdaily:id/edt\_email")  
 ele.send\_keys('jcyrss@163.com')  
 ele **=** driver.find\_element\_by\_id("io.manong.developerdaily:id/edt\_password")  
 ele.send\_keys('sdfsdf')  
  
 time.sleep(2)  
 *# 点击登录* driver.find\_element\_by\_id(

'io.manong.developerdaily:id/btn\_login').click()  
   
except**:** print traceback.format\_exc()  
  
  
input('\*\*\*\* Press to quit..')  
driver.quit()

# 查看appPackage 和appActivity （5min）

## 方法1

把api保存在电脑上

e:\androidsdk\build-tools\28.0.2\aapt.exe dump badging d:\apk\toutiao.apk

## 方法2

如果你apk已经安装在手机上了，可以直接打开手机上该应用，进入到你要操作的界面

然后执行

adb shell dumpsys activity recents | find "intent={" > d:\info.txt && d:\info.txt

下面就是最近的N 个 activity 信息，

看第一个

其中高亮部分就是我们要的

intent={act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] flg=0x10200000 cmp=com.android.mms/.ui.ConversationList}

adb shell dumpsys activity recents | find "Activities=[" > d:\info.txt && d:\info.txt

=======================

Appium 自动化和Selenium自动化都是界面的自动化操作，所以其原理和操作方式非常相似。

我们要操作移动app界面元素，首先需要 找到app界面元素，返回对应这个元素的对象。随后我们的代码才可以通过这个web元素的对象 对其 进行操作。

问题是移动app怎么找到元素呢？

# 查看界面元素工具uiautomatorviewer （25 min）

## 打开 4min

安卓查看界面元素，最常用的就是 Android SDK 中的 uiautomateviewer，它在目录

.\Android\sdk\tools\bin\

## 功能介绍 20min

现在我们就可以开始查看app界面（打开开发者头条，界面），

点击这个按钮可以看到界面图像显示在左边，

同时右边是界面的Layout布局的树形图的展示。有各种分支，从根看起，就像一颗树

uiautomatorviewer 查看的是手机当前界面的元素，我们可以点击手机的界面，进入到不同的界面，查看我们要查看界面的元素。

进入到一个新界面的时候，需要再次点击按钮。 和web 不同，它不能自动刷新，

不能实时跟踪界面的变化。需要我们手动点击重新获取当前界面

这个两个按钮的区别，右边是compressed（看tooltip），也就是精简了的界面。

我们要查找某个元素，也是这样把 鼠标移动到某个元素上，某个元素就有虚线红框子，同时右边会高亮当前的元素，

如果确定了要查看的元素，就单击一下该元素， 虚线变成实线， 就是确定了查看这个元素的信息。 同时，右边下面就显示出该元素的具体的属性信息。

-----------------------------

元素的属性：

index：就是该元素是其父元素的第几个节点，从0开始

text：就是如果该元素对应的界面上有文本内容，就是其文本内容的字符串值，

显然这是我们将来非常关心的属性，因为要分析内容，判断是否与预期相同。

比如，最便宜的商品列表

resource-id： 是该元素的资源id，一般来说唯一的决定了一个元素，

有点像web的id，优先根据它查找。

但是并非每一个元素都会有。如果这样，就需要其他的方法选择

class ：就是该元素的类型，这是安卓界面元素对应的代码里面的java类，类似html 的tag名，比如layout，textview，

package：就是 所在app的包名

content-desc: 这个属性用来描述该元素的用途，特性等。

bounds ： 就是这个元素在界面上的位置， 左上角和右下角的坐标。

以后如果我们黔驴技穷，实在选择不到元素的时候，就根据这个属性直接去根据坐标操作元素了。

# 选择元素的代码 （3 min）

选择元素的代码也和Selenium基本相同，如下

可以通过

find\_element\_by\_XXX 符合条件的第一个元素，找不到抛出异常

find\_elements\_by\_XXX 符合条件的所有元素的列表，找不到返回空列表

通过WebDriver对象调用这样的方法，查找范围是整个界面树形结构

通过WebElement对象调用这样的方法，查找范围是该节点的子节点 ，

# 选择元素方式

## 7 ID （10 min）

比如：

ele **=** driver.find\_element\_by\_id('com.example.jcy.wvtest:id/name1')  
ele.send\_keys('hello world')

---------------------------------

但是我们光靠id定位也远远是不够的。

有的元素没有ID，

而且有的时候id也并非唯一，比如

打开 开发者头条，用uiautomatorviewer工具查看一下

下面的 几个按钮，id都是相同的。这应该是开发人员的不规范造成的。就像我们web中也会出现不小心有id相同的元素



如果有这种错误，我们只能通过其他方式去定位了。

我们怎么知道是唯一的？

只能通过把这个界面save button 把内容导出到文件中， 打开该文件，复制黏贴查找一下，搜索是否是唯一。

## 8 CLASS NAME （12 min）

class属性决定了界面元素的类型。

通常class 就是类似web里面的tagname， 所以通常不是唯一的。

所以大部分情况，我们根据classname 是要选择多个而不是一个。

如果你确定，我们要查找的是某种类型的界面元素，而且这种类型的界面元素在当前界面中只有一个，就可以根据它来查找

写法是：

driver.find\_element\_by\_class\_name('android.widget.DatePicker')

大家看一个例子：开发者头条里面的 底部的大加号

android.widget.ImageButton，这个在当前界面上就是一个唯一的.

我们可以根据它查找，

driver.find\_element\_by\_class\_name("android.widget.ImageButton").click()

----------------------------------

当然，我们可以根据classname查找所有匹配的元素，这时可以用

driver.find\_elements\_by\_class\_name

*# 这一步是为了防止 界面刚出来的时候，还没显示全，就去获取所有的textview了*driver.find\_element\_by\_class\_name("android.widget.ImageButton")  
  
time.sleep(2)  
tvs **=** driver.find\_elements\_by\_class\_name("android.widget.TextView")  
for tv in tvs**:** print(tv.text)

## 9 ACCESSIBILITY ID （5 min）

content-desc属性时用来描述该元素的作用的。

要查询的界面元素的 content-desc属性 在当前界面中唯一 我们可以通过它来定位。

写法

driver.find\_element\_by\_accessibility\_id('xxx')