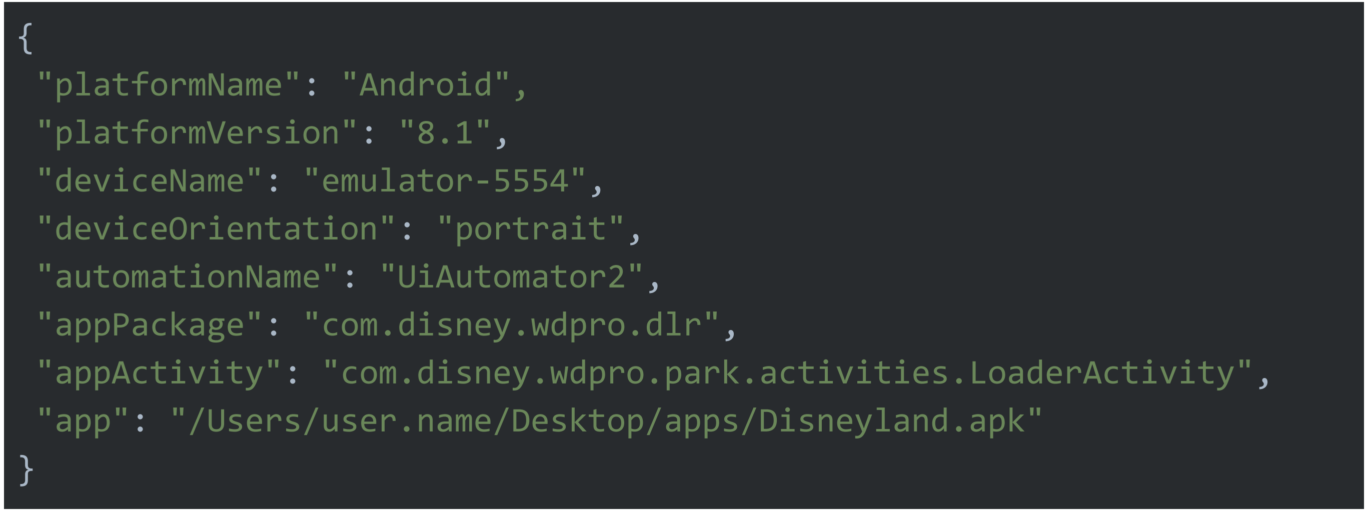
Globant

Android Locator Strategies

Appium Android Capabilities

Android

Appium Android Capabilities



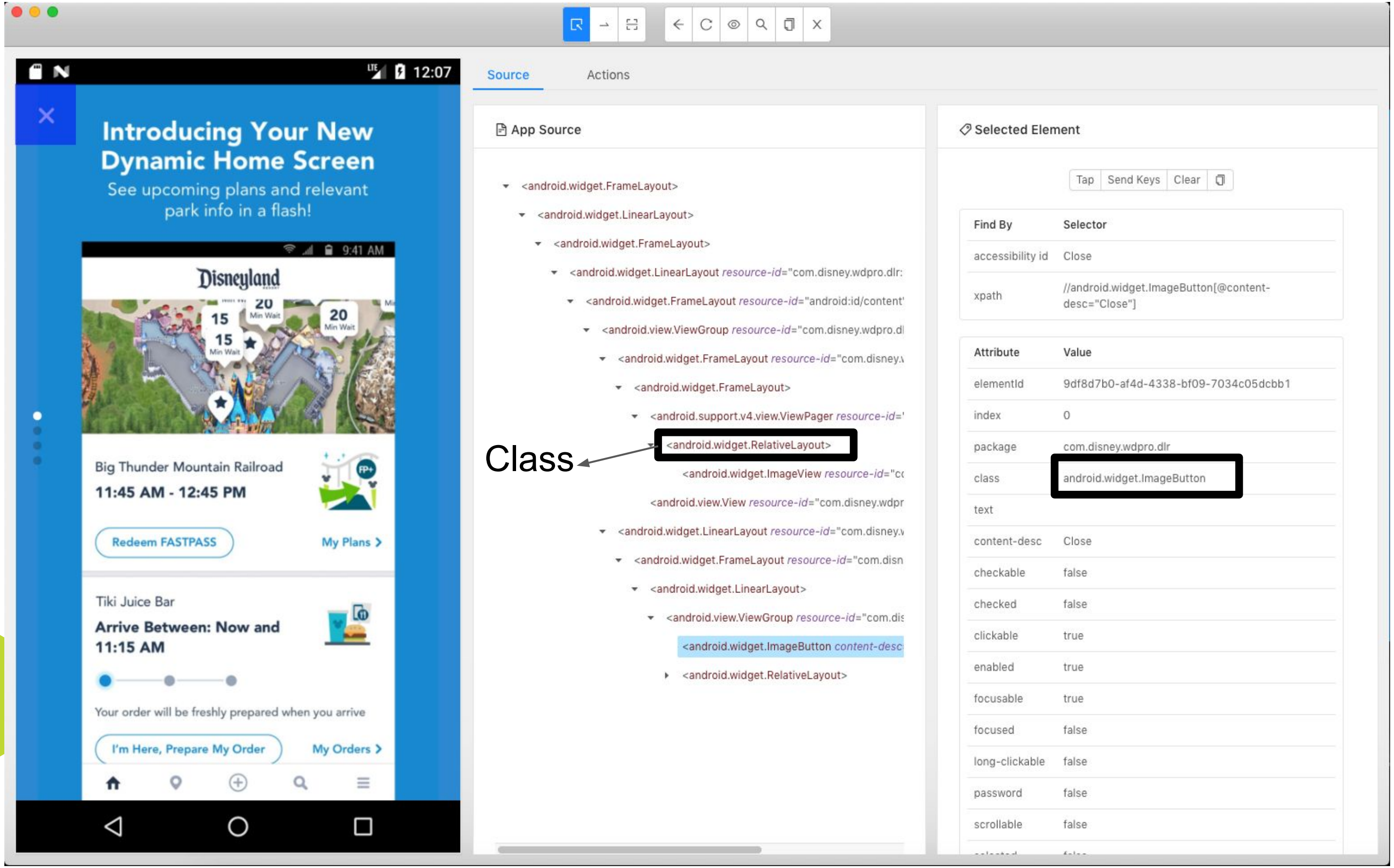
```
{
 "platformName": "Android",
 "platformVersion": "8.1",
 "deviceName": "emulator-5554",
 "deviceOrientation": "portrait",
 "automationName": "UiAutomator2",
 "appPackage": "com.disney.wdpro.dlr",
 "appActivity": "com.disney.wdpro.park.activities.LoaderActivity",
 "app": "/Users/user.name/Desktop/apps/Disneyland.apk"
}
```

Appium Documentation

Appium Interface

Android

Locator Strategies



12:07
Introducing Your New Dynamic Home Screen
See upcoming plans and relevant park info in a flash!
9:41 AM
Disneyland
15
15
20
20
Min Wait
Big Thunder Mountain Railroad
11:45 AM - 12:45 PM
Redeem FASTPASS
My Plans >
Tiki Juice Bar
Arrive Between: Now and 11:15 AM
Your order will be freshly prepared when you arrive
I'm Here, Prepare My Order
My Orders >
Source
Actions
App Source
Selected Element
Tap
Send Keys
Clear

```
<android.widget.FrameLayout>
  <android.widget.LinearLayout>
    <android.widget.FrameLayout>
      <android.widget.LinearLayout resource-id="com.disney.wdpro.dlr:
        <android.widget.FrameLayout resource-id="android:id/content'
          <android.view.ViewGroup resource-id="com.disney.wdpro.dl
            <android.widget.FrameLayout resource-id="com.disney.
              <android.widget.FrameLayout>
                <android.support.v4.view.ViewPager resource-id='
                  <android.widget.RelativeLayout>
                    <android.widget.ImageView resource-id="c
                  <android.view.View resource-id="com.disney.wdpr
              <android.widget.LinearLayout resource-id="com.disney.v
                <android.widget.FrameLayout resource-id="com.disn
                  <android.widget.LinearLayout>
                    <android.view.ViewGroup resource-id="com.dis
                      <android.widget.ImageButton content-desc
                      <android.widget.RelativeLayout>
```

