**APPIUM Tutorial for Android & iOS Mobile Apps Testing**

**What is Appium?**

APPIUM is a freely distributed open source mobile application UI Testing framework. Appium allows native, hybrid and web application testing and supports automation test on physical devices as well as an emulator or simulator both. It offers cross-platform application testing, i.e. single API works for both Android and iOS platform test scripts.

It has **NO** dependency on Mobile device OS. Because APPIUM has framework or wrapper that translate Selenium Webdriver commands into UIAutomation (iOS) or UIAutomator (Android) commands depending on the device type, not any OS type.

Appium supports all languages that have Selenium client libraries like- Java, Objective-C, JavaScript with node.js, PHP, Ruby, Python, C#, etc.

In this tutorial, we will learn about

* How APPIUM Works?
* Prerequisite to use APPIUM
* Install Appium Desktop
* APPIUM Inspector
* Attach Android Emulator to Appium
* APPIUM Test Case for Native Android App(Calculator)
* Limitations using APPIUM

**How APPIUM Works?**

* Appium is an 'HTTP Server' written using a Node.js platform and drives iOS and an Android session using Webdriver JSON wire protocol. Hence, before initializing the Appium Server, Node.js must be pre-installed on the system.
* When Appium is downloaded and installed, then a server is set up on our machine that exposes a REST API.
* It receives connection and command request from the client and executes that command on mobile devices (Android / iOS).
* It responds back with HTTP responses. Again, to execute this request, it uses the mobile test automation frameworks to drive the user interface of the apps. A framework like:-
  + Apple Instruments for iOS (Instruments are available only in Xcode 3.0 or later with OS X v10.5 and later)
  + Google UIAutomator for Android API level 16 or higher
  + Selendroid for Android API level 15 or less

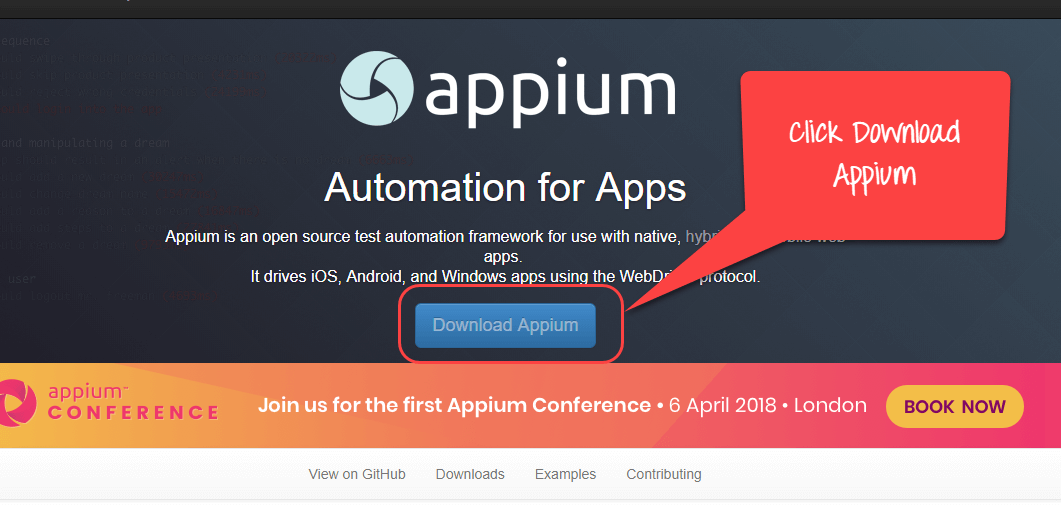
**Prerequisite to use APPIUM**

1. Install ANDROID SDK (Studio)[[Link](http://developer.android.com/sdk/index.html)]
2. Install JDK (Java Development Kit)
3. Install Eclipse
4. Install TestNg for Eclipse
5. Install Selenium Server JAR
6. Appium Client Library[[Link](http://appium.io/docs/en/about-appium/appium-clients/index.html)]
7. APK App Info on Google Play [[Link](https://play.google.com/store/apps/details?id=de.migali.soft.apkinfo&hl=en)]
8. js (Not Required - Whenever Appium server is installed, it by default comes with "Node.exe" & NPM. It's included in Current version of Appium.)
9. Install Appium Desktop

