**APPIUM**

**What is Appium?**

* It is an automation tool used to automate & test mobile applications (Native app, mobile web app, Built-in app & hybrid app) on both Android and IOS platform.
* It is developed by Sauce Labs company.
* It is open-source mobile automation tool.

**Appium architecture.**



* When user (selenium client) sent a request(consists Desired Capabilities i.e., package name, activity name, platform name & version , application name ) through JSON protocol to Appium server It will create a session id and sent it back as response to client.
* Based on this session id client(Selenium) and server(Appium) will communicate each other.
* Appium server creates service request and send it to UI Automation(IOS) or UI Automator(Android).
* These UI Automator or Automation will communicate with Bootstrap.jar which is running in emulator , simulator & real device and convert request into operations.

**Advantages :**

* It supports all languages
* Platform independent (android, IOS & Windows)
* Supports all application i.e., Built-in apps, native apps, web apps & hybrid apps.
* Open source

**Disadvantages :**

* Server should be up and running before executing script.
* Slow execution on virtual devices
* Supports only android API levels which are greater than 17.
* Parallel execution will not support on single device.
* Can’t handle Toast messages

**Emulator & Simulator:**

* These are the virtual devices which have same functionality of real devices except camera.
* An android virtual device called emulator and an IOS virtual device called simulator.

**Different types of mobile applications :**

1. **Built-in application or System applications.**

Applications which are installed as part of operation system are called built-in applications.

We can’t un-install these applications from mobile, but we can disable or enable it.

Example : Camera, Calculator, Alarm, Calendar etc.

1. **Native application:**

Applications which are developed on base of particular platform (Android & IOS) are called native applications.

Example : WhatsApp, Facebook, Gmail etc.

1. **Mobile web application:**

These are web applications or websites which are opened with the help of any browser in mobile.

Example : Chrome Browser, Firefox Browser etc.

1. **Hybrid application:**

These are combination of Native and Web applications.

Example : Amazon, Flipkart, Facebook, Instagram, Daily hunt etc.

**Different types of mobile Operating Systems:**

1. Android OS by Google
2. IOS OS by Macintosh
3. Windows by Microsoft

**List of Android versions.**

**Installations & Setup:**

1. Android Studio ( Required java pre-installed)
2. Appium Desktop application
3. Jar files :

* Selenium standalone jar files
* Appium client server jar files
* Common lang3

1. Environment variable

System Variables

* JAVA\_HOME
* ANDROID\_HOME (C:\Users\admin\AppData\Local\Android\Sdk)

Edit path & add

* % JAVA\_HOME%/bin
* % ANDROID\_HOME%/platform-tools
* % ANDROID\_HOME%/tools
* % ANDROID\_HOME%/tools/bin
* % ANDROID\_HOME%/build-tools
* C:\Program Files\Android\Android Studio\bin

**What is APK ?**

* APK stands for Android Package Kit and it's the file format that Android uses distribute and install apps.
* APK is an archive file contain all the elements that an app needs to install correctly on android device.

**What is IPA ?**

* IPA stands for iOS App Store Package and it's the file format that IOS uses distribute and install apps.
* IPA is an archive file contain all the elements that an app needs to install correctly on IOS device.

**What is ADB :**

ADB stands for Android debugging bridge.it is a command line tool which is used to communicate with emulators and android devices connected through USB or WIFI.

Advantages :

1. We can get all devices information of emulators and Real devices (connected with USB or WIFI).

**Command line :** adb devices

1. We can connect real device with WIFI

**Command line :** adb tcpip 5555

adb connect <mobile ip>

1. We can get all packages information

**Command line :** adb shell pm list packages

1. We can get specific apk from device

**Command line :** adb shell pm list packages

adb shell pm path <package-name>

adb pull <package-location> <path-on-computer-to-store-APK>

1. We can install APK on device

**Command line :** adb install <apk path>

1. We can kill adb server

**Command line** : adb kill-server

1. We can restart adb server

**Command line** : adb start-server

What is AVD Manager?

AVD stands for Android virtual device.

AVD Manager is a platform present in android studio that helps to create and manage android virtual devices.

What is SDK Manager?

The SDK manager is a command line tool that allows you to view, install, update, and uninstall packages for the Android SDK

What Is Appium Inspector ?

It is a tool which is used to find element on both android and Ios devices.

What is uiautomator & uiautomation?

Install APK in Virtual Device & Real Device using Appium.

