**Preconditions (Set up Step) – One Time Activity**

1) Java Installed with Path variable set

a. In the machine search for Edit environment variables and Search for Path variable

b. In Path variable define the Java path (C:\Program Files\Java\jdk-11\bin)

c. Update the location of java in path variable

2) Android Studio installed with 3 Path variables

a. Make sure SDK is installed in the android either (8.1 or 9)

3) Eclipse Editor

4) Setting up path variable

a. JAVA\_HOME - C:\Program Files\Java\jdk-11\

**b.** ANDROID\_HOME - **C:\Users\harish.pandalangatt\AppData\Local\Android\Sdk**

c. Set the location for SDK, platform tools

i. C:\Users\harish.pandalangatt\AppData\Local\Android\Sdk

ii. C:\Users\harish.pandalangatt\AppData\Local\Android\Sdk\platform-tools

iii. C:\Users\harish.pandalangatt\AppData\Local\Android\Sdk\tools\bin

5) Install Node js in the machine - https://nodejs.org/en/download/package-manager

6) Install Appium (npm install -g appium)

7) Create a new device in Android studio to execute the Appium scripts

a. In Android studio – Click on Device Manager

b. Click on Create Device

c. Select any device Ex: Pixel XL

d. Select the Android version

e. Click on Finish

8) Install the driver for Appium, for Android uiautomator2, and for iOS xcuitest

Command: appium driver install uiautomator2

9) Two ways we can start the appium server

1) appium -> Enter

2) appium –allow-cors

Cors (Cross origin resource sharing)

\*\*Need to give ctrl +c to stop the server\*\*\*

10) Install Appium inspector -<https://github.com/appium/appium-inspector/releases>

[Appium-Inspector-windows-2024.3.1-x64.exe](https://github.com/appium/appium-inspector/releases/download/v2024.3.1/Appium-Inspector-windows-2024.3.1-x64.exe)

**Verification Steps**

1) To check appium version – appium –version

2) To check if the devices are connected and running – adb devices in cmd

3) appium driver list (To check if the appropriate driver is installed)

**Steps for creating a project and dependency:**

Step 1) Create a new Java Project (Provide project Name, uncheck the module-info file)

Step 2) Convert the java project into a Maven Project

Right click on the project ->Configure ->Convert to Maven Project

Step 3) Add the appium dependency

<!-- https://mvnrepository.com/artifact/io.appium/java-client -->

<dependency>

<groupId>io.appium</groupId>

<artifactId>java-client</artifactId>

<version>9.2.3</version>

</dependency>

Types of Mobile Applications

1) Native App –

a. Android Amazon App (Play store)

b. Amazon App iOS (App Store)

2) Web Apps

a. Apps or applications which can be opened in a browser( Chrome – Android, Safari – iOS)

3) Hybrid Apps – It has the UI of the native apps but the platform is developed based on the web technology (cordova or ionic)

\*\*Use API Demo App for testing\*\*\*

<https://github.com/appium/appium/blob/master/packages/appium/sample-code/apps/ApiDemos-debug.apk>

**Steps to establish the connection between the inspector and device (emulator)**

**The following attributes has to be given before starting the connection**

**1)** **Remote host (127.0.0.1)**

**2)** **Remote port (4723)**

**3)** **Remote path (/)**

**Define the capabilities in the caps builder**

**{**

**"appium:automationName": "UiAutomator2",**

**"appium:platformName": "Android",**

**"appium:platformVersion": "14",**

**"appium:deviceName": "emulator-5554"**

**}**

**Universal Relattive xpath**

**//Tagname[@attribute = ‘value’]**

<dependency>

<groupId>io.appium</groupId>

<artifactId>java-client</artifactId>

<version>7.6.0</version>

</dependency>

Program 1:

**package** com.appiumtestcases;

**import** io.appium.java\_client.AppiumDriver;

**import** io.appium.java\_client.remote.options.BaseOptions;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**public** **class** BasicInteractions {

**public** **static** AppiumDriver *driver*;

**public** **static** **void** main(String args[]) **throws** MalformedURLException

{

BaseOptions options = **new** BaseOptions();

options.setPlatformName("Android");

options.setAutomationName("UiAutomator2");

options.amend("deviceName", "emulator-5554");

options.amend("platformVersion", "14");

options.amend("appPackage", "io.appium.android.apis");

options.amend("appActivity", "io.appium.android.apis.ApiDemos");

*driver* = **new** AppiumDriver(**new** URL("http://127.0.0.1:4723"),options);

WebElement AccessibilityElement = *driver*.findElement(AppiumBy.*accessibilityId*("Accessibility"));

AccessibilityElement.click();

*driver*.quit();