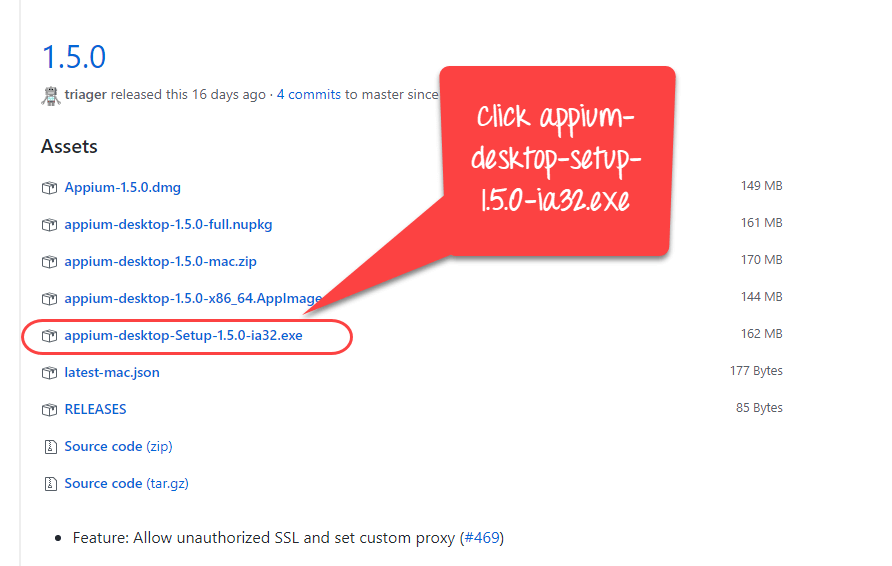
**Install Appium Desktop:**

Appium Studio is an Open source GUI app to install Appium Server. It comes bundled with all the pre-requisites to install and use Appium Server. It also has an Inspector to get basic information on your Apps. It comes with a Recorder to create boilerplate code to automate your mobile apps.

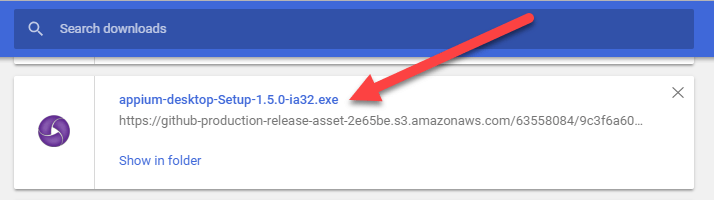
**Step 1)**Go to <http://appium.io/> and click on Download Appium.

[](https://www.guru99.com/images/1/image001.png)

**Step 2)**For Windows, select the exe file and download. The file is around 162MB will take time to download based on your internet speed.

[](https://www.guru99.com/images/1/image002.png)

**Step 3)**Click on the downloaded exe.

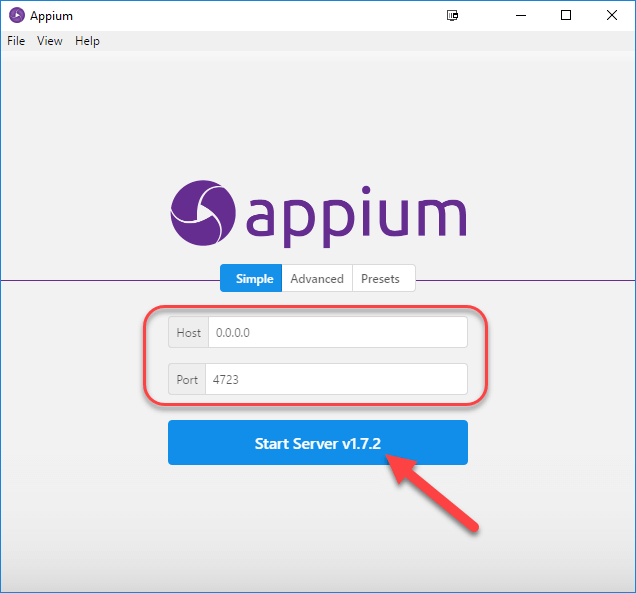
[](https://www.guru99.com/images/1/image003.png)

**Step 4)**On a Windows machine, there is no need to install Appium. It runs directly from the exe. Once you click the exe you will see the following image for few minutes.

