

# 目录

前言	1.1
简介	1.2
环境搭建	1.3
核心功能	1.4
监听	1.4.1
查找元素	1.4.2
xpath	1.4.2.1
操作元素	1.4.3
点击元素	1.4.3.1
输入内容	1.4.3.2
当前屏幕	1.4.4
相关	1.5
weditor	1.5.1
adb	1.5.2
android-uiautomator-server	1.5.3
uiautomator	1.5.4
常见问题	1.6
NAF	1.6.1
源码分析	1.7
通用代码段	1.8
工具类函数	1.8.1
adb相关	1.8.2
设备相关	1.8.3
其他	1.8.4
附录	1.9
参考资料	1.9.1

# 安卓自动化测试利器：uiautomator2

- 最新版本： v1.0
- 更新时间： 20201211

## 简介

总结安卓设备自动化测试领域好用的库uiautomator2，包括简介；如何搭建环境；有哪些核心功能，比如监听、用xpath等查找元素、以及常见的操作元素，比如点击元素、输入内容等、如何获取当前屏幕的截图和xml源码；以及与u2相关的内容，比如辅助调试的weditor、adb、android-uiautomator-server、uiautomator；以及常见的一些问题，比如NAF等；以及一些源码分析；和通用代码段，包括工具类函数、adb相关、设备相关等；最后给出参考资料和文档。

## 源码+浏览+下载

本书的各种源码、在线浏览地址、多种格式文件下载如下：

### Gitbook源码

- [crifan/android\\_automation\\_uiautomator2: 安卓自动化测试利器：uiautomator2](#)

如何使用此Gitbook源码去生成发布为电子书

详见：[crifan/gitbook\\_template: demo how to use crifan gitbook template and demo](#)

### 在线浏览

- [安卓自动化测试利器：uiautomator2 book.crifan.com](#)
- [安卓自动化测试利器：uiautomator2 crifan.github.io](#)

### 离线下载阅读

- [安卓自动化测试利器：uiautomator2 PDF](#)
- [安卓自动化测试利器：uiautomator2 ePUB](#)
- [安卓自动化测试利器：uiautomator2 Mobi](#)

### 版权说明

此电子书教程的全部内容，如无特别说明，均为本人原创和整理。其中部分内容参考自网络，均已备注了出处。如有发现侵犯您版权，请通过邮箱联系我 `admin 艾特 crifan.com`，我会尽快删除。谢谢合作。

## 鸣谢

感谢我的老婆陈雪的包容理解和悉心照料，才使得我 `crifan` 有更多精力去专注技术专研和整理归纳出这些电子书和技术教程，特此鸣谢。

## 更多其他电子书

本人 `crifan` 还写了其他 100+ 本电子书教程，感兴趣可移步至：

[crifan/crifan\\_ebook\\_readme: Crifan的电子书的使用说明](#)

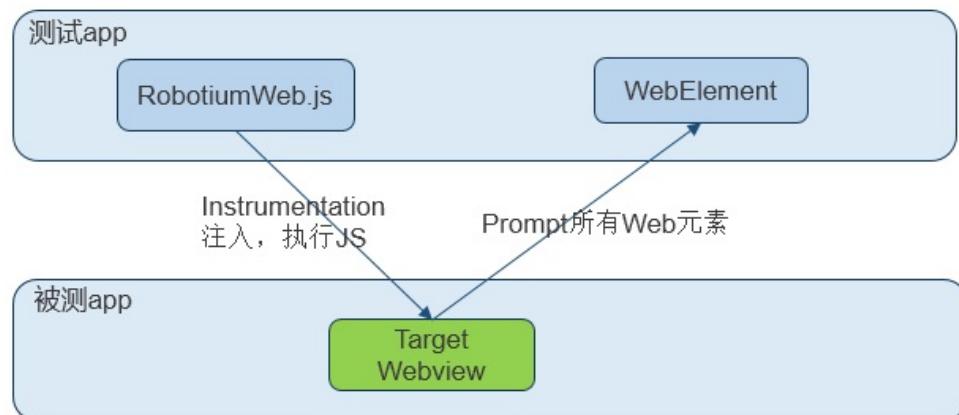
crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2021-01-16 19:56:59

## uiautomator2简介

- uiAutomator2
  - 简称: u2
  - 是什么: 使用Python对Android设备进行UI自动化的库
  - 作用: 自动化操作安卓设备, 用于测试或抓包等
  - 语言: Python
  - 主页
    - [openatx/uiAutomator2: Android UiAutomator2 Python Wrapper](#)
    - 其中 openatx 中的
      - ATX = AutomatorX
  - 竞品=其他安卓自动化测试框架
    - Robotium
    - Selendroid
    - Espresso

## 基本原理

- 背景
  - Android内置的支持测试的框架
    - Android 4.2+: UiAutomator
    - Android 2.3 ~ 4.1: Instrumentation
- uiAutomator2的原理
  - 图



- 文字
  - 采用 Instrumentation 注入被测app后, 执行 js 脚本, 提取并封装成拥有 Web 元素的文本信息、id 或 class 等属性、坐标信息等等的WebElement对象
  - 通过 js 注入的方式, 可以获取网页中的包括文字、tag标签、属性、坐标等等信息。
    - Android

- `WebChromeClient` 类在 Android 中，主要用于辅助 `WebView` 处理 js 的对话框、提示框等等

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 环境搭建

下面介绍如何搭建uiautomator2的开发环境，去测试安卓设备。

### 准备工作：安卓手机

#### 确保手机中开启了USB安装

安卓手机中开启 开发者选项 -> USB调试 -> USB安装



否则后续无法正常安装uiautomator2相关的ATX等软件和服务。

## 安装

```
pip3 install -U uiautomator2
```

- 如果包管理器是 pipenv，则用：

- ```
pipenv install uiautomator2
```

再去安装相关依赖的东西：

```
python3 -m uiautomator2 init
```

## 测试连接

再去测试连接：

```
import uiautomator2 as u2
d = u2.connect() # connect to device
print(d.info)
```

其中：

u2.connect()可以换成wifi或usb：

- wifi
  - ```
d = u2.connect('10.0.0.1')
```
- usb
  - ```
d = u2.connect('8c8a4d4d')
```

    - 其中 8c8a4d4d 是 adb devices 列出的当前（用USB数据线连接到Mac中的）安卓设备的 ID

```
→ ~ adb devices
List of devices attached
8c8a4d4d    device
```

输出举例：

```
→ autoTestAndroidGameHappyBigBattle python
Python 3.7.3 (default, May 22 2019, 10:55:14)
[Clang 10.0.1 (clang-1001.0.46.4)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import uiautomator2 as u2
>>> d = u2.connect('8c8a4d4d')
conn <urllib3.connection.HTTPConnection object at 0x1077f4da0 ,method GET,url /version,tim
eout_obj Timeout(connect 2, read 2, total None),body None,headers={'User-Agent': 'python-req
uests/2.22.0', 'Accept-Encoding': 'gzip, deflate', 'Accept': '*/*', 'Connection': 'keep-
alive'},chunked False
```

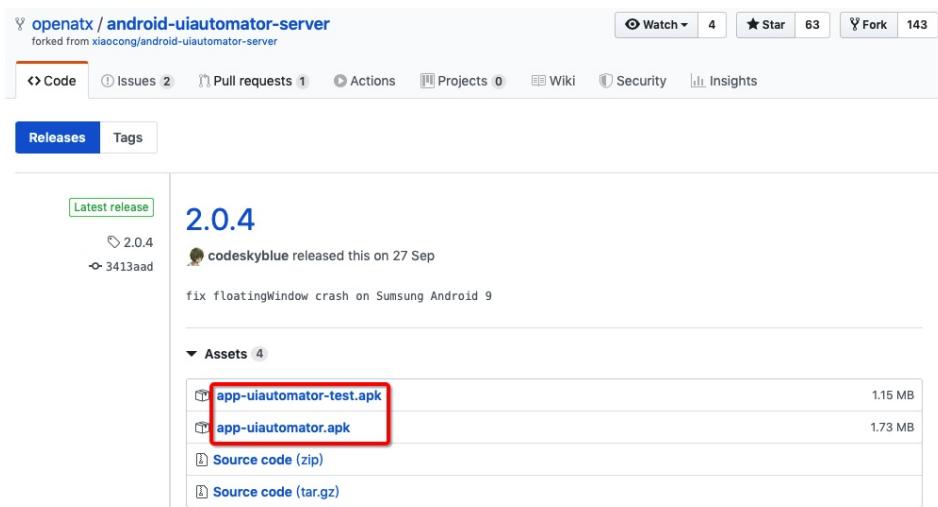
## 说明：安装细节

## 安装内容

上述命令会安装相关工具到你安卓手机中：

- uiautomator-server

- 作用：包含http rpc服务的apk
  - 2个apk
  - 图解



- 框架要求2个apk，缺一不可

- app-uiautomator-test.apk：测试程序
  - uiautomator这个框架允许我们测试第三方应用
  - 包名：com.github.uiautomator.test
- app-uiautomator.apk：被测应用
  - 基本就是个傀儡
    - 只要别轻易的死掉，就算是一个合格的应用了
  - 包名：com.github.uiautomator

- 地址：<https://github.com/openatx/android-uiautomator-server/releases>

- atx-agent

- 地址：<https://github.com/openatx/atx-agent>

- openstf/minicap

- 地址：<https://github.com/openstf/minicap>

- openstf/minitouch

- 地址：<https://github.com/openstf/minitouch>

## 安装log日志

期间如果开启了uiautomator2的debug后，可以看到更详细的信息。

比如安装路径（小米9中安装期间显示安装的东西有）：

- minicap、minitouch
  - [https://tool.appetizer.io/openatx/stf-binaries/raw/master/node\\_modules/minitouch-prebuilt/prebuilt/arm64-v8a/bin/minitouch](https://tool.appetizer.io/openatx/stf-binaries/raw/master/node_modules/minitouch-prebuilt/prebuilt/arm64-v8a/bin/minitouch)
- com.github.uiautomator, com.github.uiautomator.test 2.0.3

- <https://tool.appetizer.io/openatx/android-uiautomator-jsonrpcserver/releases/download/v0.1.6/bundle.jar>
- <https://tool.appetizer.io/openatx/android-uiautomator-jsonrpcserver/releases/download/v0.1.6/uiautomator-stub.jar>
- <https://tool.appetizer.io/openatx/android-uiautomator-server/releases/download/2.0.3/app-uiautomator.apk>
- <https://tool.appetizer.io/openatx/android-uiautomator-server/releases/download/2.0.3/app-uiautomator-test.apk>

安卓6的 华为畅享6S , 重新初始化的log是:

```
[200218 13:55:44][DevicesMethods.py 11 ] start init driver
[I 200218 13:55:45 init:132] uiautomator2 version: 2.5.3
[I 200218 13:55:45 init:317] Install minicap, minitouch
[I 200218 13:55:45 init:330] Install com.github.uiautomator, com.github.uiautomator.test 2
.1.1
[I 200218 13:56:02 init:300] - app-uiautomator.apk installed
[I 200218 13:56:14 init:300] - app-uiautomator-test.apk installed
[I 200218 13:56:14 init:308] Install atx-agent 0.8.2
[I 200218 13:56:19 init:342] Check atx-agent version
Successfully init AdbDevice(serial DWH9X17124W03779)
```

安卓9的 红米Note8Pro 的初始化log是:

```
[200217 14:45:33][DevicesMethods.py 11 ] start init driver
[I 200217 14:45:37 init:132] uiautomator2 version: 2.5.3
[I 200217 14:45:37 init:317] Install minicap, minitouch
minicap.so | 67.1K/67.1K
[I 200217 14:45:37 init:330] Install com.github.uiautomator, com.github.uiautomator.test 2
.1.1
[I 200217 14:45:38 init:300] - app-uiautomator.apk installed
[I 200217 14:45:38 init:300] - app-uiautomator-test.apk installed
[I 200217 14:45:38 init:308] Install atx-agent 0.8.2
[I 200217 14:45:39 init:342] Check atx-agent version
Successfully init AdbDevice(serial hmucae175ptk7zs)
```

分别对应着去安装:

- minicap和minitouch
- com.github.uiautomator和com.github.uiautomator.test
  - 对应着: app-uiautomator.apk和app-uiautomator-test.apk
- atx-agent

## 安装后的app

不过, 实际上 (安卓10的小米9, 安卓9的小米Note8Pro) 只安装了, 最核心的2个:

- ATX
  - 桌面图标



- 安装期间需要手动点击 继续安装



- com.github.uiautomator.test
  - 桌面图片：无
  - 安装期间，需要手动点击：继续安装



安装后，可以在应用管理中找到，刚才安装的2个应用：

- 红米Note8Pro 安卓9

- - 华为畅享6S 安卓6

◦

ATX

关于ATX，启动后的主界面：



点击 启动UIAUTOMATOR 后，会显示： ATX: Uiautomator started



crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2020-08-09 10:14:28

## 核心功能

接着介绍uiautomator2的一些常用的核心的功能。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2020-08-09 10:14:28

## 监听

其中一个很常用的功能就是：监听

即，注册了要监听的条件，满足后，就会自动触发。

典型应用比如，希望界面中出现 好的、确定 等按钮，就自动点击。

比如：

- 大众点评安装期间的 安装

◦

- 弹框中的 允许

◦

则需要去注册监听器，其核心逻辑是：

- 之前用：`watcher`
- 现改用：`xpath`

详细解释：

`watcher` 实现监听：

```
# 注册单个监听器
d.watcher("安装").when(text="安装").click()
# 等价于
d.watcher("安装").when(text="安装").click(text="安装")

# (此刻) 单次运行 (一次)
d.watchers.run()
```

```
# 后台长期的运行  
d.watcher.watched = True
```

其中的：

- `d.watcher.watched` 在 `uiautomator2 >=1.0.0` 版本后已废弃。
  - 推荐换用下面的 `xpath` 的写法：`xpath.watch_background`

`xpath` 实现监听：

```
# 注册单个监听器  
d.xpath.when(text("安装")).click()  
  
# 单次运行一次  
d.xpath.run_watchers()  
  
# 后台长期的运行=开启后台监控模式  
d.xpath.watch_background() # 默认每4s检查一次  
# 或手动设置间隔时间  
d.xpath.watch_background(2.0) # 2.0表示每2秒检查一次  
  
# 如果需要，再去停止后台监听  
d.xpath.watch_stop()
```

更多关于xpath的细节和用法，详见：

[uiautomator2/uiautomator2/ext/xpath at master · openatx/uiautomator2](#)

(注：不在主页的readme中，所以一般很少人能找到。我是从[raw的readme.md](#)中反推才找到的)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

# 查找元素

安卓测试期间，最常用的要属于，查找和定位页面中的相关元素了。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## xpath

xpath本身是一套独立的技术，常用于web领域内。

此处uiautomator2也支持xpath，用于元素定位，可以实现复杂条件的元素的查找。

### xpath常见操作

#### 定位节点和操作节点

```
tbsNodeList = self.driver.xpath("//com.tencent.tbs.core.webkit.WebView").all()
```

#### 获取属性content-desc的值

```
eachTbsNode.attrib.get("content-desc", "")
```

#### 给属性content-desc设置值

```
eachTbsNode.attrib["content-desc"] = "add something to avoid NAF"
```

#### 删除一个属性

```
eachTbsNode.attrib.pop("NAF")
```

## 文档

关于Xpath的详细用法，见官网中的xpath的文档：

[uiautomator2/uiautomator2/ext/xpath at master · openatx/uiautomator2](#)

-» 文档已经移至：

[uiautomator2/XPATH.md at master · openatx/uiautomator2](#)

其内部用的lxml，具体功能和语法都可以参考：

[The lxml.etree Tutorial](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28



## 操作元素

找到元素后，往往会涉及到操作元素，其中常见的一些操作有：

- 点击元素
- （给元素）输入内容

下面详细介绍如何操作。

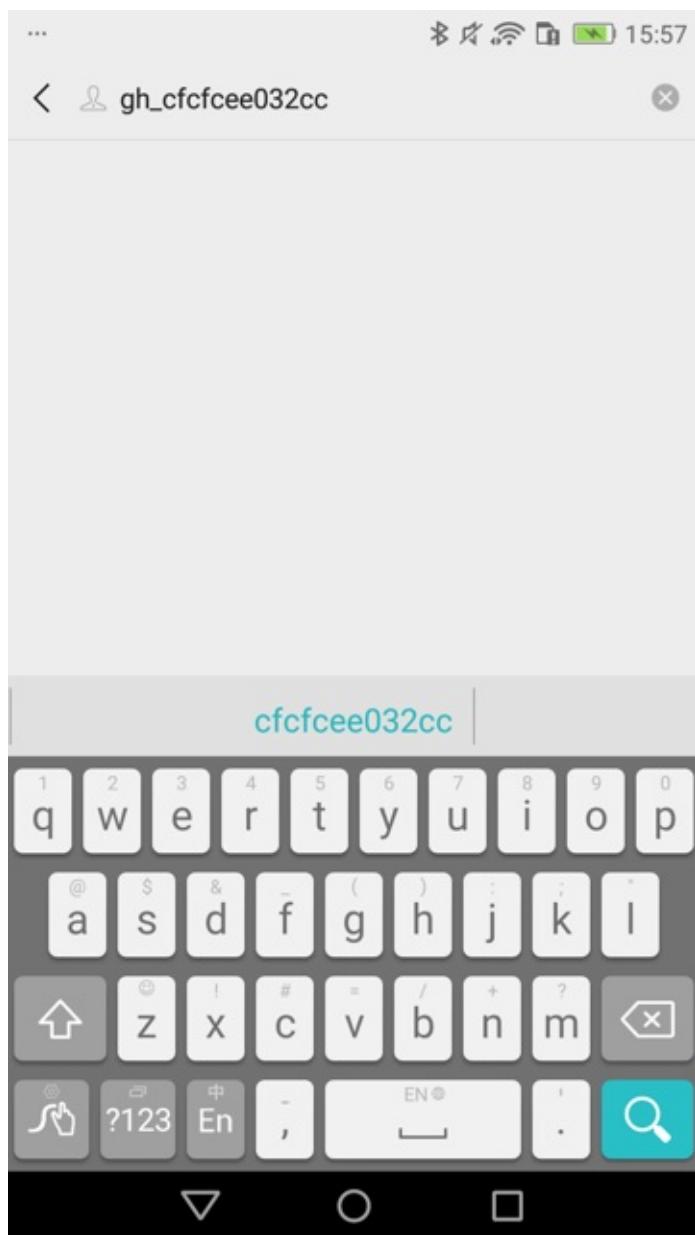
crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 点击元素