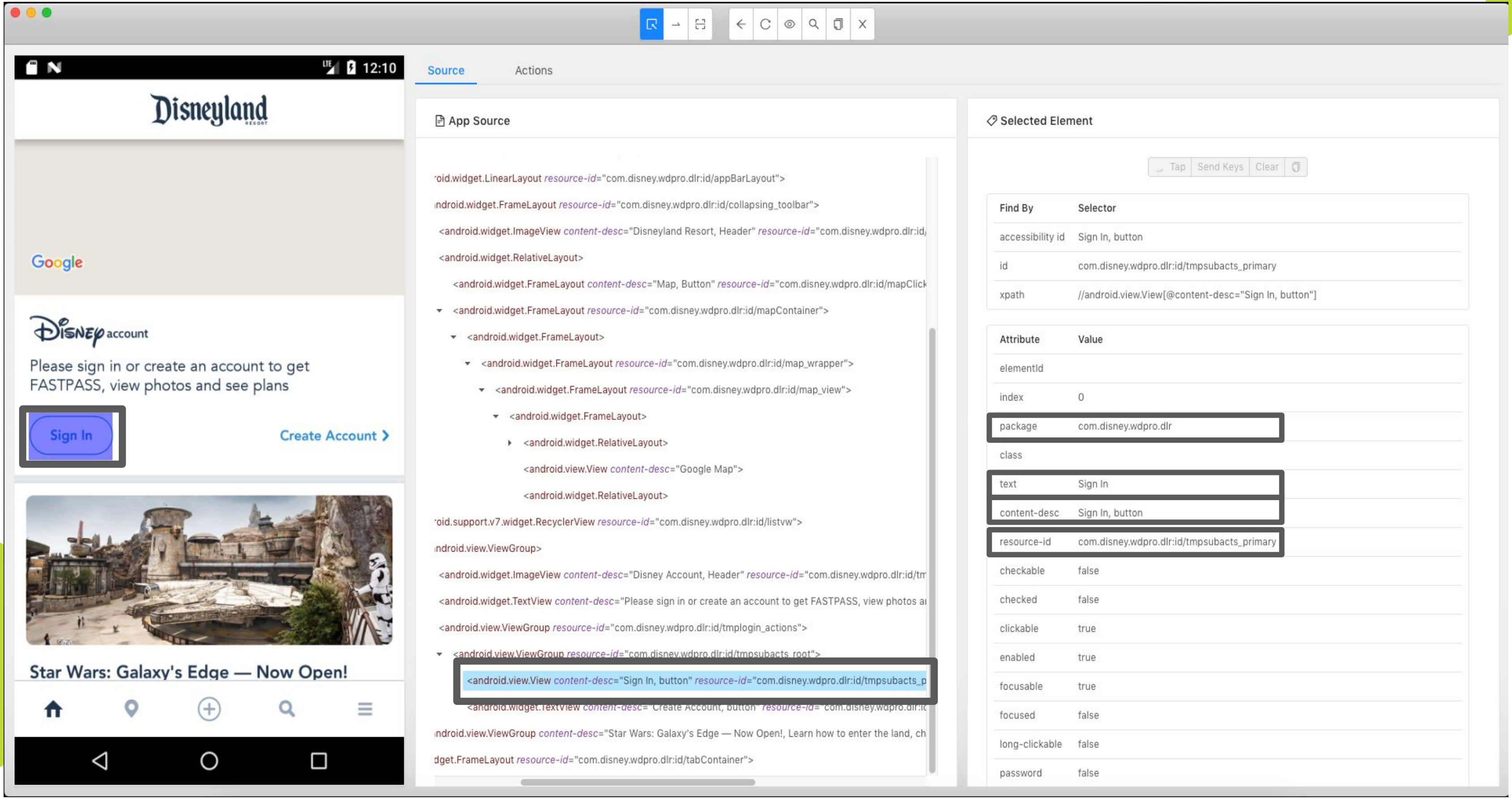
Class

Find By	Selector
accessibility id	Close
xpath	//android.widget.ImageButton[@content-desc="Close"]