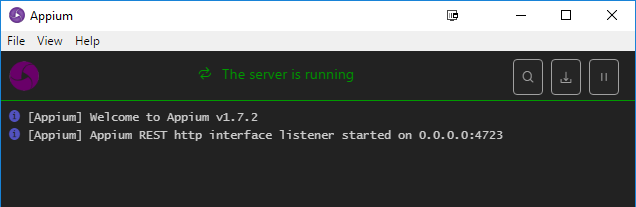
[](https://www.guru99.com/images/1/image004.gif)

For Mac, you need to install the dmg

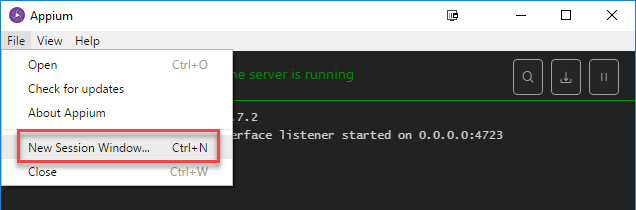
**Step 5)**Next you will see the Server Start Window. It populates the default host and port option which you can change. It also mentions the version of Appium being used.

[](https://www.guru99.com/images/1/image005.png)

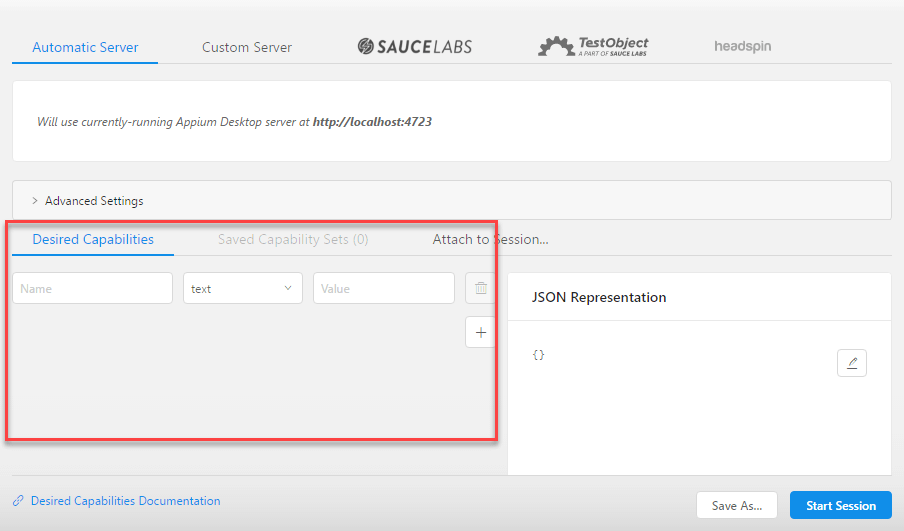
**Step 6)**On clicking the Start Server Button, a new server is launched on the specified host and port. Server log output is shown.

[](https://www.guru99.com/images/1/image006.png)

**Step 7)** Click New Session Window**.**

[](https://www.guru99.com/images/1/image007.png)

**Step 8)** You can enter the Desired Capabilities and start a session.

[](https://www.guru99.com/images/1/image008.png)