找到元素后，往往涉及到点击元素。

### 举例：点击搜索按钮

此处，对于安卓手机，华为畅享6S DIG-AL00，当前微信的公众号搜索界面中，已经处于系统自带输入法：华为Skype输入法时



用对应代码：

```
self.driver.send_action("search")
```

可以实现点击对应的 蓝色搜索 按钮，触发搜索，进入搜索结果页面。

详见：

【已解决】uiautomator2中点击华为手机中系统自带Swype的输入法中的搜索按钮

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 输入内容

找到元素后，也会遇到需要输入内容的情况。

典型用法是：

```
# 方式1：xpath的set_text方式
searchElementSelector = self.driver.xpath(locatorText)
searchElementSelector.set_text(text)
```

即可输入文字。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 当前屏幕

针对于当前屏幕，最常见的几个动作是：

- 截图=截屏
- 获取(当前)页面源码(xml)

### 给当前屏幕截图

核心代码：

```
fullImgFilePath = self.driver.screenshot(fullImgFilePath)
```

举例：

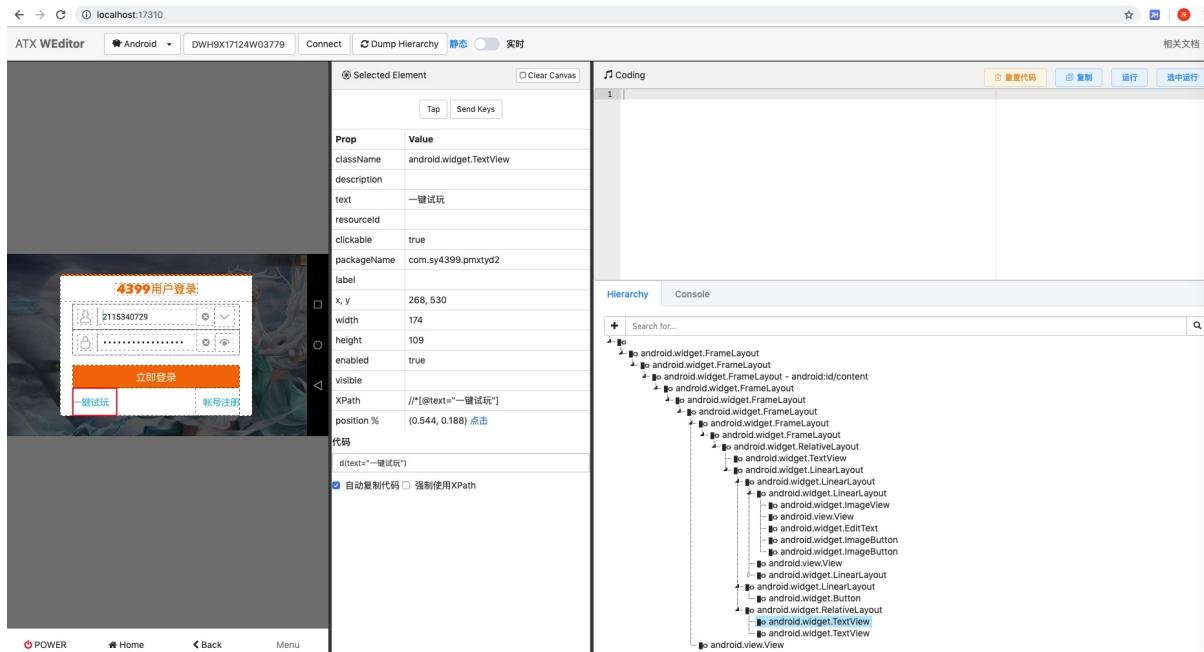
```
fullImgFilePath = 'debug/GameScreenshot/20191209_171115.png'  
fullImgFilePath = self.driver.screenshot(fullImgFilePath)
```

详见：

【已解决】 uiautomator2中如何获取到当前画面的截图文件

### 获取当前屏幕画面对应的xml源码

对于下图中左边的登录界面：



用：

```
page_source = self.driver.dump_hierarchy(compressed=False, pretty=False)
```

导出的源码是：

```
<?xml version='1.0' encoding='UTF-8' standalone='yes'?>
<hierarchy rotation='1'>
    <node index='0' text=''' resource-id=''' class='android.widget.FrameLayout' package='com.sy4399.pmxtyd2' content-desc=''' checkable='false' checked='false' clickable='false' enabled='true' focusable='false' focused='false' scrollable='false' long-clickable='false' password='false' selected='false' bounds='[0,0][1196,720]'>
        ...
            <node NAF='true' index='4' text=''' resource-id=''' class='android.widget.ToggleButton' package='com.sy4399.pmxtyd2' content-desc=''' checkable='true' checked='false' clickable='true' enabled='true' focusable='true' focused='false' scrollable='false' long-clickable='false' password='false' selected='false' bounds='[834,314][906,386]' />
                </node>
            </node>
            <node index='1' text=''' resource-id=''' class='android.widget.LinearLayout' package='com.sy4399.pmxtyd2' content-desc=''' checkable='false' checked='false' clickable='false' enabled='true' focusable='false' focused='false' scrollable='false' long-clickable='false' password='false' selected='false' bounds='[268,440][928,530]'>
                <node index='0' text='立即登录' resource-id=''' class='android.widget.Button' package='com.sy4399.pmxtyd2' content-desc=''' checkable='false' checked='false' clickable='true' enabled='true' focusable='true' focused='false' scrollable='false' long-clickable='true' password='false' selected='false' bounds='[268,440][928,530]' />
                    </node>
            ...
                </node>
            </node>
            </node>
        </node>
        <node index='1' text=''' resource-id=''' class='android.view.View' package='com.sy4399.pmxtyd2' content-desc=''' checkable='false' checked='false' clickable='false' enabled='true' focusable='false' focused='false' scrollable='false' long-clickable='false' password='false' selected='false' bounds='[218,80][978,86]' />
            </node>
        </node>
    </node>
</hierarchy>
```



## 相关

uiautomator2开放期间，往往会涉及到一些其他一些内容，此处把相对独立的部分整理出来，单独解释，方便查阅和理解。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## weditor

折腾u2期间，少不了要调试设备当前的页面，以及希望了解其中的元素和细节。

这时候，同一个作者开发的，用于辅助u2的 `weditor`，就可以派上用场了。

- 主页
  - Github
    - [openatx/weditor: web editor for atx](#)

