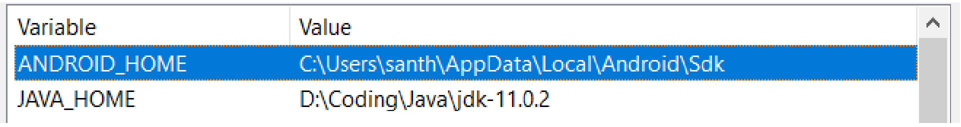
Appium Framework

Appium is the most popular open-source framework for mobile app automation testing. It allows QAs to automate tests for popular mobile platforms like Android, iOS, and Windows. Appium uses the mobile JSON wire protocol (an extension of Selenium JSON wire protocol) to drive native, mobile web and hybrid applications.

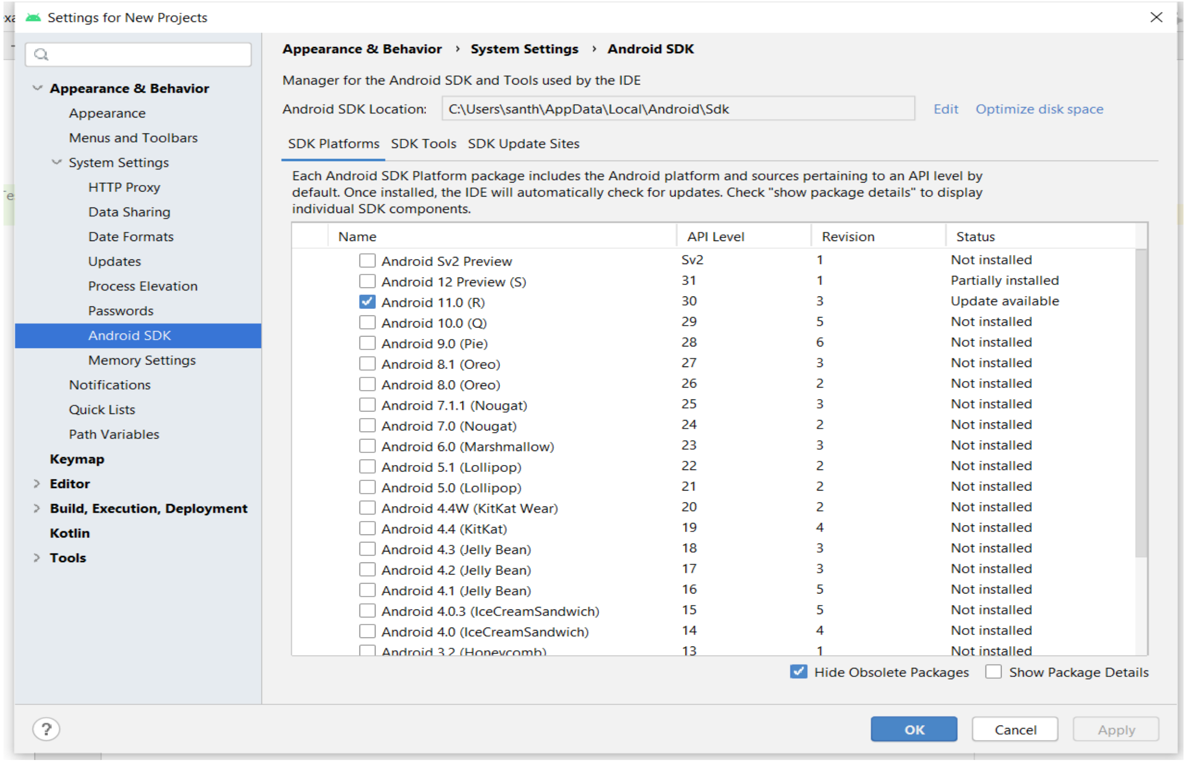
Steps to use Appium

1. Download and Install JDK and JRE from Oracle JavaSE.
    a. Set environment variable Ex -"JAVA_HOME".
2. Download and Install Android Studio from SDK.
    - Set the ANDROID_HOME.



Variable	Value
ANDROID_HOME	C:\Users\santh\AppData\Local\Android\Sdk
JAVA_HOME	D:\Coding\Java\jdk-11.0.2

3. Open Android Studio and Install necessary packages and Tools by Clicking on SDK Manager.
4. Select the options as shown below and click on Ok button.



Appearance & Behavior › System Settings › Android SDK

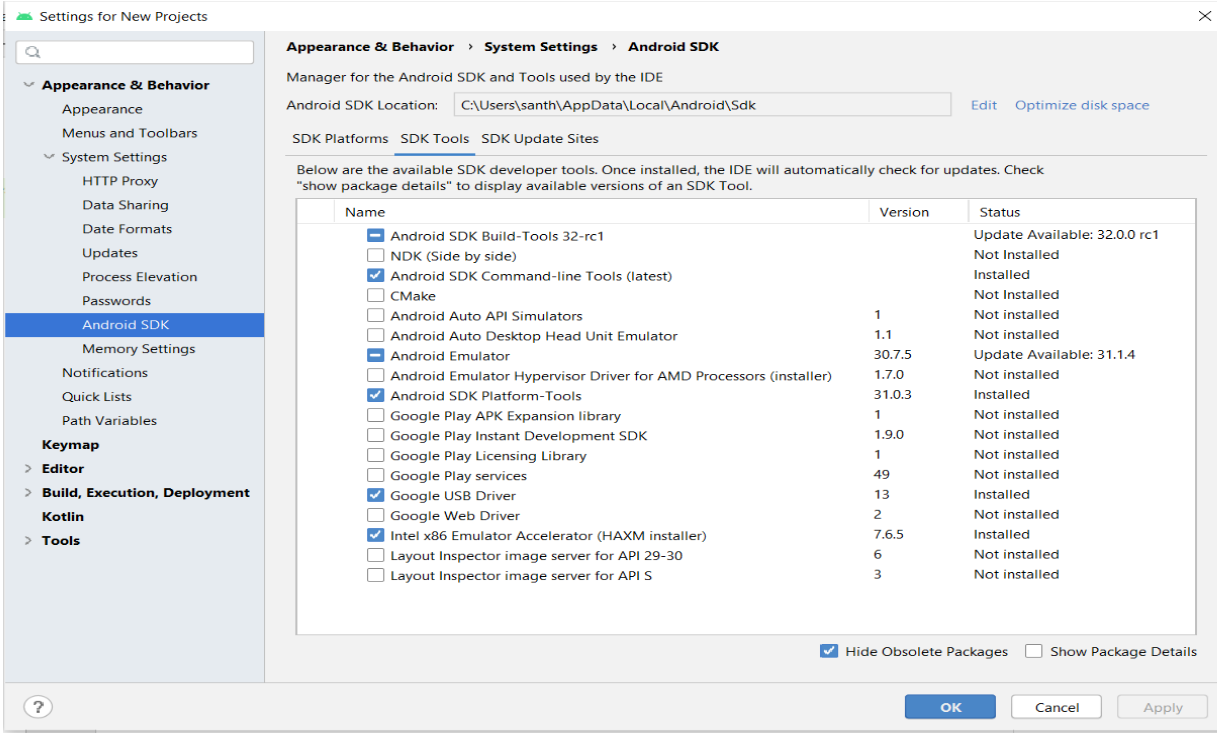
Manager for the Android SDK and Tools used by the IDE

Android SDK Location: C:\Users\santh\AppData\Local\Android\Sdk

Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, the IDE will automatically check for updates. Check "show package details" to display individual SDK components.