**APPIUM Inspector**

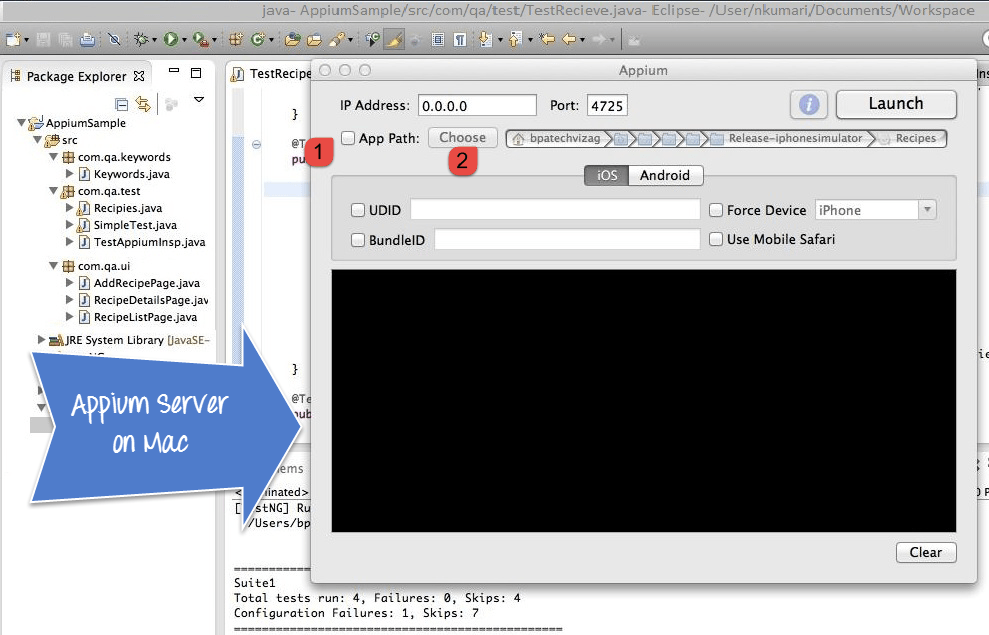
Similar to Selenium IDE record and playback tool, Appium has an 'Inspector' to record and Playback. It records and plays native application behavior by inspecting DOM and generates the test scripts in any desired language. However, currently, there is no support for Appium Inspector for Microsoft Windows. In Windows, it launches the Appium Server but fails to inspect elements. However, UIAutomator viewer can be used as an option for Inspecting elements.

Steps to start with Appium Inspector on Mac machine:-

**Step 1)** Download and start your Appium server with the default IP Address 0.0.0.0 and the port 4725.

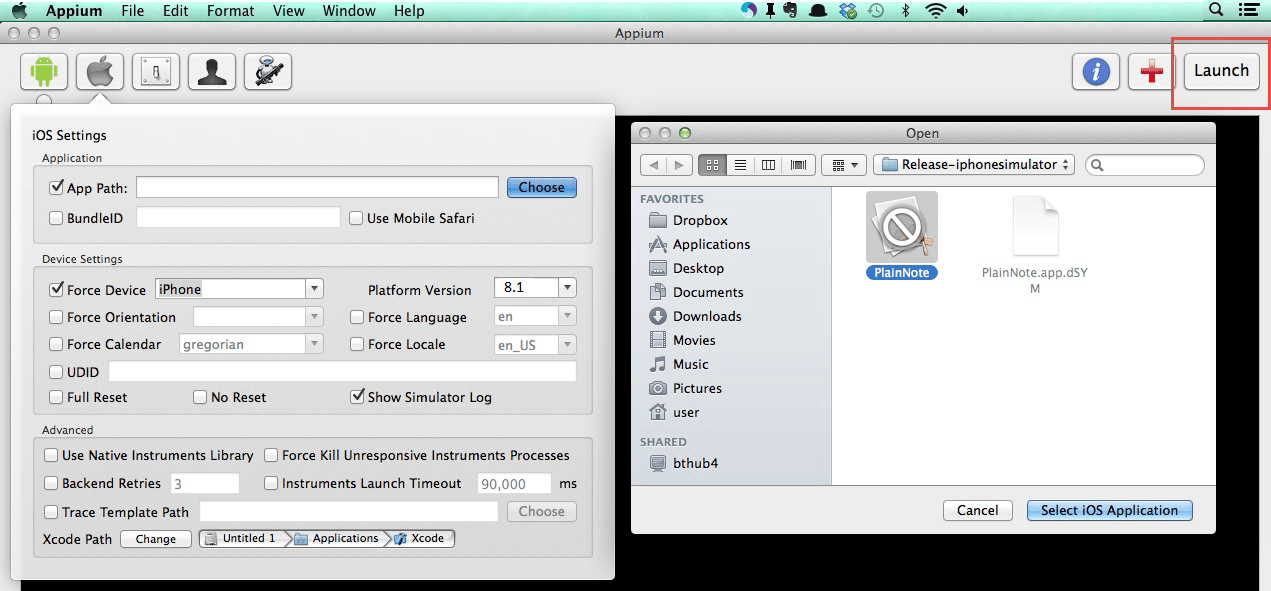
1. Select the source file or .app files from local to test.
2. Check the 'App Path' Checkbox to enable 'Choose' button.