安装：

```
pip3 install -U weditor
```

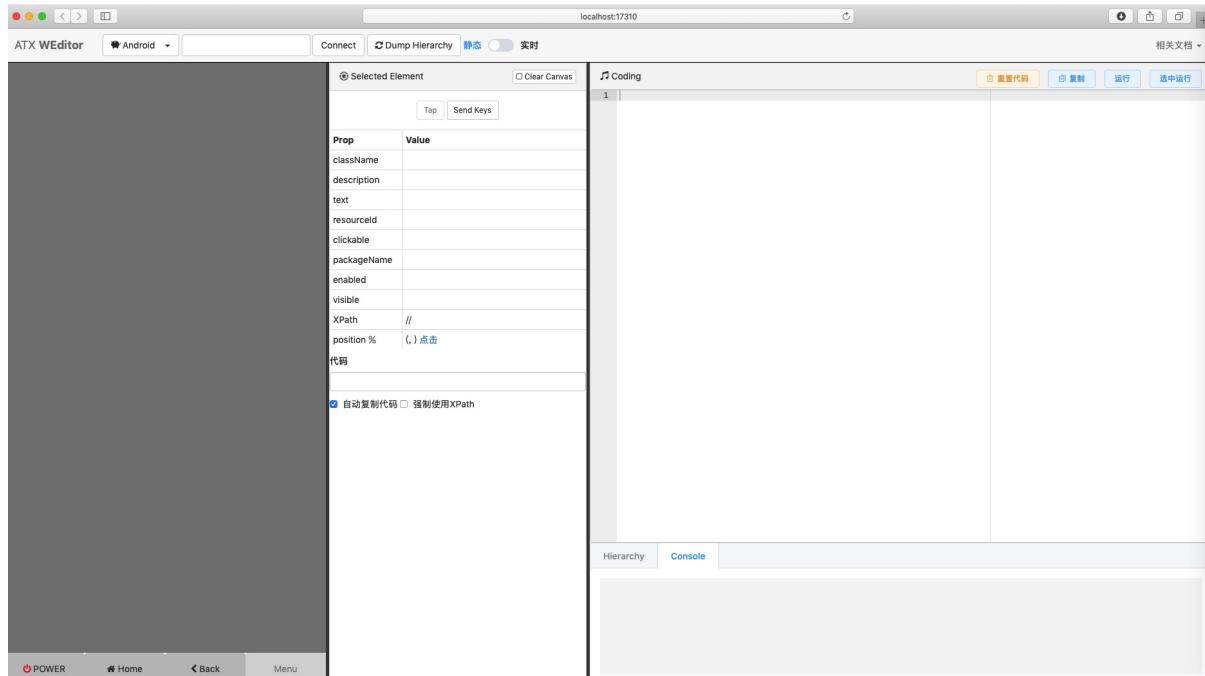
启动：

```
python -m weditor
```

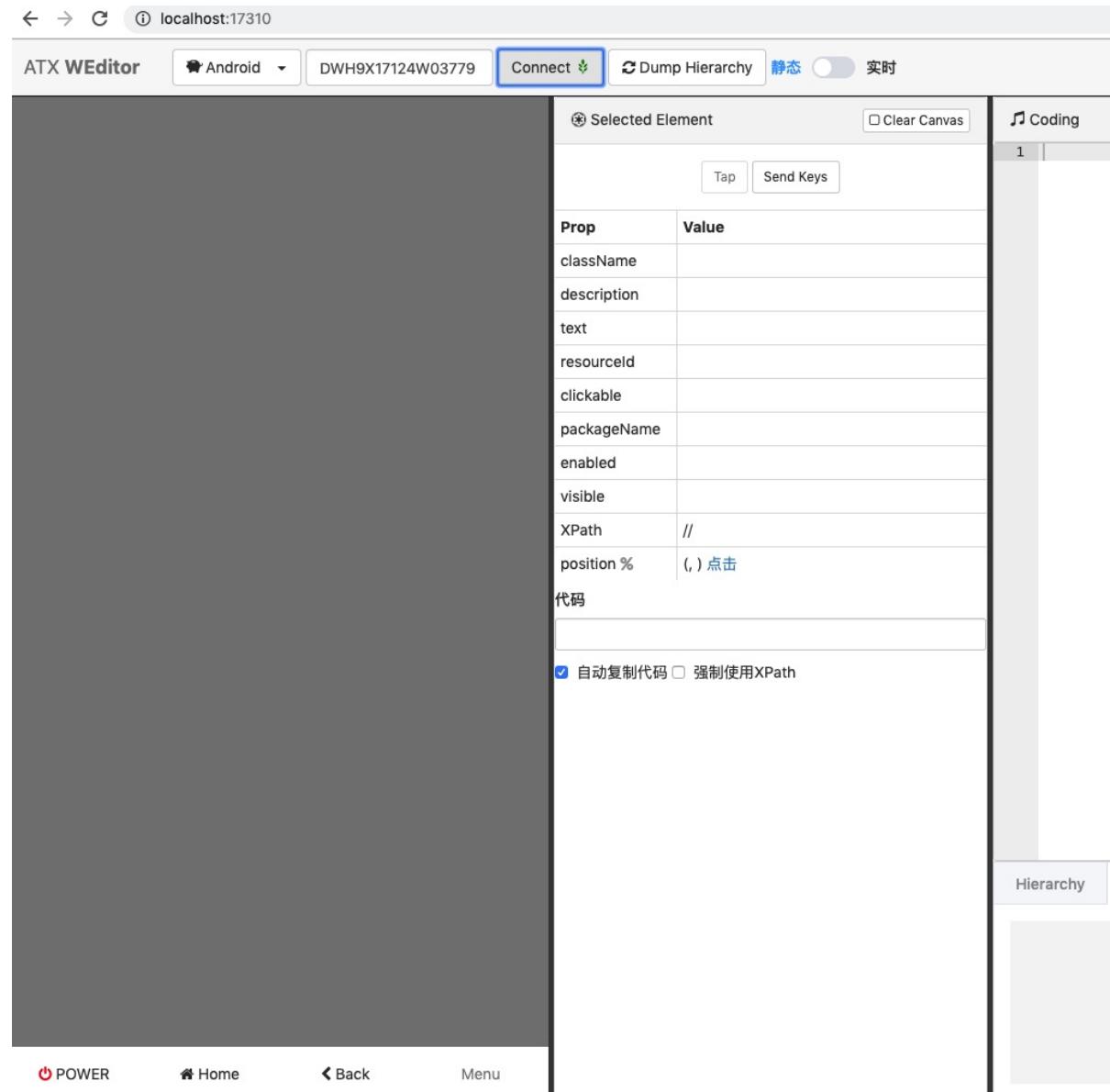
会自动调用浏览器并打开网址：

<http://localhost:17310>

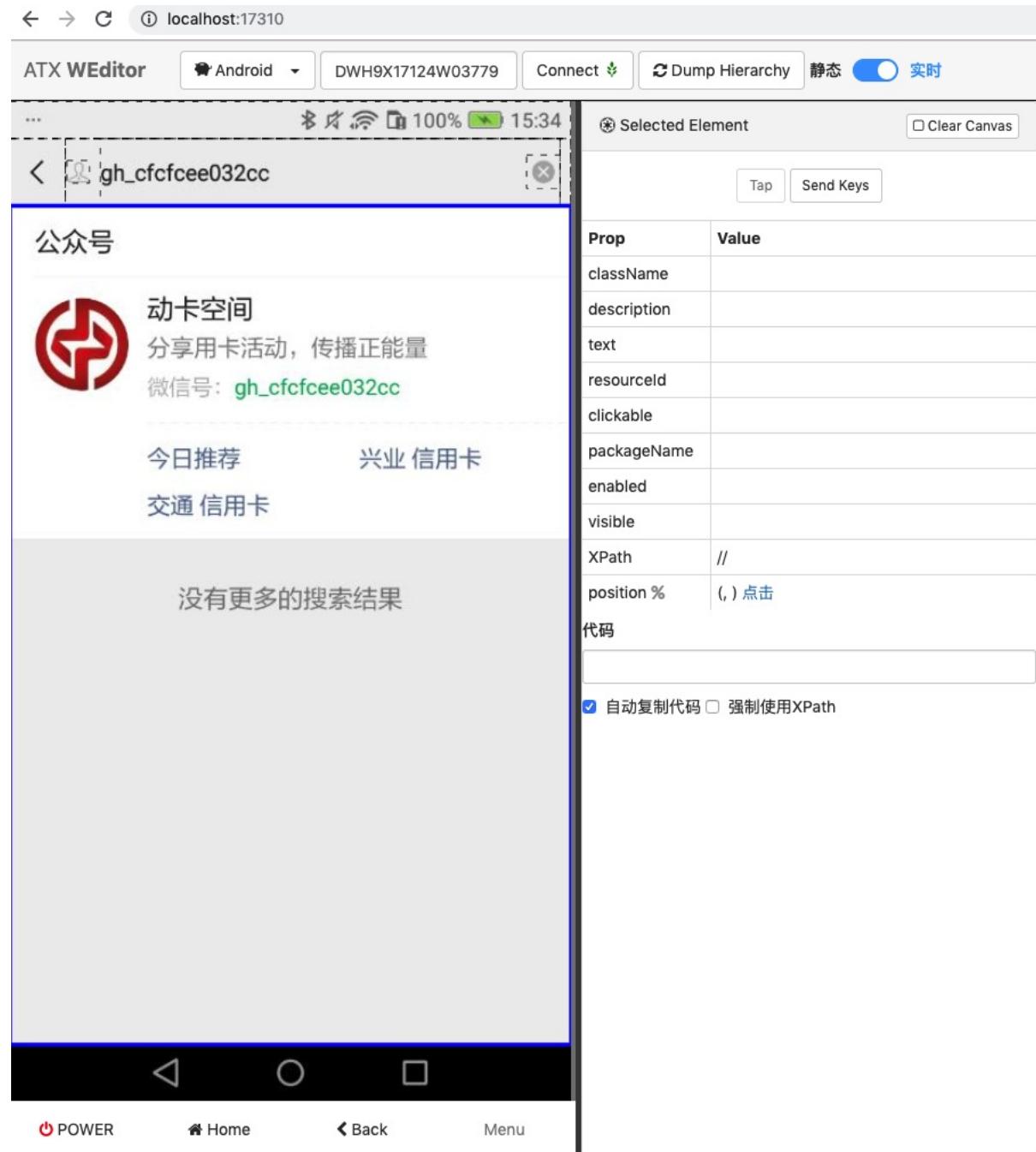
效果：



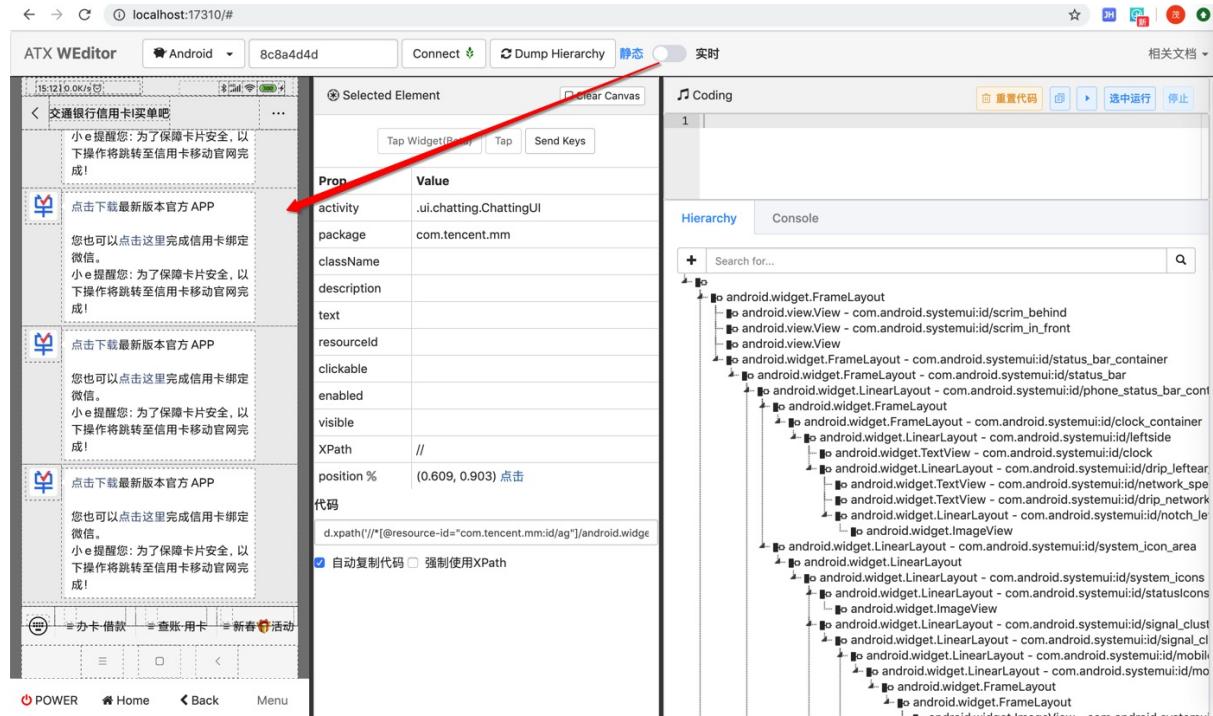
输入 安卓设备的id 后，点击 Connect 连接设备：



然后 多次在 静态 实时 直接切换几次， 最后一次点击 静态，稍等片刻，就能看到页面内容了：

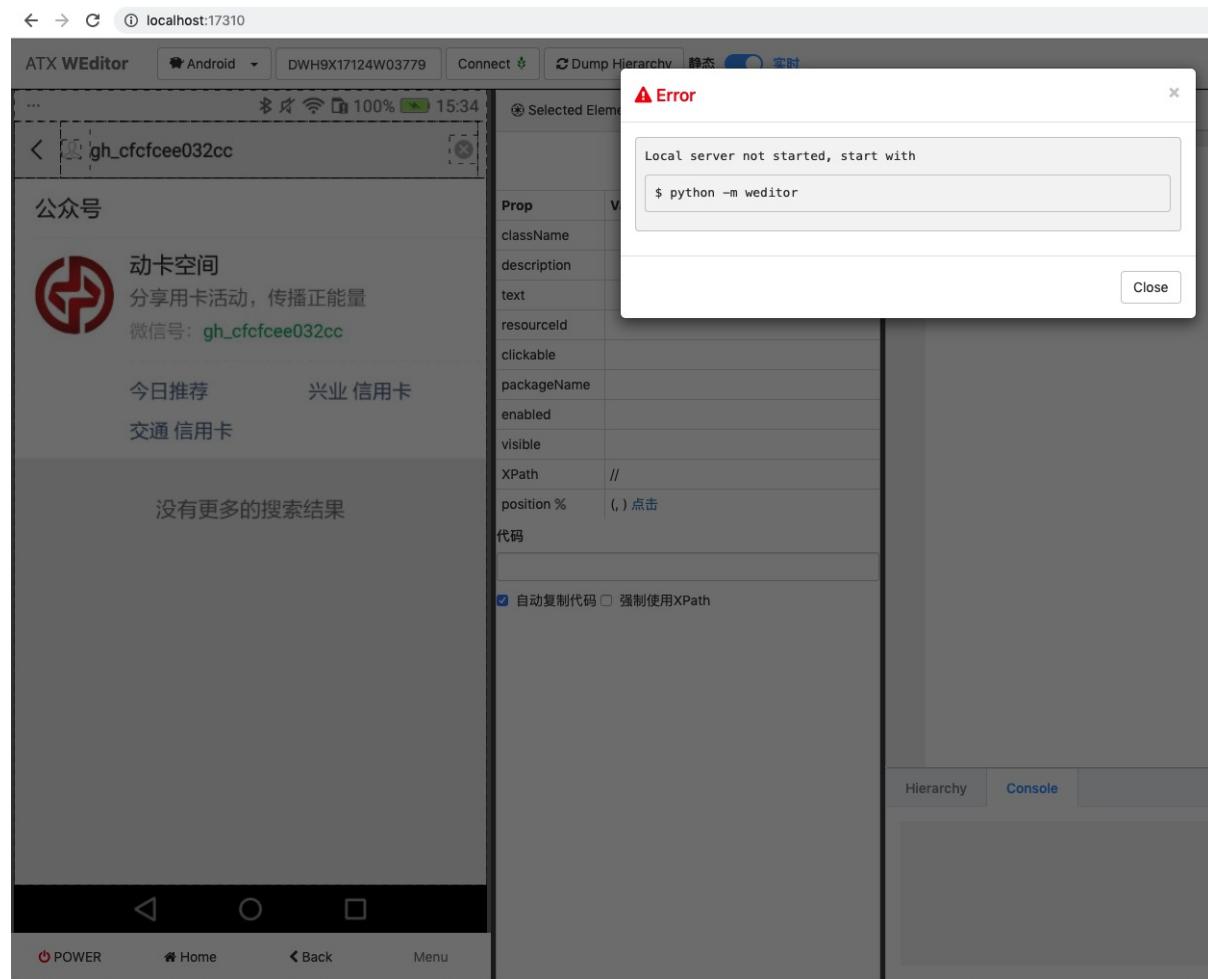


和：



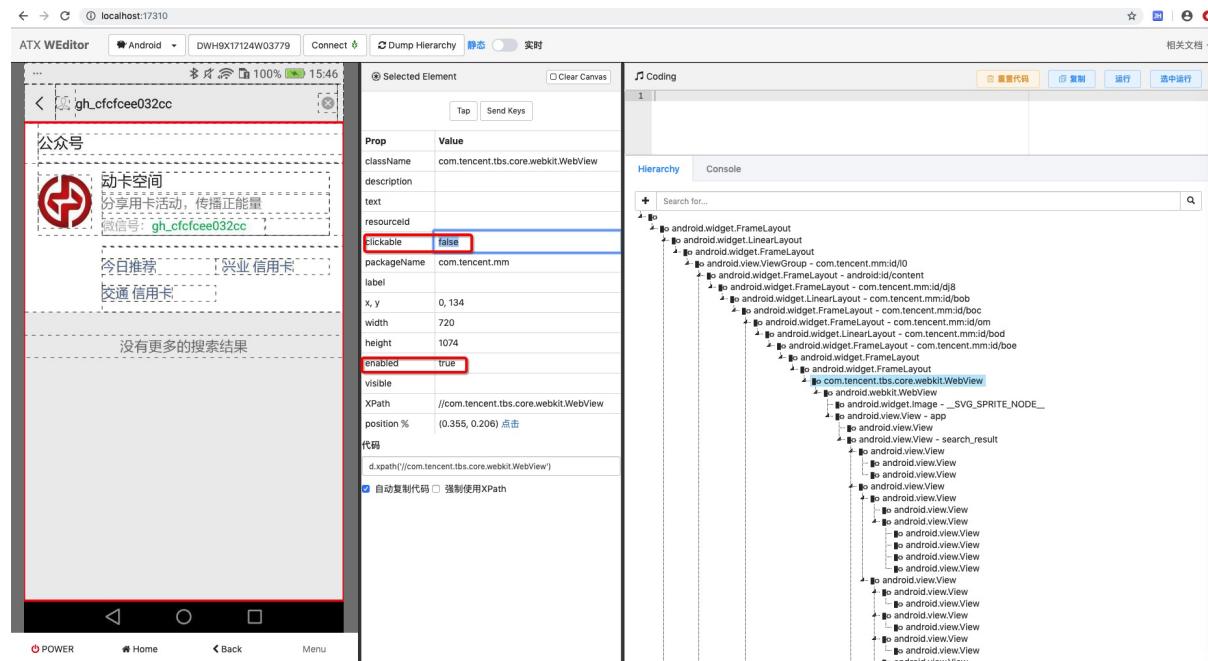
## 报错可忽略

注意，切换期间偶然会报错：



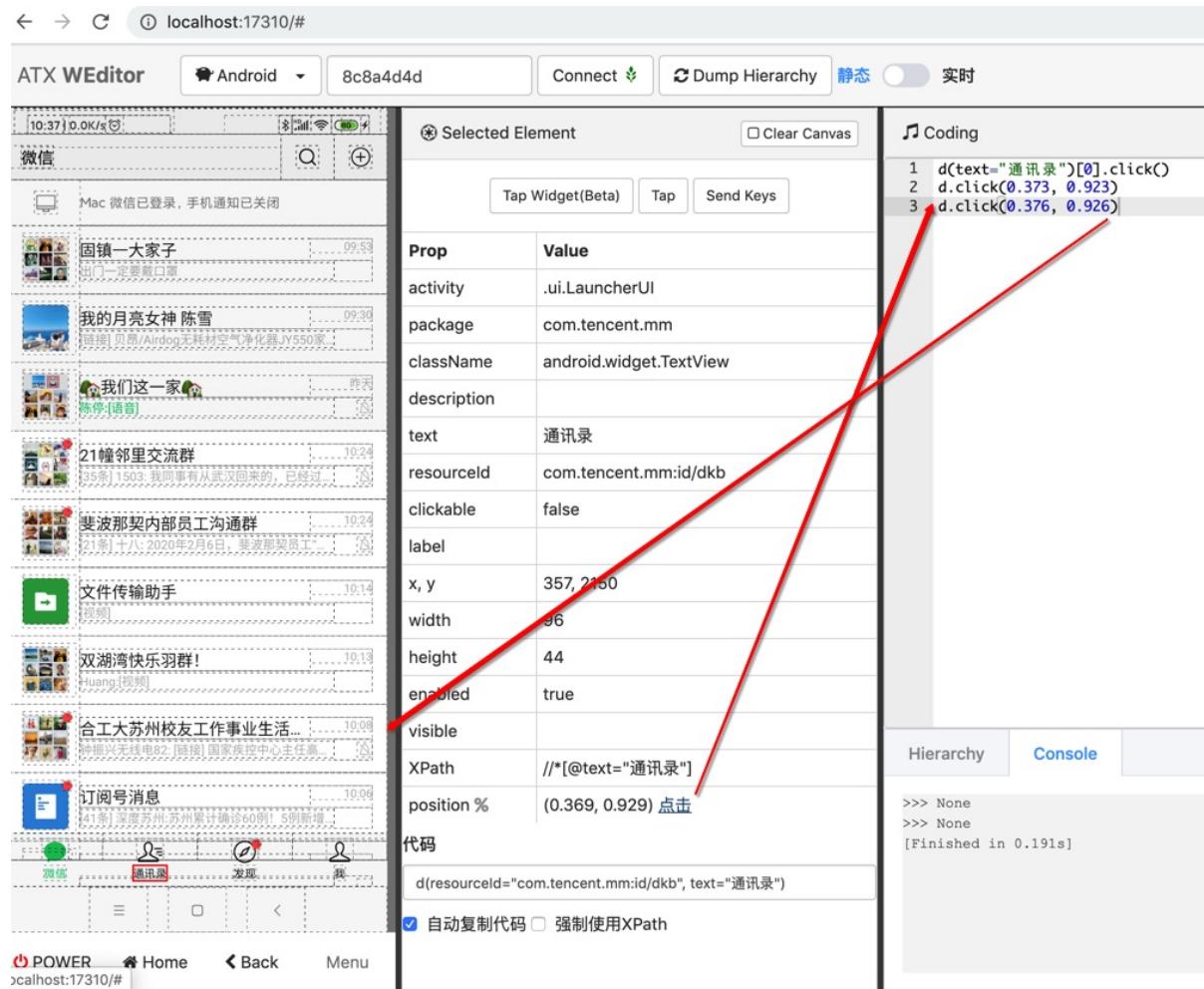
不用理会，关闭弹框，多试几次即可。

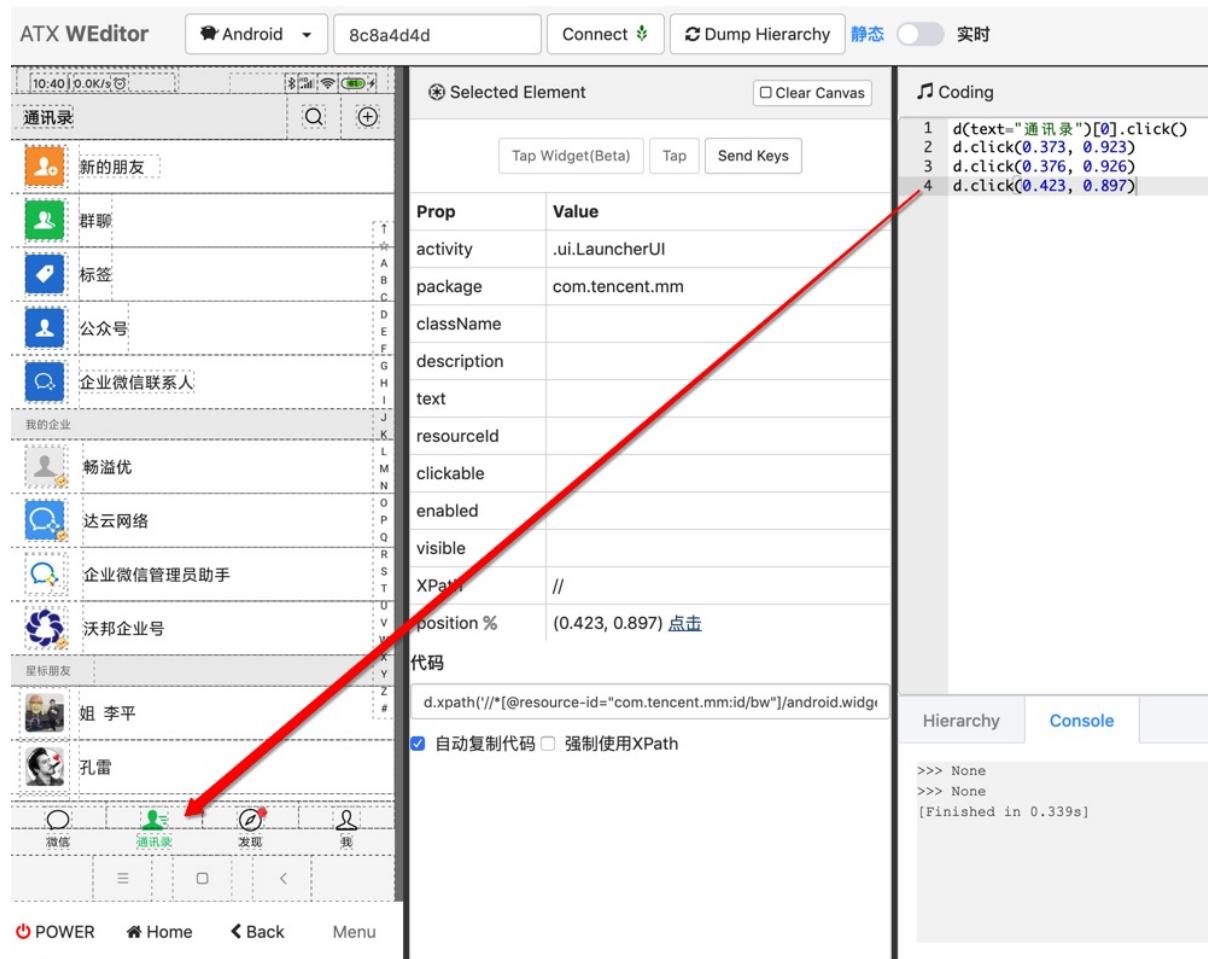
点击 Dump Hierarchy 后，能看到页面的结构：



## Coding中可以调试代码

之前有用过输入并运行代码，用于调试，效果不错：





再比如：

```

d(className="android.view.View")
d(className="android.view.View").count
    
```