Name	API Level	Revision	Status
□ Android Sv2 Preview	Sv2	1	Not installed
□ Android 12 Preview (S)	31	1	Partially installed
☑ Android 11.0 (R)	30	3	Update available
□ Android 10.0 (Q)	29	5	Not installed
□ Android 9.0 (Pie)	28	6	Not installed
□ Android 8.1 (Oreo)	27	3	Not installed
□ Android 8.0 (Oreo)	26	2	Not installed
□ Android 7.1.1 (Nougat)	25	3	Not installed
□ Android 7.0 (Nougat)	24	2	Not installed
□ Android 6.0 (Marshmallow)	23	3	Not installed
□ Android 5.1 (Lollipop)	22	2	Not installed
□ Android 5.0 (Lollipop)	21	2	Not installed
□ Android 4.4W (KitKat Wear)	20	2	Not installed
□ Android 4.4 (KitKat)	19	4	Not installed
□ Android 4.3 (Jelly Bean)	18	3	Not installed
□ Android 4.2 (Jelly Bean)	17	3	Not installed
□ Android 4.1 (Jelly Bean)	16	5	Not installed
□ Android 4.0.3 (IceCreamSandwich)	15	5	Not installed
□ Android 4.0 (IceCreamSandwich)	14	4	Not installed
□ Android 3.2 (Honeycomb)	13	1	Not installed

☑ Hide Obsolete Packages □ Show Package Details



Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by the IDE

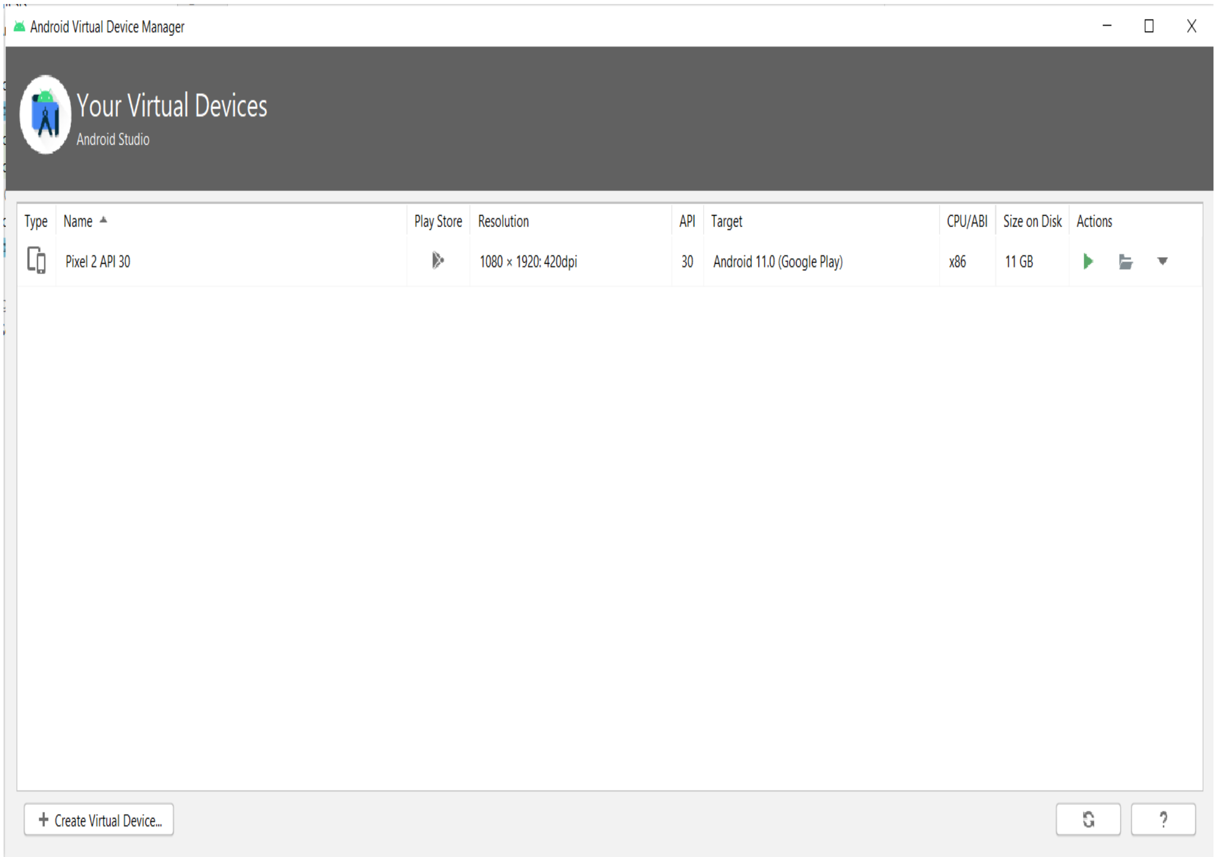
Android SDK Location: C:\Users\santh\AppData\Local\Android\Sdk

Below are the available SDK developer tools. Once installed, the IDE will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

Name	Version	Status
− Android SDK Build-Tools 32-rc1		Update Available: 32.0.0 rc1
☐ NDK (Side by side)		Not Installed
☑ Android SDK Command-line Tools (latest)		Installed
☐ CMake		Not Installed
☐ Android Auto API Simulators	1	Not installed
☐ Android Auto Desktop Head Unit Emulator	1.1	Not installed
− Android Emulator	30.7.5	Update Available: 31.1.4
☐ Android Emulator Hypervisor Driver for AMD Processors (installer)	1.7.0	Not installed
☑ Android SDK Platform-Tools	31.0.3	Installed
☐ Google Play APK Expansion library	1	Not installed
☐ Google Play Instant Development SDK	1.9.0	Not installed
☐ Google Play Licensing Library	1	Not installed
☐ Google Play services	49	Not installed
☑ Google USB Driver	13	Installed
☐ Google Web Driver	2	Not installed
☑ Intel x86 Emulator Accelerator (HAXM installer)	7.6.5	Installed
☐ Layout Inspector image server for API 29-30	6	Not installed
☐ Layout Inspector image server for API S	3	Not installed

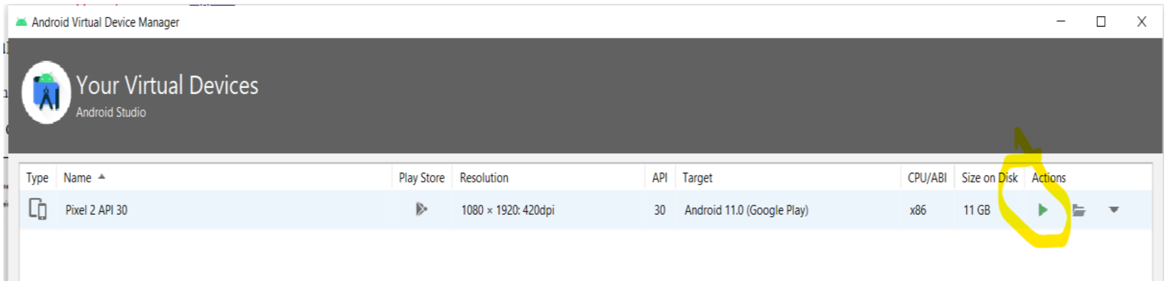
☑ Hide Obsolete Packages ☐ Show Package Details

5. Open Android Studio and click AVD Manager in the toolbar to Create Android Virtual Device.



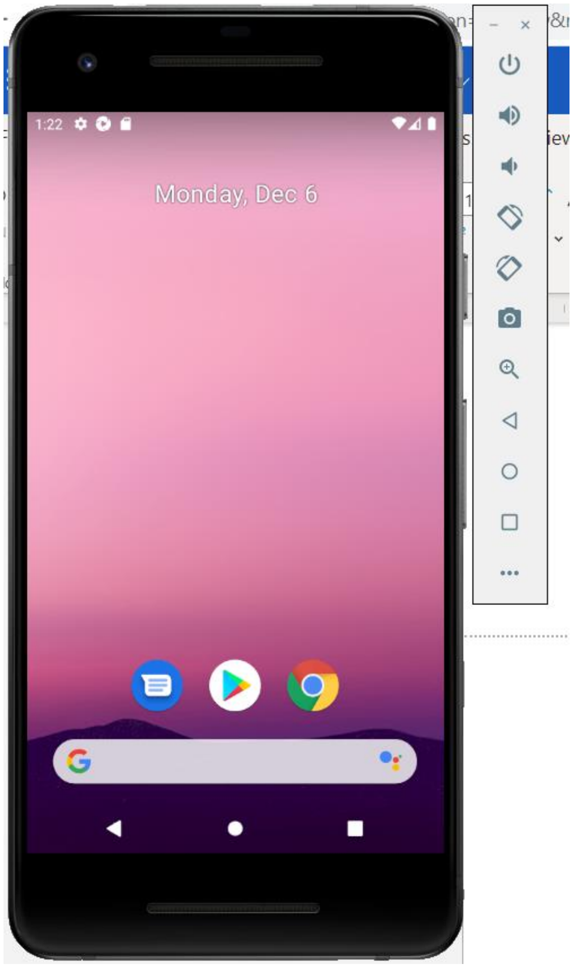
Type	Name ▲	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	Pixel 2 API 30		1080 × 1920: 420dpi	30	Android 11.0 (Google Play)	x86	11 GB	

