# 技术分享 | app自动化测试（Android）--高级定位技巧

原创 子奕 霍格沃兹测试学院 2022年06月13日 08:00 广东

> 本文节选自霍格沃兹测试开发学社内部教材

通常使用定位器定位页面上的元素会发生一些定位不到元素，或者定位失败的情况。有可能是页面上元素不唯一，有可能是页面发生变化。这节介绍定位元素的高级用法，使用层级关系定位或者多重属性定位的方式来确定元素的唯一性，从而更精准，更稳定的定位到想要的元素。

## XPath高级定位技巧

### XPath 简介

XPath 的英文全称为：XML Path Language，意旨对 XML 中的元素进行路径定位的一种语言，它可适用 XML 标记语言，Html 标记语言，app Dom 结构。XPath 是自动化工具的定位基础，可适用于 Selenium 工具，Appium 工具，Appcrawler 工具。由于前面章节已经对 XPath 进行说明，本篇只做举例说明。

### XPath 基本语法

下面是 XPath 的常用方法：

| 路径表达式 | 结果 |
| --- | --- |
| `/bookstore/book[1]` | 选取属于 bookstore 子元素的第一个 book 元素。 |
| `/bookstore/book[last()]` | 选取属于 bookstore 子元素的最后一个 book 元素。 |
| `/bookstore/book[last()-1]` | 选取属于 bookstore 子元素的倒数第二个 book 元素。 |
| `/bookstore/book[position()<3]` | 选取最前面的两个属于 bookstore 元素的子元素的 book 元素。 |
| `//title[@lang]` | 选取所有拥有名为 lang 的属性的 title 元素。 |
| `//title[@lang='eng']` | 选取所有 title 元素，且这些元素拥有值为 eng 的 lang 属性。 |
| `/bookstore/book[price>35.00]` | 选取 bookstore 元素的所有 book 元素，且其中的 price 元素的值须大于 35.00。 |
| `/bookstore/book[price>35.00]/title` | 选取 bookstore 元素中的 book 元素的所有 title 元素，且其中的 price 元素的值须大于 35.00。 |

| 表达式 | 描述 |
| --- | --- |
| nodename | 选取此节点的所有子节点。 |
| / | 从根节点选取。 |
| // | 从匹配选择的当前节点选择文档中的节点，而不考虑它们的位置。 |
| . | 选取当前节点。 |
| .. | 选取当前节点的父节点。 |
| @ | 选取属性。 |

> "/"还可表示子元素
> "//"还可表示子孙元素

## XPath 模糊定位技巧

contains( )方法是模糊匹配的定位方法，对于一个元素的属性不固定，就可以模糊匹配。如：`//[contains(@content-desc, '帮助')]`，示例代码：

**PYTHON 版本**

```
driver.find_element(By.XPATH,
'//*[contains(@text, "注册")]')

driver.find_element(By.XPATH,
'//*[contains(@content-desc, "搜索")]')

driver.find_element(By.XPATH,
'//*[contains(@resource-id, "login_phone")]')
```

**JAVA 版本**

```
driver.findElement(By.xpath(
        "//*[contains(@text, \"注册\")]"));

driver.findElement(By.xpath(
        "//*[contains(@content-desc, \"搜索\")]"));
```

```
6
7  driver.findElement(By.xpath(
8          "//*[contains(@resource-id, \"login_phone\")]"));
```

## XPath 组合定位技巧

通过 XPath 可以同时匹配 2 个甚至多个属性来完成元素定位。这里常用的属性有 text、resource-id、class、index、content-desc 等，这些属性任意组合完成定位，示例代码：

**PYTHON 版本**

```
1  driver.find_element(
2      By.XPATH,'//*[@text="我的" and @resource-id="tab_name"]'
3      ).click()
4
5  driver.find_element(
6      By.XPATH,'//*[@text="注册/登录" and @index="1"]'
7      ).click()
```

**JAVA 版本**

```
1  driver.findElement(By.xpath(
2          "//*[@text=\"我的\" and @resource-id=\"tab_name\"]")).click();
3
4  driver.findElement(By.xpath(
5          "//*[@text=\"注册/登录\" and @index=\"1\"]")).click();
```

## XPath 层级定位

通常定位元素的时候可能会涉及到通过子元素去定位父元素，或者父元素定位子元素，或者定位兄弟元素，xpath 支持父子关系，兄弟关系元素的查找。示例代码如下：

**PYTHON 版本**

```
1  # 通过子元素定位父元素
2  # 方法一：..
3  driver.find_element_by_xpath(
4      '//*[@text="手机号"]/..').tag_name
5
6  # 方法二  parent::*
7  driver.find_element_by_xpath(
```

```
    '[@text="手机号"]/parent::*').tag_name

#通过元素定位兄弟元素
driver.find_element_by_xpath(
    '//*[@text="手机号"]/../li'
    ).tag_name
```

