目录

安卓自动化测试利器：uiautomator2

- 最新版本：`v0.6`
- 更新时间：`20200620`

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。

简介

总结安卓设备自动化测试领域好用的库uiautomator2，介绍如何搭建环境，如何连接安卓真机测试，如何用weditor去辅助调试，以及放出自己总结的各种常用函数和具体用法。

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

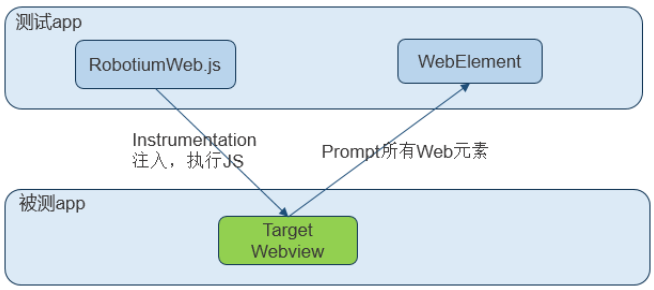
版权说明

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。

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

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
```

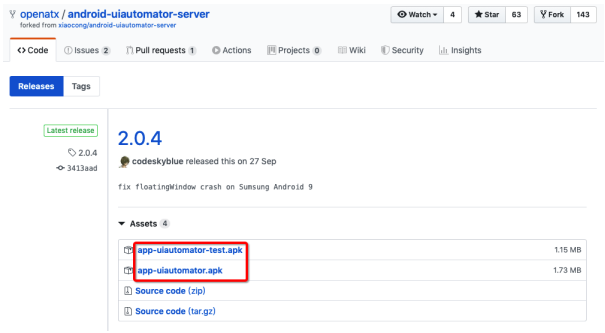
输出举例：