选中第一行后，点击 选中运行：

实时 Command+Shift+Enter

Coding

1 d(className="android.view.View")  
2 d(className="android.view.View").count

重置代码 选中运行 停止

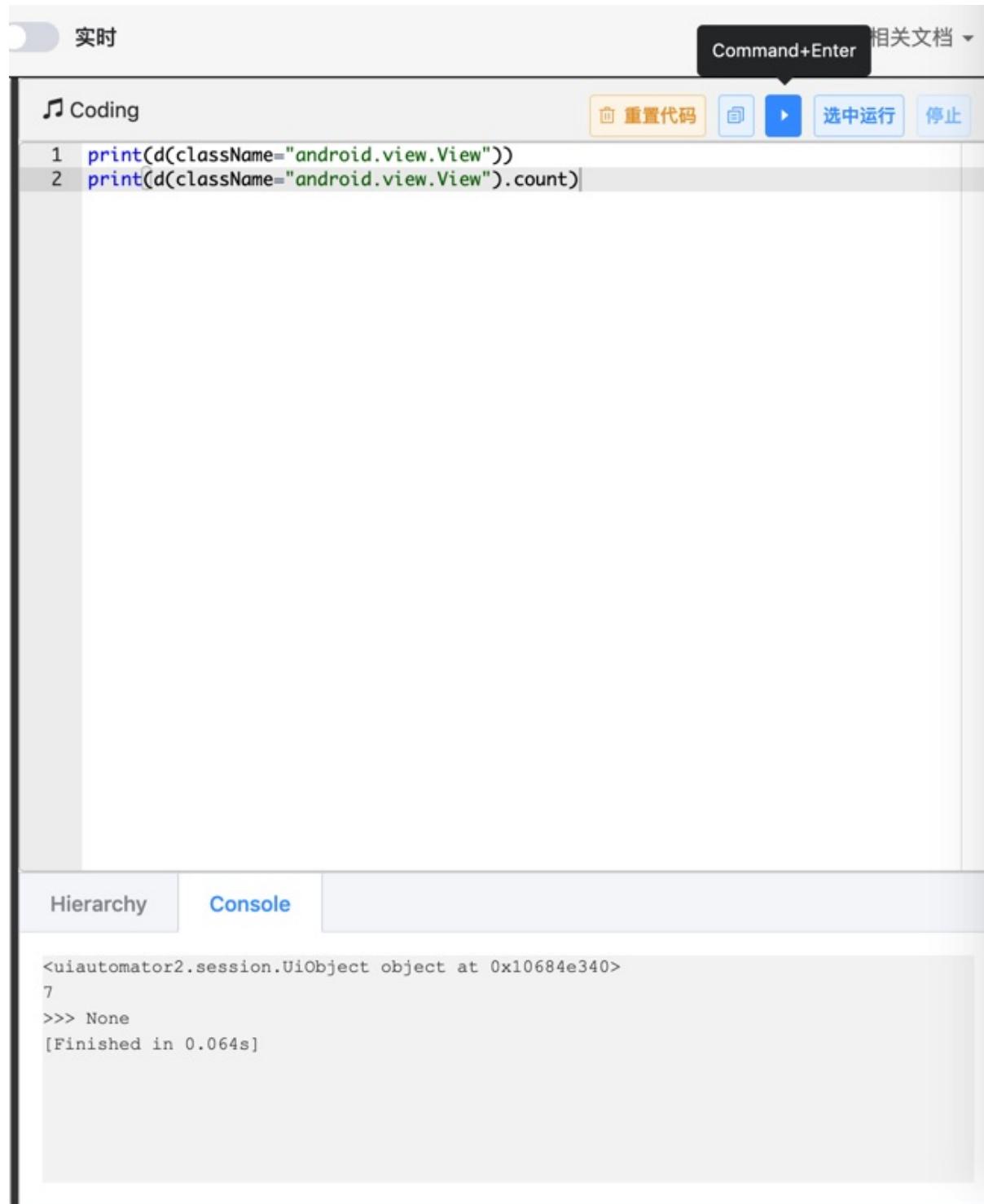
Hierarchy Console

```
>>> <uiautomator2.session.UiObject object at 0x1068653a0>
>>> None
[Finished in 0.001s]
```

加上print后

```
print(d(className "android.view.View"))
print(d(className "android.view.View").count)
```

不选中，点击 运行按钮，表示全部运行：



可以实时调试，很方便。

详见：

【未解决】自动抓包工具抓包公众号买单吧某个元素通过class+instance定位不到

【已解决】uiautomator2用click点击微信中的通讯录不起作用

## Hierarchy支持有限的搜索

对于xml中的节点：

```
<node NAF="true" index="0" text="" resource-id="com.tencent.mm:id/pq" class="android.view.View" package="com.tencent.mm" content-desc="" checkable="false" checked="false" clickable="true" enabled="true" focusable="true" focused="false" scrollable="false" long-clickable="true" password="false" selected="false" visible-to-user="true" bounds="[156,1522][912,2027]" />
```

想要去WEitor中

搜id值，即搜 `com.tencent.mm:id/pq`，结果找不到

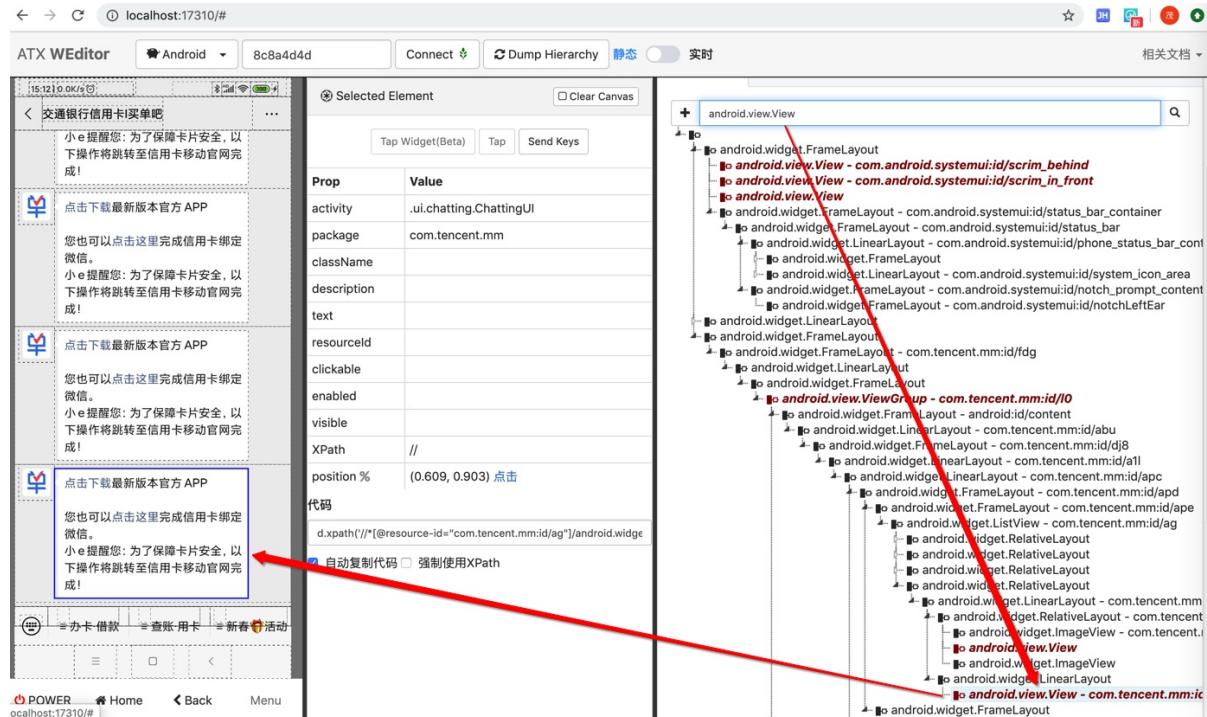
搜pq，也搜不到

后来发现，只能搜：当前显示出来的内容，即节点的class的类型

比如：`android.view.View`

是可以搜出并深红色高亮显示的对应节点的

然后才找到此处对应节点：



详见：

[【已解决】用weditor实时查看安卓当前页面中的xml源码](#)

[【已解决】Mac中安装uiautomator2的UI界面工具：weditor](#)

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28



## adb

此处整理uiautomator2开发期间，用到的和 adb 相关的东西。

### 解锁屏幕

安卓手机，华为的DIG-AL00，想要从锁屏界面，进入（等待输入密码的）解锁界面，可以用：

```
adb -s DWH9X17124W03779 shell input swipe 300 300 500 1000 100
```

或：

```
adb -s DWH9X17124W03779 shell input touchscreen swipe 300 300 500 1000 100
```

其中：

- DWH9X17124W03779：是手机的序列号
- 300 300 500 1000 100：
  - 300 300 500 1000：是屏幕坐标：X，Y，宽，高
  - 100：滑动时间，单位毫秒

即可实现进入解锁界面。

输入文字（密码）：

```
adb -s DWH9X17124W03779 shell input text 1234
```

用于解锁手机。

特殊：

- 华为畅享6S DIG-AL00：不支持
  - 相关信息
    - android版本:6.0
    - 系统：EMUI 4.1
  - 原因：估计是系统问题
  - 解决办法：无法解决

详见：

【无法解决】adb发送密码无法解锁安卓手机屏幕

### adb shell中的am start命令

android的adb调试工具，有个shell，可以执行很多命令。

其内部都是调用对应的子工具去处理具体功能的。

此处相关的有：

- 调用 Activity 管理器 ( am )
  - Activity Manager
- 调用软件包管理器 ( pm )
- 调用设备政策管理器 ( dpm )

其中am的解释是：

在 adb shell 中，您可以使用 Activity 管理器 (am) 工具发出命令以执行各种系统操作，如启动某项 Activity、强行停止某个进程、广播 intent、修改设备屏幕属性，等等。

在 shell 中，相应的语法为：

```
am command
```

您也可以直接从 adb 发出 Activity 管理器命令，无需进入远程 shell。例如：

```
adb shell am start -a android.intent.action.VIEW
```

具体参数含义解释：

command的语法=可用的 Activity 管理器命令

有很多，其中的start的语法是：

```
start [options] intent
```

- options=选项，包括：
  - -D：启用调试。
  - -W：等待启动完成。
  - --start-profiler file：启动分析器并将结果发送到 file。
  - -P file：类似于 --start-profiler，但当应用进入空闲状态时分析停止。
  - -R count：重复启动 Activity count 次。在每次重复前，将完成顶层 Activity。
  - -S：启动 Activity 前强行停止目标应用。
  - --opengl-trace：启用对 OpenGL 函数的跟踪。
  - --user user\_id | current：指定要作为哪个用户运行；如果未指定，则作为当前用户运行。
- intent：启动 intent 指定的 Activity。
  - (主要) 语法是：
    - -a action
      - 指定 intent 操作，例如 android.intent.action.VIEW（只能声明一次）。
    - -d data\_uri
      - 指定 intent 数据 URI，例如 content://contacts/people/1（只能声明一次）。
    - -t mime\_type
      - 指定 intent MIME 类型，例如 image/png（只能声明一次）。

- -c category
  - 指定 intent 类别, 例如 android.intent.category.APP\_CONTACTS。
- -n component
  - 指定带有软件包名称前缀的组件名称以创建显式 intent, 例如 com.example.app/.ExampleActivity。
- -f flags
  - 将标记添加到 setFlags() 支持的 intent。
- --esn extra\_key
  - 添加一个空 extra。URI intent 不支持此选项。
- -e | --es extra\_key extra\_string\_value
  - 将字符串数据作为键值对添加进来。
- --ez extra\_key extra\_boolean\_value
  - 将布尔型数据作为键值对添加进来。
- --ei extra\_key extra\_int\_value
  - 将整型数据作为键值对添加进来。
- --el extra\_key extra\_long\_value
  - 将长整型数据作为键值对添加进来。
- --ef extra\_key extra\_float\_value
  - 将浮点型数据作为键值对添加进来。
- --eu extra\_key extra\_uri\_value
  - 将 URI 数据作为键值对添加进来。
- --ecn extra\_key extra\_component\_name\_value
  - 添加组件名称, 该名称作为 ComponentName 对象进行转换和传递。
- --eia extra\_key extra\_int\_value[,extra\_int\_value...]
  - 添加整数数组。
- --ela extra\_key extra\_long\_value[,extra\_long\_value...]
  - 添加长整数数组。
- --efa extra\_key extra\_float\_value[,extra\_float\_value...]
  - 添加浮点数数组。

此处的：

- am start -a android.intent.action.MAIN -c android.intent.category.LAUNCHER -n com.tencent.mm/.ui.LauncherUI
  - am start : 启动
  - -a android.intent.action.MAIN : intent的动作是 android.intent.action.MAIN
  - -c android.intent.category.LAUNCHER : intent类别是 android.intent.category.LAUNCHER
  - -n com.tencent.mm/.ui.LauncherUI
    - 要启动的app=包名： com.tencent.mm
    - 也就是微信
  - 要启动的activity=界面=页面： .ui.LauncherUI
    - 也就是微信的主页面



## android-uiautomator-server

<https://github.com/openatx/android-uiautomator-server>

其发布的

<https://github.com/openatx/android-uiautomator-server/releases>

是2个apk:

- app-uiautomator-test.apk
- app-uiautomator.apk

其具体编译过程是：

```
$ ./gradlew build  
$ ./gradlew packageDebugAndroidTest
```

会生成apk，而最终的2个apk是mv生成的。

详见：.travis.yml 中的：

```
script:  
  - "./gradlew build"  
  - "./gradlew packageDebugAndroidTest"  
before_deploy:  
  - mv app/build/outputs/apk/debug/app-debug.apk app/build/outputs/apk/app-uiautomator.apk  
  - mv app/build/outputs/apk/androidTest/debug/app-debug-androidTest.apk app/build/outputs/apk/app-uiautomator-test.apk
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-12-11 21:39:52

## uiautomator

android官网的工具：uiautomator

主页：[uiautomator | Android Developers](#)

说了具体用法：

```
adb shell uiautomator dump
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 常见问题

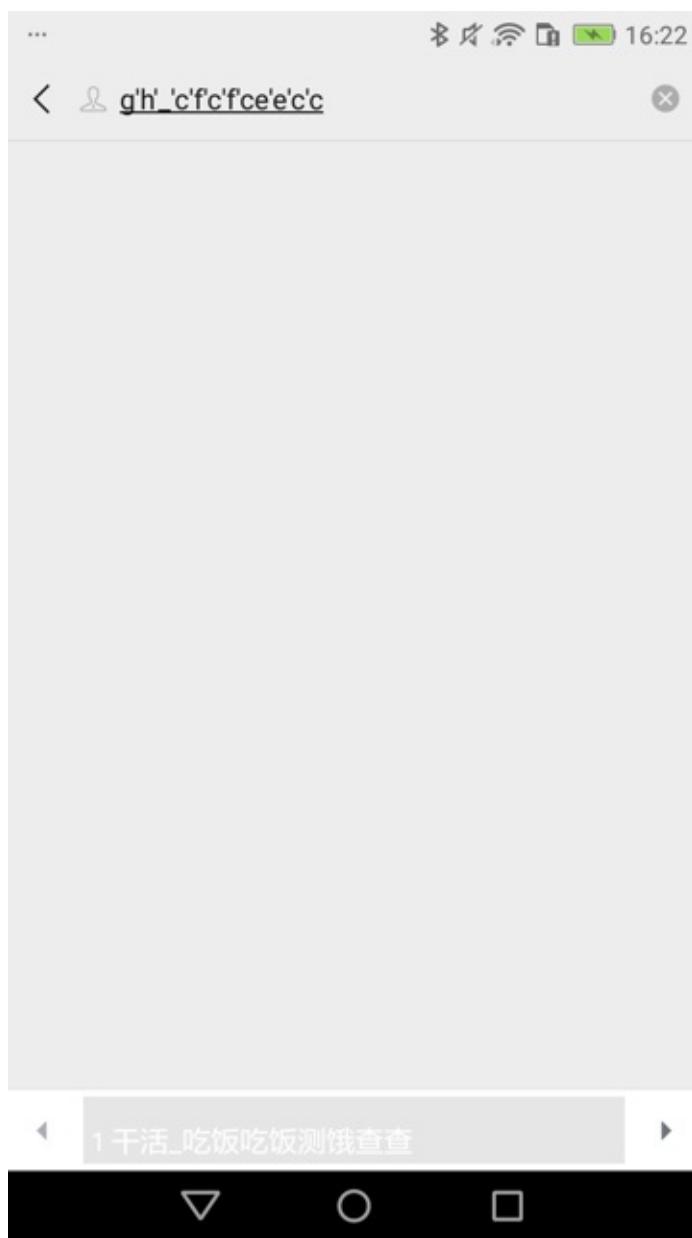
此处整理出uiautomator2开发期间，遇到的一些常见问题及其解决办法。

### 输入文字

目前发现之前最早的代码：

```
self.driver(text locator["text"]).set_text(text,timeout,WaitFind)
```