Attribute	Value
elementId	9df8d7b0-af4d-4338-bf09-7034c05dcbb1
index	0
package	com.disney.wdpro.dlr
class	android.widget.ImageButton
text	
content-desc	Close
checkable	false
checked	false
clickable	true
enabled	true
focusable	true
focused	false
long-clickable	false
password	false
scrollable	false

Android

Locator Strategies



12:10
Disneyland
Google
Disney account
Please sign in or create an account to get FASTPASS, view photos and see plans
Sign In
Create Account ›
Star Wars: Galaxy's Edge — Now Open!
Source
Actions
Tap
Send Keys
Clear

App Source

```
'oid.widget.LinearLayout resource-id="com.disney.wdpro.dlr:id/appBarLayout">
ndroid.widget.FrameLayout resource-id="com.disney.wdpro.dlr:id/collapsing_toolbar">
 <android.widget.ImageView content-desc="Disneyland Resort, Header" resource-id="com.disney.wdpro.dlr:id/
 <android.widget.RelativeLayout>
   <android.widget.FrameLayout content-desc="Map, Button" resource-id="com.disney.wdpro.dlr:id/mapClick
 ▾ <android.widget.FrameLayout resource-id="com.disney.wdpro.dlr:id/mapContainer">
   ▾ <android.widget.FrameLayout>
     ▾ <android.widget.FrameLayout resource-id="com.disney.wdpro.dlr:id/map_wrapper">
       ▾ <android.widget.FrameLayout resource-id="com.disney.wdpro.dlr:id/map_view">
         ▾ <android.widget.FrameLayout>
           ▸ <android.widget.RelativeLayout>
             <android.view.View content-desc="Google Map">
             <android.widget.RelativeLayout>
'oid.support.v7.widget.RecyclerView resource-id="com.disney.wdpro.dlr:id/listvw">
ndroid.view.ViewGroup>
 <android.widget.ImageView content-desc="Disney Account, Header" resource-id="com.disney.wdpro.dlr:id/tm
 <android.widget.TextView content-desc="Please sign in or create an account to get FASTPASS, view photos ai
 <android.view.ViewGroup resource-id="com.disney.wdpro.dlr:id/tmplogin_actions">
 ▾ <android.view.ViewGroup resource-id="com.disney.wdpro.dlr:id/tmpsubacts_root">
     <android.view.View content-desc="Sign In, button" resource-id="com.disney.wdpro.dlr:id/tmpsubacts_p
     <android.widget.TextView content-desc="Create Account, button" resource-id="com.disney.wdpro.dlr:ic
ndroid.view.ViewGroup content-desc="Star Wars: Galaxy's Edge — Now Open!, Learn how to enter the land, ch
dget.FrameLayout resource-id="com.disney.wdpro.dlr:id/tabContainer">
```

