Mobile Browser Tests With Appium

Creation Date: 2/9/2023

Table Of Contents

Introduction 1

Pre-Requisites 1

Setup On Android Mobile 2

Starting Appium Server 3

Create Script 7

Creating NestedFlow Automation Script 9

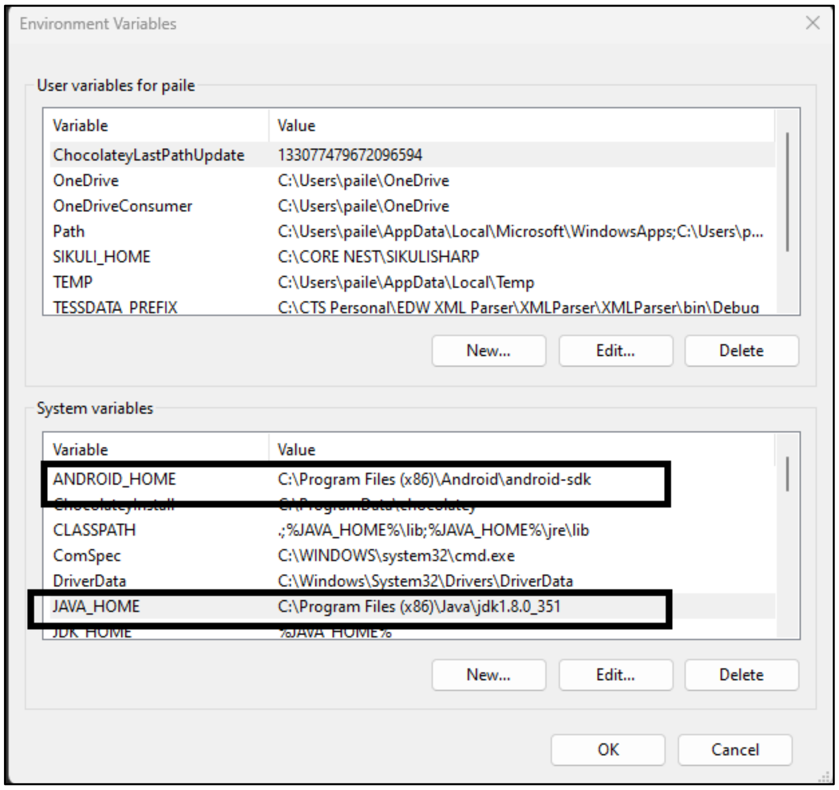
Testing App on Android Emulator 11

Introduction

This document explains how to execute browser based automation tests on real android devices using NestedFlowAutomation tool and Appium.

Pre-Requisites

1. Java and Android SDK are installed on your machine.
2. ANDROID_HOME and JAVA_HOME variables are created in the Environment variables.



Variable	Value
ChocolateyLastPathUpdate	133077479672096594
OneDrive	C:\Users\paile\OneDrive
OneDriveConsumer	C:\Users\paile\OneDrive
Path	C:\Users\paile\AppData\Local\Microsoft\WindowsApps;C:\Users\p...
SIKULI_HOME	C:\CORE NEST\SIKULISHARP
TEMP	C:\Users\paile\AppData\Local\Temp
TESSDATA_PREFIX	C:\CTS Personal\EDW XML Parser\XMLParser\XMLParser\bin\Debug

Variable	Value
ANDROID_HOME	C:\Program Files (x86)\Android\android-sdk
ChocolateyInstall	C:\ProgramData\chocolatey
CLASSPATH	.;%JAVA_HOME%\lib;%JAVA_HOME%\jre\lib
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
JAVA_HOME	C:\Program Files (x86)\Java\jdk1.8.0_351
JDK_HOME	%JAVA_HOME%

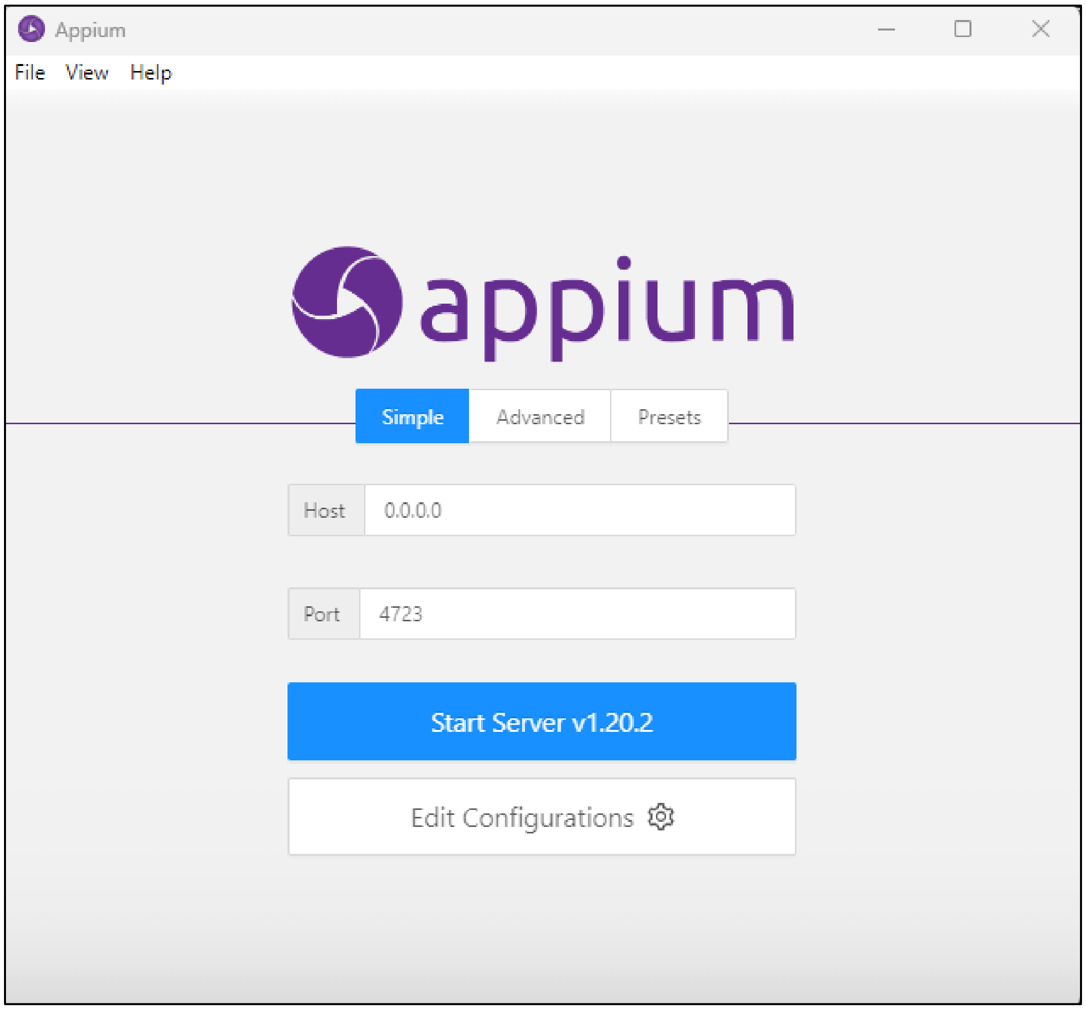
3. Install Appium on your machine

Setup On Android Mobile

1. Click on **Settings** App on your android phone
2. Navigate to **About phone** and click on it
3. There will be an entry called **Build number** towards the end of it
4. Tap the **Build number** multiple times till you see a message **You are now a developer!**
5. We get again to settings app and search for **USB debugging** and enable it
6. Connect the phone to machine and choose **USB tethering** in the **Use USB for** options