6. You should now see the AVD you created in the AVD Manager. Launch the Emulator by clicking on below button.

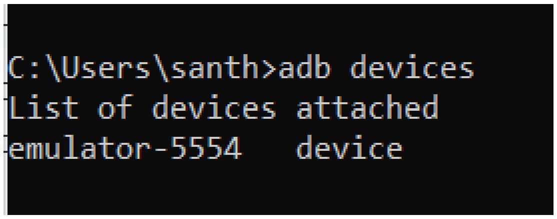


Type	Name ▲	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	Pixel 2 API 30		1080 × 1920: 420dpi	30	Android 11.0 (Google Play)	x86	11 GB	

7. Emulator will be opened.

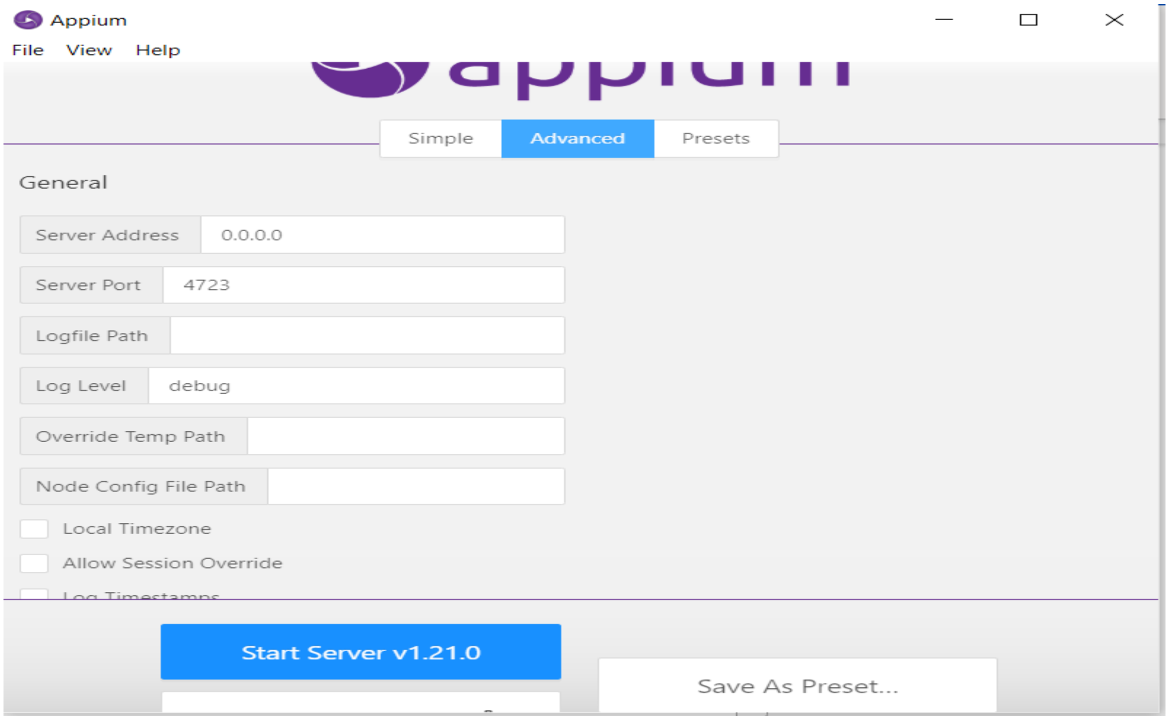


8. To know List of devices attached.



```
C:\Users\santh>adb devices
List of devices attached
emulator-5554   device
```