**JAVA 版本**

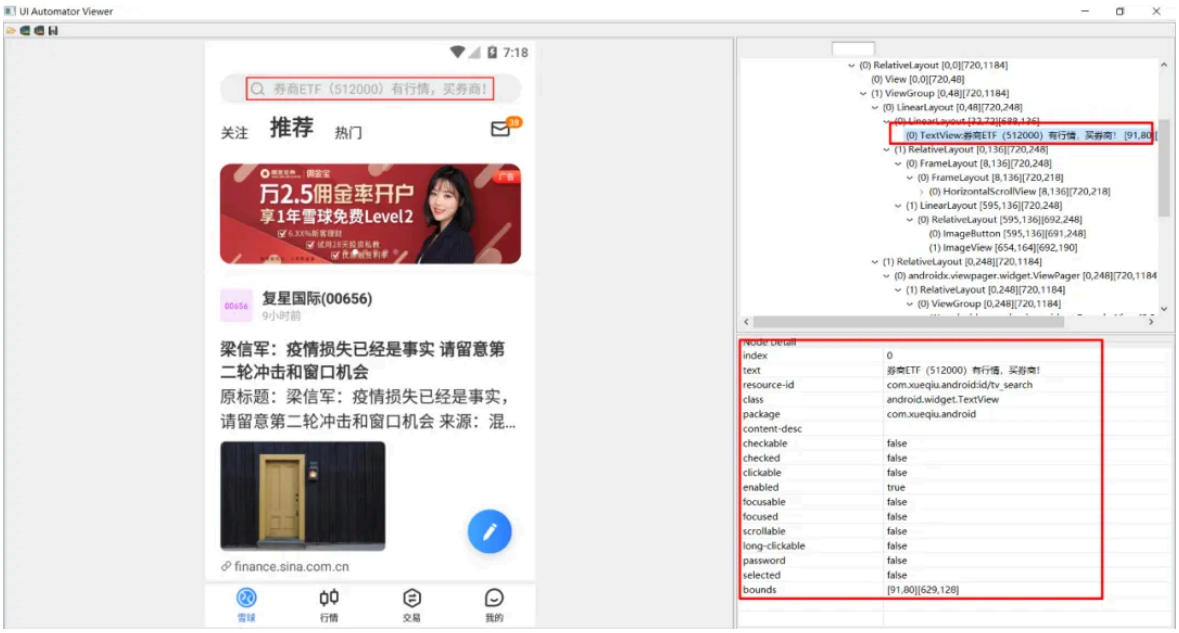
```
// 通过子元素定位父元素
// 方法一：..
driver.findElement(By.xpath(
    "//*[@text=\"手机号\"]/..")).getTagName();

// 方法二  parent::*
driver.findElement(By.xpath(
    "[@text=\"手机号\"]/parent::*")).getTagName();

// 通过元素定位兄弟元素
driver.findElement(By.xpath(
    "//*[@text=\"手机号\"]/../li"
)).getTagName();
```

## 案例

**场景一：**

应用：雪球 apk

可以使用 uiautomatorviewer 工具进行 dom 分析，然后对分析到的元素进行 XPath 定位，比如下面的搜索框，可以使用元素的多种属性进行定位，常用的有 text，resource-id，class，content-desc 等属性。