Starting Appium Server

Open Appium tool

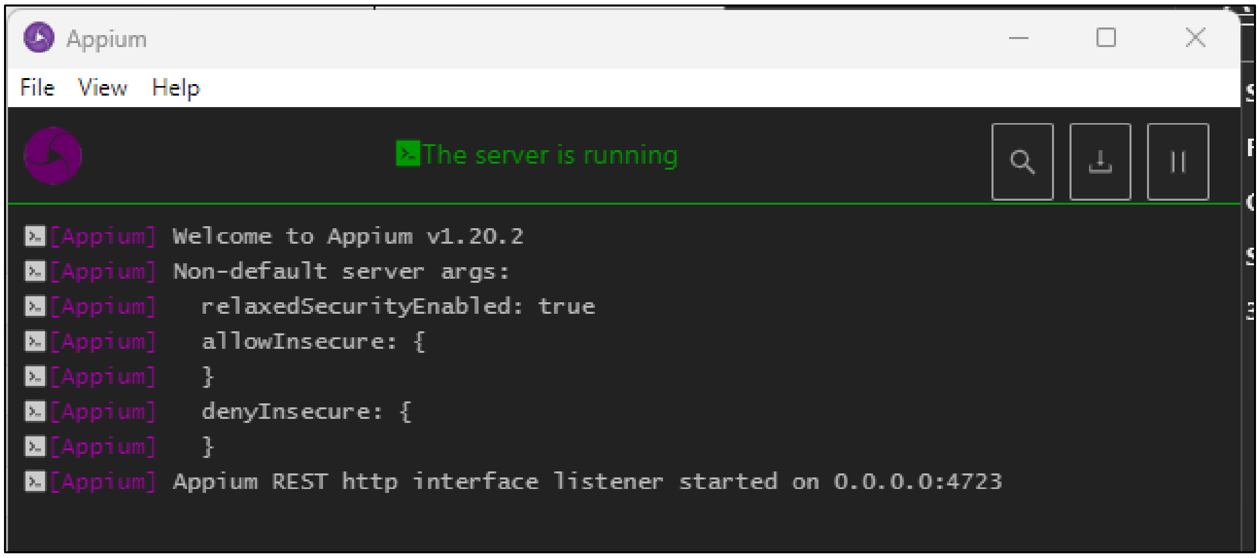


Host 0.0.0.0

Port 4723

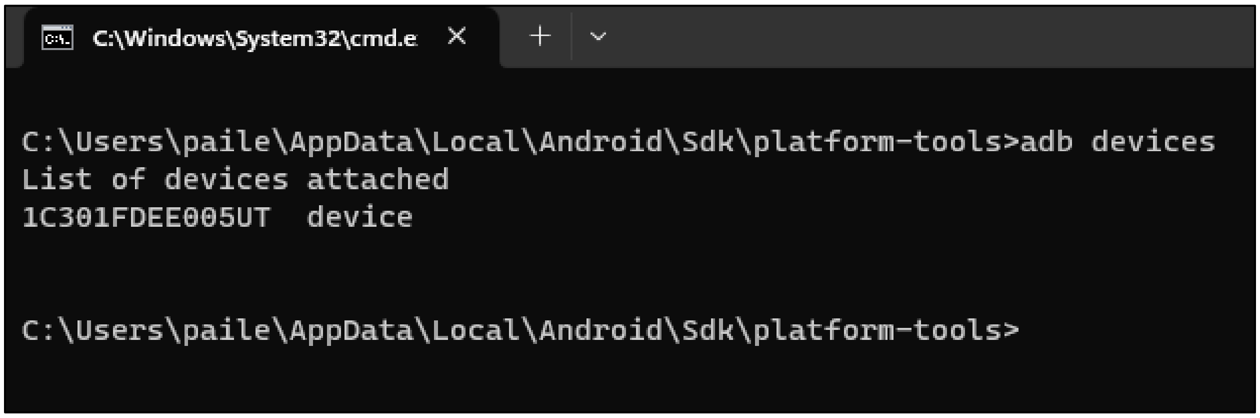
Click on **Start Server** button

(In this example it starts on 0.0.0.0:4723)



```
[Appium] Welcome to Appium v1.20.2
[Appium] Non-default server args:
[Appium]   relaxedSecurityEnabled: true
[Appium]   allowInsecure: {
[Appium]   }
[Appium]   denyInsecure: {
[Appium]   }
[Appium] Appium REST http interface listener started on 0.0.0.0:4723
```

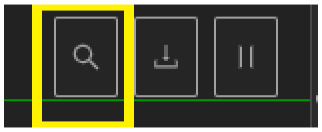
Open a windows command terminal and run command adb devices (open terminal for the same folder in which adb.exe exists)