**Step 2)**Now, click on 'Choose' button will give the option to browse and select test file from the local drive.

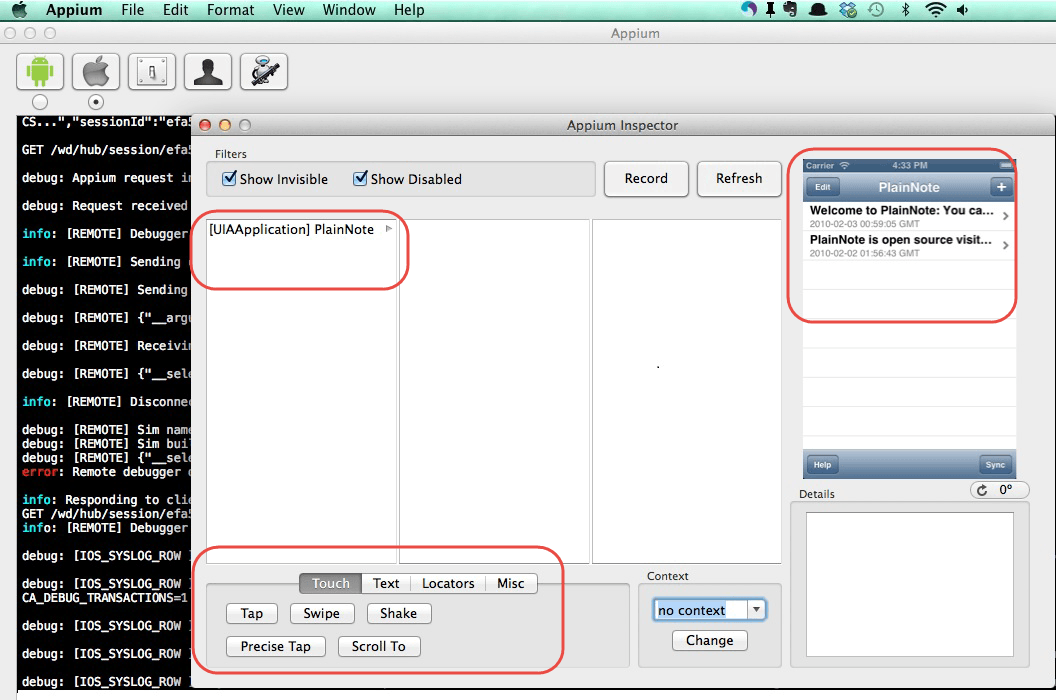
[](https://www.guru99.com/images/1-2015/012315_1021_Introductio1.png)

**Step 3)** Start Simulator on Mac machine.

**Step 4)** Click 'Launch' button from a top right corner, which enables a blue color icon. Again, click on this blue color icon, it will open the Appium inspector and Simulator with a pre-selected application.

[](https://www.guru99.com/images/1-2015/012315_1021_Introductio2.png)

**Step 5)**- Launching your Appium Inspector will show the element hierarchy in column-wise structure. Also, a user can apply actions using buttons like Tap, Swipe, etc.

