**1. What is Appium?**   
Appium is an open source,cross-platform test automation tool used for native,hybrid and mobile web apps,which can test on simulators(iOS,FirefoxOS), emulators(Android), and real devices(iOS,Android,FirefoxOS).

**2. Why Appium?**

* Appium is a free open-source test automation framework for mobile testing.
* Using Appium we can automate Android Applications as well as iOS Applications i.e it is a “Cross-Platform” automation tool
* It is an extension to selenium(If the people who already have knowledge on selenium can quickly get the things on Appium and start working).
* Appium supports multiple platforms, i.e, Android, iOS, FirefoxOS
* Appium supports multiple languages with Selenium Webdriver API, i.e, Java, PHP, Python, Ruby, C# etc…
* We can use any testing framework.
* If we use Apple’s UIAutomation library without Appium we can only write tests using JavaScript and we can only run tests through the Instruments application.Similarly, with Google’s UIAutomator we can only write tests in Java.Appium opens up the possibility of true cross-platform native mobile automation.

**3. Which language is used to write tests in Appium?**  
We can write our Appium tests in any language because Appium is nothing more than an HttpServer. Most common languages and development frameworks are C#.net, Java, Ruby, Python and Javascript.

**4. What are the pre-requisites to use Appium?**(Important: Appium Interview Questions)

Below is the list of pre-requisites to start mobile automation:

* Android SDK
* Eclipse IDE
* Java Development Kit(JDK)
* Selenium Server Jars
* WebDriver language binding library
* TestNG
* Appium for Windows
* APK app info on google play
* Java Script

**5. What underlying API does Appium follow?**

Appium uses underlying selenium API to automate Test Cases, i.e. Appium is an extension to selenium because all the Selenium API’s are present in Appium also.

**6. What are Desired Capabilities?**

Desired Capabilities is bunch of keys and it's value. Using this set of keys and values we can tell to appium server that what kind of automation session we are going to create. Example : If you set platformName = Android then appium will understand that you wants to create automation session to run test on Android.

**7. How would you inspect elements of Native Android App?**

Using UIAutomator tool in android SDK we can able to get access to the object locators of the Android Native Apps.

Or

Using Appium Inspector We can inspect the Native Android app.

**8. How will you scroll down in App?**

By using scrollTo() method we can scroll down in App.This method will scroll automatically until the specific text is not matching.(Deprecated for java-client 1.5 and Above)

By using UiScrollable class and scrollIntoView() method

Usage:

driver.findElementByAndroidUIAutomator("new UiScrollable(new UiSelector())

.scrollIntoView(text(\"WebView\"))");

**9. Can you start Appium Server programmatically?**

Yes, we can start Appium Server programmatically.

AppiumDriverLocalService service= AppiumDriverLocalService.*buildService*(

**new** AppiumServiceBuilder().usingDriverExecutable(**new** File("C:\\Program Files\\nodejs\\node.exe"))

.withAppiumJS(**new** File("C:\\Program Files(x86)\\Appium\\resources\\app\\node\_modules\\appium\\build\\lib\\main.js")).withLogFile(**new** File("D:\\log.txt")));

service.start();

**10. How would you inspect elements on iOS Apps?**

By using Appium Inspector which is a GUI based tool we can identify the objects.

**11. How would you identify Mobile Browser objects(Elements)?**   
Ans:

https://developers.google.com/web/tools/chrome-devtools/remote-debugging/?utm\_source=dcc&utm\_medium=redirect&utm\_campaign=2016q3

**12. How would you identify object uniquely when there are multiple objects with same class name using appium**  
By using driver.findElements(By.className) and take the list and get through the index.Using list mechanism and findElements method we will be traversing through the indexes and handle the object uniquely with the class name.

**13. What are the Limitations of Appium?**

* Appium doesn’t support testing of Andriod version which is lower than 4.2
* It has no support to run appium inspector on Microsoft windows
* It has limited support for Hybrid App testing.

**14. What is the difference between Emulator and Simulator?**  
The emulator is used to call Android virtual devices where we use in our PC on a windows machine, and Simulator is used to call for iOS virtual devices which will launch from Xcode in MAC.

**15. How do you detect the device got connected to the PC?**  
By using the command ADB devices we can get the information whether there are any objects or devices detected by the PC.

**16. Will Appium work for Mobile Browser Automation?**  
Yes, we can use Appium to work on Mobile Browsers also.

**17. What are Native Apps?**  
Native Apps or Applications are those written by using the Android or iOS SDK’s.

**18. What are Mobile Web Apps?**  
Mobile Web Applications are those which are accessed using a Mobile browser.

**19. What test frameworks are supported by Appium?**

Appium does not support test frameworks because there is no need to support, we can use appium with all test frameworks which ever we want like NUnit etc…

**20. What are the basic requirements to write Appium tests?**

* We require driver client to write Appium tests.It drives mobile applications as a user.Using client library we write Appium tests which wrap our test steps and sends to appium server over HTTP.
* We need to initialize a session firstly as such appium takes place in the session.Once the automation is done for one session it can be ended and wait for other sessions.
* We need to define certain parameters known as desired capabilities like platform name, platform version, device name and so on in order to initialize an appium session.
* By using a large and expressive vocabulary of driver commands we can write our tests.