推荐使用 resource-id 进行定位，通常情况下，它是页面唯一的属性，其 XPath 如下：

**PYTHON 版本**

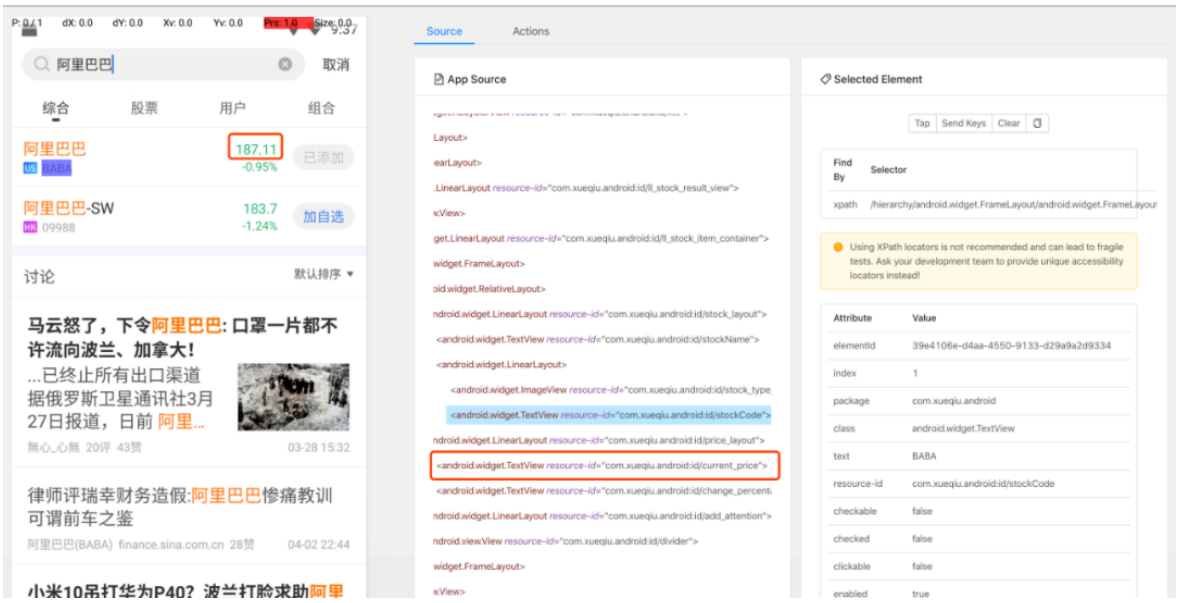
```
driver.find_element(
    By.XPATh, '//*[contains(@resource-id, "tv_search")]')
# 或者也可写成下面这样
driver.find_element(By.ID, 'tv_search')
```

**JAVA 版本**

```
driver.findElement(By.xpath("//*[contains(@resource-id,
        \"tv_search\")]"));
// 或者也可写成下面这样
driver.findElement(By.id("tv_search"));
```

## 场景二：

如下图，获取 “BABA” 所对应的股票价格 “187.11”，可以使用 XPath 父子关系来进行元素定




代码如下：

**PYTHON 版本**

```
curr_price = self.driver.find_element(
    MobileBy.XPATH,"//*[@text='BABA']/../../..\
    //*[@resource-id='com.xueqiu.android:id/current_price']")
```

**JAVA 版本**

```
MobileElement curr_price = driver.findElement(
            By.xpath("//*[@text=\"BABA\"]/../../..//\
            *[@resource-id='com.xueqiu.android:id/current_price']"));
```

## Android UiAutomator定位技巧

UiAutomator 是 Android SDK 自带的一个测试框架，这个测试框架提供了一系列的 API，可以与 Android APP 进行交互，例如打开菜单，点击，滑动等。当 Appium 的 Caps 参数 `uiautomationName` 设置为 UiAutomator2 时，就能够实现与手机端的 UiAutomator 进行通信并且使用 UiAutomator 执行测试代码。如果不进行设置，默认也是使用 UiAutomator2 工作引擎。UiAutomator1 是较老的工作引擎，如果想测试较老版本的 Android 系统（低于 Android4.4 版本）需要设置 `uiautomationName="UiAutomator1"` 。

由于 Android UiAutomator 是 Android SDK 中自带的工作引擎，使用这种定位方式，速度上要比 Xpath 定位方式快很多。但由于写法比较特殊，调试起来要相对麻烦，如果定位语句编写不当，脚本编辑器也不会给出任何提示信息。只能在运行的时候校验对错。

下面就单独介绍基于 uiautomator 定位元素的方法，基本语法如下：

- Python 版本

```
driver.find_element_by_android_uiautomator()
```

- Java 版本

```
driver.findElement(MobileBy.AndroidUIAutomator());
```

常用的方法有：

```
UiSelector()  # 实现元素定位
UiScrollable() # 实现滚动查找元素
```

### 通过 TEXT 文本定位

UiSelector( ) 与 XPath 类似，可以通过元素的 text 属性来定位元素。语法格式如下：

```
new UiSelector().text("text文本")
```

同样也能用模糊查询的用法去定位元素

例如： `new UiSelector().textContains("手机")` 示例代码：

- Python 版本

```
driver.find_element_by_android_uiautomator(
    'new UiSelector().textContains("手机")').click()
```

- Java 版本

```
driver.findElementByAndroidUIAutomator(\
    "new UiSelector().textContains(\"手机\")").click();
```

### 通过 RESOURCEID 定位

uiautomator 同样也能进行 id 定位，格式为 `new UiSelector().resourceId("resource-id属性")`，示例代码：

- Python 版本

```
driver.find_element_by_android_uiautomator(
    'new UiSelector().resourceId("rl_login_phone")').click()
```

- Java 版本

```
driver.findElementByAndroidUIAutomator("new UiSelector().\
resourceId(\"rl_login_phone\")").click();
```

### 通过 CLASSNAME 定位

页面上的 class 属性一般不唯一，此时可以根据下标进行定位，格式为 `new UiSelector().className("className")`，一般会使用 `find_elements` 完成定位，示例代码：

- Python 版本

```
driver.find_elements_by_android_uiautomator(
    'new UiSelector().\
    className("android.widget.TextView")')[5].click()
```