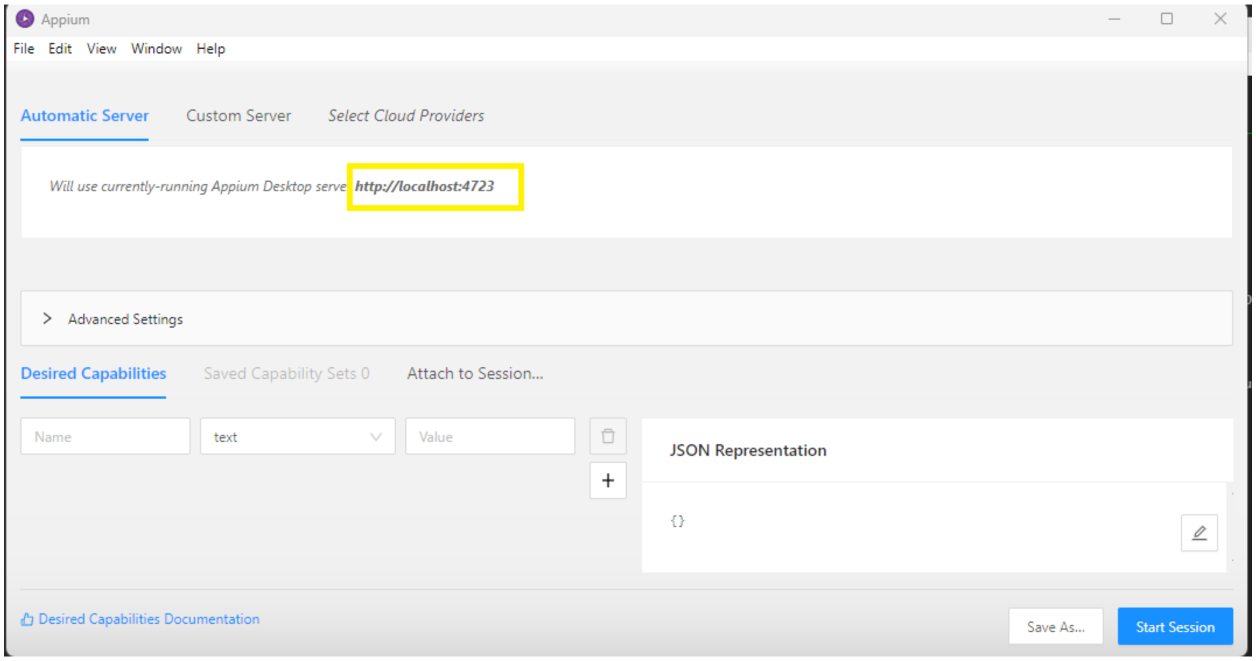


```
C:\Users\paile\AppData\Local\Android\Sdk\platform-tools>adb devices
List of devices attached
1C301FDEE005UT   device


C:\Users\paile\AppData\Local\Android\Sdk\platform-tools>
```

The device name will be displayed (in this case IC301FDEE005UT)

Once the name is captured, go back to Appium Server window and click on lens button

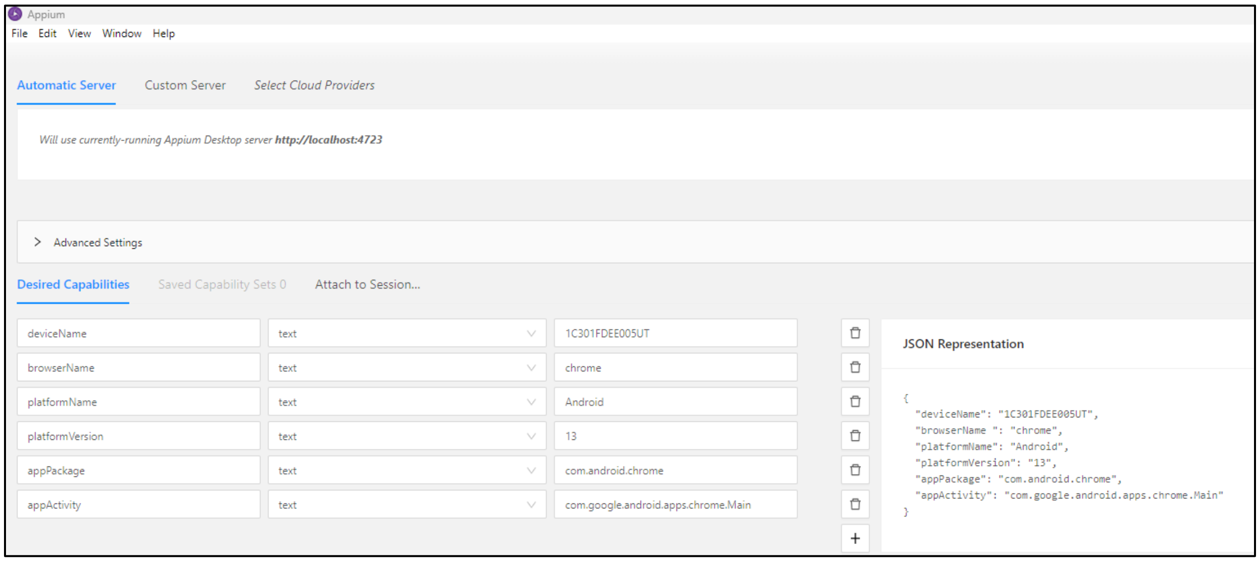




Will use currently-running Appium Desktop serve http://localhost:4723

```
{}
```

Create below Desired capabilities based on your device details



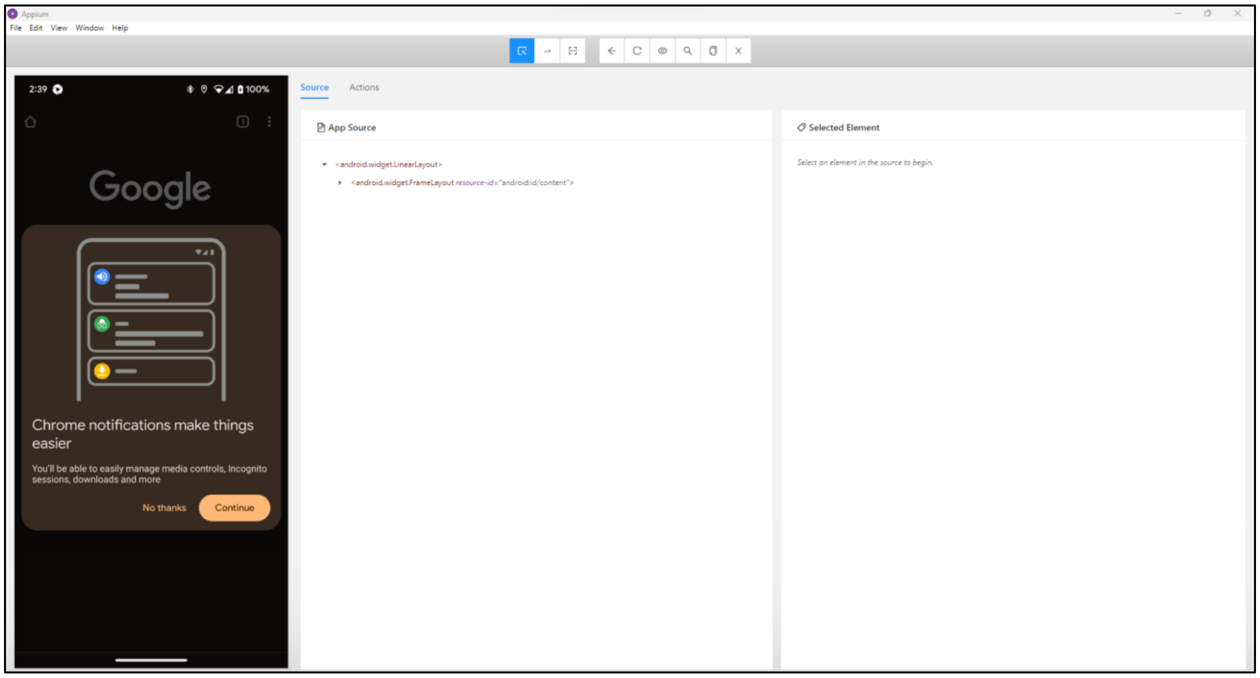
Will use currently-running Appium Desktop server http://localhost:4723

deviceName	text	1C301FDEE005UT
browserName	text	chrome
platformName	text	Android
platformVersion	text	13
appPackage	text	com.android.chrome
appActivity	text	com.google.android.apps.chrome.Main

```
{
  "deviceName": "1C301FDEE005UT",
  "browserName ": "chrome",
  "platformName": "Android",
  "platformVersion": "13",
  "appPackage": "com.android.chrome",
  "appActivity": "com.google.android.apps.chrome.Main"
}
```