**21. What are Hybrid Apps?**  
Hybrid Apps are those which have a wrapper around a webview and a native control which interacts with web content.

**22. What is the difference between Hybrid Apps and Native Apps?**  
Hybrid apps are web based apps that work well on the mobile browser, where as Native apps are written for particular OS i.e, Android, iOS, etc…

**23. What is Appium Inspector?**  
Appium Inspector is similar to selenium IDE.It is a record and playback tool.It records and plays native application behavior by inspection DOM(Documentation Object Model).By DOM Inspection it generates the test scripts in any desired language.Appium uses UIAutomator viewer in its option as appium does not support Windows.

**24. Does Appium support Emulators in iOS?**  
No, Appium doesn’t support Emulators in iOS.

**25. Explain the design of Appium?**(Important: Appium Interview Questions)

Appium is an HTTP server which is written by using node js platform and drivers.It drives Android and iOS session by using webdriver JSON wire protocol. A server is a setup on our machine which exposes REST API when appium is downloaded and installed.Appium receives command and connection request from the client and executes those commands on mobile devices.

Mobile test automation frameworks are apple instruments for iOS, Google UIAutomator for Android API and Selendroid for Android API level 15 or less.

**26. Can Appium run on windows to test iOS app?**

No, Appium will not be able to test iOS apps on the windows server which is hosted locally, because appium relies on OS X-only libraries to support iOS testing.

**27. What kind of platforms supported by Appium?**(Important: Appium Interview Questions)

Appium supports below platforms:

* Android
* Firefox mobile OS
* iOS

**28. Which kind of applications can we automate using Appium?**

We can automate below applications using Appium

* Mobile Web Browser
* Hybrid Applications
* Native Applications

**29. What is the default port number used for Appium Server?**

4723 is the default port number used for Appium Server.

**30. To perform app automation can we use JavaScript in Appium?**

Yes, we can use JavaScript to automate applications using Appium.Whatever the language that supports HTTP request can be used with Appium.

**31. Name the list of Appium Client Libraries?**

Ruby, Java, Python, PHP, and C# are the appium client libraries which are used for extension WebDriver protocol.

**32. Do you need an app’s .apk in Android to automate using Appium or you also need the app in your workspace?**

We need only .apk file in order to automate using Appium in Android.

**33. Can we run our tests in a multithreaded environment while using Appium?**

Yes, we can run our test in a multithreaded environment while using appium, but we have to make sure that not to run more than one test at the same time against the same Appium server.

**34. How we find DOM element in a mobile application?**

By using “UIAutomateviewer” we can able to find DOM element in a mobile application.

**35. Does Appium have the same approach as WebDriver?**  
Yes, Appium has the same approach as WebDriver.

**36. What are the advantages of Appium?** Appium supports multiple programming languages

* It is an open source tool
* It supports iOS and Android platforms, it drives iOS and Android Apps using WebDriver protocol.
* Appium can automate Mobile Web Browser, Hybrid Applications, and Native Applications
* It supports multiple frameworks
* It is a cross-platform automation tool
* It supports Selenium and JSON wire protocol.

**37. What are the disadvantages of Appium?**

* Appium does not support testing of Android versions lower than 4.1
* Doesn’t support Image Comparison

**38. Does a server node is required to execute Appium tests?**

No, we don’t need a server node to execute appium tests.

**39. What are the basic selenium commands to use with appium?**

* Managing Alert boxes
* Elements locate commands by using class name, ID
* Text commands like type(). Syntax: wdBacked.type(locator,value);

**40. The difference between findElement and findElements method in selenium?**

* Find Element means to Capture Single WebElement
* Find Elements means to Capture Multiple WebElements from the List of Webelement

**41. What is the command used to start and stop appium service?**

Once we are done with tests as ‘appiumservice.start()’ and ‘appiumservice.stop()’ we can start appium and stop the appium service. With the help of ‘buildDefaultService()’ we can easily start and stop appium service.

**42. What are the steps to configure and run Mobile Automation Testcases using Appium?** (Important: Appium Interview Questions)

* Download Java and set Java\_Home in environmental variables.
* Download Android Studio.
* Check Android installation path in machine
* Set Android\_Home Environmental variables path to SDK location and include bin folder paths in PATH variable
* Open Android Studio and configure Virtual device/Emulator
* Open Emulator and check whether it is working
* Download Node.js
* Set Node\_Home Environmental variables path
* Set npm Environmental variables path
* Download Appium Server from Node
* Download Appium Java client library
* Install Eclipse and Create a Project in Eclipse and configure Appium libraries
* Start Appium Server

**43. Difference between Selenium and Appium?**(Important: Appium Interview Questions)

Selenium is an open source tool which allows automation of the elements of Web Applications as well as Mobile Applications, where as Appium is also an open source tool to test the web applications which runs on mobile browsers.

**44. How will we run parallel iOS tests in Appium?**(Important: Appium Interview Questions)

By using Sauce Labs we can run iOS tests parallel, we just need to upload our Appium test to sauce labs and it will run as many iOS tests parallel.