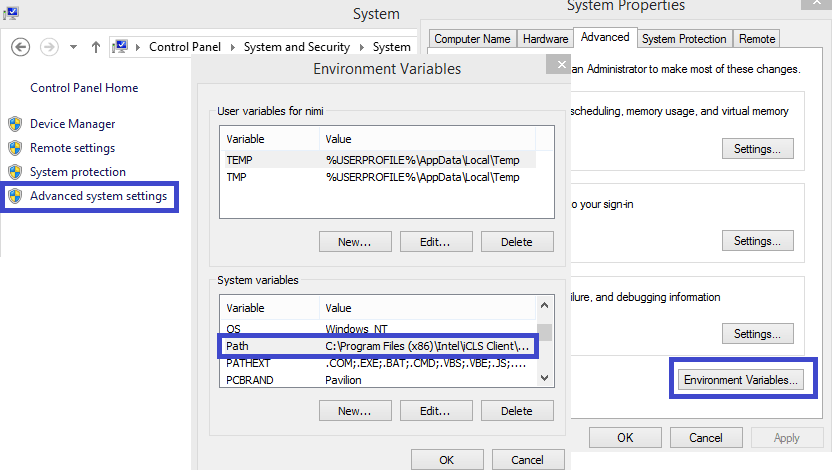
[](https://www.guru99.com/images/1-2015/012315_1021_Introductio3.png)

**Step 6)** Click on 'Stop' button to stop recording.

**Attach Android Emulator to Appium**

**Step 1)** Install Android SDK in your system.

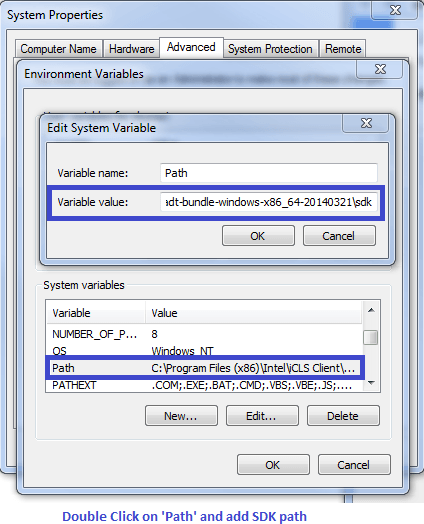
Go to Control panel >> System and Security >> System and from the left panel click on 'Advanced System Settings'. From 'System Properties' pop up, click on 'Advanced' tab and then click on "Environment Variables" button.

[](https://www.guru99.com/images/1-2015/012315_1021_Introductio4.png)

**Step 2)** Now, from 'Environment variables' pop up, 'double click on 'Path' and set ANDROID\_HOME variable that point to your SDK directory. In the path append the whole SDK folder path.

e.g. -

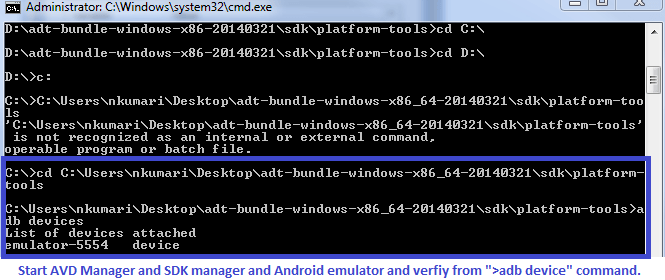
C:\User\ABC\Desktop\adt-bundled-windows-x86\_64-20140321\sdk

[](https://www.guru99.com/images/1-2015/012315_1021_Introductio5.png)

**Step 3)** Start your Android emulator or any attach any Android device to your system (Make sure you have Android Debugging option enabled in your Android device. To check Debugging Option. Go to Device Settings >> Developer Options >> Check "Debugging Option").

**Step 4)** Open Command Prompt and navigate to your Android SDK's \platform-tools\ directory (E.g. D:\adt-bundle-windows-x86\_64-20130514\sdk\platform-tools).