Click on **Start Session** button

Device mirror is shown



Chrome notifications make things easier

You'll be able to easily manage media controls, Incognito sessions, downloads and more

```
<android.widget.LinearLayout>
  <android.widget.FrameLayout resource-id="android:id/content">
```

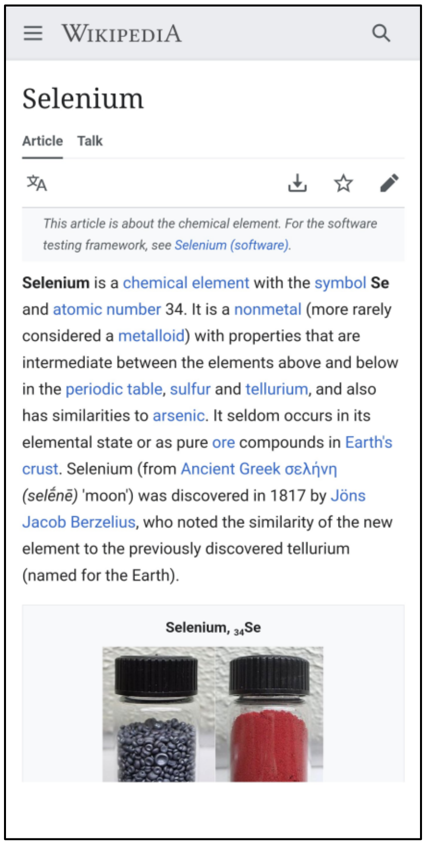
Select an element in the source to begin.

Create Script

Below is a sample script using visual studio which is just a Wikipedia sample:-

```
using System;
using OpenQA.Selenium;
using OpenQA.Selenium.Appium;
using OpenQA.Selenium.Appium.Interfaces;
using OpenQA.Selenium.Appium.MultiTouch;
using OpenQA.Selenium.Interactions;
using OpenQA.Selenium.Remote;
using OpenQA.Selenium.Appium.Android;


namespace Mobile_Testing_POC
{

    public static class Class1
    {
        private static Uri testServerAddress = new
Uri("http://localhost:4723/wd/hub"); // If Appium is running locally
        private static TimeSpan INIT_TIMEOUT_SEC = TimeSpan.FromSeconds(180); /*
Change this to a more reasonable value */
        private static TimeSpan IMPLICIT_TIMEOUT_SEC = TimeSpan.FromSeconds(10); /*
Change this to a more reasonable value */
        public static void Main()
        {


            IWebDriver driver;
            DesiredCapabilities capabilities = new DesiredCapabilities();
            capabilities.SetCapability("device", "Android");
            capabilities.SetCapability("browserName", "chrome");
            capabilities.SetCapability("deviceName", " 1C301FDEE005UT");
            capabilities.SetCapability("platformName", "Android");


            driver = new RemoteWebDriver(new Uri("http://localhost:4723/wd/hub"),
capabilities, TimeSpan.FromSeconds(180));
            driver.Navigate().GoToUrl("https://wikipedia.com");
            System.Threading.Thread.Sleep(1000);
            IWebElement IWebe =
driver.FindElement(By.XPath("//*[@id='searchInput']"));
            IWebe.SendKeys("Selenium");
            IWebElement IWebe1 = driver.FindElement(By.XPath("//*[@id='search-
form']/fieldset/button"));
            IWebe1.Click();
            Screenshot SR = (driver as ITakesScreenshot).GetScreenshot();
            SR.SaveAsFile(@"c:\km\phoneimage.png", ScreenshotImageFormat.Png);
        }
    }
}
```

The script will run on the actual android machine linked to Appium server and gives back the screenshot.



Selenium

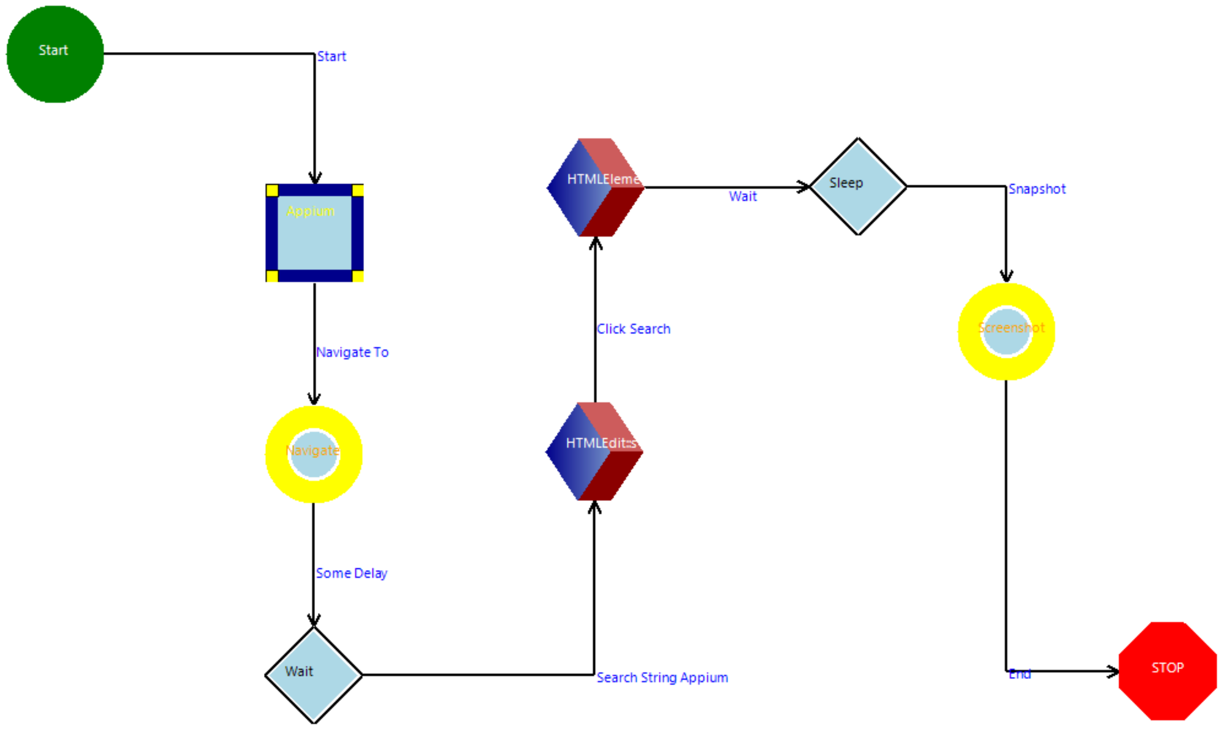
This article is about the chemical element. For the software testing framework, see Selenium (software).