```
➜  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
>>> import uiautomator2 as u2
>>> d = u2.connect('8c8a4d4d')
conn=<urllib3.connection.HTTPConnection object at 0x1077f4
```

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httprpc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

- 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.uiautomato
[I 200218 13:56:02 init:300] - app-uiautomator.apk installe
[I 200218 13:56:14 init:300] - app-uiautomator-test.apk ins
[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：

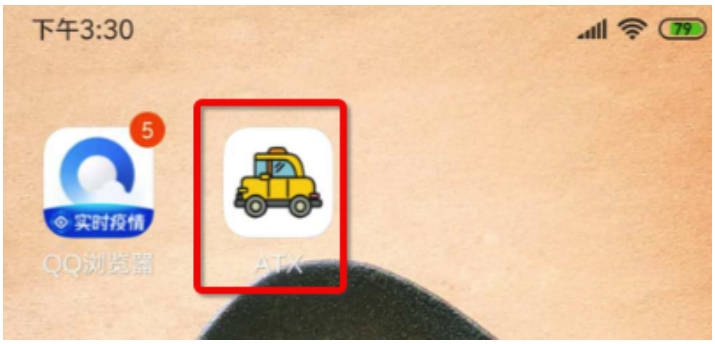
```
[200217 14:45:33][DevicesMethods.py 11 ] start init driver
[I 200217 14:45:37 init:132] uiautomator2 version: 2.5.3
[I 200217 14:45:37 init:317] Install minicap, minitouch
minicap.so |##########| 67.1K/67.1K
[I 200217 14:45:37 init:330] Install com.github.uiautomato
[I 200217 14:45:38 init:300] - app-uiautomator.apk installe
[I 200217 14:45:38 init:300] - app-uiautomator-test.apk ins
[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=hmucaei75ptk7szs)
```

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：

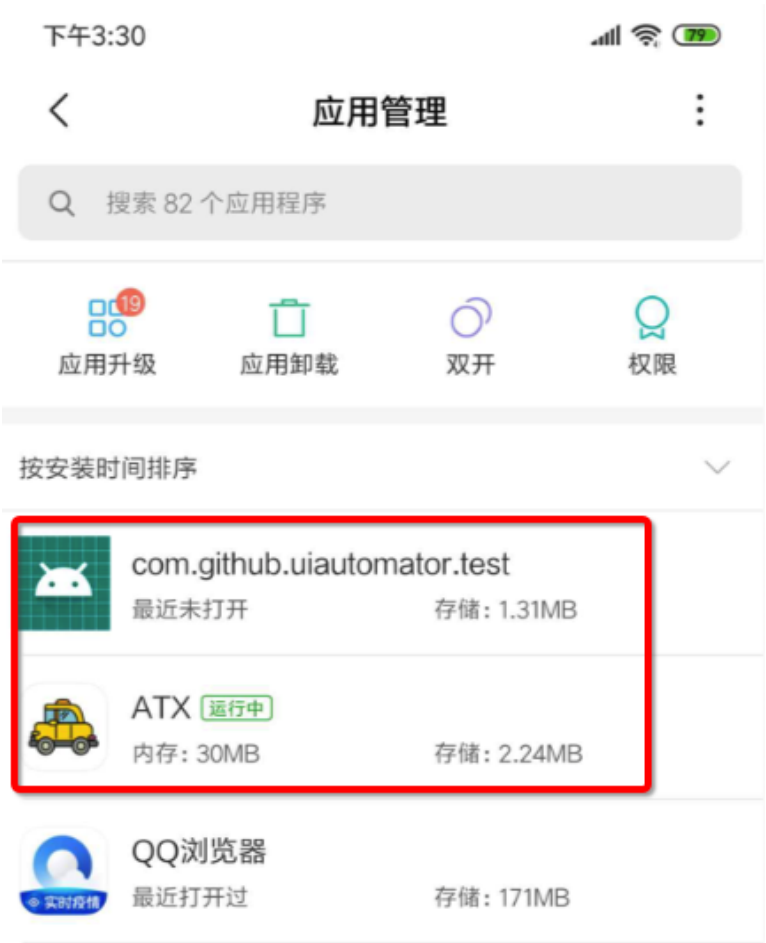
- ATX
  - 桌面图标
    - 

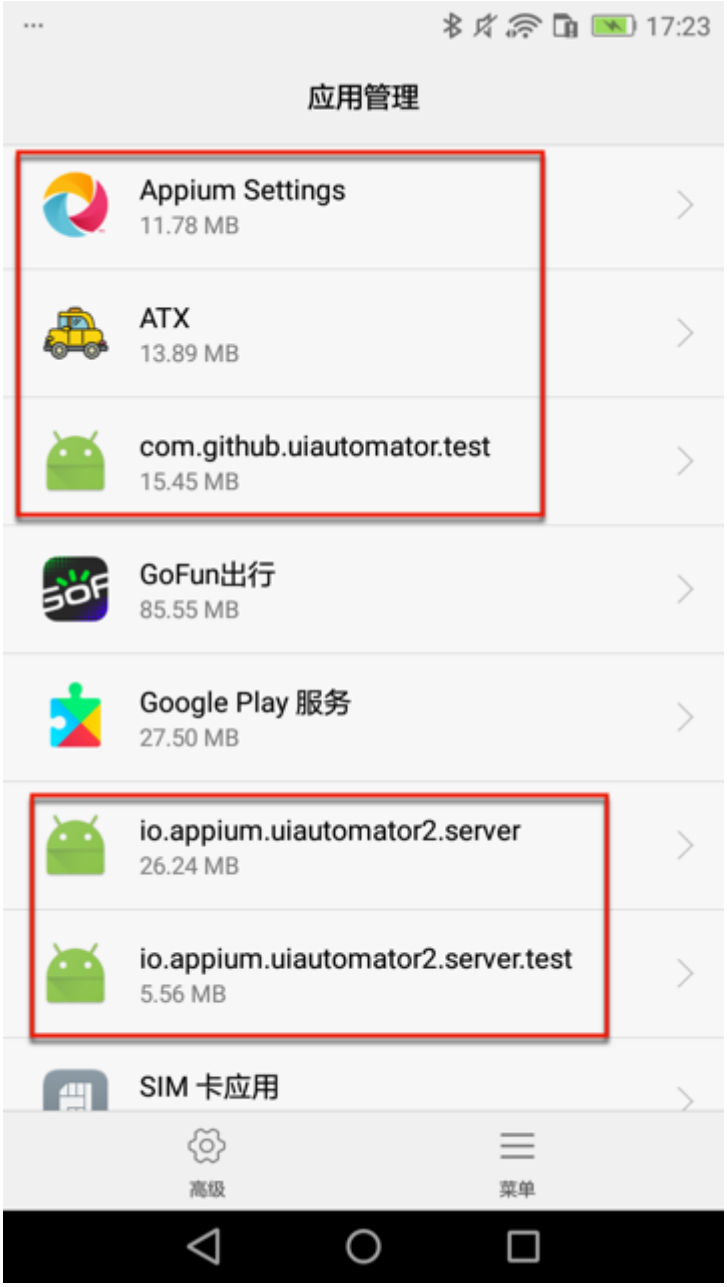
  - 安装期间需要手动点击 继续安装
    - 

- 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`

核心功能

接着介绍uiautomator2的一些常用的核心的功能。

监听

其中一个很常用的，很好用的功能就是：**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。

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。

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。

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。

点击元素

找到元素后，往往涉及到点击元素。

输入内容

找到元素后，也会遇到需要输入内容的情况。

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。

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。

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。

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。

通用功能函数

开发uiautomator2期间，把一些常用的功能，封装成了函数，贴出来，并给出具体调用方式即实际案例，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。

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, sh
        try:
            resultStr = consoleOutputByte.decode("utf-8")
        except UnicodeDecodeError:
            # TODO: use chardet auto detect encoding
            # consoleOutputStr = consoleOutputByte.decode('
            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
```