- Java 版本

```
driver.findElementsByAndroidUIAutomator("new UiSelector().\
className(\"android.widget.TextView\")")[5].click();
```

### 通过 DESCRIPTION 定位

同样的，也支持 contenet-desc 定位方式，格式为：`new UiSelector().description("contenet-des属性")`，示例代码：

- Python 版本

```
driver.find_element_by_android_uiautomator(
```

```
    'new UiSelector().description("搜索")').click()
```

- Java 版本

```
driver.findElementByAndroidUIAutomator("new \
UiSelector().description(\"搜索\")").click();
```

### 组合定位方式

Uiautomator 也支持组合元素查找功能，示例代码：

- Python 版本

```
driver.find_element_by_android_uiautomator(
    'new UiSelector().resourceId(\
    "com.xueqiu.android:id/tv_login_phone").text("手机号")').click()
```

- Java 版本

```
driver.findElementByAndroidUIAutomator("new UiSelector().resourceId(\
\"com.xueqiu.android:id/tv_login_phone\").text(\"手机号\")").click();
```

### 滚动查找元素

Uiautomator 使用 `UiScrollable()` 实现了滚动查找元素的功能，可以指定滑动到某个元素，示例代码：

- Python 版本

```
driver.find_element_by_android_uiautomator(
    'new UiScrollable(new UiSelector().scrollable(true)\
    .instance(0)).scrollIntoView(new UiSelector()\
    .text("我的").instance(0));').click()
```

- Java 版本

```
driver.findElementByAndroidUIAutomator(\
    "new UiScrollable(new UiSelector().scrollable(true)\
    .instance(0)).scrollIntoView(new UiSelector().\
    text(\"我的\").instance(0));").click();
```

上面的代码，在当前的页面滚动的查找 text 文本是“我的”这个元素，找到之后执行点击操作。

## css selector元素定位

Appium Server 从 1.19.0 这个版本开始，元素定位增加了 css selector 的支持。appium-uiautomator2-driver 会将 css selector 定位器转化成 `android uiautomator` 定位方式。

注意：appium inspector 暂时没有添加这种定位方式。

由于 UiSelector() 的表达式是 Java 的语法格式，编写定位元素的表达式很复杂，代码编写工具（比如 Pycharm，VSCode，IntelliJ IDEA等）也不会有任何提示错误信息。只能是运行时才能发现表达式的错误。官方提供了 css selector 的语法，会自动转成 `android uiautomator` 的语法结构，这种原生的定位元素的方式，定位速度要更快一些。

详情参考官方： `https://github.com/appium/appium-uiautomator2-driver/pull/410`

源码地址： `https://github.com/appium/appium-uiautomator2-driver/blob/master/lib/css-converter.js`

### id 定位

可以使用 css selector 语法定位。如下代码， `#igk` 表示 css selector 定位符

- Python 版本

```
driver.find_element_by_css_selector('#igk')
```

对应 ID 定位器代码如下：

```
driver.find_element_by_id('android:id/igk')
```

- Java 版本

```
driver.findElementByCssSelector("#igk").click();
```

对应 ID 定位器代码如下：

```
driver.findElementById("android:id/igk").click();
```

### class name 定位

如下代码，表示 css selector 定位符为 `.android.widget.ImageView` 的元素

- Python 版本

```
driver.find_element_by_css_selector('.android.widget.ImageView')
```

对应 class name 定位器代码如下：

```
driver.find_element_by_class_name("android.widget.ImageView")
```

- Java 版本

```
driver.findElementByCssSelector(".android.widget.ImageView");
```

对应 class name 定位器代码如下：

```
driver.findElementByClassName("android.widget.ImageView");
```

### text 定位

如下代码，表示 css selector 定位符为 `*[text='工作台']` 的元素：

- Python 版本

```
driver.find_element_by_css_selector("*[text='工作台']")
```

对应 xpath 定位器代码如下：

```
driver.find_element_by_xpath("//*[@text='工作台']")
```

- Java 版本

```
driver.findElementByCssSelector("*[text=\"工作台\"]");
```

对应 xpath 定位器代码如下：

```
driver.findElementByXPath("//*[@text=\"工作台\"]");
```

### description 定位

如下代码，表示 css selector 定位符为 `*[description="ContentDescription"]` 的元素：

- Python 版本

```
driver.find_element_by_css_selector('*[description="ContentDescription"]')
```

对应 accessibility id 定位器代码如下：

```
driver.find_element_by_accessibility_id("ContentDescription")
```

- Java 版本

```
driver.findElementByCssSelector("*[description=\"ContentDescription\"]");
```

对应 accessibility id 定位器代码如下：

```
driver.findElementByAccessibilityId("ContentDescription");
```