**python + uiautomatorX 自动化培训**

**作者：      李金洲**

**日期:   2017/10/17**

* **Connect an device**

# coding: utf-8

import atx

d = atx.connect() # 如果多个手机连接电脑，则需要填入对应的设备号

* **Take screenshot**

     python -m atx gui

* **四个环境变量**

ATX\_ADB\_SERIALNO

ATX\_ADB\_HOST

ATX\_ADB\_PORT

ATX\_PLATFORM 默认是 android

* **App start and stop**

package\_name = 'com.example.game'

d.stop\_app(package\_name)

# d.stop\_app(package\_name, clear=True) # stop and remove app data (only Android)

d.start\_app(package\_name)

* **Execute shell command (Only Android)**

 d.adb\_cmd(['pull', '/data/local/tmp/hi.txt']) # default timeout 30s, use timeout=None to set unlimited time

 d.adb\_shell(['uptime'])

 print d.wlan\_ip # 获取手机的Wlan IP

 print d.current\_app() # 获取当前运行应用的package name和activity以及运行的pid

 # Expect: AppInfo(package='com.miui.mihome2', activity='com.android.launcher2.Launcher', pid=634)

* **图片查找与点击**

# find image position

if d.exists('button.png'): # 判断截图是否在屏幕中出现, 反馈查找到的坐标

    print 'founded'

# take screenshot

d.screenshot('screen.1920x1080.png') # Save screenshot as file

# click position

d.click(50, 100) # 模拟点击 x, y

# long click

d.long\_click(50, 100) # only works on android for now

* **click\_image函数**

# click image, if "button.png" not found, exception will be raise.

 d.click\_image("button.png")

 # add description (also used for report generate)

 d.click\_image("button.png", desc="I love click")

 # click image, if "button.png" not found, will return None

 d.click\_image("button.png", safe=True)

 # click image with long click

 d.click\_image("button.png", action='long\_click')

 # 不等待的图片点击, 如果图片不存在直接返回None

 d.click\_nowait('button.png')

 # 文件名添加截图手机的分辨率, 脚本运行在其他分辨率的手机上时可以自动适应

 d.click\_image("button.1920x1080.png")

 # 等价于

 d.click\_image(atx.Pattern('button.png', rsl=(1080, 1920)))

 # 文件名中添加偏移量, 格式为 <L|R><number><T|B><number>.png

 # 其中 L: Left, R: Right, T: Top, B: Bottom

 # number为百分比

 # 所以 R20T50代表，点击为止从图片中心向右偏移20%并且向上偏移50%

 d.click\_image("button.R20T50.png")

 # same as

 d.click\_image("button.png", offset=(0.2, -0.5))

# Full example

d.click\_image("button.png",

       offset=(0.2, 0.5),

       action="long\_click",

       safe=True,

       desc="I love click",

       method='template',

       threshold=0.8)

 # if image not show in 10s, ImageNotFoundError will raised

 try:

       d.click\_image('button.png', timeout=10.0)

 except atx.ImageNotFoundError:

       print('Image not found')

 # 在特定的区域内查找匹配的图像(IDE暂时还不支持如此高级的操作)

 nd = d.region(atx.Bounds(50, 50, 180, 300))

 print nd.match('folder.png')

* **点击和滑动**

# click by UI component

d(text='Enter').click()

d(text='Enter').sibling(className='android.widget.ImageView').click()

# swipe from (sx, sy) to (ex, ey)

d.swipe(sx, sy, ex, ey)

# swipe from (sx, sy) to (ex, ey) with 10 steps

d.swipe(sx, sy, ex, ey, steps=10)

* **文本的输入 (only Android)**

 d.type("hello world")

 d.type("atx", enter=True) # perform enter after input

* **Common settings**

# 配置截图图片的手机分辨率

 d.resolution = (1920, 1080)

 print d.resolution

 # expect output: (1080, 1920) 实际获取到的值会把小的放在前面

 # this is default (first check minicap and then check uiautomator)

 d.screenshot\_method = atx.SCREENSHOT\_METHOD\_AUTO # 默认

 # d.screenshot\_method = atx.SCREENSHOT\_METHOD\_UIAUTOMATOR # 可选

 # d.screenshot\_method = atx.SCREENSHOT\_METHOD\_MINICAP # 可选

 d.image\_match\_method = atx.IMAGE\_MATCH\_METHOD\_TMPL # 模版匹配, 默认

 # d.image\_match\_method = atx.IMAGE\_MATCH\_METHOD\_SIFT # 特征点匹配, 可选

 # d.image\_match\_threshold = 0.8 # 默认(模版匹配相似度)

 d.rotation = None # default auto detect, 这个配置一下比较好，自动识别有时候识别不出来

 # 0: home key bottom(normal)

 # 1: home key right

 # 2: home key top

 # 3: home key left

 # 图片路径查找(实验性功能)

 d.image\_path = ['.'] # 默认

 # 主要用在希望代码和图片放在不同目录的情况, 如代码结构

 # /--

 #   |-- test.py

 #   |-- images/

 #          |- photo1.png

 #          `- photo2.png

 #

 # test.py 中的关键性代码

 d.image\_path = ['.', 'images']

 d.click\_image('photo1.png')

 d.click\_image('photo2.png')

* **events函数调用事件**

 def my\_listener(event):

  print 'out:', event

 d.add\_listener(my\_listener, atx.EVENT\_SCREENSHOT)

 d.screenshot()

 # expect output:

 # out: HookEvent(flag=8, args=(), kwargs={})

* **Command line tools**

python -m atx --help

**python -m atx gui**

**python -m atx apkparse demo.apk**

**python -m atx install demo.apk**

**python -m atx install --start demo.apk**

**python -m atx screencap -o screen.png**

**python -m atx info**

* **Retrieve the device info**

d.info

* **Trun on/off screen**

# Turn on screen

d.screen.on()