}

}

**8th June**

<dependency>

<groupId>io.appium</groupId>

<artifactId>java-client</artifactId>

<version>7.6.0</version>

</dependency>

**Basic Interactions**

**package** com.appiumtestcases;

**import** io.appium.java\_client.AppiumDriver;

**import** io.appium.java\_client.android.AndroidDriver;

**import** io.appium.java\_client.android.AndroidElement;

**import** io.appium.java\_client.android.nativekey.AndroidKey;

**import** io.appium.java\_client.android.nativekey.KeyEvent;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.remote.DesiredCapabilities;

**public** **class** BasicInteractions {

**public** **static** AndroidDriver<AndroidElement> *driver*;

**public** **static** **void** main(String args[]) **throws** MalformedURLException, InterruptedException

{

DesiredCapabilities caps = **new** DesiredCapabilities();

caps.setCapability("deviceName", "emulator-5554");

caps.setCapability("platformVersion", "14");

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

caps.setCapability("automationName", "UiAutomator2");

caps.setCapability("appPackage", "io.appium.android.apis");

caps.setCapability("appActivity", "io.appium.android.apis.ApiDemos");

*driver* = **new** AndroidDriver(**new** URL("http://127.0.0.1:4723"),caps);

Thread.*sleep*(6000);

WebElement AccessibilityElement = *driver*.findElementByAccessibilityId("Accessibility");

AccessibilityElement.click();

Thread.*sleep*(5000);

//To navigate back to the app

*driver*.pressKey(**new** KeyEvent().withKey(AndroidKey.***BACK***));

Thread.*sleep*(3000);

*driver*.findElement(By.*xpath*("//android.widget.TextView[@content-desc='Media']")).click();

*driver*.pressKey(**new** KeyEvent().withKey(AndroidKey.***BACK***));

AndroidElement OSElement = *driver*.findElementByAccessibilityId("OS");

OSElement.click();

Thread.*sleep*(3000);

AndroidElement SMSReciepnt = *driver*.findElementByAccessibilityId("SMS Messaging");

SMSReciepnt.click();

Thread.*sleep*(3000);

AndroidElement smsmessage = *driver*.findElementById("io.appium.android.apis:id/sms\_recipient");

smsmessage.sendKeys("Jagadesh");

Thread.*sleep*(3000);

*driver*.pressKey(**new** KeyEvent(AndroidKey.***SPACE***));

*driver*.pressKey(**new** KeyEvent(AndroidKey.***A***));

//to hide the keyboard opened

*driver*.hideKeyboard();

*driver*.quit();

System.***out***.println("Exceution completed");

}

}

**<!-- https://mvnrepository.com/artifact/org.testng/testng -->**

**<dependency>**

**<groupId>org.testng</groupId>**

**<artifactId>testng</artifactId>**

**<version>7.10.2</version>**

**<scope>test</scope>**

**</dependency>**

**package** com.appiumtestcases;

**import** java.net.MalformedURLException;

**import** java.net.URL;

**import** org.openqa.selenium.WebElement;

**import** org.openqa.selenium.remote.DesiredCapabilities;

**import** org.testng.annotations.Test;

**import** io.appium.java\_client.android.AndroidDriver;

**import** io.appium.java\_client.android.AndroidElement;

**public** **class** CalculatorExample {

**public** **static** AndroidDriver<AndroidElement> *driver*;

@Test

**public** **void** CalculatorTestcase() **throws** MalformedURLException, InterruptedException

{

DesiredCapabilities caps = **new** DesiredCapabilities();

caps.setCapability("deviceName", "emulator-5554");

caps.setCapability("platformVersion", "14");

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

caps.setCapability("automationName", "UiAutomator2");

caps.setCapability("appPackage", "io.appium.android.apis");

caps.setCapability("appActivity", "io.appium.android.apis.ApiDemos");

*driver* = **new** AndroidDriver(**new** URL("http://127.0.0.1:4723"),caps);

Thread.*sleep*(6000);

WebElement AccessibilityElement = *driver*.findElementByAccessibilityId("Accessibility");

AccessibilityElement.click();

*driver*.closeApp();

}

}

[**https://www.browserstack.com/users/sign\_in**](https://www.browserstack.com/users/sign_in)

**TestNG – Used to manage the Test Automation project, it uses Annotations for structuring your test cases. We can run multiple cases using Test NG, using a file called TestNG.xml (Any file)**

**Browser stack – Cloud provider**

**1)** **Launching a browser in a device**

**2)** **Launching an apk in a device**

**3)** **Launching inspector for native apps**

[**https://www.browserstack.com/docs/automate/capabilities**](https://www.browserstack.com/docs/automate/capabilities)