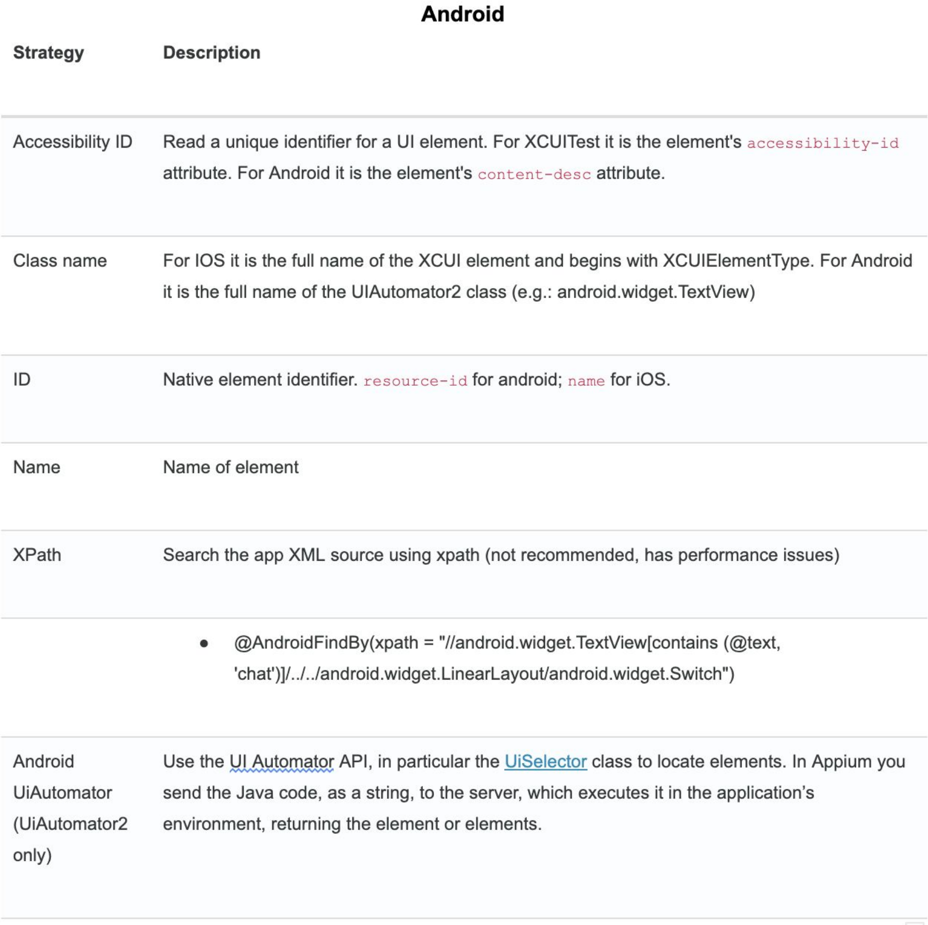
Selected Element

Find By	Selector
accessibility id	Sign In, button
id	com.disney.wdpro.dlr:id/tmpsubacts_primary
xpath	//android.view.View[@content-desc="Sign In, button"]

Attribute	Value
elementId	
index	0
package	com.disney.wdpro.dlr
class	
text	Sign In
content-desc	Sign In, button
resource-id	com.disney.wdpro.dlr:id/tmpsubacts_primary
checkable	false
checked	false
clickable	true
enabled	true
focusable	true
focused	false
long-clickable	false
password	false

UiSelector()