9. Download Appium.
10. Launch Appium server and start the server.



General

Server Address 0.0.0.0

Server Port 4723

Logfile Path

Log Level debug

Override Temp Path

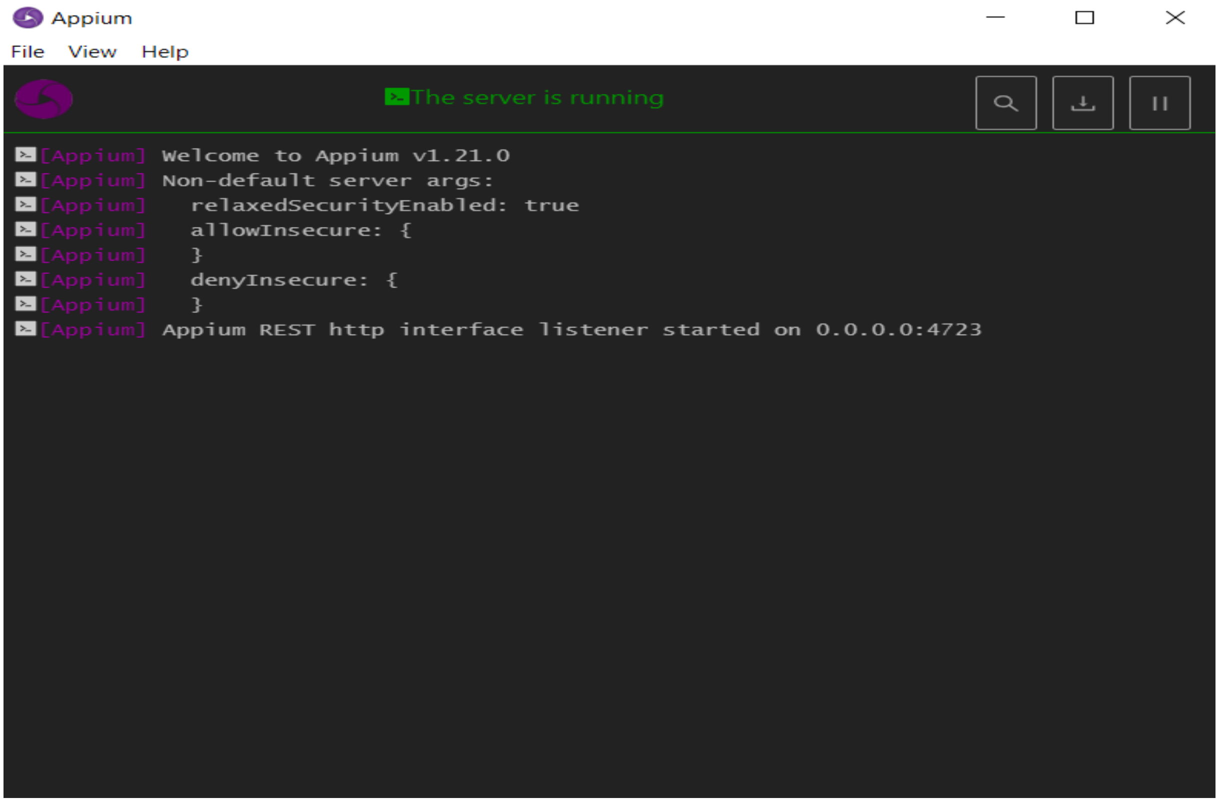
Node Config File Path

□ Local Timezone

□ Allow Session Override

□ Log Timestamps

11. Server Running as shown below.

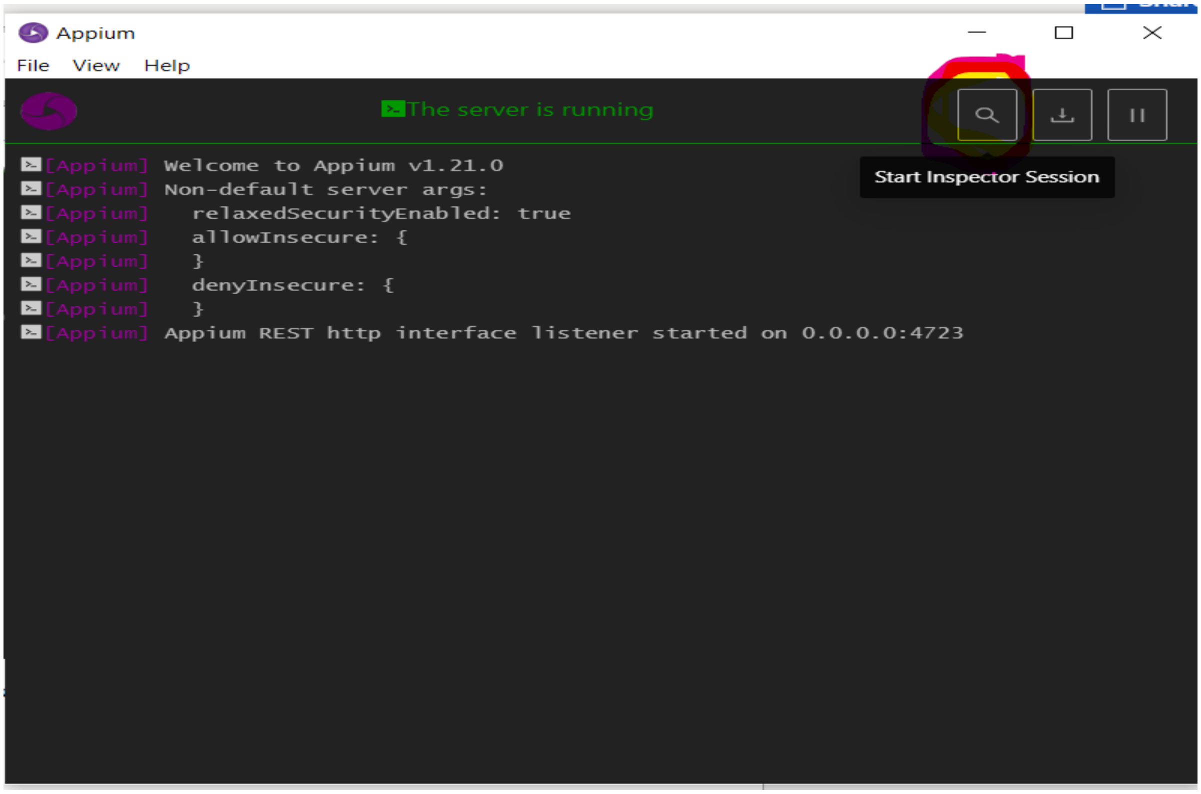


The server is running

```
[Appium] Welcome to Appium v1.21.0
[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
```

Test Android Device with Appium

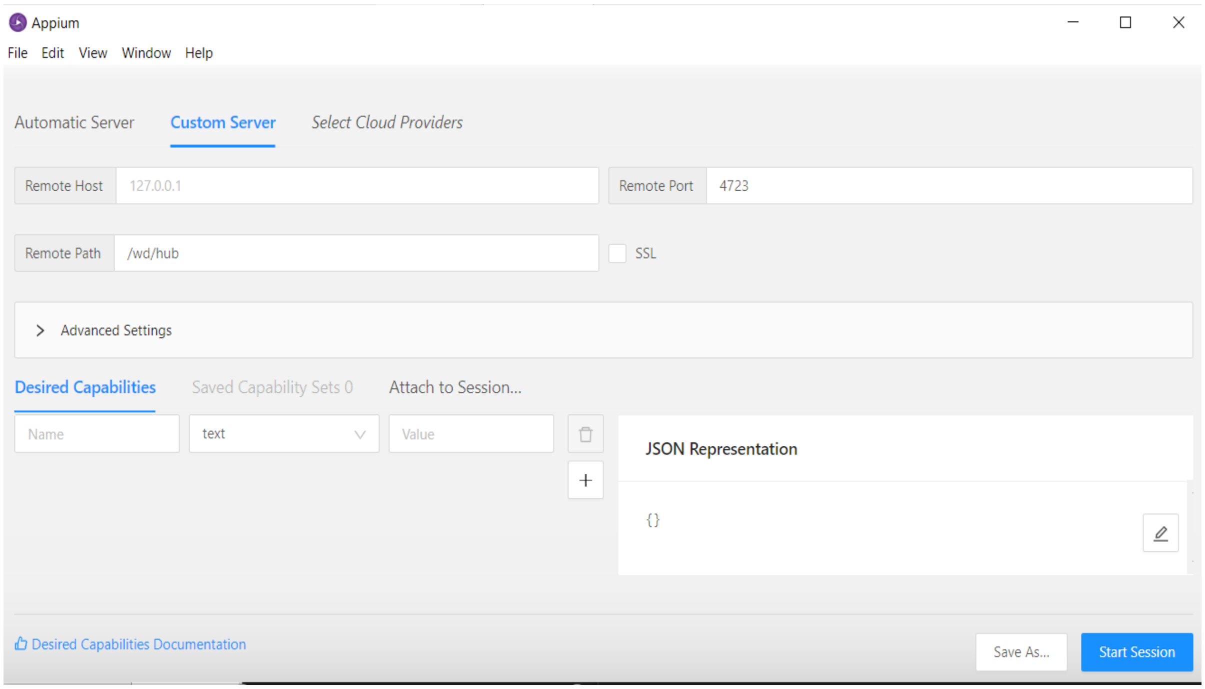
12. Test an Android Device with Appium Click on Start Inspector Session on Appium Page.



The server is running

```
[Appium] Welcome to Appium v1.21.0
[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
```