结果会无法完整输入内容：处于，中文输入法中，输入了字母，但是丢失了数字的效果：



且输入法此时已经也被换了（换成了FastInputIME或系统自带（华为Swype）输入法了）

注：

```
self.driver(text_locator["text"]).set_text(text, timeout=WaitFind)
```

内部是调用的uiautomator2的session的set\_text：

文件： /Users/limao/.pyenv/versions/3.8.0/lib/python3.8/site-packages/uiautomator2/session.py

```
def set_text(self, text, timeout=None):
    self.must_wait(timeout=timeout)
    if not text:
        return self.jsonrpc.clearTextField(self.selector)
    else:
        return self.jsonrpc.setText(self.selector, text)
```

(除了额外支持timeout参数外)

而换用另外的：

(1) xpath 的 set\_text

```
searchElementSelector = self.driver.xpath(searchKeyText)
searchElementSelector.set_text(text)
```

内部调用的：

文件： /Users/limao/.pyenv/versions/3.8.0/lib/python3.8/site-packages/uiautomator2/xpath.py

```
def set_text(self, text: str = ""):
    el = self.get()
    self._parent.send_text() # switch ime
    el.click() # focus input-area
    self._parent.send_text(text)
```

或

(2) send\_keys

```
self.driver.send_keys(text)
self.driver.set_fastinputime(False) # 关掉FastInputIME输入法，切换回系统默认输入法（此处华为手机默认输入法是华为Swype输入法）
```

其中，是否加上 打开FastInputIME

```
self.driver.set_fastinputime(True) # # 切换成FastInputIME输入法
self.driver.send_keys(text)
self.driver.set_fastinputime(False) # 关掉FastInputIME输入法，切换回系统默认输入法（此处华为手机默认输入法是华为Swype输入法）
```

经测试，感觉没区别。

结果都是：

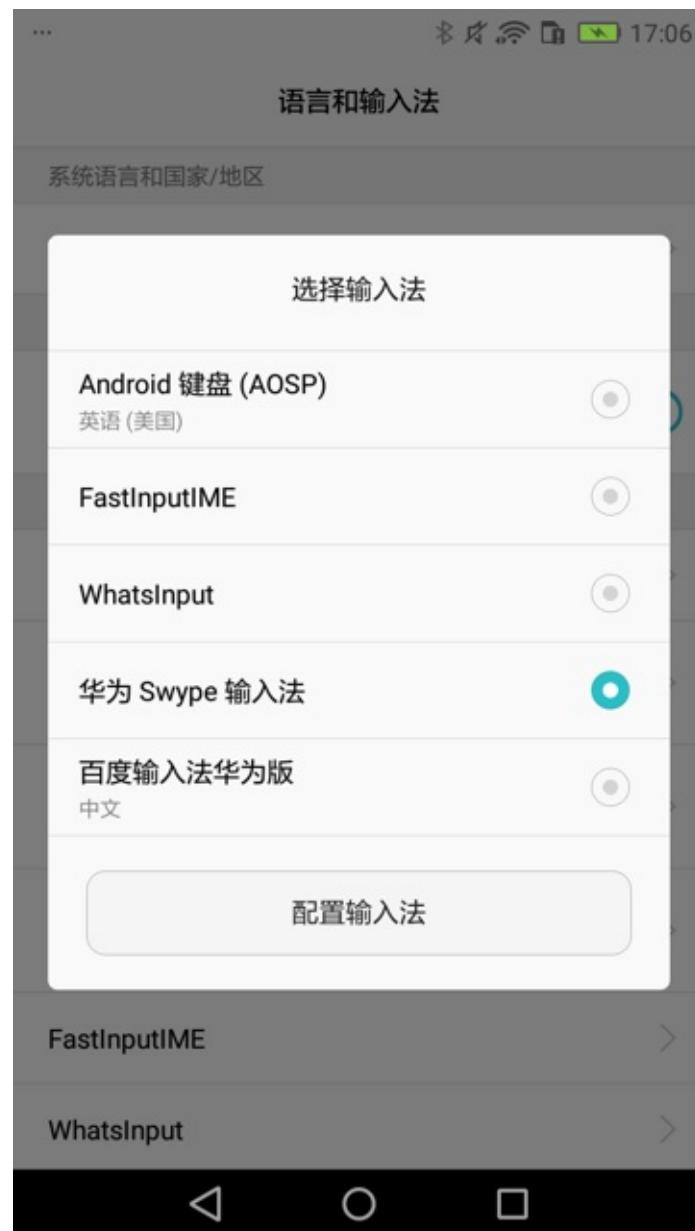
- 可以成功输入文字
  - 此处的：gh\_cfcfcee032cc
- 但是输入法会被切换掉
  - 我之前设置的是：百度的输入法

```
* I[android_input_method_baidu](../assets/img/android_input_method_baidu.png)
```

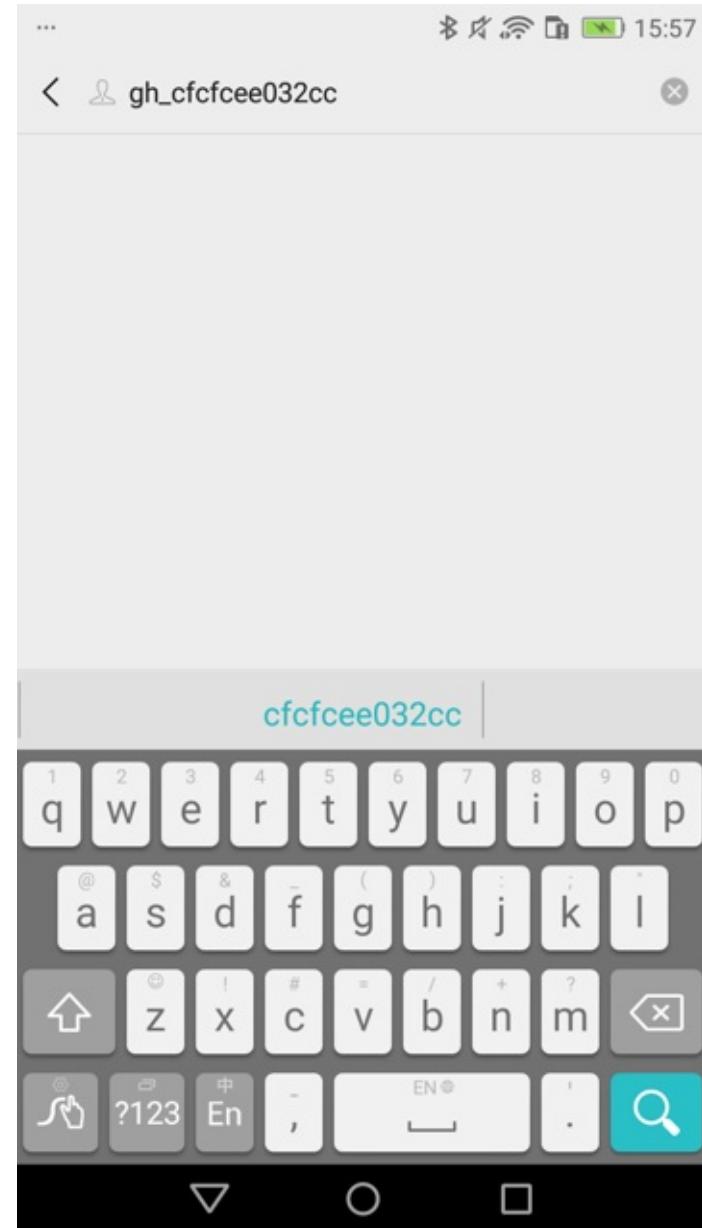
- 对应着，输入文字之前，应该是



- 会被换成：当前系统默认自带输入法
  - 当前系统是：华为的畅享6S手机 DIG-AL00
  - 自带输入法是：华为Swype输入法



- 效果是：



结论：

- 基本上实现了自己的：要输入文字的目的
- 但是：却把之前设置的（百度）输入法切换成系统的（华为）输入法了。
  - 问题不大，但是很不爽
    - 但是没办法改变和保留原有输入法

详见：

【部分解决】python的uiautomator2中set\_text导致输入法变化无法顺利输入文字

## 权限问题导致long\_click不工作

之前小米9中用long\_click

```
self.driver(text locator="text")).click(timeout WaitFind)
```

报错：

```
uiautomator2.exceptions.JsonRpcError: 0 Unknown error: > Injecting to another application requires INJECT_EVENTS permission> data: {'exceptionTypeName': 'java.lang.SecurityException', 'message': 'Injecting to another application requires INJECT_EVENTS permission'}, method: click
```



即： INJECT\_EVENTS 问题=权限问题

解决办法：去开启权限 USB调试（安全设置） -> 允许通过USB调试修改权限或模拟点击

17:41 | 0.9K/s ☺

4G 90%



## 开发者选项

### 信任状态结束时锁定屏幕

启用后，系统会在最后一个可信代理结束信任状态时锁定设备



调试

#### USB 调试

连接 USB 后启用调试模式



#### 撤销 USB 调试授权



#### USB安装

允许通过USB安装应用



#### USB调试（安全设置）

允许通过USB调试修改权限或模拟点击



#### 选择模拟位置信息应用



尚未设置模拟位置信息应用

#### 强制启用 GNSS 测量结果全面跟踪

在停用工作周期的情况下跟踪所有 GNSS 星座和频率



#### 启用视图属性检查功能



注：期间会3次提醒你

- 因为这个权限很重要
  - 如果随便给了其他坏的应用
    - 可能会滥用，而导致你手机被恶意操控
    - 所以多次提醒你确认
    - 自己此处是调试手机，自动抓包，所以没问题，是打算开启此权限

新: 2020-08-09 10:14:28

## NAF

此处所用安卓手机： 华为畅享6S DIG-AL00



对于最新版的 v7.0.8 的微信，公众号搜索结果的页面，去导出源码，发现：

- `uiautomator2` 中用代码：`self.driver.dump_hierarchy()`
  - 只能导出部分页面的源码
    - 其中红框内的源码无法导出



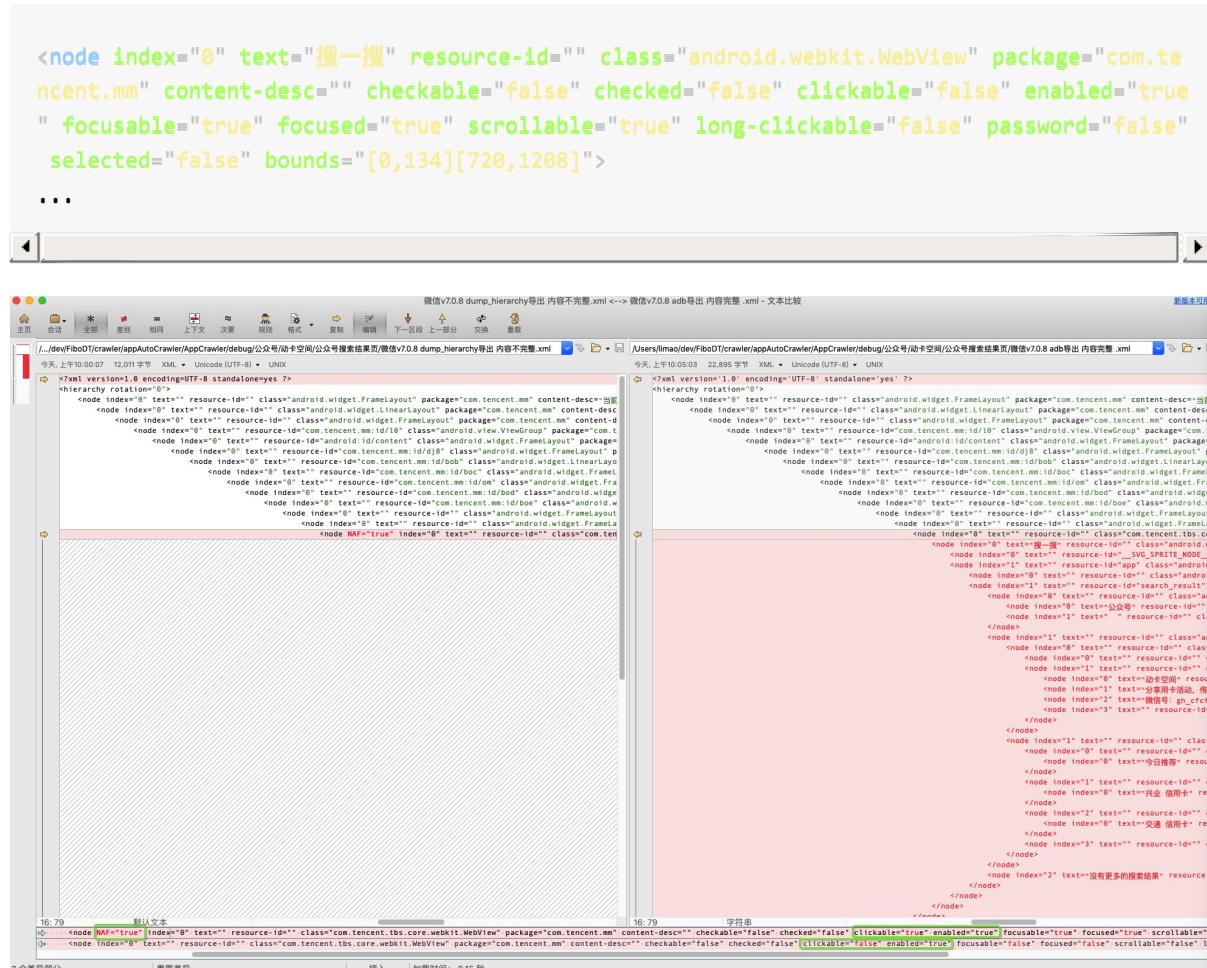
- Mac中终端运行adb命令: `adb shell uiautomator dump`
  - 能导出完整页面的源码

核心代码差异是:

```
<node NAF="true" index="0" text="" resource-id="" class="com.tencent.tbs.core.webkit.WebView" package="com.tencent.mm" content-desc="" checkable="false" checked="false" clickable="true" enabled="true" focusable="true" focused="true" scrollable="false" long-clickable="true" password="false" selected="false" bounds="[0,134][720,1208]" />
```

和

```
<node index="0" text="" resource-id="" class="com.tencent.tbs.core.webkit.WebView" package="com.tencent.mm" content-desc="" checkable="false" checked="false" clickable="false" enabled="true" focusable="false" focused="false" scrollable="false" long-clickable="false" password="false" selected="false" bounds="[0,134][720,1208]">
```