Selenium is a chemical element with the symbol **Se** and atomic number 34. It is a nonmetal (more rarely considered a metalloid) with properties that are intermediate between the elements above and below in the periodic table, sulfur and tellurium, and also has similarities to arsenic. It seldom occurs in its elemental state or as pure ore compounds in Earth's crust. Selenium (from Ancient Greek σελήνη *(selḗnē)* 'moon') was discovered in 1817 by Jöns Jacob Berzelius, who noted the similarity of the new element to the previously discovered tellurium (named for the Earth).

Selenium, $_{34}$Se

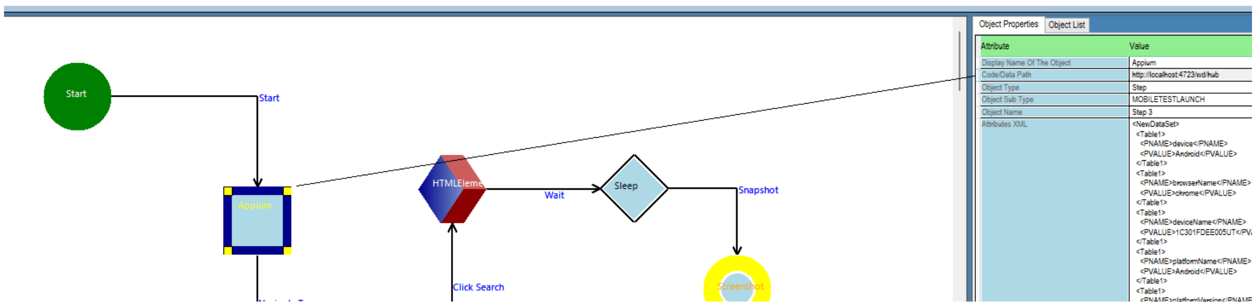
Creating NestedFlow Automation Script

Now the same script will be recreated using NestedFlowAutomation tool.



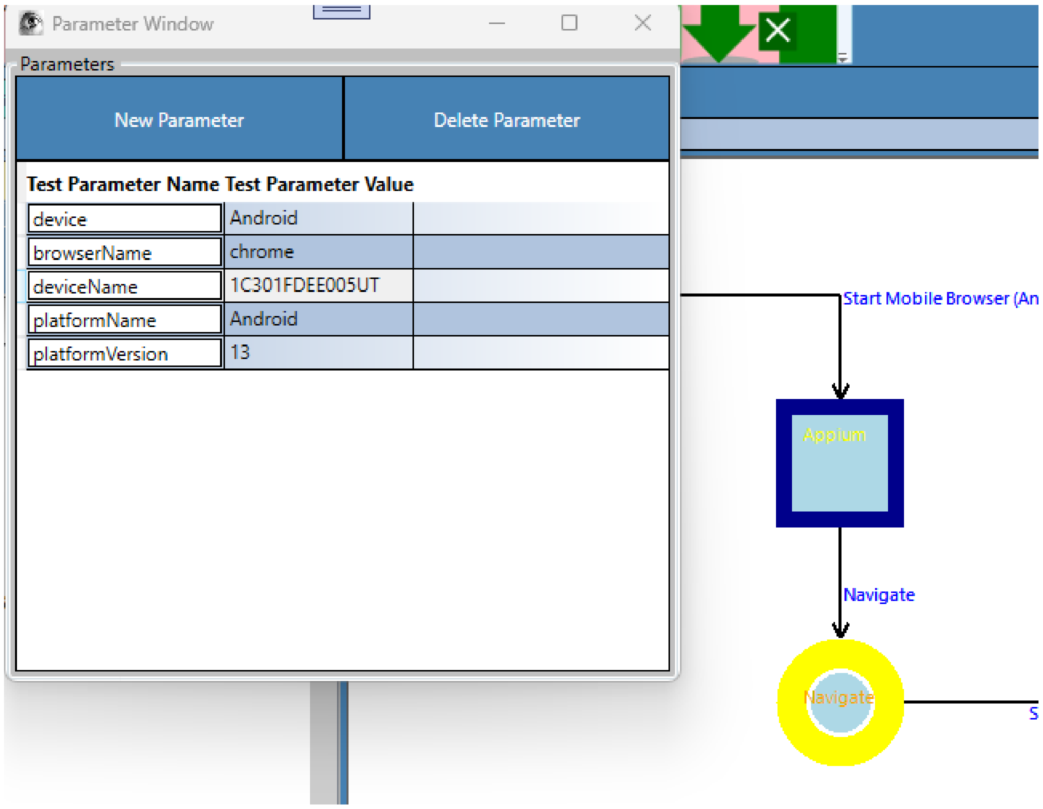
Appium server is connected using Android browser invoke function under **HTML UI Automation Controls**

Code/DataPath of the step is set to local Appium server `http://localhost:4723/wd/hub`



Attribute	Value
Display Name Of The Object	Appium
Code/Data Path	http://localhost:4723/wd/hub
Object Type	Step
Object Sub Type	MOBILETESTLAUNCH
Object Name	Step 3
Attributes XML	`<NewDataSet>` `<Table1>` `<PNAME>device</PNAME>` `<PVALUE>Android</PVALUE>` `</Table1>` `<Table1>` `<PNAME>browserName</PNAME>` `<PVALUE>chrome</PVALUE>` `</Table1>` `<Table1>` `<PNAME>deviceName</PNAME>` `<PVALUE>1C301FDEE005UT</PV` `</Table1>` `<Table1>` `<PNAME>platformName</PNAME>` `<PVALUE>Android</PVALUE>` `</Table1>` `<Table1>`