13. Below page will open.



Remote Host 127.0.0.1

Remote Port 4723

Remote Path /wd/hub

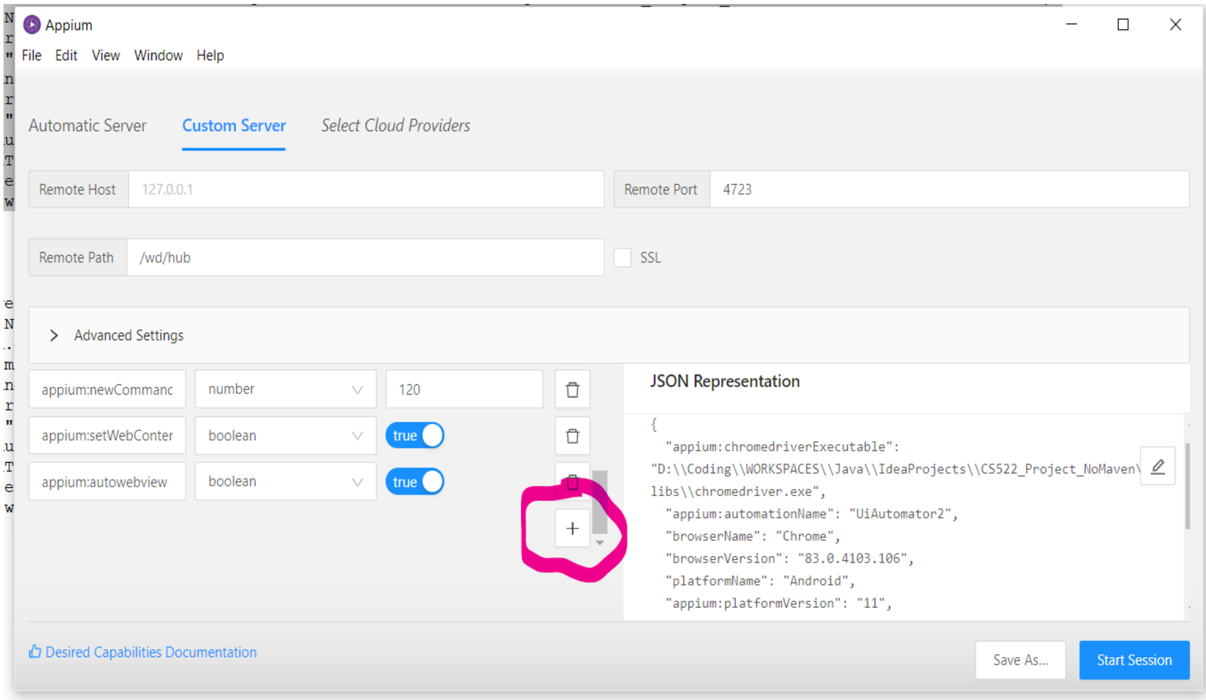
□ SSL

Name | text | Value

JSON Representation

{}

14. Add Capabilities under Advanced Settings tab to test Mobile web application or App (Chrome Browser or messages) in Android device using Appium (By default Appium Page will open in browser.)
15. (Json Code also available when you add Capabilities).



Remote Host 127.0.0.1

Remote Port 4723

Remote Path /wd/hub

□ SSL

appium:newCommanc	number	120
appium:setWebConter	boolean	true
appium:autowebview	boolean	true

```
{
  "appium:chromedriverExecutable":
"D:\\Coding\\WORKSPACES\\Java\\IdeaProjects\\CS522_Project_NoMaven\
libs\\chromedriver.exe",
  "appium:automationName": "UiAutomator2",
  "browserName": "Chrome",
  "browserVersion": "83.0.4103.106",
  "platformName": "Android",
  "appium:platformVersion": "11",
```

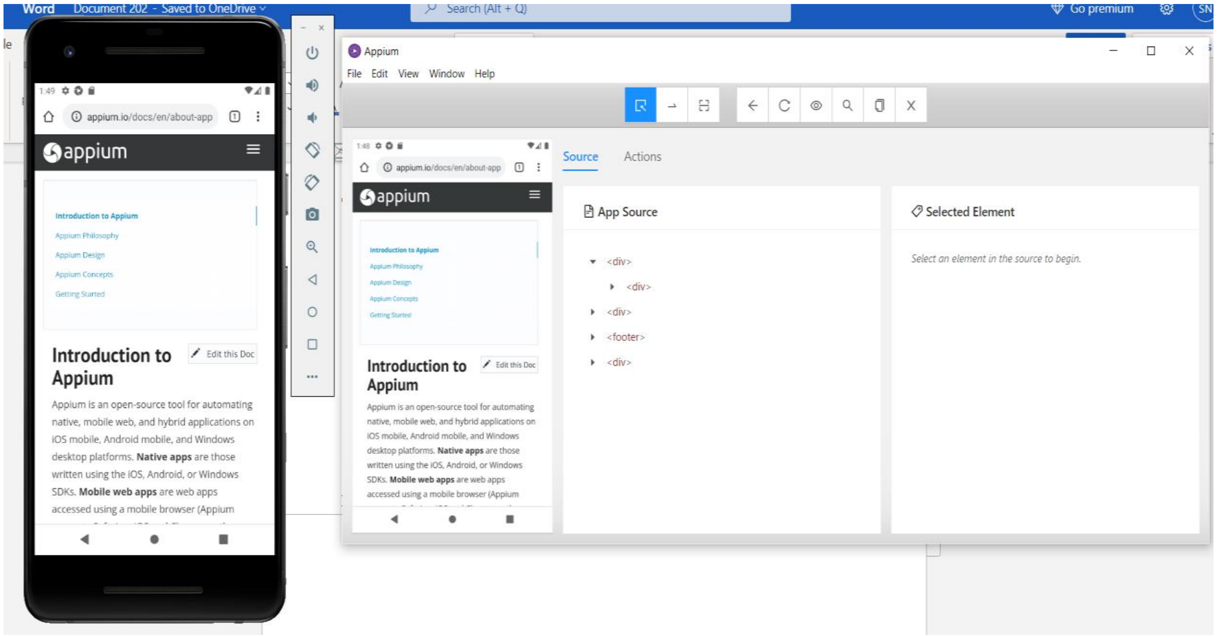
Sample JSON Codes

- **To test web application sample JSON Code.**

JSON Represenation code - //If you want to use JSON Code edit and paste the code then save it.

```
{
  "appium:chromedriverExecutable":
"D:\\Coding\\WORKSPACES\\Java\\IdeaProjects\\CS522_Project_NoMaven\\libs\\chromedriver.exe",

  "appium:automationName": "UiAutomator2",

  "browserName": "Chrome",

  "browserVersion": "83.0.4103.106",

  "platformName": "Android",

  "appium:platformVersion": "11",

  "appium:deviceName": "emulator-5554",

  "appium:udid": "emulator-5554",

  "appium:newCommandTimeout": 120,
  "appium:setWebContentsDebuggingEnabled": true,
  "appium:autowebview": true
}
```

16. Click on Start Session Button. Chrome Browser will launch and opened Appium page in AVD Device.



Introduction to Appium

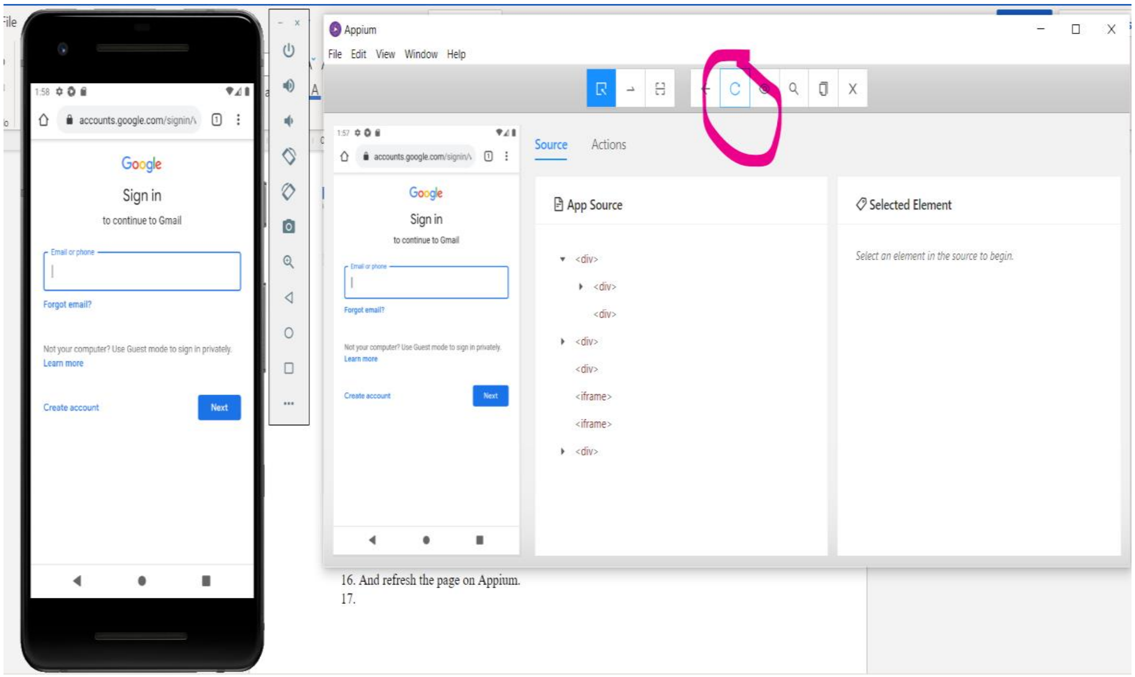
Appium is an open-source tool for automating native, mobile web, and hybrid applications on iOS mobile, Android mobile, and Windows desktop platforms. **Native apps** are those written using the iOS, Android, or Windows SDKs. **Mobile web apps** are web apps accessed using a mobile browser (Appium