另外：

之前旧版本 v6.7.3 的微信，是可以正常导出的。

所以去研究：

微信版本升级前后的页面源码的变化：

- 升级前 = 微信 v6.7.3

- 2018微信v6.7.3老旧历史版本安装包官方免费下载\_豌豆荚
- 页面源码：

```

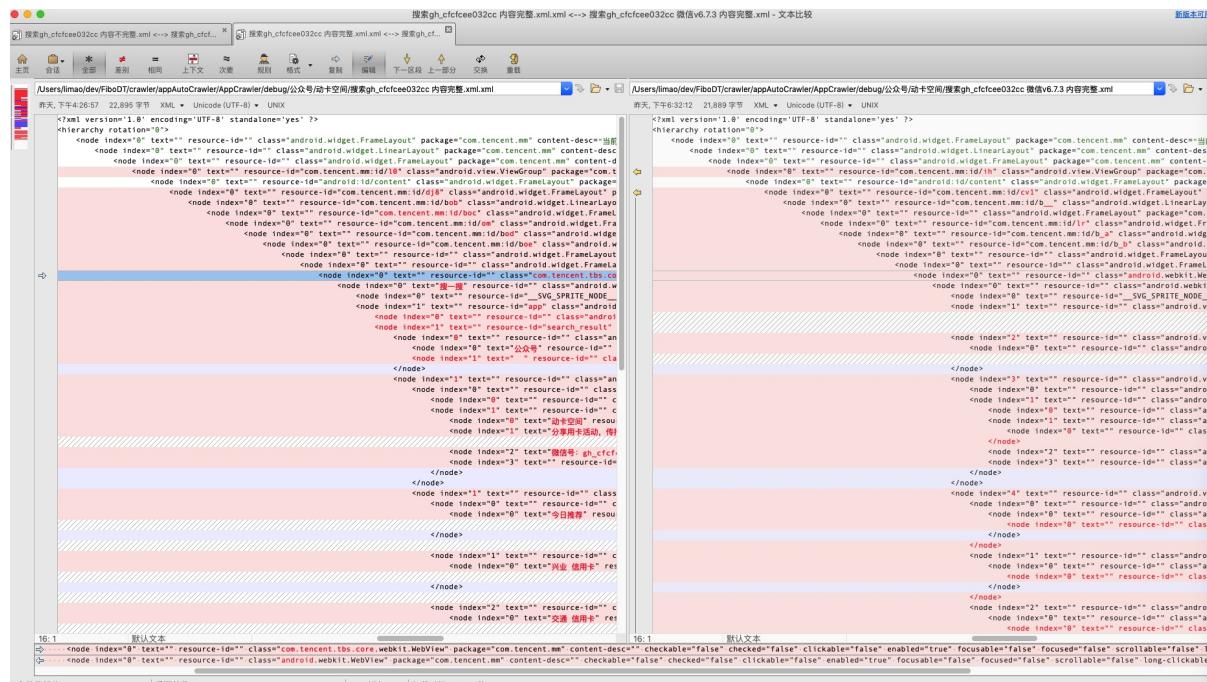
<node index="0" text="" resource-id="" class="android.webkit.WebView" package="com.tencent.mm" content-desc="" checkable="false" checked="false" clickable="false" enabled="true" focusable="true" focused="false" scrollable="false" password="false" selected="false" bounds="[0,144][720,1208]">
<node index="0" text="" resource-id="" class="android.webkit.WebView" package="com.tencent.mm" content-desc="搜一搜" checkable="false" checked="false" clickable="false" enabled="true" focusable="true" focused="true" scrollable="true" long-clickable="false" password="false" selected="false" bounds="[0,144][720,1208]">

<node index="0" text="" resource-id="__SVG_SPRITE_NODE__" class="android.widget.ImageView" package="com.tencent.mm" content-desc="" checkable="false" checked="false" clickable="false" enabled="true" focusable="false" focused="false" scrollable="false" bounds="0,144,720,1208">

```



对比的区别：



主要是class的不同：

- v6.7.3 : class="android.webkit.WebView"
- v7.0.8 : class="com.tencent.tbs.core.webkit.WebView"

而 tbs = 腾讯浏览器服务 = X5浏览器 = X5内核

即，新版本微信的浏览器的内核，用上了新的X5内核

- 什么是NAF

- = Not Accessibility Friendly
  - 直译：不可方便地访问（的节点）
    - Accessibility = 可访问性 = 可及性
      - 与之相对的是：（元素节点） Accessible 可访问
        - 其他普通的节点都是属于可访问的
        - 被谁访问：被其他的工具或软件等读取和操作
          - 其他的工具和软件：用来查看和研究Android页面源码的工具
            - 比如
              - android 自带的 uiautomatorviewer
                - 其还支持选项 "Toggle NAF Nodes"，打开后，可以查看到 NAF的节点
                  - 默认不能查看到NAF节点
                - 详见：
                  - [UI Testing | Android Developers](#)
    - Accessibility可访问性：主要指的是描述内容 content description 和 text 文本
      - 只有描述内容或文本有内容
        - 普通用户，才能从页面上才能看到该元素
          - 否则对于普通用户就看不到该元素了，也就没太大意义了
            - 或许可以被视为不可见元素了
    - 节点=元素=xml中的节点=某个UI控件=Android程序中页面上的某个控件
      - =Android的页面源码=xml代码
  - 判定NAF的逻辑=如何判定一个节点是NAF
    - 根据上面的代码中的 !nafExcludedClass(node) && !nafCheck(node) 可以看出：
      - 先判断节点的类型
        - !nafExcludedClass(node)：是属于那些可能被当做NAF节点的类型
          - 哪些节点，可能会被当做NAF节点呢？
            - class="xxx" 中 xxx，即类名不在 NAF\_EXCLUDED\_CLASSES 范围内的
              - 而 NAF\_EXCLUDED\_CLASSES 包括哪些呢？，包括如下：
                - android.widget.GridLayout
                - android.widget.GridView
                - android.widget.ListView
                - android.widget.TableLayout
            - 可见：除了上面4种节点，其他类型的节点，（只要符合特定条件）都可能被判定为NAF
        - 再判断节点的内容是否符合条件

- !nafCheck(node) :
  - 是可点击的
    - XML代码中: `clickable="true"`
  - 是已启用的=是有效的
    - XML代码中: `enabled="true"`
  - 描述内容是空的
    - XML代码中: `content-desc=""`
  - 文本是空的
    - XML代码中: `text=""`
- 如果上述2个条件都满足则判定是：NAF节点
  - 输出的节点中，会加上：`NAF="true"`
  - 这类节点，往往 `resource-id` 也是空
    - 典型的XML源码：
 

```
<node NAF="true" ... text="" resource-id="" class="com.tencent.tbs.core.webkit.WebView" package="com.tencent.mm" content-desc="" ... clickable="true" enabled="true" ... />
```
- 思考：
  - 为何对于：可点击的、已启用的，但是描述内容是空的、文本是空的 节点，被当做 NAF，认为不能被访问到呢？
    - 因为，这类节点，从Android的界面上，往往是看不到的，但是却又能被点击，所以基本上处于不可用状态
      - 所以（代码的作者）认为这类节点，属于（从设计角度来说，就是故意）不想被普通用户看到，接触到
        - 所以被判定为NAF，不应该被访问到
          - 在导出页面源码时，被忽略掉，不导出NAF节点
  - 为何上述4种节点：`GridLayout`，`GridView`，`ListView`，`TableLayout`，不会被当做 NAF呢？
    - 因为：满足了前面的 可点击的、已启用的，但是描述内容是空的、文本是空的 节点
      - 如果是本身属于（Android系统自带的）列表、表格等类型的节点，
        - 看起来就是：属于正常的节点了，因为这类节点，本身是可以没有描述内容，文本是空的
          - 而列表、表格等节点，就是Android中的：`GridLayout`, `GridView`, `ListView`, `TableLayout`等类型的节点
            - 当然，作者自己也说了，这4个类型，未必完整
              - 理论上，你也可以把其他的，合理的节点类型加到这个
                - `NAF_EXCLUDED_CLASSES` =不应该被认为是NAF的节点的类型中

详见：

**【已解决】uiautomator2中导出页面源码中NAF是什么意思**

而关于NAF如何规避解决，详见：

**【未解决】如何确保uiautomator2的dump\_hierarchy能导出页面中NAF的元素节点**

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2020-08-09 10:14:28

## 源码分析

折腾uiautomator2期间，分析了其中部分源码。现把过程整理如下供参考。

**uiautomator-server中最终的底层实现dumpWindowHierarchy的处理返回页面数据的逻辑**

先启动了底层的jsonrpc的服务，监听发送过来的，要执行的动作

文件： app/src/androidTest/java/com/github/uiautomator/stub/Stub.java

```
import com.github.uiautomator.JsonRpcServer;

public class Stub {
    ...
    int PORT = 9008;
    AutomatorHttpServer server = new AutomatorHttpServer(PORT);

    @Before
    public void setUp() throws Exception {
        launchService();
        JsonRpcServer jrs = new JsonRpcServer(new ObjectMapper(), new AutomatorServiceImpl(),
        AutomatorService.class);
        ...
        server.route("/jsonrpc/0", jrs);
        server.start();
    }
}
```

其中对于launchService：

```
private void launchService() throws RemoteException {
    UiDevice device = UiDevice.getInstance(InstrumentationRegistry.getInstrumentation());
    Context context = InstrumentationRegistry.getContext();
    device.wakeUp();

    // Wait for launcher
    String launcherPackage = device.getLauncherPackageName();
    Boolean ready = device.wait(Until.hasObject(By.pkg(launcherPackage).depth(0)), LAU
NCH_TIMEOUT);
    if (ready) {
        Log.i(TAG, "Wait for launcher timeout");
        return;
    }
}
```

```

        Log.d("Launch service");
        startMonitorService(context);
    }

    private void startMonitorService(Context context) {
        Intent intent = new Intent("com.github.uiautomator.ACTION_START");
        intent.setPackage("com.github.uiautomator"); // fix error: Service Intent must be explicit
        context.startService(intent);
    }

```

去启动了 `com.github.uiautomator`，应该就是在后台运行的uiautomator的服务了。

而前面的 `JsonRpcServer` 的 `jrs`，则是：

- 负责监听 `/jsonrpc/0`
  - 对应着之前 `uiautomator2` 中发送过来的请求
    - Shell\$ curl -X POST -d 'b'{"jsonrpc": "2.0", "id": "1f056baf5d6b2ea2cb7e546efb7cd64f", "method": "dumpWindowHierarchy", "params": [true, null]}' http://127.0.0.1:64445/jsonrpc/0
      - 中的 `jsonrpc/0`
- 其具体实现的类是 `AutomatorServiceImpl` 的 `AutomatorService`
  - 下面就来介绍 `AutomatorServiceImpl`

文件： `app/src/androidTest/java/com/github/uiautomator/stub/AutomatorServiceImpl.java`

```

public class AutomatorServiceImpl implements AutomatorService {

    /**
     * It's to test if the service is alive.
     *
     * @return 'pong'
     */
    @Override
    public String ping() {
        return "pong";
    }

    /**
     * Get the device info.
     *
     * @return device info.
     */
    @Override
    public DeviceInfo deviceInfo() {
        return DeviceInfo.getDeviceInfo();
    }
}

```

```
...
}
```

上面是最基本的几个函数：

- ping
  - 返回 pong
    - 表示服务还在，有效、alive
- deviceInfo
  - 对应着之前调试：
    - d = u2.connect('8c8a4d4d')
  - 期间输出的：
    - conn=<urllib3.connection.HTTPConnection object at  
0x1077f4be0>,method=POST,url=/jsonrpc/0,timeout\_obj=Timeout(connect=2, read=2,  
total=None),body={"jsonrpc": "2.0", "id": 1, "method": "deviceInfo"},headers=  
'User-Agent': 'python-requests/2.22.0', 'Accept-Encoding': 'gzip, deflate',  
'Accept': '\*/\*', 'Connection': 'keep-alive', 'Content-Length': '51'},chunked=False
  - 中的
    - "method": "deviceInfo"
  - 用于返回设备信息

而AutomatorServiceImpl中海油其他很多很多功能的具体实现。下面分别介绍一下之前接触过的。

```
public boolean click(int x, int y) {  
public boolean drag(int startX, int startY, int endX, int endY, int steps) throws NotImplementedException {  
public boolean swipe(int startX, int startY, int endX, int endY, int steps) {  
...  
}
```

都是常见的基础功能。

```
// Multi touch is a little complicated  
@Override  
public boolean injectInputEvent(int action, float x, float y, int metaState) {  
    MotionEvent e = MotionEvent.obtain(SystemClock.uptimeMillis(),  
        SystemClock.uptimeMillis(),  
        action, x, y, metaState);  
    e.setSource(InputDevice.SOURCE_TOUCHSCREEN);  
    boolean b = uiAutomation.injectInputEvent(e, true);  
    e.recycle();  
    return b;  
}
```

之前就遇到过多次，上层调用一些函数会报错，其中就会提到这个

比如：

【已解决】python的uiautomator2报错： uiautomator2.exceptions.JsonRpcError -32601 Method not found data injectInputEvent

中的

```
obj.jsonrpc.injectInputEvent(ACTION_DOWN, x, y, 0)
```

其他还有很多很多：

```
/**  
 * Simulates a short press using key name.  
 *  
 * @param key possible key name is home, back, left, right, up, down, center, menu, se  
arch, enter, delete(or del), recent(recent apps), volume_up, volume_down, volume_mute, cam  
era, power  
 * @return true if successful, else return false  
 * @throws RemoteException  
 */  
@Override  
public boolean pressKey(String key) throws RemoteException {  
    boolean result;  
    key = key.toLowerCase();  
    if ("home".equals(key)) result = device.pressHome();  
    else if ("back".equals(key)) result = device.pressBack();  
    else if ("left".equals(key)) result = device.pressDPadLeft();  
    else if ("right".equals(key)) result = device.pressDPadRight();  
    else if ("up".equals(key)) result = device.pressDPadUp();  
    else if ("down".equals(key)) result = device.pressDPadDown();  
    else if ("center".equals(key)) result = device.pressDPadCenter();  
    else if ("menu".equals(key)) result = device.pressMenu();  
    else if ("search".equals(key)) result = device.pressSearch();  
    else if ("enter".equals(key)) result = device.pressEnter();  
    else if ("delete".equals(key) || "del".equals(key)) result = device.pressDelete();  
    else if ("recent".equals(key)) result = device.pressRecentApps();  
    else if ("volume_up".equals(key)) result = device.pressKeyCode(KeyEvent.KEYCODE_VOL  
UME_UP);  
    else if ("volume_down".equals(key))  
        result = device.pressKeyCode(KeyEvent.KEYCODE_VOLUME_DOWN);  
    else if ("volume_mute".equals(key))  
        result = device.pressKeyCode(KeyEvent.KEYCODE_VOLUME_MUTE);  
    else if ("camera".equals(key)) result = device.pressKeyCode(KeyEvent.KEYCODE_CAMERA  
);  
    else result = "power".equals(key) && device.pressKeyCode(KeyEvent.KEYCODE_POWER);  
  
    return result;  
}  
  
public boolean pressKeyCode(int keyCode) {  
public boolean pressKeyCode(int keyCode, int metaState) {
```

```

public void clearTextField(Selector obj) throws UiObjectNotFoundException {
}

/**
 * Reads the text property of the UI element
 *
 * @param obj the selector of the UiObject.
 * @return text value of the current node represented by this UiObject
 * @throws UiObjectNotFoundException
 */
@Override
public String getText(Selector obj) throws UiObjectNotFoundException {
    if (obj.toUiObject2() == null) {
        return device.findObject(obj.toUiSelector()).getText();
    } else {
        return obj.toUiObject2().getText();
    }
}

/**
 * Sets the text in an editable field, after clearing the field's content. The UiSelector
 * selector of this object must reference a UI element that is editable. When you call this
 * method, the method first simulates a click() on editable field to set focus. The method
 * then clears the field's contents and injects your specified text into the field. If you want
 * to capture the original contents of the field, call getText() first. You can then modify
 * the text and use this method to update the field.
 *
 * @param obj the selector of the UiObject.
 * @param text string to set
 * @return true if operation is successful
 * @throws UiObjectNotFoundException
 */
@Override
public boolean setText(Selector obj, String text) throws UiObjectNotFoundException {
    try {
        obj.toUiObject2().click();
        obj.toUiObject2().setText(text);
        return true;
    } catch (NullPointerException | StaleObjectException e) {
        return device.findObject(obj.toUiSelector()).setText(text);
    }
}

/**
 * Performs a click at the center of the visible bounds of the UI element represented
 * by this UiObject.
 *
 * @param obj the target ui object.

```

```

    * @return true if successful else false
    * @throws UiObjectNotFoundException
    */
@Override
public boolean click(Selector obj) throws UiObjectNotFoundException {
    if (obj.toUiObject2() == null) {
        return device.findObject(obj.toUiSelector()).click();
    } else {
        obj.toUiObject2().click();
        return true;
    }
}

/**
 * Clicks the bottom and right corner or top and left corner of the UI element
 *
 * @param obj    the target ui object.
 * @param corner "br"/"bottomright" means BottomRight, "tl"/"topleft" means TopLeft, "center" means Center.
 * @return true on success
 * @throws UiObjectNotFoundException
 */
@Override
public boolean click(Selector obj, String corner) throws UiObjectNotFoundException {
    return click(device.findObject(obj.toUiSelector()), corner);
}

private boolean click(UiObject obj, String corner) throws UiObjectNotFoundException {
    if (corner == null) corner = "center";
    corner = corner.toLowerCase();
    if ("br".equals(corner) || "bottomright".equals(corner)) return obj.clickBottomRight();
    else if ("tl".equals(corner) || "topleft".equals(corner)) return obj.clickTopLeft();
    else if ("c".equals(corner) || "center".equals(corner)) return obj.click();
    return false;
}

public boolean dragTo(Selector obj, Selector destObj, int steps) throws UiObjectNotFoundException,
    NotImplementedException {
}

/**
 * Performs the swipe up/down/left/right action on the UiObject
 *
 * @param obj    the target ui object.
 * @param dir    "u"/"up", "d"/"down", "l"/"left", "r"/"right"
 * @param steps indicates the number of injected move steps into the system. Steps are

```

```

injected about 5ms apart. So a 100 steps may take about 1/2 second to complete.
 * @return true if successful
 * @throws UiObjectNotFoundException
 */
@Override
public boolean swipe(Selector obj, String dir, int steps) throws UiObjectNotFoundException {
    return swipe(device.findObject(obj.toUiSelector()), dir, steps);
}

private boolean swipe(UiObject item, String dir, int steps) throws UiObjectNotFoundException {
    dir = dir.toLowerCase();
    boolean result = false;
    if ("u".equals(dir) || "up".equals(dir)) result = item.swipeUp(steps);
    else if ("d".equals(dir) || "down".equals(dir)) result = item.swipeDown(steps);
    else if ("l".equals(dir) || "left".equals(dir)) result = item.swipeLeft(steps);
    else if ("r".equals(dir) || "right".equals(dir)) result = item.swipeRight(steps);
    return result;
}

...
...
...