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'.forr
    cmd = 'adb -s {} shell getprop ro.product.name'.format(
    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 r
```

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 it
    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 {0} shell pm list packages'.format(se
        lines = CommonUtils.get_cmd_lines(cmd)
        return {line.split(":")[1].strip() for line in line
```

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 alread installed".forr
    else:
        logging.info("start to install app in {}".format(os
        os.system("adb -s {0} install {1}".format(self.devi
```

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
    logging.info("uninstall app {} end".format(item[1]))
```

调用：

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

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_RESTA
        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", dis
    # 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", screer
    if driverInfo["displayRotation"]:
        curScreenWidth = screenHeight
        curScreenHeight = screenWidth
    else:
        curScreenWidth = screenWidth
        curScreenHeight = screenHeight
    logging.debug("curScreenWidth=%s, curScreenHeight=%s",

    return (curScreenWidth, curScreenHeight)
```

调用：

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

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=

    # output, exitCode = self.driver.shell(["adb", "shell",
    # output, exitCode = self.driver.shell(["uiautomator",
    # output, exitCode = self.driver.shell("uiautomator dur
    # output, exitCode = self.driver.shell(["shell", "uiaut
    # 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=
    if clickableParentNode:
        foundNodeAttrib = clickableParentNode.attrib
        clickableParentNode.click()
        foundAndClicked = True
        logging.info("clicked element [%s] found by [xpath=
    else:
        logging.warning("Fail click %s for not found %s(par

    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, maxUpLeve
    """
        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 '
        # curNodeInfo = curNode.info # .info not contain '
        isCurMatch = self.isMatchNode(curNodeAttrib, matchD
        if isCurMatch:
            # current is match
            matchNode = curNode
        else:
            # try parent nodes
            curUpLevel = 1
            curParentNodeXpath = curNodeXpath
            while(curUpLevel <= maxUpLevel):
                curParentNodeXpath += "/.."
                curParentNode = self.driver.xpath(curParent
                curParentNodeAttrib = curParentNode.attrib
                isCurParentMatch = self.isMatchNode(curPare
                if isCurParentMatch:
                    matchNode = curParentNode
                    break

                curUpLevel += 1

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

    if not matchNode:
        logging.warning("Not found match parent for xpath=%

    return matchNode
```

isMatchNode：节点是否匹配

```
def isMatchNode(self, curNodeAttrib, toMathInfo):
    """判断当前节点属性是否满足条件"""
    isAllMatch = True
    for eachKey, eachToMatchValue in toMathInfo.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=x
            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
        """
        curTextNodeXpath = "//android.widget.TextView[@text
        self.findAndClickNode(curTextNodeXpath)
```

xpathFindElement：用xpath查找元素

```
def xpathFindElement(self, curClass=None, curId=None, curBo
    """
        find element by xpath

        return value type
            is: u2.xpath.XMLElement
            not: u2.session.UiObject
    """
    foundElement = None
    curXpath = self.generateElementXpath(curClass=curClass,
    try:
        foundElement = self.driver.xpath(curXpath).get()
    except XPathElementNotFoundError as xpathNotFoundErr:
        logging.error("XPathElementNotFoundError: %s from %

    return foundElement
```

调用：

（1）

```
foundElement = self.xpathFindElement(curClass=locatorClass,
```

相关函数：

generateElementXpath：生成元素xpath

```
def generateElementXpath(self, curClass=None, curId=None, c
    """generate element xpath"""
    # nodeXpath = ""
    # if locatorClass:
    #     nodeXpath =  "//%s[@bounds='%s']" % (locatorClass
    # elif locatorId:
    #     nodeXpath =  "//*[@resource-id='%s' and @bounds=
    # 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

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

        propertyRule += "@bounds='%s'" % curBounds
        # "@resource-id='com.netease.newsreader.activity:id

    # TODO: add other support: text, desc, instance, ...
    curXpath = "//%s[%s]" % (classRule, propertyRule)
    # "//android.widget.ImageView[@resource-id='com.netease

    return curXpath
```

调用：

```
curClassname = None
curResId = None
curBoundsStr = None

# curAttrib = foundElement.attrib
# AttributeError: 'UiObject' object has no attribute 'attr
if hasattr(foundElement, "attrib"):
    curAttrib = foundElement.attrib
    # {'index': '0', 'text': '', 'resource-id': 'com.neteas
    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, 'to
if not curClassname:
    curClassname = curInfo["className"] # 'android.widget.I

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.n

curNodeXpath = self.generateElementXpath(
    curClass=curClassname,
    curId=curResId,
    curBounds=curBoundsStr,
)
```

附录

下面列出相关参考资料。

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
-
- 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
-