**45. What are the advantages of using Appium over Sauce Labs?**

* You don’t have to install/configure the mobile emulators/simulators in your local environment.
* Modifications to the source code of the application are not required.
* You can start scaling your tests instantly.
* Setup time for Appium server locally is reduced.

**46. Can we able to test iOS applications on Linux machine using Appium?**

No, we won’t be able to test iOS applications on Linux machines by using Appium.

**47. What is a Xcode in Appium?**

Xcode is an integrated development environment for OS X and iOS.

**48. Does Appium supports ‘C’ language?**

No, Appium doesn’t support ‘C’ language.

**49. What is POM and what is the basic use of POM?**

POM is basically an XML file which is used for configuring the project in MAVEN.

**50. How will we inspect elements using Appium?**

By using UIAutomatorViewer and Appium Inspector, we can inspect elements in Appium.

**51. What is the Full form of apk extension?**

apk stands for Android Application Package File.

**52. What tools can be used for debugging Android?**

Android Monitor.bat tool is used for debugging Android.

**53. How can we test screen sizes for different devices?**

By using Emulators we can test different devices screen sizes.

**54. What is the extension used for Android files?**

.apk(Android application package) is used for Android files.

**55. What is the extension used for iOS files?**

.ipa(iOS App Store Package) is used for iOS files.

**56. Which is the complex task to perform using Appium?**

The data exchange part is the complex task to perform using Appium.

**57. What are the possible errors you might encounter while using Appium?**

* Fail to start appium session
* Nothing is happening when Clicking on Inspector icon

**58. What tools can be used for debugging iOS?**

iPhone Configuration Utility tool is used for debugging on iOS

**59. What is main advantage of cross platform support of Appium?**  
Using cross platform support of Appium, You can reduce your software automation test-suite's code size as you can reuse code between Android and IOS test-suites.

**60. In which language appium is written?**  
Answer : Appium server is written in Node.js. You can install it directly from Node Package Manager(NPM) or you can build and install it from source.

**61. What is Node.js?**  
It is cross-platform runtime environment which is used to develop server-side software web applications. It is open-source so you can use it without any cost.

**62. Is there any way to access Android SDK tools from eclipse?**

Yes, You can access Android SDK tools from eclipse using it's ADT Plugin.

**63. How to configure android device to run software test using appium in devices?**

In order to run software automation tests using appium in android device, You need to enable USB debugging in your android device.

**64. Tell me the command name for windows to verify that android device is connected with PC.**  
You can run **adb devices** command in windows to make sure that device is connected properly with PC to run appium tests on it.

**65. Appium support multiple languages?**  
Yes, Appium support multiple languages to create test cases. List of appium supported languages are

* Java
* Objective-C
* JavaScript with Node.js
* PHP
* Python
* Ruby
* C#
* Clojure
* Perl

**66. Which tool you are using to locate elements of android software app?**  
I am using UI Automator Viewer tool which is provided by android SDK to locate elements of android app.

**67. Can we locate android app's elements from emulator using UI Automator Viewer tool?**Yes, UI Automator Viewer tool support physical android devices as well as android emulators so you can use it to inspect app's elements from android emulators too.

**68. Tell me different ways to locate android app elements in appium automation.**

You can locate android app's elements using bellow given different ways.

1. XPath
2. ID
3. Name
4. className
5. AccessibilityId
6. *XPath using class and text attribute :*

In above image you can see that button 5 has **text**attribute with value = 5 which is unique. Class name is **android.widget.Button** which is same for all number buttons. So we can create XPath using text attribute value with class name as bellow.

xpath("//android.widget.Button[@text='5']")

1. *XPath using class and resource-id :*

Here resource-id for button 5 is com.android.calculator2:id/digit5. Trailing part (e.g digit5, digit4, digit7) of **resource-id** is changing for every number buttons so we can use contains function to get relative XPath of button 5 as bellow.

xpath("//android.widget.Button[contains(@resource-id,'digit5')]")

1. *XPath using class, text attribute and resource-id :*

We can use combination of **text**attribute and **resource-id** with class name to create unique XPath of element button 5 as bellow.

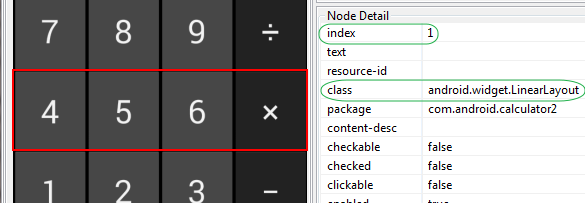
xpath("//android.widget.Button[contains(@resource-id,'digit5') and @text='5']")

1. *XPath using class, text attribute and index :*

We can use same approach as described in point 3 but with index number. Button 5 has index 1 but it is not unique as number buttons 0,2 and 8 has also same index with same class name. So we can include **text**attribute with **index**attribute to locate number button 5.

xpath("//android.widget.Button[@text='5' and @index='1']")

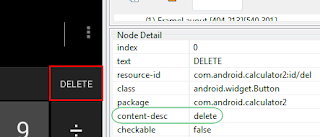
1. *XPath using parent and child class hierarchy*