Right Click on the step and click on **Invoke Mobile Automation Options**



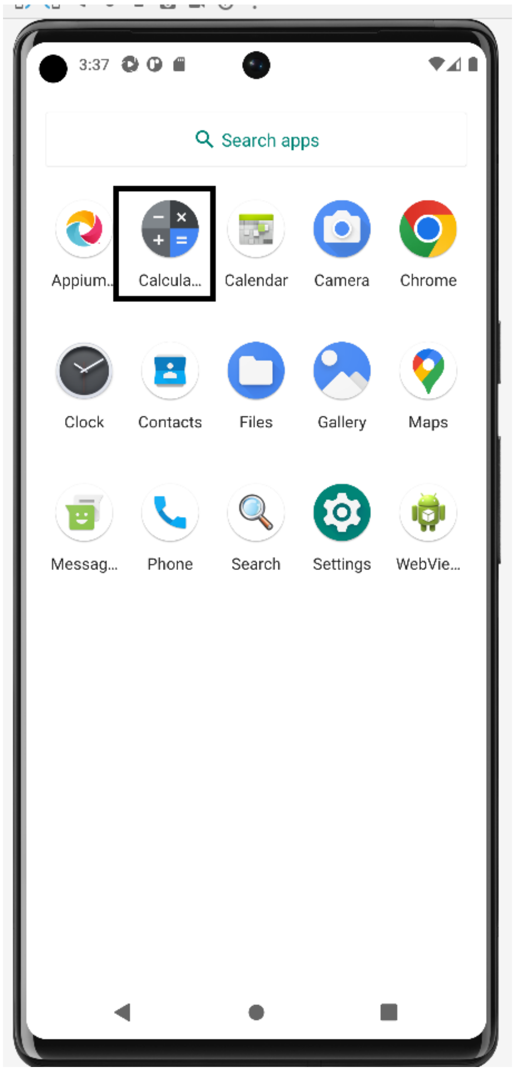
Test Parameter Name	Test Parameter Value
device	Android
browserName	chrome
deviceName	1C301FDEE005UT
platformName	Android
platformVersion	13

This will allow you to provide automation options. In this case I have provided device, browserName, deviceName,platformName, platformVersion

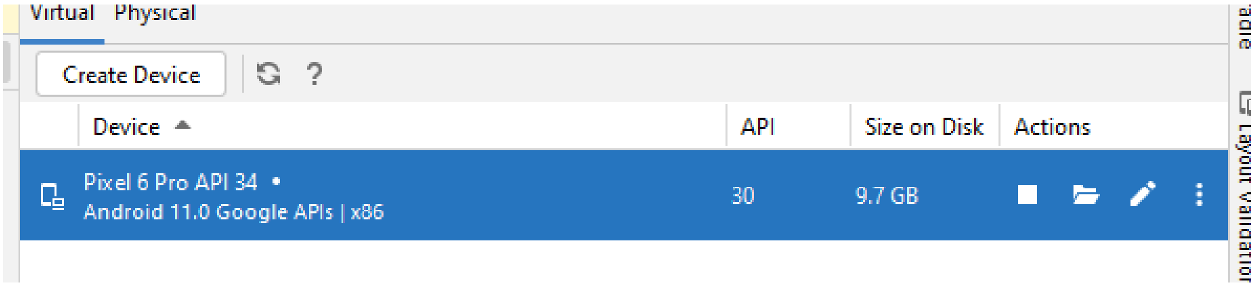
This is trying to execute the script on chrome browser on actual android device. The deviceName is obtained by what is provided from **adb devices** command

Testing App on Android Emulator

In this example we will be testing Calculator app on the Android emulator



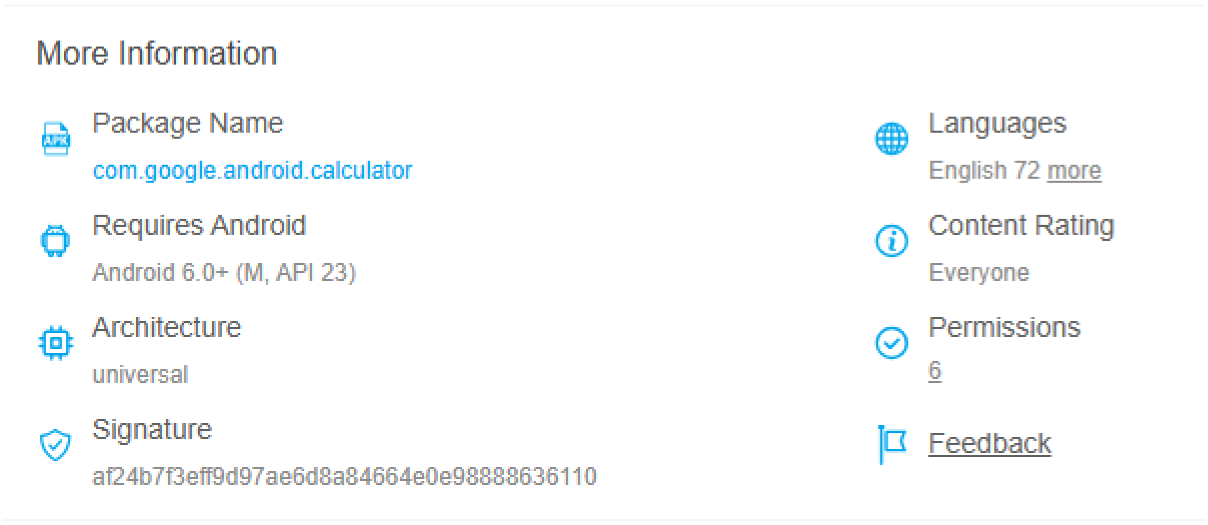
We used Android Studio to create Pixel 6 Pro simulated device



Device	API	Size on Disk	Actions
Pixel 6 Pro API 34 • Android 11.0 Google APIs | x86	30	9.7 GB	

Downloaded the calculator apk from [Calculator APK for Android Download (apkpure.com)]

And dragged the apk on to emulator. App gets installed successfully


More Information

Package Name
com.google.android.calculator

Requires Android
Android 6.0+ (M, API 23)