- **To test an app sample JSON Code.**

```
{
  "appium:chromedriverExecutable": "D:\\Coding\\WORKSPACES\\Java\\IdeaProjects\\CS522_Project_NoMaven\\libs\\chromedriver.exe",
  "appium:automationName": "UiAutomator2",
  "appPackage": "com.google.android.apps.messaging",
  "appActivity": "com.google.android.apps.messaging.ui.ConversationListActivity",
  "platformName": "Android",
  "appium:platformVersion": "11",
  "appium:deviceName": "emulator-5554",
  "appium:udid": "emulator-5554",
  "appium:newCommandTimeout": 120,
  "appium:setWebContentsDebuggingEnabled": true,
  "appium:autowebview": true
}
```

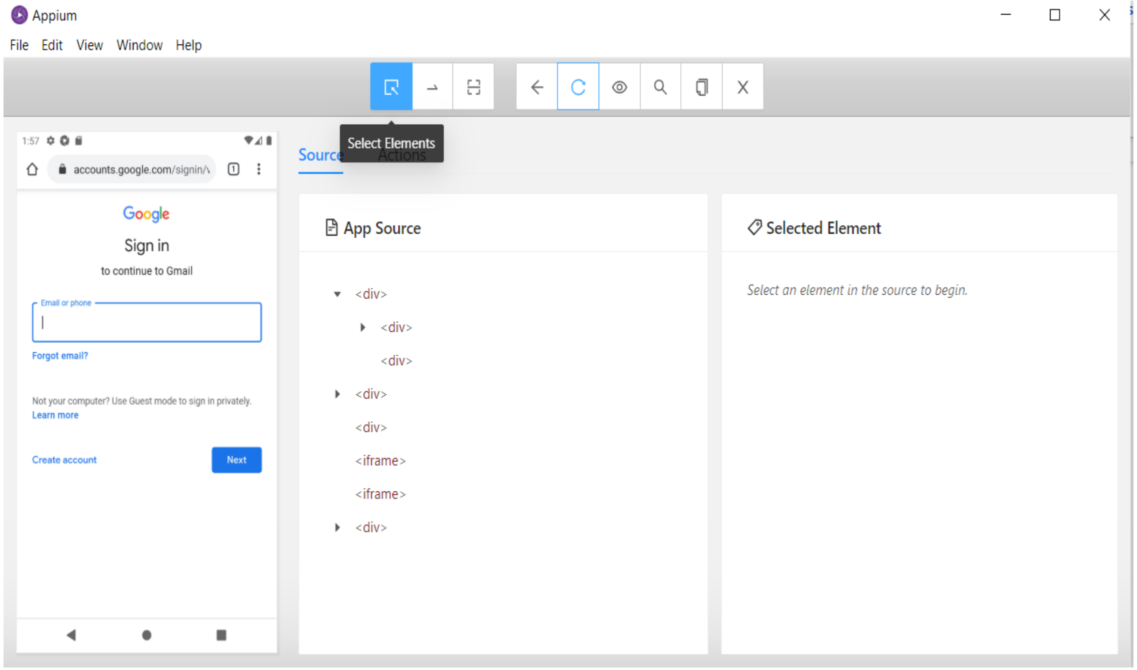
Locate Elements on AVD device using Appium

17. Continuation of above step open gmail.com in AVD Device.
18. And refresh the page on Appium to get sync with AVD Device.

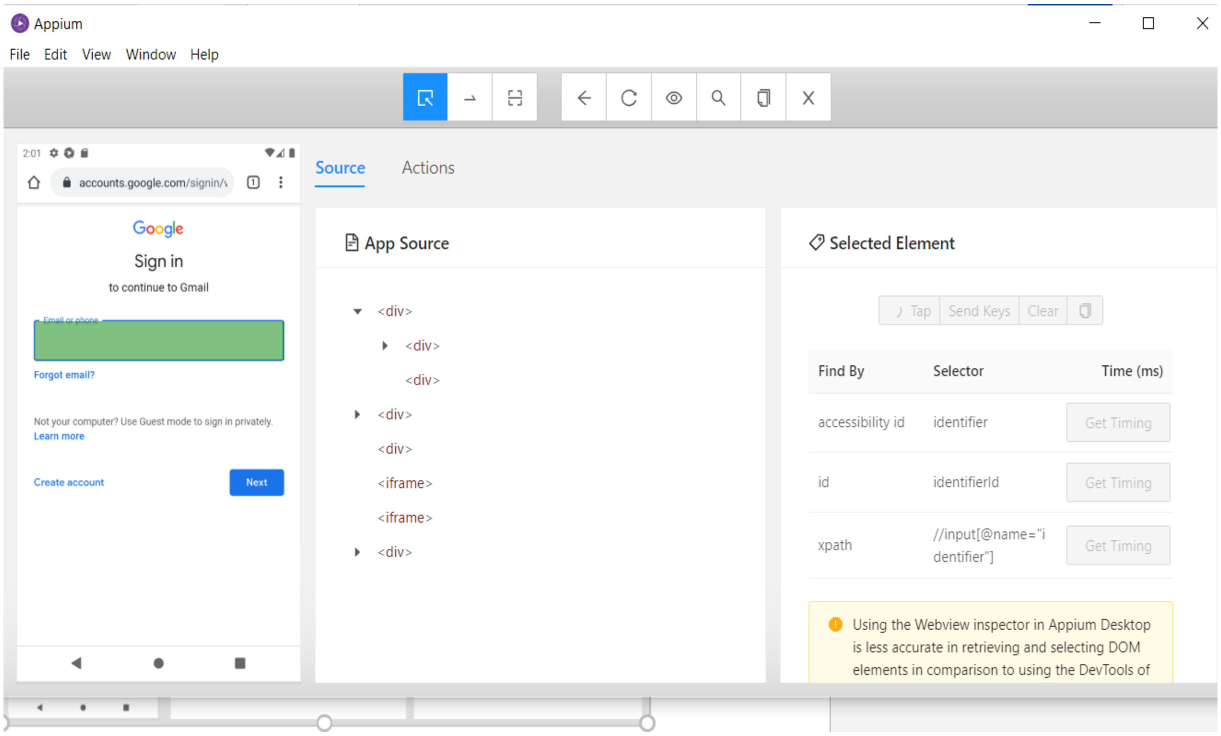


16. And refresh the page on Appium.
17.

19. Select Elements using below button to locate web Elements.



20. In below page you can see the locators xpath, id for email field on gmail web page.
21. Copy Xpath and use it in the Selenium Web driver code.



Find By	Selector	Time (ms)
accessibility id	identifier	Get Timing
id	identifierId	Get Timing
xpath	//input[@name="identifier"]	Get Timing