Android Selector Strategies



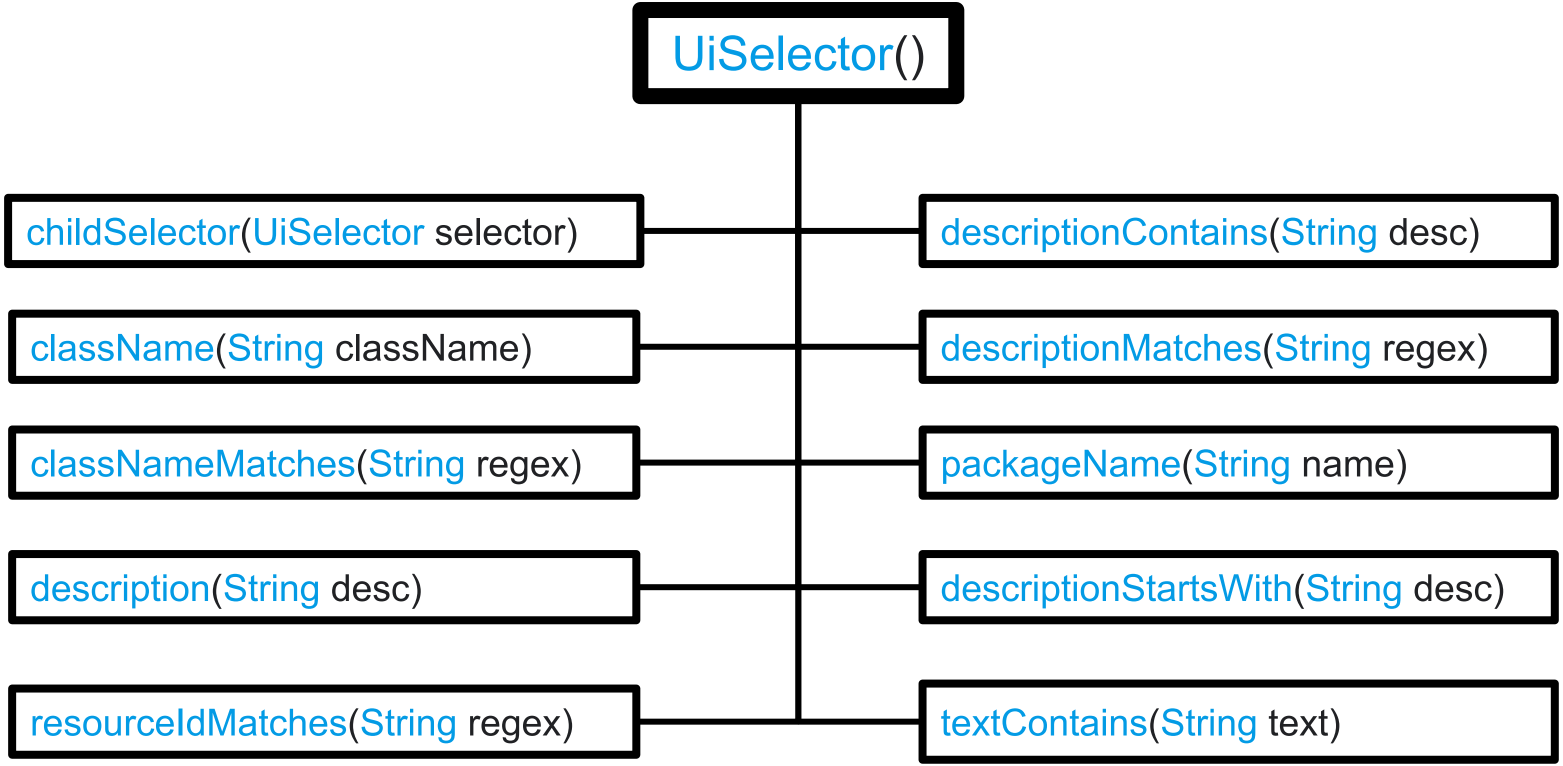
Android

Strategy	Description
Accessibility ID	Read a unique identifier for a UI element. For XCUITest it is the element's `accessibility-id` attribute. For Android it is the element's `content-desc` attribute.
Class name	For IOS it is the full name of the XCUI element and begins with XCUIElementType. For Android it is the full name of the UIAutomator2 class (e.g.: android.widget.TextView)
ID	Native element identifier. `resource-id` for android; `name` for iOS.
Name	Name of element
XPath	Search the app XML source using xpath (not recommended, has performance issues)
	● @AndroidFindBy(xpath = "//android.widget.TextView[contains (@text, 'chat')]/../../android.widget.LinearLayout/android.widget.Switch")
Android UiAutomator (UiAutomator2 only)	Use the UI Automator API, in particular the UiSelector class to locate elements. In Appium you send the Java code, as a string, to the server, which executes it in the application's environment, returning the element or elements.