# Turn off screen

d.screen.off()

# wakeup the device

d.wakeup()

# sleep the device, same as turning off the screen.

d.sleep()

* **Press hard/soft key**

# press home key

d.press.home()

# press back key

d.press.back()

# the normal way to press back key

d.press("back")

# press keycode 0x07('0') with META ALT(0x02) on

d.press(0x07, 0x02)

home     back     left     right     up     down     center     menu     search     enter     delete(or del)

recent     volume\_up     volume\_down     camera     power

* **Drag**

# drag from (sx, sy) to (ex, ey)

d.drag(sx, sy, ex, ey)

# drag from (sx, sy) to (ex, ey) with 10 steps

d.drag(sx, sy, ex, ey, steps=10)

* **Long click**

# long click (x, y) on screen

d.long\_click(x, y)

* **Dump windows hierarchy**

# dump the widown hierarchy and save to local file "hierarchy.xml"

d.dump("hierarchy.xml")

# or get the dumped content(unicode) from return.

xml = d.dump()

* **open notification or quick settings**

# open notification, can not work until Android 4.3.

d.open.notification()

# open quick settings, can not work until Android 4.3.

d.open.quick\_settings()

* **Selector is to identify specific ui object in current window.**

# To seleted the object ,text is 'Clock' and its className is 'android.widget.TextView'

d(text='Clock', className='android.widget.TextView')

1. text, textContains, textMatches, textStartsWith
2. className, classNameMatches
3. description, descriptionContains, descriptionMatches, descriptionStartsWith
4. checkable, checked, clickable, longClickable
5. scrollable, enabled,focusable, focused, selected
6. packageName, packageNameMatches
7. resourceId, resourceIdMatches
8. index, instance

* **child**

# get the child or grandchild

d(className="android.widget.ListView").child(text="Bluetooth")

* **sibling**

# get sibling or child of sibling

d(text="Google").sibling(className="android.widget.ImageView")

* **relative position**

we can use the relative position methods to get the view: left, right, top, bottom.

* d(A).left(B), means selecting B on the left side of A.
* d(A).right(B), means selecting B on the right side of A.
* d(A).up(B), means selecting B above A.
* d(A).down(B), means selecting B under A.
* **instances**

d(text="Add new", instance=0) # which means the first instance with text "Add new"

* **uiautomator provides list like methods to use it.**

# get the count of views with text "Add new" on current screen

d(text="Add new").count

# same as count property

len(d(text="Add new"))

# get the instance via index

d(text="Add new")[0]

d(text="Add new")[1]

...

# iterator

for view in d(text="Add new"):

view.info # ...

Set/Clear text of editable field

d(text="Settings").clear\_text() # clear the text

d(text="Settings").set\_text("My text...") # set the text

Perform click on the specific ui object

# click on the center of the specific ui object

d(text="Settings").click()

# click on the bottomright corner of the specific ui object

d(text="Settings").click.bottomright()

# click on the topleft corner of the specific ui object

d(text="Settings").click.topleft()

# click and wait until the new window update

d(text="Settings").click.wait()

Perform long click on the specific ui object

# long click on the center of the specific ui object

d(text="Settings").long\_click()

# long click on the bottomright corner of the specific ui object

d(text="Settings").long\_click.bottomright()

# long click on the topleft corner of the specific ui object

d(text="Settings").long\_click.topleft()

Drag the ui object to another point or ui object

# notes : drag can not be set until Android 4.3.

# drag the ui object to point (x, y)

d(text="Settings").drag.to(x, y, steps=100)

# drag the ui object to another ui object(center)

d(text="Settings").drag.to(text="Clock", steps=50)

Swipe from the center of the ui object to its edge

(text="Settings").swipe.right()

d(text="Settings").swipe.left(steps=10)

d(text="Settings").swipe.up(steps=10)

d(text="Settings").swipe.down()

Two point gesture from one point to another

d(text="Settings").gesture((sx1, sy1), (sx2, sy2)) \

                  .to((ex1, ey1), (ex2, ey2))

# notes : pinch can not be set until Android 4.3.

# from edge to center. here is "In" not "in"

d(text="Settings").pinch.In(percent=100, steps=10)

# from center to edge

d(text="Settings").pinch.Out()

Wait until the specific ui object appears or gone

# wait until the ui object appears

d(text="Settings").wait.exists(timeout=3000)

# wait until the ui object gone

d(text="Settings").wait.gone(timeout=1000)

Perform fling on the specific ui object(scrollable)

# fling forward(default) vertically(default)

d(scrollable=True).fling()

# fling forward horizentally

d(scrollable=True).fling.horiz.forward()

# fling backward vertically

d(scrollable=True).fling.vert.backward()

# fling to beginning horizentally

d(scrollable=True).fling.horiz.toBeginning(max\_swipes=1000)

# fling to end vertically

d(scrollable=True).fling.toEnd()

Perform scroll on the specific ui object(scrollable)

# scroll forward(default) vertically(default)

d(scrollable=True).scroll(steps=10)

# scroll forward horizentally

d(scrollable=True).scroll.horiz.forward(steps=100)

# scroll backward vertically

d(scrollable=True).scroll.vert.backward()

# scroll to beginning horizentally

d(scrollable=True).scroll.horiz.toBeginning(steps=100, max\_swipes=1000)

# scroll to end vertically

d(scrollable=True).scroll.toEnd()

# scroll forward vertically until specific ui object appears

d(scrollable=True).scroll.to(text="Security")