Architecture
universal

Signature
af24b7f3eff9d97ae6d8a84664e0e98888636110

Languages
English 72 more

Content Rating
Everyone

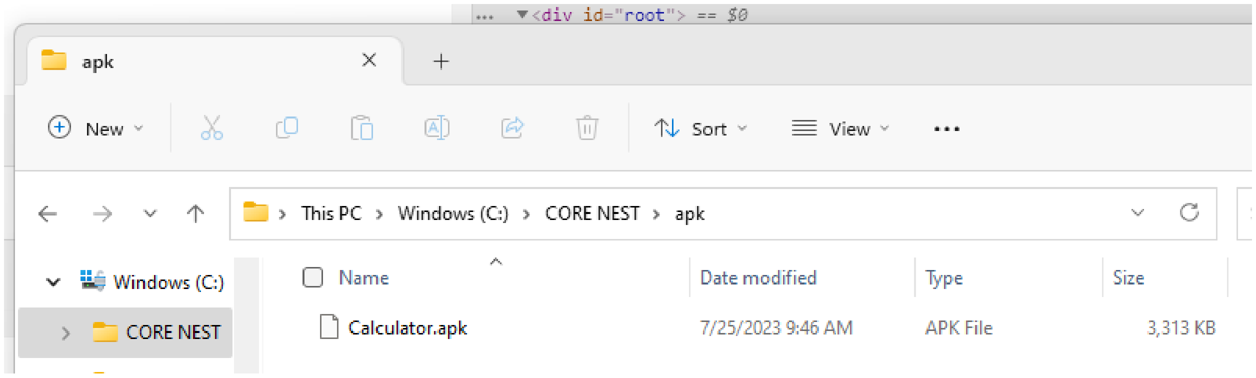
Permissions
6

Feedback

The capabilities for the test

```
{
  "platformName": "Android",
  "platformVersion": "11",
  "deviceName": "emulator-5554",
  "appPackage": "com.google.android.calculator",
  "app": "c:\\CORE NEST\\apk\\Calculator.apk"
}
```

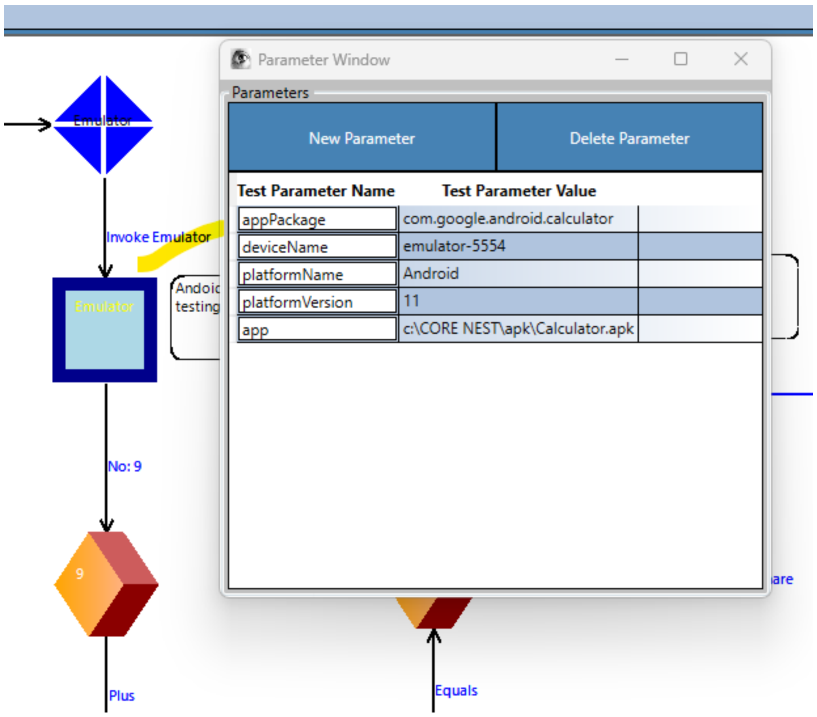
App capability is nothing but the location in which the apk file is kept on the machine



Name	Date modified	Type	Size
Calculator.apk	7/25/2023 9:46 AM	APK File	3,313 KB

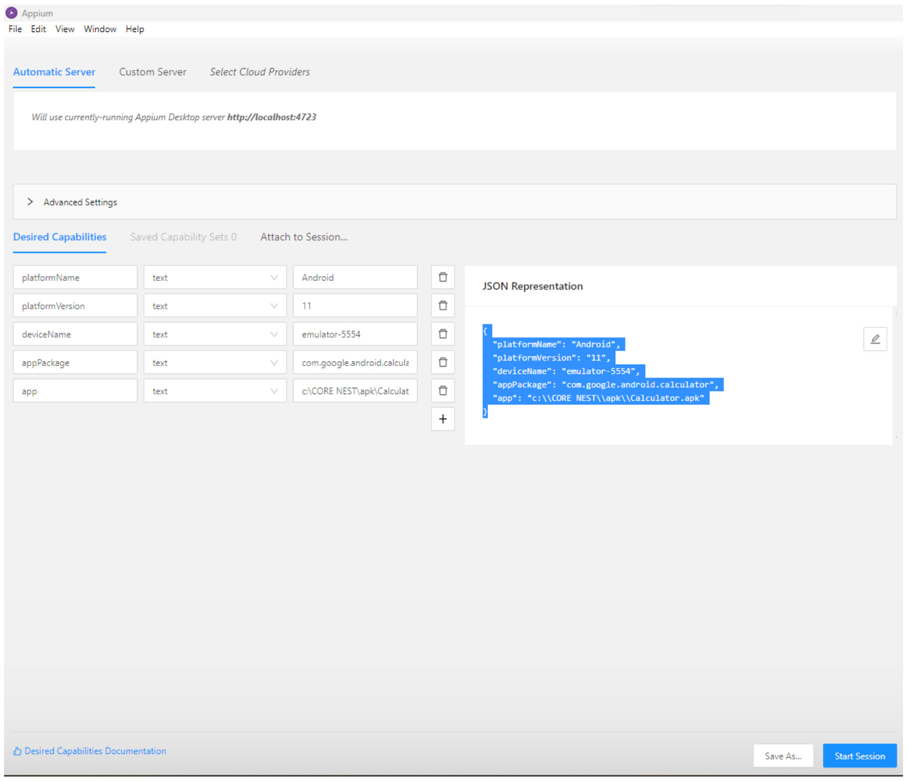
DeviceName is derived from **adb devices** command

The same capabilities are represented as below on the NestedFlowAutomation



Test Parameter Name	Test Parameter Value
appPackage	com.google.android.calculator
deviceName	emulator-5554
platformName	Android
platformVersion	11
app	c:\CORE NEST\apk\Calculator.apk

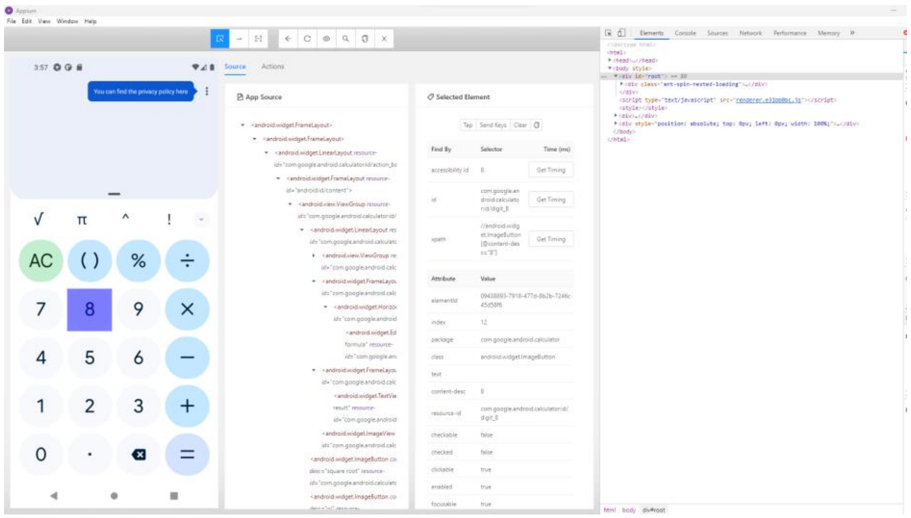
Object properties are derived by opening the app using Appium inspector tool by clicking **Start Session** button