Step1: Start emulator or connect real device with USB or Wifi

Step2. Launch Appium and start server

Step3: Click on start inspector icon

Step4: Under desired Capabilities tab add respective capabilities like platformName, devicename, app, platformversion ect.

Step5: Click on start session.

Install APK in Virtual Device & Real Device using adb.exe

Step1: Start emulator or connect real device with USB or Wifi

Step2. Launch Appium and start server

Step3: open command prompt and execute below line

adb install <apk path>

Install APK in Virtual Device & Real Device by script.

**Script:**

DesiredCapabilities ds = **new** DesiredCapabilities();

ds.setCapability(MobileCapabilityType.***AUTOMATION\_NAME***, "Appium");

ds.setCapability(MobileCapabilityType.***PLATFORM\_NAME***, "Android");

ds.setCapability(MobileCapabilityType.***PLATFORM\_VERSION***, "7.1");

ds.setCapability(MobileCapabilityType.***DEVICE\_NAME***, "Android Emulator"); //or

// ds.setCapability(MobileCapabilityType.***DEVICE\_NAME***, "Android"); // Real Device

// ds.setCapability(MobileCapabilityType.APP, "D:\\Selenium\_SS\\ ApiDemos-debug.apk");

or

ds.setCapability(MobileCapabilityType.APP, "https://github.com/appium/appium/raw/master/sample-code/apps/ApiDemos-debug.apk");

ds.setCapability("appPackage", "com.finance.emi.calculate");

ds.setCapability("appActivity", "com.finance.emi.calculate.ui.SplashActivity");

URL url = **new** URL("http://0.0.0.0:4723/wd/hub");

AndroidDriver<WebElement> driver = **new** AndroidDriver<WebElement>(url,ds);

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

Thread.*sleep*(10000);

driver.findElementById("com.finance.emi.calculate:id/btn\_skip").click();

driver.findElementById("com.finance.emi.calculate:id/rootView").click();

driver.findElementByXPath("//android.widget.EditText[@resource-id='com.finance.emi.calculate:id/principal']").sendKeys("1000000");

driver.findElementById("com.finance.emi.calculate:id/interest").sendKeys("10.25");

driver.findElementById("com.finance.emi.calculate:id/loan\_tenure").sendKeys("5");

driver.findElementByXPath("//android.widget.Button[@text='Calculate']").click();

Thread.*sleep*(10000);

driver.quit();

**Different types of locators in Appium?**

Ref: https://appium.io/docs/en/commands/element/find-elements/index.html#selector-strategies

1. **ID**

Each element has a unique ID which makes easier to identify.

**resource-id** is used as an element identifier for Android and **name** is used for iOS.

**Example:**

driver.findElementById("IntegerA"); // for **iOS**

driver.findElementById("android:id/text1")).click(); //for **Android**

1. **Accessibility ID**

It is a highly preferred locator strategy, especially in the case of automating Android and iOS test cases. Developers can explicitly set the Accessibility ID during development.

For iOS, the default **Accessibility ID** is set to the **name** of the UI element. For Android, the value of Accessibility is same as the value of the attribute “**content-desc**”.

**Example:**

driver.findElementByAccessibilityId("Accessibility").click();

1. **ClassName**

Finding an element using class name is very common, but multiple elements may have the same class name, and this creates a problem in finding one particular element.

So, we needs to use a combination of multiple attributes, for example, combining text with the class name to identify the element.

For iOS, Class Name is represented as the full name of the XCUI element and begins with XCUIElementType.

**For example** – UIAButton, UIARadioButton

In the case of Android, the Class Name is called out as the full name of the UIAutomator2 class. **For example** – android.widget.TextView

**Example** :

List<WebElement> buttons = driver.findElementsByClassName("android.widget.TextView"); for(WebElement button : buttons)

{

System.out.println(button.getText());

if(button.getText().equals("Animation"))

{

button.click();

}

}

1. **Xpath**

We should go xpath only when there is no ID, Name, or accessibility ID assigned to a specific UI element.



**Example:**

driver.findElementByXPath("(//XCUIElementTypeButton)[1]");

driver.findElementByXPath("//android.widget.Button[@text='Calculate']").click();

driver.findElementByXPath("//android.widget.EditText[@resource-id='com.finance.emi.calculate:id/principal']").click();

1. **Android UI Automator**

This locator is Android-specific. we needs to use the UI Automator API, i.e. **UISelector** Class to search for specific elements.