As you can see in above image, Parent class **android.widget.LinearLayout** class with **index**= 1 has buttons 4, 5, 6 and X. So we can locate that specific row by it's class index. Child element is member of class **android.widget.Button** with **index** = 1. So we can format XPath using parent and child class hierarchy as bellow.

xpath("//android.widget.LinearLayout[@index='1']/android.widget.Button[@index='1']")

1. *XPath using content-desc*

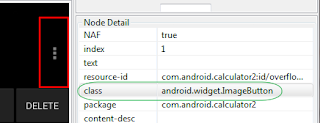
[](http://3.bp.blogspot.com/-cgeWA_z4umQ/VdaJDsNfs7I/AAAAAAAACB8/ooujFuvcqJI/s1600/xpath+using+content-desc.png)

Supposing element has **content-desc** which is unique then you can use it to create XPath. Example : DELETE button has unique **content-desc** = delete. So we can write XPath for delete button using it as bellow.

xpath("//android.widget.Button[@content-desc='delete']")

1. *XPath using class name*

**Note**: Using class name In By.className or By.xpath will works only if it is unique for element. If same class name is provided to multiple elements then it will not work.

[](http://3.bp.blogspot.com/-BdyWWtpJ-UY/VdaM8pMeV1I/AAAAAAAACCQ/nucF53DdEZs/s1600/xpath+using+class+name.png)

If software application's element has a class name and it is unique for element then you can use class name only to create XPath. See above image. Options button has class name **android.widget.ImageButton** which is unique. So we can use it only to create xpath as bellow.

xpath("//android.widget.ImageButton")

1. *Locating Android App Element By ID*

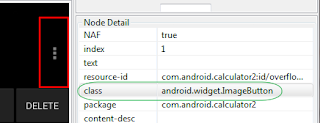
Earlier we learnt how to locate element by ID for web application in [**THIS POST**](http://www.software-testing-tutorials-automation.com/2014/01/how-to-locate-elements-by-id-in.html). We can locate elements by ID in android app too. We can use resource-id as a id. Example is as bellow.

resource-id for button 5 is "com.android.calculator2:id/digit5" so we can locate to it by id as bellow.

id("com.android.calculator2:id/digit5")

1. *Locating Android App Element By className*

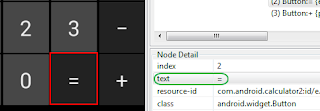
Same way, We can use class name to locate element if it is unique.

[](http://3.bp.blogspot.com/-BdyWWtpJ-UY/VdaM8pMeV1I/AAAAAAAACCQ/nucF53DdEZs/s1600/xpath+using+class+name.png)

Class name for above option button is "android.widget.ImageButton". We can locate it using className as bellow.

By.className("android.widget.ImageButton")

1. *Locating Android App element by Name*It is possible to locate element by name if element contains unique text.

[](http://3.bp.blogspot.com/-iwvrEx_rm0s/Vdp12fDZSuI/AAAAAAAACEQ/XtTc9ydVz40/s1600/find+android+element+by+name.png)

If you see in above image, button Equals has text value "=". So we can locate it by name using bellow given syntax.

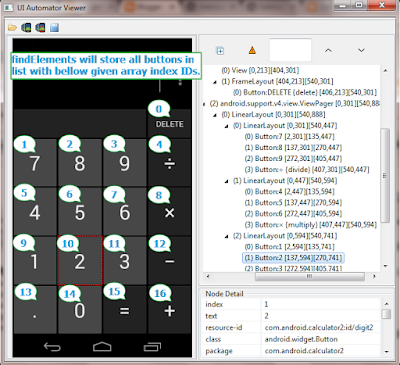
By.name("=")

1. *Locating element by findElements*

If you inspect all buttons of android calculator app, All the buttons has same class "android.widget.Button". If you know, findElements method is useful to get list of all the matching elements from current page as per given element locator mechanism. [**VIEW MORE DETAIL**](http://www.software-testing-tutorials-automation.com/2014/02/selenium-webdriver-difference-between.html) on findElements. So here we can collect list of all buttons using findElements and store list using java

List interface. Then we access required element using get() method of List interface.

Array list Index IDs of button elements of "android.widget.Button" class is as bellow.

[](http://3.bp.blogspot.com/-l4sSNl0yn-o/VdqJFdQT8gI/AAAAAAAACEo/kQ0E8RZcM_Q/s1600/findelements+to+store+list+of+android+elements.png)

List<WebElement> calcButtons = driver.findElements(By.xpath("//android.widget.Button"));

Above given syntax will store list of all buttons in list **calcButtons**with bellow given array list index ids. Bellow given table illustrate array index ids for different buttons of calculator application.

| **Button** | **Array index ID** |
| --- | --- |
| DELETE | 0 |
| 7 | 1 |
| 8 | 2 |
| 9 | 3 |
| ÷(divide) | 4 |
| 4 | 5 |
| 5 | 6 |
| 6 | 7 |
| ×(multiply) | 8 |
| 1 | 9 |
| 2 | 10 |
| 3 | 11 |
| −(minus) | 12 |
| .(point) | 13 |
| 0 | 14 |
| =(equals) | 15 |
| +(plus) | 16 |

So now we can access required button using it's index which is stored in list **calcButtons**. Means if you wants to tap on button 5 then you can use bellow given syntax.

calcButtons.get(6).click();

Here 6 is list index id of button 5 as shown in above table.