Will use currently-running Appium Desktop server **http://localhost:4723**

platformName	text	Android
platformVersion	text	11
deviceName	text	emulator-5554
appPackage	text	com.google.android.calculz
app	text	c:\CORE NEST\apk\Calculat

```
{
  "platformName": "Android",
  "platformVersion": "11",
  "deviceName": "emulator-5554",
  "appPackage": "com.google.android.calculator",
  "app": "c:\\CORE NEST\\apk\\Calculator.apk"
}
```

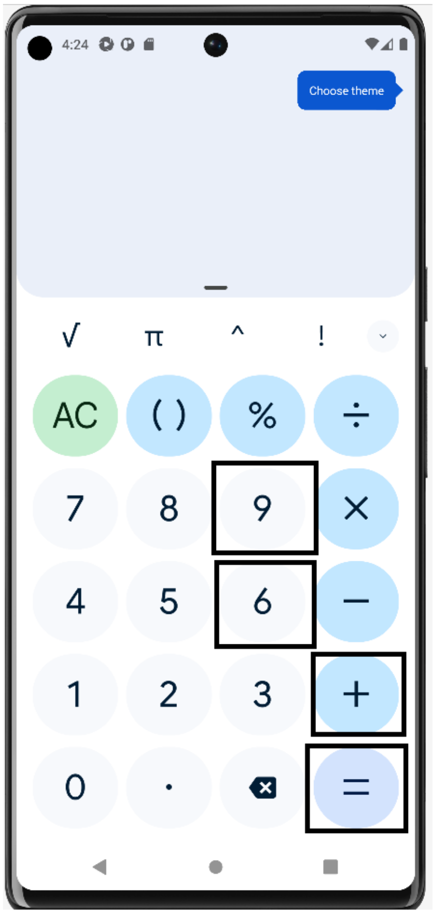


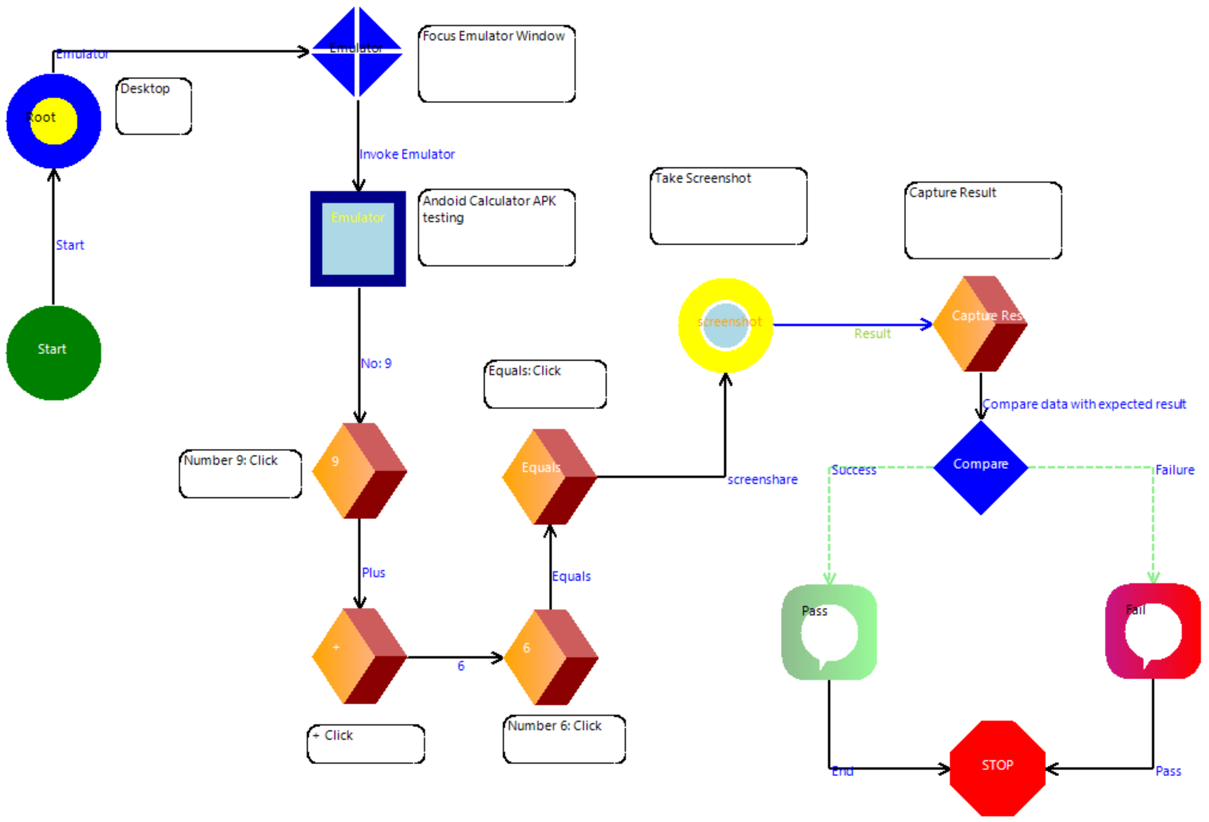
Selected Element

Find By	Selector	Time (ms)
accessibility id	8	Get Timing
id	com.google.android.calculator:id/digit_8	Get Timing
xpath	//android.widget.ImageButton[@content-desc="8"]	Get Timing

Attribute	Value
elementId	09438893-7918-477d-8b2b-7246c45d58f6
index	12
package	com.google.android.calculator
class	android.widget.ImageButton
text	
content-desc	8
resource-id	com.google.android.calculator:id/digit_8
checkable	false
checked	false
clickable	true
enabled	true
focusable	true

Performing 9+6 = 15 and validating result




Emulator
Focus Emulator Window
Desktop
Root
Invoke Emulator
Start
Emulator
Andoid Calculator APK testing
Take Screenshot
Capture Result
Start
screenshot
Result
Capture Res
No: 9
Equals: Click
Compare data with expected result
Number 9: Click
9
Equals
screenshare
Success
Compare
Failure
Plus
Equals
Pass
Fail
+
6
6
+ Click
Number 6: Click
End
STOP
Pass