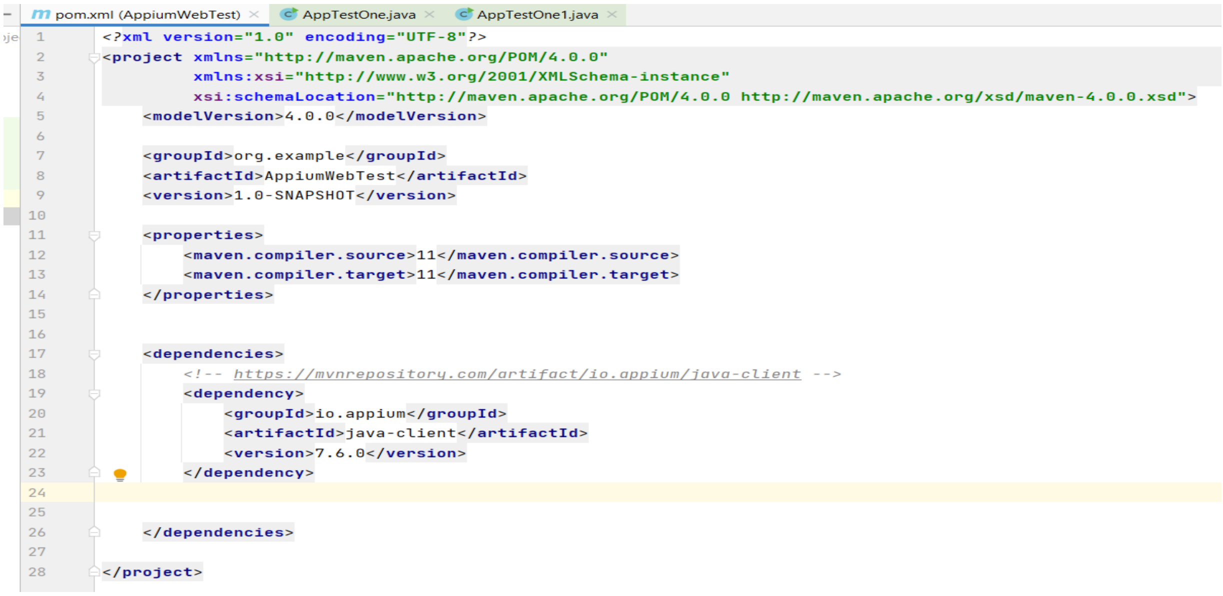
Using the Webview inspector in Appium Desktop is less accurate in retrieving and selecting DOM elements in comparison to using the DevTools of

Test a Mobile Web Application with Appium

22. It’s time to write Selenium Webdriver code (I used IntelliJ).

Steps to follow to write Code -

- Create a Maven Project and add dependencies Ex – Appium Jar file in POM File.



```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>AppiumWebTest</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
    </properties>


    <dependencies>
        <!-- https://mvnrepository.com/artifact/io.appium/java-client -->
        <dependency>
            <groupId>io.appium</groupId>
            <artifactId>java-client</artifactId>
            <version>7.6.0</version>
        </dependency>


    </dependencies>

</project>
```

POM File -

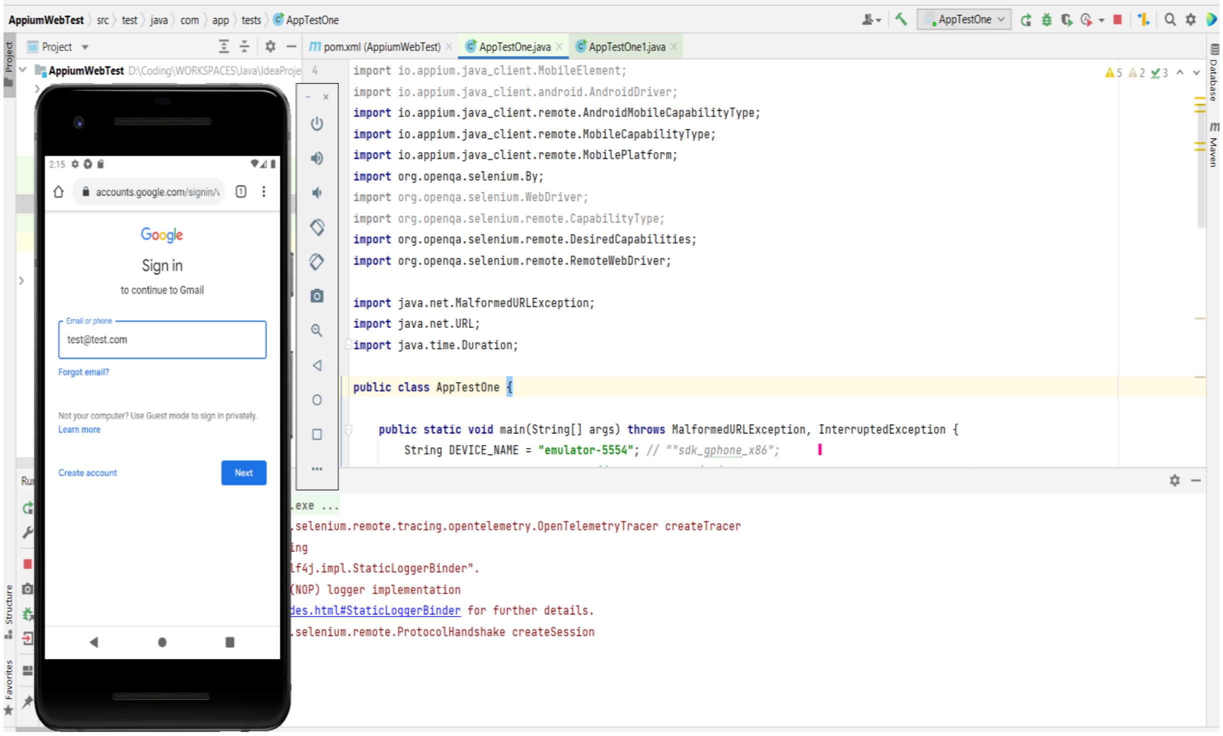
```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>org.example</groupId>
    <artifactId>AppiumWebTest</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
    </properties>


    <dependencies>
        <!-- https://mvnrepository.com/artifact/io.appium/java-client -->
        <dependency>
            <groupId>io.appium</groupId>
            <artifactId>java-client</artifactId>
            <version>7.6.0</version>
        </dependency>

    </dependencies>

</project>
```

- Create. Java Class file and add below code to run a web application (to launch browser) in AVD device using Appium (Selenium Web driver code).
- To run the below code, make sure you have to use chrome browser version should be same in AVD Device and in your local machine.

-
  ```
  package com.app.tests;

  import io.appium.java_client.AppiumDriver;
  import io.appium.java_client.MobileElement;
  import io.appium.java_client.android.AndroidDriver;
  import io.appium.java_client.remote.AndroidMobileCapabilityType;
  import io.appium.java_client.remote.MobileCapabilityType;
  import io.appium.java_client.remote.MobilePlatform;
  import org.openqa.selenium.By;
  import org.openqa.selenium.WebDriver;
  import org.openqa.selenium.remote.CapabilityType;
  import org.openqa.selenium.remote.DesiredCapabilities;
  import org.openqa.selenium.remote.RemoteWebDriver;
  ```

```
import java.net.MalformedURLException;
import java.net.URL;
import java.time.Duration;

public class AppTestOne {

    public static void main(String[] args) throws MalformedURLException,
InterruptedException {
        String DEVICE_NAME = "emulator-5554"; // ""sdk_gphone_x86";
        String APPIUM_SRV_URL = "http://127.0.0.1:4723/wd/hub";
        String CHROME_DRIVER_PATH =
"D:\\Coding\\WORKSPACES\\Java\\IdeaProjects\\CS522_Project_NoMaven\\libs\\chromedr
iver.exe";



        DesiredCapabilities capabilities = new DesiredCapabilities();

        capabilities.setCapability(MobileCapabilityType.PLATFORM_NAME,
MobilePlatform.ANDROID);
        capabilities.setCapability(MobileCapabilityType.AUTOMATION_NAME,
"UiAutomator2");
        capabilities.setCapability(MobileCapabilityType.PLATFORM_VERSION, "11");
        capabilities.setCapability(MobileCapabilityType.DEVICE_NAME, DEVICE_NAME);
        // capabilities.setCapability(MobileCapabilityType.UDID, DEVICE_NAME);
        capabilities.setCapability(MobileCapabilityType.BROWSER_NAME, "Chrome");
        capabilities.setCapability(MobileCapabilityType.BROWSER_VERSION,
"83.0.4103.106");

capabilities.setCapability(AndroidMobileCapabilityType.CHROMEDRIVER_EXECUTABLE,
CHROME_DRIVER_PATH);

        RemoteWebDriver driver = new RemoteWebDriver(new URL(APPIUM_SRV_URL),
capabilities);
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(30));
        driver.get("http://gmail.com");

driver.findElement(By.xpath("//input[@name=\"identifier\"]")).sendKeys("test@test.
com");

        Thread.sleep(5000);
        driver.quit();

    }
}
```