So all these are different ways to locate android software app elements. We will use all these element locator methods practically in our test scripts which I will publish in upcoming steps.

**69. What is use of package and launcher activity name of android app in appium automation test?**

**Answer**: In appium automation test, We need to provide android app's package and launcher activity name so that appium can understand which application to launch for test when you start running appium test. We can set package and launcher activity name as desired capability.

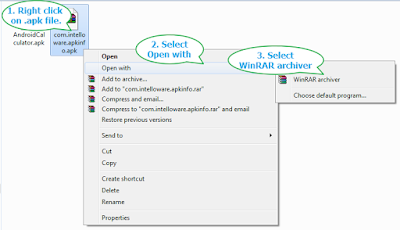
**70.  How to get android app's package and launcher activity name?**

*Method 1 : Using AndroidManifest.xml*

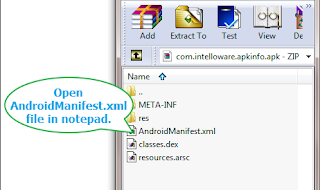
Every .apk file contains **AndroidManifest.xml** file which contains android app's package and activity name. Bellow given steps will tell you how to open AndroidManifest.xml file and read package and activity name.

**Prerequisites**:  1. .apk file for which you wants to get package and activity name. 2. WinRAR should be installed in your PC.

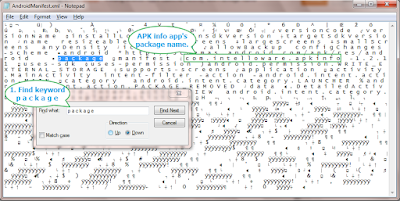
* **Right click on .apk file** of APK Info app(or any other .apk file for which you wants to know package and activity name).
* Select **Open with** -> **WinRAR archiver** option from right click context menu.

[](http://4.bp.blogspot.com/-0c07A3lv6p0/VeupPzjv-SI/AAAAAAAACOw/uQAMuTAYRoU/s1600/open+manifest+xml.png)

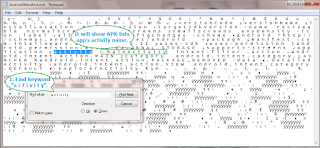
* It will open .apk file in WinRAR window and you can see AndroidManifest.xml there as shown in bellow image.

[](http://2.bp.blogspot.com/-lDwl4vcMLDQ/VeuqtIoZ2lI/AAAAAAAACO8/wfDW7s3V5E8/s1600/open+androidmanifest+file.png)

* Open AndroidManifest.xml file in notepad.
* Now find keyword "p a c k a g e" In notepad. It will show you package name of your app as shown in bellow image.

[](http://4.bp.blogspot.com/-kqFp5UDyOUc/VeusgznXrPI/AAAAAAAACPI/VZ-9poeurUA/s1600/get+package+name+using+androidmanifest.png)

* Package name for my APK Info app is : **com.intelloware.apkinfo**
* Find keyword "a c t i v i t y" In notepad. It will show you activity name of your app as shown in bellow image.

[](http://1.bp.blogspot.com/-SPdr8zOY4FE/VeuuWFU3-AI/AAAAAAAACPU/8BJRs06TIPU/s1600/get+activity+name+using+androidmanifest.png)

* Activity name for my APK Info app is display: .**MainActivity**. Here you need to append package name ahead of activity name so full activity name is : **com.intelloware.apkinfo.MainActivity**

*Method 2 : Using hierarchyviewer of SDK*

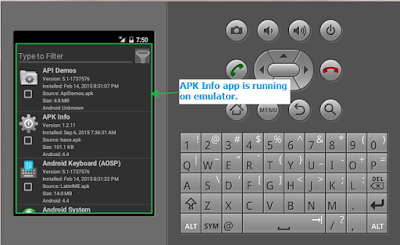
**Prerequisites**:

1. Android SDK should be installed and configured as described [**HERE**](http://www.software-testing-tutorials-automation.com/2015/09/steps-to-download-and-install-android.html).
2. Android emulator should be created as described [**HERE**](http://www.software-testing-tutorials-automation.com/2015/10/how-to-create-and-start-android-virtual.html).
3. .apk should be installed in emulator as described [**HERE**](http://www.software-testing-tutorials-automation.com/2015/10/installuninstall-app-in-emulator-avd-of.html). I will use APK Info app in this example so i have installed it in my emulator.

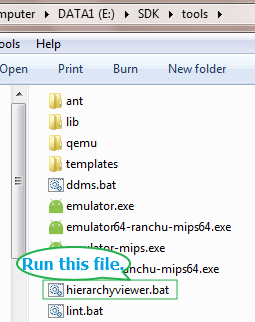
hierarchyviewer is a tool provided by android SDK. You will find it(hierarchyviewer.bat) inside **tools**folder of SDK. Here we will use it to get package and launcher activity name of APK Info app. Follow bellow given steps.