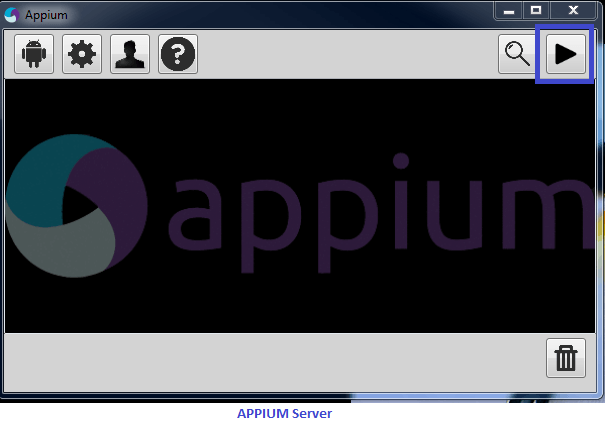
**Step 5)**- Run 'adb devices' command. You can see your connected device listed in Command Prompt window. (In CMD write '>adb devices'- This command will list the connected emulator instances. E.g.: adb –s emulator-5554 install <Location of .apk file>)

[](https://www.guru99.com/images/1-2015/012315_1021_Introductio6.png)

**Step 6)**- Run 'adb start-server' command. It will start ADB server that will be used by Appium to send commands to your Android device.

**Step 7)** Now, navigate to Appium directory in your system and start Appium by clicking an Appium.exe file.

**Step 8)** Do not alter the IP address or port number and click 'Launch' button. Your Appium console starts at 127.0.0.1:4723 as shown in below.

[](https://www.guru99.com/images/1-2015/012315_1021_Introductio7.png)

**Step 9)** Click on 'Start' button, Appium server started running on your system.

**APPIUM Test Case for Native Android App(Calculator)**

**Step 1)**) Download ADT eclipse plugin or download ADT bundled separately [here](http://developer.android.com/tools/help/adt.html)

**Step 2)** Open Eclipse and Create a new Project >> Package >> Class

**Step 3)** Import Selenium library and[Testng](https://www.guru99.com/all-about-testng-and-selenium.html)inside that new project.

**Step 4)** Now Create a small test Program for 'Calculator.app' to sum two numbers.

package src\_Appium;

import java.net.MalformedURLException;

import java.net.URL;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

//import org.openqa.selenium.remote.CapabilityType;

import org.openqa.selenium.remote.DesiredCapabilities;

import org.openqa.selenium.remote.RemoteWebDriver;

import org.testng.annotations.\*;

public class Calculator {

WebDriver driver;

@BeforeClass

public void setUp() throws MalformedURLException{

//Set up desired capabilities and pass the Android app-activity and app-package to Appium

DesiredCapabilities capabilities = new DesiredCapabilities();

capabilities.setCapability("BROWSER\_NAME", "Android");

capabilities.setCapability("VERSION", "4.4.2");

capabilities.setCapability("deviceName","Emulator");

capabilities.setCapability("platformName","Android");

capabilities.setCapability("appPackage", "com.android.calculator2");

// This package name of your app (you can get it from apk info app)

capabilities.setCapability("appActivity","com.android.calculator2.Calculator"); // This is Launcher activity of your app (you can get it from apk info app)

//Create RemoteWebDriver instance and connect to the Appium server

//It will launch the Calculator App in Android Device using the configurations specified in Desired Capabilities

driver = new RemoteWebDriver(new URL("http://127.0.0.1:4723/wd/hub"), capabilities);

}

@Test

public void testCal() throws Exception {

//locate the Text on the calculator by using By.name()

WebElement two=driver.findElement(By.name("2"));

two.click();

WebElement plus=driver.findElement(By.name("+"));

plus.click();

WebElement four=driver.findElement(By.name("4"));

four.click();

WebElement equalTo=driver.findElement(By.name("="));

equalTo.click();

//locate the edit box of the calculator by using By.tagName()

WebElement results=driver.findElement(By.tagName("EditText"));

//Check the calculated value on the edit box

assert results.getText().equals("6"):"Actual value is : "+results.getText()+" did not match with expected value: 6";

}

@AfterClass

public void teardown(){

//close the app

driver.quit();

}

}

Appium Server and Android Emulator from 'AVD Manager' and Click Run >> TestNG. Above program will run the 'Calculator.app' on selected emulator and Result displayed under Eclipse console using a TestNG framework.

**Limitations using APPIUM**

1. Appium does not support testing of Android Version lower than 4.2
2. Limited support for hybrid app testing. e.g.: not possible to test the switching action of application from the web app to native and vice-versa.
3. No support to run Appium Inspector on Microsoft Windows.