Appium Documentation

Android Documentation

UiSelector Location Strategies



UiSelector()

childSelector(UiSelector selector)

className(String className)

classNameMatches(String regex)

description(String desc)

resourceIdMatches(String regex)

descriptionContains(String desc)

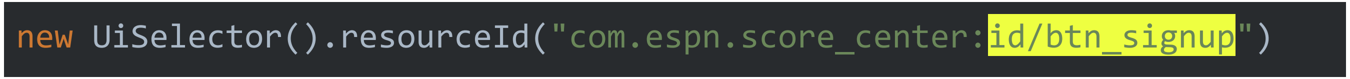
descriptionMatches(String regex)

packageName(String name)

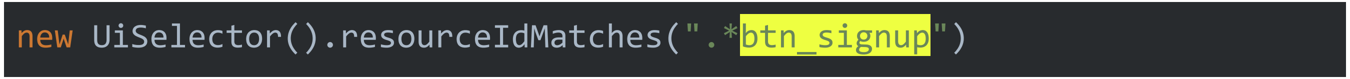
descriptionStartsWith(String desc)

textContains(String text)

Using ResourceId Attribute



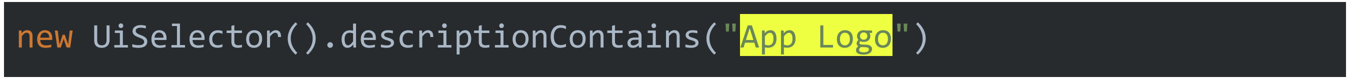
```
new UiSelector().resourceId("com.espn.score_center:id/btn_signup")
```



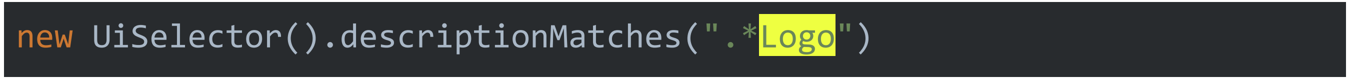
```
new UiSelector().resourceIdMatches(".*btn_signup")
```

Android Documentation

Using Description Attribute



```
new UiSelector().descriptionContains("App Logo")
```



```
new UiSelector().descriptionMatches(".*Logo")
```

Android Documentation

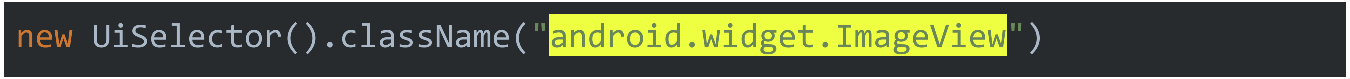
Using Text

```
new UiSelector().textContains("LATER")
```

```
new UiSelector().text("SIGN UP LATER")
```

Android Documentation

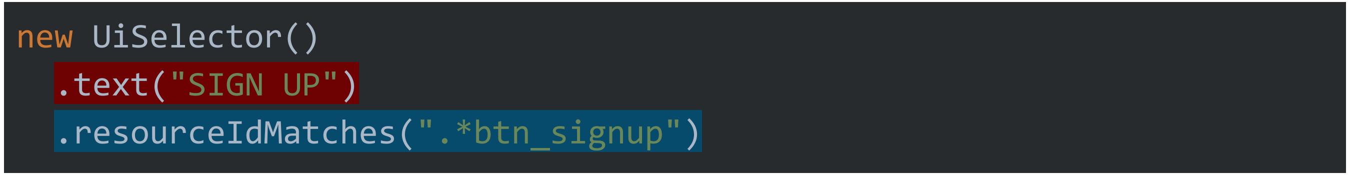
Using ClassName



```
new UiSelector().className("android.widget.ImageView")
```

Android Documentation

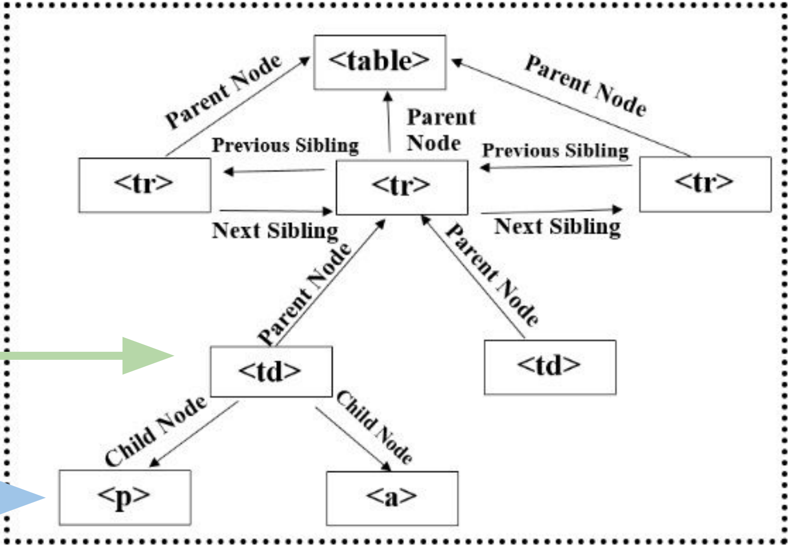
Chaining Attributes



```
new UiSelector()
   .text("SIGN UP")
   .resourceIdMatches(".*btn_signup")
```