**Note**: hierarchyviewer is works well with emulators only for me. So try to use it will emulators only.

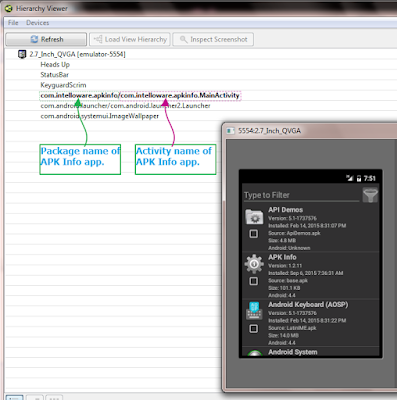
* Launch android emulator and open APK Info app in emulator.

[](http://4.bp.blogspot.com/-s7GdPVuaPls/Veu5FGTc6VI/AAAAAAAACPk/0jAIOp4r1xQ/s1600/start+apk+info+app+in+emulator.png)

* Open **hierarchyviewer.bat** file from **tools**folder of SDK.

[](http://1.bp.blogspot.com/-BviXN28fXjM/Veu8WQEP4-I/AAAAAAAACPw/KWYJl6MTiKM/s1600/run+hierarchyviewer+bat+file.png)

* It will show hierarchyviewer interface. It will show you package and activity name of you app as shown in bellow image.

[](http://2.bp.blogspot.com/-K4X4aZ9Dh_s/Veu9-94myLI/AAAAAAAACP8/WYiM_RKgIKQ/s1600/hierarchy+view+of+apk+info.png)

This way, You can get Android App's Launcher Activity And Package Name.

**71. Can you tell me usage of findElements() method?**

We can use findElements() method to get the list of elements from page.

Example : Supposing there are 5 buttons on page with same class name. In this case, We can use findElements() method and provide reference of class name in xpath(Example : findElements(By.xpath("//android.widget.Button"))). So it will return list of all button elements. Then we can use get() method and provide reference of button's index to locate specific button(Example : findElements(By.xpath("//android.widget.Button")).get(0)). View **[findElements() Example](http://www.software-testing-tutorials-automation.com/2015/10/appium-android-test-using-different.html)**.

**72. How to perform drag and drop action in appium?**

Drag And Drop Is one of the common action of any android app. You will see many android mobile apps where you can drag and drop element from one place to other place. If you are automating android mobile app where you have to perform drag and drop then you need to use **[TouchAction class](http://appium.github.io/java-client/index.html?io/appium/java_client/TouchAction.html)**. TouchAction class provides us facility to **automate mobile gestures** of **android app using appium**. Let's take very simple example to perform drag and drop operation on android application using appium.

**PREREQUISITES** : Previous 18 steps (Listed In [**PART 1**](http://www.software-testing-tutorials-automation.com/2015/09/appium-tutorials.html)and [**PART 2**](http://www.software-testing-tutorials-automation.com/2015/10/appium-tutorials-part-2.html)) of appium tutorials should be completed.

**Download Drag-Sort Demos app**

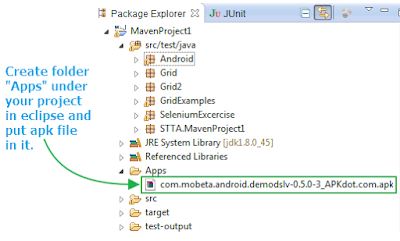
We will use **Drag-Sort Demos** app to perform drag and drop operation in android device. You can download it from [**Google Play Store**](https://play.google.com/store/apps/details?id=com.mobeta.android.demodslv&hl=en) or [**THIS PAGE**](http://www.software-testing-tutorials-automation.com/2015/11/test-apps-to-use-in-appium-automation.html).

**Create New Folder Under Project Folder To Store APK File**

You need to create new folder "Apps" under your project folder and put downloaded Drag-Sort Demos's .apk file inside it.

To create new folder under project in eclipse,

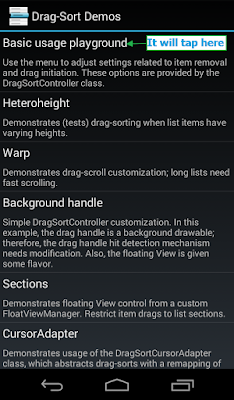
* Right click on project(MavenProject1 for me) folder -> New -> Folder.
* Give folder name = **Apps** and click on Finish button. It will add Apps folder under project folder.
* Copy-paste downloaded Drag-Sort Demos apk file inside it as shown in bellow image.

[](http://4.bp.blogspot.com/-tXgQpSPwwqw/VfRQRZuUuYI/AAAAAAAACQU/rDDmeUqIX6I/s1600/set+apk+file+under+project.png)

**Aim Of This Appium Android Test**

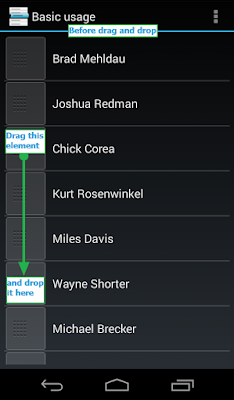
Our appium automation test script will,

* Install Drag Sort Demos app in android mobile device automatically.
* Then It will launch Drag Sort Demos app in your android mobile device.
* It will tap on "Basic usage playground" text.