Ref : https://developer.android.com/training/testing/ui-automator#java

**Example:**

String selector = "new UiSelector().text(“Cancel”)).className(“android.widget.Button”))"; MobileElement element = (MobileElement)driver.findElement(MobileBy.AndroidUIAutomator(selector));

UiSelector class : Represents a query for one or more target UI elements on a device.

**methods:**

1. text()
2. new UiSelector().textContains(“xxxx”)
3. new UiSelector().textStartsWith(“xxxx”)
4. new UiSelector().textMatches(“.\*app”)
5. new UiSelector().description(“xxxx”)
6. new UiSelector().descriptionContains(“xxxx”)
7. new UiSelector().descriptionStartsWith(“xxxx”)
8. new UiSelector().descriptionMatches(“.\*app”)
9. new UiSelector().resourceId(“xxxx”)
10. new UiSelector().resourceIdMatches(“.\*app”)
11. new UiSelector().className(“xxxx”)
12. new UiSelector().classNameMatches(“.\*app”)
13. new UiSelector().className(“xxxx”).instance(index)
14. new UiSelector().scrollable(boolean)

**Example:**

driver.findElementByAndroidUIAutomator("new UiSelector().resourceId(\"com.finance.emi.calculate:id/rootView\").index(1)").click();

driver.findElementByAndroidUIAutomator("new UiSelector().className(\"android.widget.EditText\").resourceId(\"com.finance.emi.calculate:id/principal\")").sendKeys("1000000");

driver.findElementByAndroidUIAutomator("new UiSelector().resourceId(\"com.finance.emi.calculate:id/extraRepaymentCard\").clickable(true)").click();

UiScrolable class : Provides support for searching for items in a scrollable UI container.

**methods:**

1. scrollIntoView(new UiSelector().className(“xxxx”))
2. getChildByText(new UiSelector().className(“xxxx”),”text”)
3. getChildByDescription(new UiSelector().className(“xxxx”),”description”)
4. setAsHorizontalList()

**Example**:

driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector()).scrollIntoView(new UiSelector().description(\"android.widget.Buttn\"))").click();

driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector().scrollable(true)).scrollIntoView(new UiSelector().classNameMatches(\".\*android.widget.Button\").index(2))").click();

driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector().resourceId(\"com.finance.emi.calculate:id/emi\_calculator\_content\_pro\")).getChildByText(new UiSelector().className(\"android.widget.Button\"),\"Share Result\")").click();

driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector().resourceId(\"com.finance.emi.calculate:id/emi\_calculator\_content\_pro\")).scrollIntoView(new UiSelector().text(\"Share Result\").instance(0)").click();

driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector().resourceId(\"com.finance.emi.calculate:id/emi\_calculator\_content\_pro\")).setAsHorizontalList().scrollIntoView(new UiSelector().text(\"Share Result\").instance(0)").click();

driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector().resourceId(\"com.finance.emi.calculate:id/emi\_calculator\_content\_pro\")).setAsHorizontalList().getChildByText(new UiSelector().className(\"android.widget.Button\"),\"Share Result\")").click();

1. **Android View Tag**

This is also an Android platform-specific locator. We use this to locate elements using its view tag.

**Example:**

driver.findElementByAndroidViewTag("android.widget.Button ").click();

AndroidDriver class methods:

1. launchApp();
2. installApp(java.lang.String appPath);
3. isAppInstalled(java.lang.String bundleId);
4. resetApp();
5. runAppInBackground(java.time.Duration duration);
6. removeApp(java.lang.String bundleId);
7. closeApp();
8. activateApp(java.lang.String bundleId);
9. terminateApp(java.lang.String bundleId);
10. getPlatformName()
11. getCurrentPackage()
12. currentActivity()
13. getDeviceTime()
14. getScreenshotAs()
15. rotate(ScreenOrientation)
16. isBrowser()
17. isDeviceLocked()
18. isKeyboardShown()
19. lockDevice()
20. lockDevice(Duration)
21. unlockDevice();
22. openNotifications()
23. getKeyboard()
24. hideKeyboard()

TouchAction Class:

Touchaction class used to perform interactive action on mobile application.

1. lognPress(PointOption)
2. lognPress(longPressOptions)
3. tap(PointOption)
4. tap(tapoptions)
5. waitOption()
6. waitOption(WatiOptions)
7. press(PointOption)
8. release()
9. cancel()
10. moveTo(PointOption)

scrolling in Appium

alerts in appium

TextBox, checkbox, radio button, tagle button & Toast message