- Save the file and run. (To run the code AVD Device and Appium Server should be up and running, Stop the session on Appium page if you run something before)
- Gmail account will be opened in chrome browser and enters the Email field.



```
import io.appium.java_client.MobileElement;
import io.appium.java_client.android.AndroidDriver;
import io.appium.java_client.remote.AndroidMobileCapabilityType;
import io.appium.java_client.remote.MobileCapabilityType;
import io.appium.java_client.remote.MobilePlatform;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.remote.CapabilityType;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.remote.RemoteWebDriver;

import java.net.MalformedURLException;
import java.net.URL;
import java.time.Duration;

public class AppTestOne {

    public static void main(String[] args) throws MalformedURLException, InterruptedException {
        String DEVICE_NAME = "emulator-5554"; // ""sdk_gphone_x86";
```

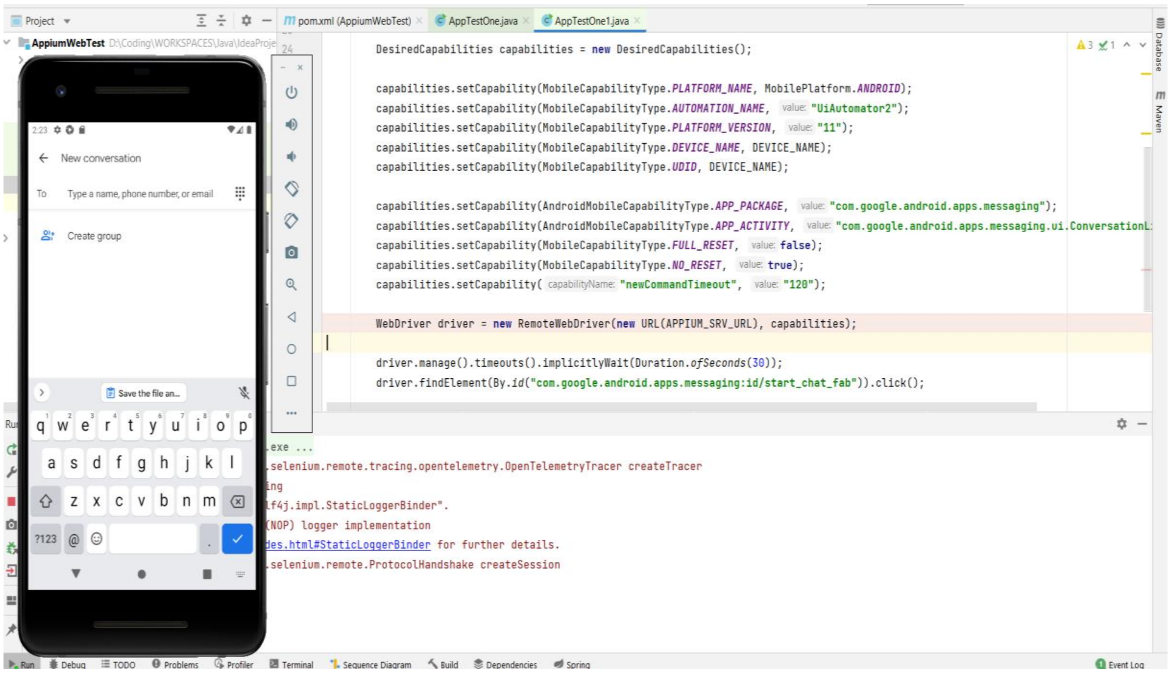
```
.exe ...
.selenium.remote.tracing.opentelemetry.OpenTelemetryTracer createTracer
ing
lf4j.impl.StaticLoggerBinder".
(NOP) logger implementation
des.html#StaticLoggerBinder for further details.
.selenium.remote.ProtocolHandshake createSession
```

Test a Mobile Native App with Appium

- Create. Java Class file and add below code to run an app (Ex – Open Messages App) in AVD device using appium(Selenium Web driver code).
- Trying to open a messages app in AVD and clicking on start chat button.

```
package com.app.tests;
import io.appium.java_client.AppiumDriver;
import io.appium.java_client.android.AndroidDriver;
import io.appium.java_client.remote.AndroidMobileCapabilityType;
import io.appium.java_client.remote.MobileCapabilityType;
import io.appium.java_client.remote.MobilePlatform;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.remote.DesiredCapabilities;
import org.openqa.selenium.remote.RemoteWebDriver;
import java.net.MalformedURLException;
import java.net.URL;
import java.time.Duration;

public class AppTestOne1 {
    public static void main(String[] args) throws MalformedURLException,
InterruptedException {
        String DEVICE_NAME = "emulator-5554"; // ""sdk_gphone_x86";
        String APPIUM_SRV_URL = "http://0.0.0.0:4723/wd/hub";
        String CHROME_DRIVER_PATH =
"D:\\Coding\\WORKSPACES\\Java\\IdeaProjects\\CS522_Project_NoMaven\\libs\\chromedriver.ex
e";

    DesiredCapabilities capabilities = new DesiredCapabilities();

capabilities.setCapability(MobileCapabilityType.PLATFORM_NAME, MobilePlatform.ANDROID);
 capabilities.setCapability(MobileCapabilityType.AUTOMATION_NAME, "UiAutomator2");
        capabilities.setCapability(MobileCapabilityType.PLATFORM_VERSION, "11");
        capabilities.setCapability(MobileCapabilityType.DEVICE_NAME, DEVICE_NAME);
        capabilities.setCapability(MobileCapabilityType.UDID, DEVICE_NAME);
capabilities.setCapability(AndroidMobileCapabilityType.APP_PACKAGE,
"com.google.android.apps.messaging");
        capabilities.setCapability(AndroidMobileCapabilityType.APP_ACTIVITY,
"com.google.android.apps.messaging.ui.ConversationListActivity");
        capabilities.setCapability(MobileCapabilityType.FULL_RESET, false);
        capabilities.setCapability(MobileCapabilityType.NO_RESET, true);
        capabilities.setCapability("newCommandTimeout", "120");

        WebDriver driver = new RemoteWebDriver(new URL(APPIUM_SRV_URL), capabilities);
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(30));

driver.findElement(By.id("com.google.android.apps.messaging:id/start_chat_fab")).click();
        Thread.sleep(5000);
        driver.quit();

    }
}
```

- Save the file and run.



```
DesiredCapabilities capabilities = new DesiredCapabilities();

capabilities.setCapability(MobileCapabilityType.PLATFORM_NAME, MobilePlatform.ANDROID);
capabilities.setCapability(MobileCapabilityType.AUTOMATION_NAME,  value: "UiAutomator2");
capabilities.setCapability(MobileCapabilityType.PLATFORM_VERSION,  value: "11");
capabilities.setCapability(MobileCapabilityType.DEVICE_NAME, DEVICE_NAME);
capabilities.setCapability(MobileCapabilityType.UDID, DEVICE_NAME);

capabilities.setCapability(AndroidMobileCapabilityType.APP_PACKAGE,  value: "com.google.android.apps.messaging");
capabilities.setCapability(AndroidMobileCapabilityType.APP_ACTIVITY,  value: "com.google.android.apps.messaging.ui.ConversationL
capabilities.setCapability(MobileCapabilityType.FULL_RESET,  value: false);
capabilities.setCapability(MobileCapabilityType.NO_RESET,  value: true);
capabilities.setCapability( capabilityName: "newCommandTimeout",  value: "120");

WebDriver driver = new RemoteWebDriver(new URL(APPIUM_SRV_URL), capabilities);

driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(30));
driver.findElement(By.id("com.google.android.apps.messaging:id/start_chat_fab")).click();
```

```
.exe ...
.selenium.remote.tracing.opentelemetry.OpenTelemetryTracer createTracer
ing
lf4j.impl.StaticLoggerBinder".
(NOP) logger implementation
des.html#StaticLoggerBinder for further details.
.selenium.remote.ProtocolHandshake createSession
```