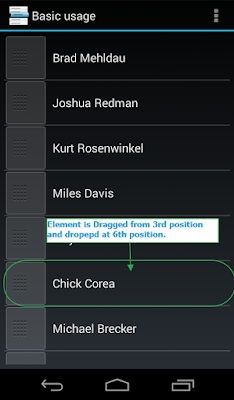
[](http://2.bp.blogspot.com/-hogga1rnb_Q/VfRU155pW0I/AAAAAAAACQg/EL8Ec1-fV3g/s1600/tap+on+basic+usage+playground.png)

* Then It will locate and drag 3rd element and drop it to 6th position as shown in bellow images.

Before drag and drop

[](http://1.bp.blogspot.com/-6PsZca90fTg/VfTKqNymmHI/AAAAAAAACQw/JAypXKXPolY/s1600/before+draga+and+drop.png)

After drag and drop

[](http://2.bp.blogspot.com/-Hz-TeHYneWo/VfTL-txkh7I/AAAAAAAACQ8/oz0BtmgmpSA/s1600/after+drag+and+drop.png)

This is our aim to achieve from bellow given test script.

**Create And Run Test**

Now you already knows what we wants to do In this test. Let's create and run test in eclipse.

**PREREQUISITES** :

1. Android mobile device should be connected with PC with USB debugging mode enabled. View [**THIS POST**](http://www.software-testing-tutorials-automation.com/2015/09/connect-android-device-with-pc-in-usb.html).
2. Drag-Sort Demos app should be there in Apps folder as described above.
3. There should be enough free space in your mobile device to install Drag-Sort Demos app.
4. appium node server should be launched and running mode. View [**THIS POST**](http://www.software-testing-tutorials-automation.com/2015/10/appium-run-first-android-automation.html).
5. TestNG should be installed in eclipse. View [**THIS POST**](http://www.software-testing-tutorials-automation.com/2014/03/steps-of-downloading-and-installing.html).

Now create new class file DragAndDropAction.java under package(for me, package name is Android) of your project and copy paste bellow given test script in it.  
  
**Note**: Please replace deviceName and platformVersion as per you actual in bellow given test.

**DragAndDropAction.java**

package Android;

import io.appium.java\_client.MobileDriver;

import io.appium.java\_client.TouchAction;

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

import java.io.File;

import java.net.MalformedURLException;

import java.net.URL;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.remote.DesiredCapabilities;

import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

public class DragAndDropAction {

**//Object reference of AndroidDriver.**

AndroidDriver driver;

@BeforeTest

public void setUp() throws MalformedURLException {

**//Set Drag-Sort Demos app folder path. This statement will refer project's folder path.**

File classpathRoot = new File(System.getProperty("user.dir"));

**//Set folder name "Apps" where .apk file is stored.**

File appDir = new File(classpathRoot, "**/Apps**");

**//Set Drag-Sort Demos .apk file name.**

File app = new File(appDir, "**com.mobeta.android.demodslv-0.5.0-3\_APKdot.com.apk**");

**// Created object of DesiredCapabilities class.**

DesiredCapabilities capabilities = new DesiredCapabilities();

**// Set android deviceName desired capability. Set your device name.**

capabilities.setCapability("deviceName", "**ZX1B32FFXF**");

**// Set BROWSER\_NAME desired capability. It's Android in our case here.**

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

**// Set android VERSION desired capability. Set your mobile device's OS version.**

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

**// Set android platformName desired capability. It's Android in our case here.**

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

**//Set .apk file's path capabilities.**

capabilities.setCapability("app", app.getAbsolutePath());

**// Set app Package desired capability of Drag-Sort Demos app.**

capabilities.setCapability("appPackage", "**com.mobeta.android.demodslv**");

**// Set app Activity desired capability of Drag-Sort Demos app.**

capabilities.setCapability("appActivity", "**com.mobeta.android.demodslv.Launcher**");

**// Created object of AndroidDriver and set capabilities.**

**// Set appium server address and port number in URL string.**

**// It will launch Drag-Sort Demos app in emulator.**

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

driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);

}

@Test

public void dragDrop() {

**//Tap on Basic usage Playground.**

driver.findElementByName("Basic usage playground").click();

**//Locate 3rd element(Chick Corea) from list to drag.**

WebElement **ele1** = (WebElement) driver.findElementsById("com.mobeta.android.demodslv:id/drag\_handle").get(2);

**//Locate 6th element to drop dragged element.**

WebElement **ele2** = (WebElement) driver.findElementsById("com.mobeta.android.demodslv:id/drag\_handle").get(5);

**//Perform drag and drop operation using TouchAction class.**

**//Created object of TouchAction class.**

TouchAction action = new TouchAction((MobileDriver) driver);

System.out.println("It Is dragging element.");

**//It will hold tap on 3rd element and move to 6th position and then release tap.**

action.longPress(**ele1**).moveTo(**ele2**).release().perform();

System.out.println("Element has been droped at destination successfully.");

}

@AfterTest

public void End() {

**// Quit**

driver.quit();

}

}

Comments are already there with each sentence of test script. Let me describe you few Important class and methods which are used in above test script.