Using ChildSelector

```
new UiSelector()
  .className("android.widget.GridView")
  .childSelector(
    new UiSelector()
    .resourceIdMatches(".*animation_view")
  )
```



Android Documentation

UiScrollable()

Android

UiScrollable()

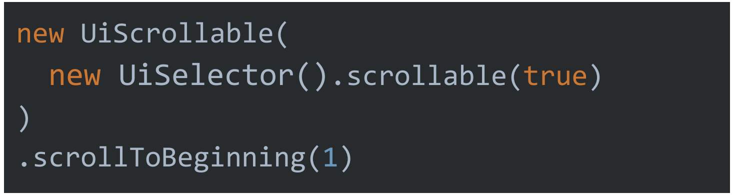
Filtering by attribute

```
new UiScrollable(
  new UiSelector().scrollable(true)
)
.scrollIntoView(
  new UiSelector()
  .resourceIdMatches(".*tmpsubacts_primary")
  .descriptionContains("See Ticket Option")
)
```

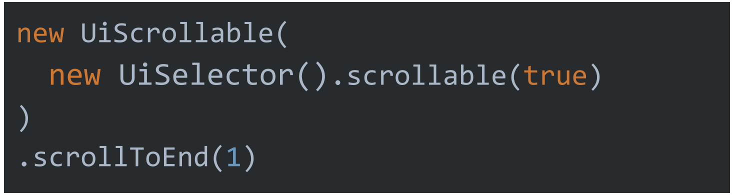
```
new UiScrollable(
  new UiSelector().scrollable(true)
)
.scrollIntoView(
  new UiSelector().textContains("Terms of use")
)
```

Appium Documentation

Scrolling without any specification



```
new UiScrollable(
  new UiSelector().scrollable(true)
)
.scrollToBeginning(1)
```



```
new UiScrollable(
  new UiSelector().scrollable(true)
)
.scrollToEnd(1)
```

Android Documentation

TouchAction API

Android
TouchAction

What is TouchAction?

- Is an Appium API that allows you to build up arbitrary gestures with multiple actuators.

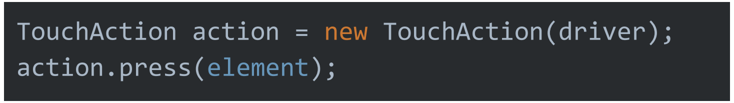
- Is a solution that allows you to apply the same gestures for Android and iOS, avoiding the creation of different flows/interactions for each one. For example, we are not able to use UiAutomation (UiScrollable) on iOS.

Appium Documentation

Appium Github Documentation

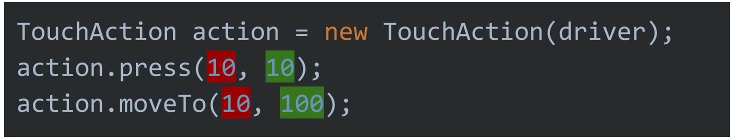
TouchAction examples

- Touching an specific element



```
TouchAction action = new TouchAction(driver);
action.press(element);
```

- Touching over specific coordinates and moving to specific coordinates



```
TouchAction action = new TouchAction(driver);
action.press(10, 10);
action.moveTo(10, 100);
```

X coordinates
Y coordinates

Appium Documentation

THANKS FOR
YOUR ATTENTION!