**We need have the following details for estabilishing the session**

**1)** **Browser stack Userid and API key (pass it in the code)**

**2)** **Correct device name and the version of the OS**

**3)** **APK key – for native apps**

**4) Inspection of elements can be done in Real device session**

**Cloud Browser Test**

**package com.appiumtestcases;**

**import org.openqa.selenium.By;**

**import org.openqa.selenium.WebDriver;**

**import org.openqa.selenium.WebElement;**

**import org.openqa.selenium.remote.DesiredCapabilities;**

**import org.openqa.selenium.remote.RemoteWebDriver;**

**import org.testng.annotations.Test;**

**import java.net.\*;**

**public class BrowserStackCloudSafariTest {**

**public WebDriver driver;**

**public static String *USERNAME*="automationarchit\_1qbYK3";**

**public static String *AUTOMATE\_KEY*="Px62iSPzsAhkzyuqqRo1";**

**public static String *URL* ="https://"+*USERNAME*+":"+*AUTOMATE\_KEY*+"@hub-cloud.browserstack.com/wd/hub";**

**@Test**

**public void CloudTestcase() throws MalformedURLException, InterruptedException**

**{**

**DesiredCapabilities caps = new DesiredCapabilities();**

**caps.setCapability("browser", "safari");**

**caps.setCapability("os\_version", "16");**

**caps.setCapability("device", "iPhone 14 Pro");**

**//Set name for our Test**

**caps.setCapability("name", "Cloud Safari Test");**

**//Set the remote driver details**

**driver = new RemoteWebDriver(new URL(*URL*),caps);**

**driver.get("https://www.amazon.in/");**

**System.*out*.println(driver.getTitle());**

**System.*out*.println(driver.getCurrentUrl());**

**WebElement searchBox = driver.findElement(By.*xpath*("//input[@name='k']"));**

**searchBox.sendKeys("Shirts");**

**WebElement SearchIcon = driver.findElement(By.*xpath*("//input[@class='nav-input']"));**

**SearchIcon.click();**

**Thread.*sleep*(6000);**

**driver.quit();**

**}**

**}**

**Native Code on BS:**

**package com.appiumtestcases;**

**import org.openqa.selenium.remote.DesiredCapabilities;**

**import org.testng.annotations.Test;**

**import io.appium.java\_client.android.AndroidDriver;**

**import io.appium.java\_client.android.AndroidElement;**

**import java.net.\*;**

**public class CloudNativeAppTest {**

**@Test**

**public void OpenAPKFile() throws MalformedURLException, InterruptedException**

**{**

**DesiredCapabilities caps = new DesiredCapabilities();**

**//Passing the BS detaisl as caps**

**caps.setCapability("browserstack.user", "automationarchit\_1qbYK3");**

**caps.setCapability("browserstack.key", "Px62iSPzsAhkzyuqqRo1");**

**caps.setCapability("os\_version", "14.0");**

**caps.setCapability("device", "Google Pixel 8");**

**//Set the apk to be launched**

**caps.setCapability("app","bs://032f345ea9f026a1a626f35c5b9f2cd2b8d51ca3");**

**caps.setCapability("name", "Appium Test");**

**AndroidDriver<AndroidElement> driver;**

**driver = new AndroidDriver<AndroidElement> (new URL("https://hub.browserstack.com/wd/hub"),caps);**

**driver.findElementByAccessibilityId("Graphics").click();**

**Thread.*sleep*(5000);**

**driver.quit();**

**}**

**}**

**Annotations Example**

**package com.appiumtestcases;**

**import org.testng.annotations.AfterClass;**

**import org.testng.annotations.AfterMethod;**

**import org.testng.annotations.AfterSuite;**

**import org.testng.annotations.AfterTest;**

**import org.testng.annotations.BeforeClass;**

**import org.testng.annotations.BeforeMethod;**

**import org.testng.annotations.BeforeSuite;**

**import org.testng.annotations.BeforeTest;**

**import org.testng.annotations.Test;**

**public class AnnotationsExamples {**

**@BeforeSuite**

**public void BeforeSuiteMethod()**

**{**

**System.*out*.println("BeforeSuite");**

**}**

**@BeforeTest**

**public void BeforeTestMethod()**

**{**

**System.*out*.println("BeforeTest");**

**}**

**@BeforeClass**

**public void BeforeClassMethod()**

**{**

**System.*out*.println("BeforeClass");**

**}**

**@BeforeMethod**

**public void BeforeMethod1()**

**{**

**System.*out*.println("BeforeMethod");**

**}**

**@Test**

**public void Testcase1()**

**{**

**System.*out*.println("Testcase1");**

**}**

**@Test**

**public void Testcase2()**

**{**

**System.*out*.println("Testcase2");**

**}**

**@AfterMethod**

**public void AfterMethod1()**

**{**

**System.*out*.println("AfferMethod");**

**}**

**@AfterClass**

**public void AfterClassMethod()**

**{**

**System.*out*.println("AfterClass");**

**}**

**@AfterTest**

**public void AfterTestMethod() {**

**System.*out*.println("AFterTest");**

**}**

**@AfterSuite**

**public void AfterSuiteMethod()**

**{**

**System.*out*.println("AfterSuite");**

**}**

**}**