```

其他更多函数就不贴代码了。

## 底层调用dumpWindowHierarchy, 处理, 返回数据

如上所述, AutomatorServiceImpl.java 中的很多功能函数, 此处最关心的dumpWindowHierarchy了:

```

/**
 * Helper method used for debugging to dump the current window's layout hierarchy. The
file root location is /data/local/tmp
 *
 * @param compressed use compressed layout hierarchy or not using setCompressedLayoutH
ierarchy method. Ignore the parameter in case the API level lt 18.
 * @param filename the filename to be stored. @deprecated
 * @return the absolute path name of dumped file.
 */
@Deprecated
@Override
public String dumpWindowHierarchy(boolean compressed, String filename) {
    return dumpWindowHierarchy(compressed);
}

/**
 * Helper method used for debugging to dump the current window's layout hierarchy.
 *
 * @param compressed use compressed layout hierarchy or not using setCompressedLayoutH

```

```

eierarchy method. Ignore the parameter in case the API level lt 18.
    * @return the absolute path name of dumped file.
    */
@Override
public String dumpWindowHierarchy(boolean compressed) {
    device.setCompressedLayoutHeirarchy(compressed);
    ByteArrayOutputStream os = null;
    try {
        os = new ByteArrayOutputStream();
        AccessibilityNodeInfoDumper.dumpWindowHierarchy(device, os);
        device.dumpWindowHierarchy(os);

        return os.toString("UTF-8");
    } catch (IOException e) {
        Log.d("dump Window Hierarchy got IOException " + e);
    }finally {
        if (os != null) {
            try {
                os.close();
            } catch (IOException e) {
                //ignore
            }
        }
    }
}

return null;
}

```

前一个：

```
public String dumpWindowHierarchy(boolean compressed, String filename) {
```

已废弃。

后一个，核心是调用：

```
AccessibilityNodeInfoDumper.dumpWindowHierarchy(device, os);
```

app/src/androidTest/java/com/github/uiautomator/stub/AccessibilityNodeInfoDumper.java

```

public static void dumpWindowHierarchy(UIDevice device, OutputStream out) throws IOException {
    XmlSerializer serializer = Xml.newSerializer();
    serializer.setFeature("http://xmlpull.org/v1/doc/features.html#indent-output", true);
    serializer.setOutput(out, "UTF-8");
    serializer.startDocument("UTF-8", true);
    serializer.startTag("", "hierarchy");
    serializer.attribute("", "rotation", Integer.toString(device.getDisplayRotation()));
}

```

```

    ;
    AccessibilityNodeInfo[] arr$ = getWindowRoots(device); // device.getWindowRoots();
    int len$ = arr$.length;

    for (int i$ = 0; i$ < len$; ++i$) {
        AccessibilityNodeInfo root = arr$[i$];
        dumpNodeRec(root, serializer, 0, device.getDisplayWidth(), device.getDisplayHeight());
    }

    serializer.endTag("", "hierarchy");
    serializer.endDocument();
}

```

最终返回的内容，就是此处的dumpWindowHierarchy函数的处理，生成xml内容后，所返回的。

比如某次调试过程：

jsonrpc的调用：

```
[191120 10:17:07][__init__.py 493] jsonrpc_call: jsonrpc_url http://127.0.0.1:64445/jsonrpc/0, method dumpWindowHierarchy, params (True, None), http_timeout 60
```

底层发送的请求是：

```
Shell$ curl -X POST -d 'b'{"jsonrpc": "2.0", "id": "5a175f3159cc1aa2e27f1cb68f5a0509", "method": "dumpWindowHierarchy", "params": [true, null]}' http://127.0.0.1:64445/jsonrpc/0
```

最终返回的结果是：

```
Output {"jsonrpc":"2.0","id":"5a175f3159cc1aa2e27f1cb68f5a0509","result":"<?xml version='1.0' encoding='UTF-8' standalone='yes' ?><hierarchy rotation=\"0\"><node index=\"0\" text=\"\" resource-id=\"\" class=\"android.widget.FrameLayout\" package=\"com.tencent.mm\" content-desc=\"当前所在页面,搜一搜\" checkable=\"false\" checked=\"false\" clickable=\"false\" enabled=\"true\" focusable=\"false\" focused=\"false\" scrollable=\"false\" long-clickable=\"false\" password=\"false\" selected=\"false\" bounds=\"[0,0][720,1208]\"><node index=\"0\" text=\"\" resource-id=\"com.tencent.mm:id/om\" class=\"android.widget.FrameLayout\" package=\"com.tencent.mm\" content-desc=\"containerFrameLayout\" checkable=\"false\" checked=\"false\" clickable=\"false\" enabled=\"true\" focusable=\"false\" focused=\"false\" scrollable=\"false\" long-clickable=\"false\" password=\"false\" selected=\"false\" bounds=\"[0,134][720,1208]\"><node index=\"0\" text=\"\" resource-id=\"com.tencent.mm:id/boe\" class=\"android.widget.FrameLayout\" package=\"com.tencent.mm\" content-desc=\"containerFrameLayout\" checkable=\"false\" checked=\"false\" clickable=\"false\" enabled=\"true\" focusable=\"false\" focused=\"false\" scrollable=\"false\" long-clickable=\"false\" password=\"false\" selected=\"false\" bounds=\"[0,134][720,1208]\"><node NAF=\"true\" index=\"0\" text=\"\" resource-id=\"\" class=\"com.tencent.tbs.core.webkit.WebView\" package=\"com.tencent.mm\" content-desc=\"\" checkable=\"false\" checked=\"false\" clickable=\"true\" enabled=\"true\" focusable=\"true\" focused=\"true\" scrollable=\"false\" long-clickable=\"true\" pass
```

```

word=false" selected=false" bounds="[0,134][720,1208]" /></node></node index
= "1" text="" resource-id="com.tencent.mm:id/m4" class="android.widget.LinearLayout"
" package="com.tencent.mm" content-desc="" checkable=false" checked=false" click
able=true" enabled=true" focusable=false" focused=false" scrollable=false"
long-clickable=false" password=false" selected=false" bounds="[0,48][71,134]" ><
node index="0" text="" resource-id="com.tencent.mm:id/m5" class="android.widget.Im
ageView" package="com.tencent.mm" content-desc="返回" checkable=false" checked=false"
clickable=false" enabled=true" focusable=false" focused=false" scrollable=false"
long-clickable=false" password=false" selected=false" bounds="[0,48][71,134]" ></node><node index="3" text="gh_cfcfcee032cc" resource-id="com.tencent.mm
:id/m7" class="android.widget.EditText" package="com.tencent.mm" content-desc=""
checkable=false" checked=false" clickable=true" enabled=true" focusable=true"
focused=false" scrollable=false" long-clickable=true" password=false" selecte
d=false" bounds="[116,48][706,134]" ><node NAF=true" index="4" text="" resour
ce-id="com.tencent.mm:id/m3" class="android.widget.ImageButton" package="com.tencent.
mm" content-desc="" checkable=false" checked=false" clickable=true" enabled=true"
focusable=true" focused=false" scrollable=false" long-clickable=false"
password=false" selected=false" bounds="[663,69][706,112]" ></node></hierarchy">

```

可见其中的xml头部的内容：

```
<?xml version='1.0' encoding='UTF-8' standalone='yes' ?><hierarchy rotation="0">
```

就是上面的 `XmlSerializer` 的代码所生成的。

而其他的node节点，则是`dumpNodeRec`所生成的。

由此，后续深入研究，才知道，最终返回的节点中，如果符合NAF条件，则会被忽略其下内容，最终返回一个NAF="true"的节点，导致后续只返回部分页面内容的。

具体细节详见：

- 【未解决】 uiautomator2中`dump_hierarchy`中只能获取到页面的部分的xml源码
- 【已解决】 搞懂uiautomator-server中最终的底层实现`dumpWindowHierarchy`的处理返回页面数据的逻辑

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 通用功能函数

开发uiautomator2期间，把一些常用的功能，常用代码段，封装成了函数并贴出来，和具体调用方式，供参考。

其中后续各种通用功能和函数，往往都会调用到一些基础的工具类函数，详见接下来的工具类函数，后续就不再赘述。

crifan.com，使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

# 工具类函数

在介绍通用功能之前，要先把常用到的基础的工具类的函数贴出来，供参考使用。

## 获取命令执行后返回的结果

```
def get_cmd_lines(cmd, text=False):
    # 执行cmd命令，将结果保存为列表
    resultStr = ""
    resultStrList = []
    try:
        consoleOutputByte = subprocess.check_output(cmd, shell=True) # b'C02Y3N10JHC8\n'
        try:
            resultStr = consoleOutputByte.decode("utf-8")
        except UnicodeDecodeError:
            # TODO: use chardet auto detect encoding
            # consoleOutputStr = consoleOutputByte.decode("gbk")
            resultStr = consoleOutputByte.decode("gb18030")

        if not text:
            resultStrList = resultStr.splitlines()
    except Exception as err:
        print("err=%s when run cmd=%s" % (err, cmd))

    if text:
        return resultStr
    else:
        return resultStrList
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2020-08-09 10:14:28

## adb

uiautomator2操作安卓设备期间，往往会涉及到，借助于安卓体系内本身就有的工具 `adb`，去实现对设备的一些操控。

此处整理出一些常见的用法和通用功能。

### 获取当前安卓手机名

```
def get_phone_name_Android(self):
    # cmd = 'adb -s {} shell getprop ro.product.model'.format(self.device)
    cmd = 'adb -s {} shell getprop ro.product.name'.format(self.device)
    text = CommonUtils.get_cmd_lines(cmd, text=True)
    # https://miuiver.com/xiaomi-device-codename/
    #     begonia -> 红米Note 8内部代号为 "begonia"
    return re.sub("\s", "", text)
    # isRunCmdOk, outputText = self.getCommandOutput(cmd)
    # if isRunCmdOk:
    #     phoneName = outputText
    # else:
    #     phoneName = ""
    # return phoneName
```

调用：

```
def get_phone_name(self):
    # 获取手机名称，以提取配置信息
    if self.isAndroid:
        return self.get_phone_name_Android()
```

### 获取当前连接的设备

```
def get_devices_Android(self):
    lines = CommonUtils.get_cmd_lines('adb devices')
    return [line.split()[0] for line in lines if line and not "devices" in line]
```

调用：

```
devices = self.get_devices_Android()
```

相关命令输出举例：

```
→ ~ adb devices
List of devices attached
```

```
8c8a4d4d      device
```

## 获取当前正在运行的app和页面activity

```
def get_PackageActivity_Android(self):
    # adb直接获取当前活跃app及activity
    package, activity = "", ""
    cmds = ['dumpsys activity |grep {}'.format(item) for item in ['mFocusedActivity', 'mResumedActivity']]
    for cmd in cmds:
        output = self.driver.shell(cmd).output
        result = re.search("u0(.*)/", output)
        package = result.group(1).strip() if result else ""
        result = re.search("/(.*)\s", output)
        activity = result.group(1).strip() if result else ""
        if package and activity:
            return package, activity
    return package, activity
```

调用：

```
package, activity = self.get_PackageActivity()
```

## 获取已安装app列表

```
def get_packages(self):
    # 获取已安装的app的appPackage列表
    if isinstance(self.driver, u2.UIAutomatorServer):
        text = self.driver.shell("pm list packages")[0]
        return re.findall(':(.*?)\n', text)
    else:
        cmd = 'adb -s {} shell pm list packages'.format(self.device)
        lines = CommonUtils.get_cmd_lines(cmd)
        return {line.split(":")[1].strip() for line in lines}
```

调用：

```
packages = self.get_packages()
```

## 安装安卓app

```
def install_app_Android(self, item, packages=None):
    if packages is None:
        packages = self.get_packages()
    if item[1] in packages:
```

```
        logging.info("AppName {0} is already installed".format(item[0]))
    else:
        logging.info("start to install app in {}".format(os.path.basename(self.arg_options.task)))
        os.system("adb -s {0} install {1}".format(self.device, item[3]))
```

调用：

```
def install_app(self, item, packages=None):
    # 安装app
    if self.isAndroid:
        return self.install_app_Android(item, packages)
```

## 卸载安卓app

```
def uninstall_app(self, item):
    # 卸载安装包
    os.system("adb -s {0} uninstall {1}".format(self.device, item[1]))
    logging.info("uninstall app {} end".format(item[1]))
```

调用：

```
if item[1] in packages:
    self.uninstall_app(item)
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 设备相关

此处整理出，和安卓设备相关的一些通用功能的函数和调用举例。

### 获取安卓设备信息

```
def getDeviceInfo(self):
    return self.driver.device_info
```

调用：

```
deviceInfo = self.getDeviceInfo()
```

### 获取安卓版本

```
def getAndroidVersion(self):
    """返回安卓版本号，float值：6.0, 9.0"""
    deviceInfo = self.getDeviceInfo()
    logging.debug("deviceInfo=%s" % deviceInfo)
    androidVersionStr = deviceInfo["version"] # '6.0'
    androidVersionFloat = float(androidVersionStr)
    return androidVersionFloat
```

调用：

```
curAndroidVersionFloat = self.getAndroidVersion()
ANDROID_VERSION_NEED_RESTART_U2 = 7.0
if curAndroidVersionFloat <= ANDROID_VERSION_NEED_RESTART_U2:
    isNeedRestartU2 = True
```

### 获取安卓屏幕分辨率

```
def getCurScreenResolution(self):
    """Get current screen resolution"""
    driverInfo = self.driver.info
    logging.debug("driverInfo=%s" % driverInfo)
    # displayWidth = driverInfo["displayWidth"]
    # displayHeight = driverInfo["displayHeight"]
    # logging.info("displayWidth=%s, displayHeight=%s", displayWidth, displayHeight)
    # deviceInfo = self.driver.device_info
    deviceInfo = self.getDeviceInfo()
    logging.debug("deviceInfo=%s" % deviceInfo)
    deviceDisplay = deviceInfo["display"]
```

```

logging.debug("deviceDisplay=%s" % deviceDisplay)
screenWidth = deviceDisplay["width"]
screenHeight = deviceDisplay["height"]
logging.debug("screenWidth=%s, screenHeight=%s", screenWidth, screenHeight)
if driverInfo["displayRotation"]:
    curScreenWidth = screenHeight
    curScreenHeight = screenWidth
else:
    curScreenWidth = screenWidth
    curScreenHeight = screenHeight
logging.debug("curScreenWidth=%s, curScreenHeight=%s", curScreenWidth, curScreenHeight)

return (curScreenWidth, curScreenHeight)

```

调用：

```
screenWidth, screenHeight = self.getCurScreenResolution()
```

输出：

```

[191213 16:16:13][AppCrawler.py 209] driverInfo {'currentPackageName': 'com.zzt1.dragon.cs',
'displayHeight': 720, 'displayRotation': 1, 'displaySizeDpX': 640, 'displaySizeDpY': 360,
'displayWidth': 1196, 'productName': 'DIG-AL00', 'sdkInt': 23, 'naturalOrientation': False}
[191213 16:16:13][AppCrawler.py 212] displayWidth 1196, displayHeight 720
[191213 16:16:13][AppCrawler.py 214] deviceInfo {'udid': 'DWH9X17124W03779-18:d2:76:f4:9e:7b-DIG-AL00',
'version': '6.0', 'serial': 'DWH9X17124W03779', 'brand': 'HUAWEI', 'model': 'DIG-AL00',
'hwaddr': '18:d2:76:f4:9e:7b', 'port': 7912, 'sdk': 23, 'agentVersion': '0.7.4',
'display': {'width': 720, 'height': 1280}, 'battery': {'acPowered': True, 'usbPowered': False,
'wirelessPowered': False, 'status': 2, 'health': 2, 'present': True, 'level': 26,
'scale': 100, 'voltage': 3892, 'temperature': 315, 'technology': 'Li-ion'}, 'memory': {'total': 2879816, 'around': '3 GB'}, 'cpu': {'cores': 8, 'hardware': 'Qualcomm Technologies, Inc MSM8940'},
'arch': '', 'owner': None, 'presenceChangedAt': '0001-01-01T00:00:00Z', 'usingBeganAt': '0001-01-01T00:00:00Z',
'product': None, 'provider': None}
[191213 16:16:13][AppCrawler.py 216] deviceDisplay {'width': 720, 'height': 1280}
[191213 16:16:13][AppCrawler.py 219] screenWidth 720, screenHeight 1280
[191213 16:16:13][AppCrawler.py 226] curScreenWidth 1280, curScreenHeight 720

```

得到了我们要的：屏幕的宽度和高度

且知道了是当前屏幕是否已旋转（从安卓手机的默认的竖屏，旋转成游戏的横屏）了

另外，当屏幕故意不去旋转，回到默认竖屏后：



此时

- 旋转为 False
  - displayRotation : 0
  - naturalOrientation : True
- 但 displayHeight 值有变化: 是 1208
  - 却不是 1280

如图:

The screenshot shows the PyCharm IDE with a Python script open. The code is related to device information and screen dimensions. On the right side of the interface, there is a screenshot of a mobile application's settings screen, which includes a 'Screen' section with various options like 'Screen orientation' and 'Screen size'.

```

197     # payYuanStr = "%(d+元)"
198     # payYuanImgPath = "debug/安卓/app/Gam
199     # foundPayYuanStr, foundResultList,
200
201     # # for debug
202     # # jianLingLongImgPath = "debug/安卓
203     # self._isAnnouncementPage(jianLing
204     # zhizhunTuLongImgPath = "debug/安卓
205     # self._isAnnouncementPage(zhizhunTuL
206
207
208     driverInfo = self.driver.info
209     logging.info("driverInfo=%s" % driverInfo)
210     displayWidth = driverInfo["displayWidth"]
211     displayHeight = driverInfo["displayHeight"]
212     logging.info("displayWidth=%s, displayHeight=%s", displayWidth, displayHeight)
213     deviceInfo = self.driver.device_info
214     logging.info("deviceInfo=%s" % deviceInfo)
215     deviceDisplay = deviceInfo["display"]
216     logging.info("deviceDisplay=%s" % deviceDisplay)
217     screenWidth = deviceDisplay["width"]
218     screenHeight = deviceDisplay["height"]
219     logging.info("screenWidth=%s, screenHeight=%s", screenWidth, screenHeight)
220     if driverInfo["displayRotation"]:
221         curScreenWidth = screenHeight
222         curScreenHeight = screenWidth
223     else:
224         curScreenWidth = screenWidth
225         curScreenHeight = screenHeight
226     logging.info("curScreenWidth=%s, curScreenHeight=%s", curScreenWidth, curScreenHeight)
227
228     # for debug
229     while self._isAnnouncementPage():

```

问题 438 输出 调试控制台 终端

[191213 16:20:47][AppCrawler.py 209] driverInfo: {'currentPackageName': 'com.huawei.android.launcher', 'displayHeight': 1208, 'displayRotation': 0, 'displaySizeDpX': 360, 'displaySizeDpY': 640, 'displayWidth': 720, 'productName': 'DIG-AL00', 'sdkInt': 23, 'naturalOrientation': True}

[191213 16:20:50][AppCrawler.py 212] displayWidth=720, displayHeight=1208

[191213 16:23:17][AppCrawler.py 214] deviceInfo: {'udid': 'DWH9X17124W03779-18:d2:76:f4:9e:7b-DIG-AL00', 'version': '6.0', 'serial': 'DWH9X17124W03779', 'brand': 'HUAWEI', 'model': 'DIG-AL00', 'hwaddr': '18:d2:76:f4:9e:7b', 'port': 7912, 'sdk': 23, 'agentVersion': '0.7.4', 'display': {'width': 720, 'height': 1280}, 'battery': {'acPowered': True, 'usbPowered': False, 'wirelessPowered': False, 'status': 2, 'health': 2, 'present': True, 'level': 26, 'scale': 100, 'voltage': 3892, 'temperature': 315, 'technology': 'Li-ion'}, 'memory': {'total': 2879816, 'around': '3 GB'}, 'cpu': {'cores': 8, 'hardware': 'Qualcomm Technologies, Inc MSM8940'}, 'arch': '', 'owner': None, 'presenceChangedAt': '0001-01-01T00:00:00Z', 'usingBeganAt': '0001-01-01T00:00:00Z', 'product': None}

[191213 16:23:19][AppCrawler.py 216] deviceDisplay={'width': 720, 'height': 1280}

[191213 16:23:21][AppCrawler.py 219] screenWidth=720, screenHeight=1280

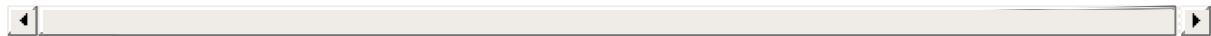
[191213 16:23:24][AppCrawler.py 226] curScreenWidth=720, curScreenHeight=1280

详细log:

```

[191213 16:20:47][AppCrawler.py 209] driverInfo: {'currentPackageName': 'com.huawei.android.launcher', 'displayHeight': 1208, 'displayRotation': 0, 'displaySizeDpX': 360, 'displaySizeDpY': 640, 'displayWidth': 720, 'productName': 'DIG-AL00', 'sdkInt': 23, 'naturalOrientation': True}
[191213 16:20:50][AppCrawler.py 212] displayWidth 720, displayHeight 1208
[191213 16:23:17][AppCrawler.py 214] deviceInfo: {'udid': 'DWH9X17124W03779-18:d2:76:f4:9e:7b-DIG-AL00', 'version': '6.0', 'serial': 'DWH9X17124W03779', 'brand': 'HUAWEI', 'model': 'DIG-AL00', 'hwaddr': '18:d2:76:f4:9e:7b', 'port': 7912, 'sdk': 23, 'agentVersion': '0.7.4', 'display': {'width': 720, 'height': 1280}, 'battery': {'acPowered': True, 'usbPowered': False, 'wirelessPowered': False, 'status': 2, 'health': 2, 'present': True, 'level': 26, 'scale': 100, 'voltage': 3892, 'temperature': 315, 'technology': 'Li-ion'}, 'memory': {'total': 2879816, 'around': '3 GB'}, 'cpu': {'cores': 8, 'hardware': 'Qualcomm Technologies, Inc MSM8940'}, 'arch': '', 'owner': None, 'presenceChangedAt': '0001-01-01T00:00:00Z', 'usingBeganAt': '0001-01-01T00:00:00Z', 'product': None}
[191213 16:23:19][AppCrawler.py 216] deviceDisplay={'width': 720, 'height': 1280}
[191213 16:23:21][AppCrawler.py 219] screenWidth 720, screenHeight 1280
[191213 16:23:24][AppCrawler.py 226] curScreenWidth=720, curScreenHeight=1280

```



详见：

【已解决】uiautomator2获取当前屏幕的宽和高即屏幕大小分辨率信息

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28

## 其他

### 坐标值

boundsToCenterPoint: 从bounds算出中间坐标值

```
def boundsToCenterPoint(self, boundsStr):
    """
        从bounds转换出中间点位置坐标

    Example:
        bounds: '[156,1522][912,2027]'
        return: [534, 1774]
    """
    filterStr = re.sub('[\[,\]]', " ", boundsStr)
    boundStrList = filterStr.split()
    boundMap = map(int, boundStrList)
    boundIntList = list(boundMap)
    x0 = boundIntList[0]
    y0 = boundIntList[1]
    x1 = boundIntList[2]
    y1 = boundIntList[3]
    centerPoint = [(x1 + x0)//2, (y1 + y0)//2]
    return centerPoint
```

调用:

```
centerPoint = self.boundsToCenterPoint(locatorBounds)
self.tap(centerPoint)
```

### xpath

getCurPageSource: 获取当前页面源码xml

```
def getCurPageSource(self):
    # curPageSrcXml = self.driver.dump_hierarchy()
    curPageSrcXml = self.driver.dump_hierarchy(compressed=False, pretty=False)

    # output, exitCode = self.driver.shell(["adb", "shell", "uiautomator", "dump"])
    # output, exitCode = self.driver.shell(["uiautomator", "dump"])
    # output, exitCode = self.driver.shell("uiautomator dump")
    # output, exitCode = self.driver.shell(["shell", "uiautomator", "dump"])
    # curPageSrcXml = output

    return curPageSrcXml
```

调用：

```
curPageSrcXml = self.getCurPageSource()
```

## 查找元素

findAndClickNode： 查找当前节点的父级符合条件的节点 并点击

```
def findAndClickNode(self, curNodeXPath):
    """
        寻找可以clickable=true的当前或父级元素，并点击

        注：主要用于当节点clickable=false，点击无效时，使用此方法
    """
    foundAndClicked = False
    matchDict = {"clickable": "true"}
    clickableParentNode = self.findParentNode(curNodeXPath, curNodeXPath, matchDict)
    if clickableParentNode:
        foundNodeAttrib = clickableParentNode.attrib
        clickableParentNode.click()
        foundAndClicked = True
        logging.info("clicked element [%s] found by [xpath=%s, match=%s]", foundNodeAttrib,
                     curNodeXPath, matchDict)
    else:
        logging.warning("Fail click %s for not found %s(parent) node", curNodeXPath, matchDict)

    return foundAndClicked
```

调用：

```
if curNodeXPath:
    foundAndClicked = self.findAndClickNode(curNodeXPath)
```

相关函数：

findParentNode： 寻找父节点

```
def findParentNode(self, curNodeXPath, matchDict, maxUpLevel=3):
    """
        寻找符合特定条件的父级节点，最多向上找3级

        如果当前节点符合条件，则返回当前节点
    """
    matchNode = None
```

```

try:
    curNode = self.driver.xpath(curNodeXPath).get()
    curNodeAttrib = curNode.attrib # .attrib contain 'clickable'
    # curNodeInfo = curNode.info # .info not contain 'clickable'
    isCurMatch = self.IsMatchNode(curNodeAttrib, matchDict)
    if isCurMatch:
        # current is match
        matchNode = curNode
    else:
        # try parent nodes
        curUpLevel = 1
        curParentNodeXPath = curNodeXPath
        while(curUpLevel <= maxUpLevel):
            curParentNodeXPath += "/.."
            curParentNode = self.driver.xpath(curParentNodeXPath).get()
            curParentNodeAttrib = curParentNode.attrib
            isCurParentMatch = self.IsMatchNode(curParentNodeAttrib, matchDict)
            if isCurParentMatch:
                matchNode = curParentNode
                break
            curUpLevel += 1

except XPathElementNotFoundError as xpathNotFoundErr:
    logging.error("XPathElementNotFoundError: %s", xpathNotFoundErr)

if not matchNode:
    logging.warning("Not found match parent for xpath=%s and match=%s", curNodeXPath,
                    matchDict)

return matchNode

```

isMatchNode: 节点是否匹配

```

def IsMatchNode(self, curNodeAttrib, toMatchInfo):
    """判断当前节点属性是否满足条件"""
    isAllMatch = True
    for eachKey, eachToMatchValue in toMatchInfo.items():
        if eachKey not in curNodeAttrib:
            isAllMatch = False
            break

        curValue = curNodeAttrib[eachKey]
        if curValue != eachToMatchValue:
            isAllMatch = False
            break

    return isAllMatch

```

findAndClickTextNode: 寻找节点并点击

```

def findAndClickTextNode(self, text):
    """
        对于text类型节点: android.widget.TextView, text=xxx
        寻找可以clickable=true的当前或父级元素，并点击

    注: 主要用于当text=xxx的节点clickable=false, 点击无效时, 使用此方法
    """

    curTextNodeXpath = "//android.widget.TextView[@text='%s']" % text
    self.findAndClickNode(curTextNodeXpath)

```

## xpathFindElement: 用xpath查找元素

```

def xpathFindElement(self, curClass=None, curId=None, curBounds=None):
    """
        find element by xpath

        return value type
        is: u2.xpath.XMLElement
        not: u2.session.UiObject
    """

    foundElement = None
    curXPath = self.generateElementXPath(curClass, curId, curBounds)
    try:
        foundElement = self.driver.xpath(curXPath).get()
    except XPathElementNotFoundError as xpathNotFoundError:
        logging.error("XPathElementNotFoundError: %s from %s", xpathNotFoundError, curXPath)

    return foundElement

```

调用:

(1)

```

foundElement = self.xpathFindElement(locatorClass, locatorId, locatorBounds)

```

相关函数:

## generateElementXPath: 生成元素xpath

```

def generateElementXPath(self, curClass=None, curId=None, curBounds=None):
    """generate element xpath"""
    # nodeXPath = ""
    # if locatorClass:
    #     nodeXPath = "//%s[@bounds='%s']" % (locatorClass, locatorBounds) # "//android.widget.TextView[@bounds='[191,2060][430,2135]']"
    # elif locatorId:

```

```

#     nodeXPath = "//*[@resource-id='%s' and @bounds='%s']" % (locatorId, locatorBounds)
# else:
#     nodeXPath = "//*[@bounds='%s'" % locatorBounds

classRule = "*"
if curClass:
    classRule = curClass # 'android.widget.ImageView'

propertyRule = ""
if curId:
    propertyRule += "@resource-id='%s'" % curId
    # "@resource-id='com.netease.newsreader.activity:id/hs'"

if curBounds:
    if propertyRule:
        propertyRule += " and "

    propertyRule += "@bounds='%s'" % curBounds
    # "@resource-id='com.netease.newsreader.activity:id/hs' and @bounds='[75,2098][141,2134]'""

# TODO: add other support: text, desc, instance, ...
curXPath = "//%s[%s]" % (classRule, propertyRule)
# "//android.widget.ImageView[@resource-id='com.netease.newsreader.activity:id/hs' and
# @bounds='[75,2098][141,2134]']"

return curXPath

```

调用:

```

curClassName = None
curResId = None
curBoundsStr = None

# curAttrib = foundElement.attrib
# AttributeError: 'UiObject' object has no attribute 'attrib'
if hasattr(foundElement, "attrib"):
    curAttrib = foundElement.attrib
    # {'index': '0', 'text': '', 'resource-id': 'com.netease.newsreader.activity:id/hs', 'package': 'com.netease.newsreader.activity', 'content-desc': '', 'checkable': 'false', 'checked': 'false', 'clickable': 'false', 'enabled': 'true', 'focusable': 'false', 'focused': 'false', 'scrollable': 'false', 'long-clickable': 'false', 'password': 'false', 'selected': 'true', 'visible-to-user': 'true', 'bounds': '[75,2098][141,2134]'}
    curResId = curAttrib["resource-id"]
    curBoundsStr = curAttrib["bounds"]
else:
    # # for debug
    # self.debugPrintElement(foundElement, "no attrib")
    logging.debug("")

curInfo = foundElement.info

```

```
# {'bounds': {'bottom': 2134, 'left': 75, 'right': 141, 'top': 2098}, 'className': 'android.widget.ImageView', 'contentDescription': '', 'enabled': 'true', 'focusable': 'false', 'focused': 'false', 'longClickable': 'false', 'packageName': 'com.netease.newsreader.activity', 'scrollable': 'false', 'selected': 'true', 'text': ''}  
if not curClassname:  
    curClassname = curInfo["className"] # 'android.widget.ImageView'  
  
if not curBoundsStr:  
    boundsDict = curInfo["bounds"]  
    x0 = boundsDict["left"]  
    y0 = boundsDict["top"]  
    x1 = boundsDict["right"]  
    y1 = boundsDict["bottom"]  
    curBoundsStr = "[%d,%d][%d,%d]" % (x0, y0, x1, y1)  
    # '[75,2098][141,2134]'  
  
if not curResId:  
    if "resourceName" in curInfo:  
        curResId = curInfo["resourceName"] # 'com.netease.newsreader.activity:id/bn5'  
  
curNodeXPath = self.generateElementXPath(  
    curClass curClassname,  
    curId curResId,  
    curBounds curBoundsStr,  
)
```

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2020-08-09 10:14:28

## 附录

下面列出相关参考资料。

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新: 2020-08-09 10:14:28

## 参考资料

- 【记录】 mac中用pipenv安装uiautomator2
- 【未解决】 给安卓手机小米9中欢乐大作战的游戏实现自动挂机
- 【已解决】 红米Note8Pro的uiautomator2初始化出错： OSError Errno Uiautomator started failed
- 【已解决】 uiautomator2获取当前屏幕的宽和高即屏幕大小分辨率信息
- 【已解决】 uiautomator2中如何获取到当前画面的截图文件
- 【部分解决】 python的uiautomator2中set\_text导致输入法变化无法顺利输入文字
- 【已解决】 uiautomator2中点击华为手机中系统自带Swype的输入法中的搜索按钮
- 【已解决】 python的uiautomator2报错： uiautomator2.exceptions.JsonRpcError -32601 Method not found data injectInputEvent
- 【未解决】 uiautomator2中dump\_hierarchy中只能获取到页面的部分的xml源码
- 【已解决】 搞懂uiautomator-server中最终的底层实现dumpWindowHierarchy的处理返回页面数据的逻辑
- 【已解决】 uiautomator2中导出页面源码中NAF是什么意思
- 【未解决】 如何确保uiautomator2的dump\_hierarchy能导出页面中NAF的元素节点
- 【无法解决】 adb发送密码无法解锁安卓手机屏幕
- 【未解决】 自动抓包工具抓包公众号买单吧某个元素通过class+instance定位不到
- 【已解决】 uiautomator2用click点击微信中的通讯录不起作用
- 【已解决】 用weditor实时查看安卓当前页面中的xml源码
- 【已解决】 Mac中安装uiautomator2的UI界面工具： weditor
- 【未解决】 如何修改Android项目android-uiautomator-server的Java代码并重新打包生成2个apk
- 
- [uiautomator | Android Developers](#)
- [Android 手机自动化测试工具有哪几种？ - 知乎](#)
- [一种 Android 端 Web 多进程情况下支持 Web 自动化测试的方法 - 云+社区 - 腾讯云](#)
- [ATX 文档 - iOS 控件操作 API · TesterHome](#)
- [Manual Init · openatx/uiautomator2 Wiki](#)
- 

crifan.com, 使用[署名4.0国际\(CC BY 4.0\)协议](#)发布 all right reserved, powered by Gitbook最后更新：2020-08-09 10:14:28