* **TouchAction**: This is class of Webdriver 3 which provide different methods(longPress, moveTo, perform, press, tap, etc..) to automate mobile gestures like drag and drop, swipe etc. It will help us to generate action chain of different actions.
* **longPress**: longPress is method to hold tap for long time on given element.
* **moveTo**: moveTo is method to move action.
* **release**: release is used to release from longPress tap action.
* **perform**: perform will execute full action chain of drag and drop.

Also If you notice in above test script, We have used AndroidDriver at place of RemoteWebDriver as we are automating android app. And also used findElementByName("") and findElementsById("") methods at place of findElement(By.Name("")) and findElements(By.Id("")) methods.

Run above test using testng and observe Drag And Drop in your mobile device.

* It will launch Drag-Sort Demos app.
* Tap on "Basic usage playground" text and
* Perform draga and drop operation as shown in above Images.

This way we can automate mobile gestures in appium android automation test. Same thing you can do with any app.

**72. What is usage TouchAction Class in Appium?**

**Answer**: TouchAction Class is used to generate actions chain to perform drag and drop, swipe element, etc.. in appium test.

**73. How to perform swipe action in appium?**

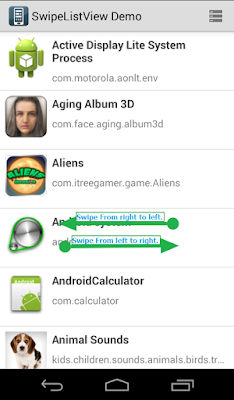
We can use TouchAction class and it's methods to generate action chain of swipe action.

*Install SwipeListView Demo App*

You can view my previous post to learn how to download and install SwipeListView Demo software app in your android mobile device.

*Aim To Achieve In This Appium Test*

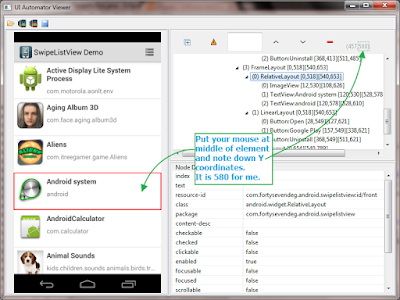
Our aim Is to swipe particular element in horizontal(Right to left and then left to right) direction. As you can see In bellow image, We wants to swipe 4th element from listing grid of SwipeListView Demo app in horizontal(Right to left and then left to right) direction.

[](http://4.bp.blogspot.com/-SW02SRWuz6g/VfzzRK9aCaI/AAAAAAAACR0/zSGla-7VrMY/s1600/swipe+using+Action+class.png)

**Get element's Y axis coordinates**

In order to swipe element in X(horizontal) direction, You need to find element's position in Y direction. To Get It,

* Connect your mobile phone with PC with USB debugging mode enabled. [**VIEW MORE**](http://www.software-testing-tutorials-automation.com/2015/09/connect-android-device-with-pc-in-usb.html).
* Open SwipeListView Demo app In mobile device.
* Open UI Automator Viewer from tools folder of SDK. View [**THIS POST**](http://www.software-testing-tutorials-automation.com/2015/09/locating-android-app-elements-using-ui.html) for more detail on UI Automator Viewer.
* Capture device's screenshot In UI Automator Viewer.
* Put mouse cursor at middle of 4th element.
* Note down Y Coordinates as shown in bellow Image.

[](http://4.bp.blogspot.com/-tVOdoG9m7l0/Vf034dwjFGI/AAAAAAAACSE/VJ3ET2UeNo8/s1600/get+coordinated+using+UIAutomatorviewer.png)

We will use this Y coordinates In our appium test script to perform swipe.

**Create And Run Android Appium Test To Swipe Element**

I have created very simple test script to swipe android app's element In horizontal direction. Create new class file SwipeAction.java under Android package of your project In eclipse and copy paste bellow given test script In It.

**Note** : Please set your device's capabilities in bellow given test.  
**SwipeAction.java**

package Android;

import io.appium.java\_client.MobileDriver;

import io.appium.java\_client.TouchAction;

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

import java.net.URL;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.Dimension;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.remote.DesiredCapabilities;

import org.openqa.selenium.support.ui.ExpectedConditions;

import org.openqa.selenium.support.ui.WebDriverWait;

import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

public class SwipeAction {

AndroidDriver driver;

Dimension size;

WebDriverWait wait;

@BeforeTest

public void setUp() throws Exception {

DesiredCapabilities capabilities = new DesiredCapabilities();

capabilities.setCapability("deviceName", "ZX1B32FFXF");

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

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

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

capabilities.setCapability("appPackage", "com.fortysevendeg.android.swipelistview");

capabilities.setCapability("appActivity","com.fortysevendeg.android.swipelistview.sample.activities.SwipeListViewExampleActivity");

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

driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);

wait = new WebDriverWait(driver, 300);

wait.until(ExpectedConditions.elementToBeClickable(By.className("android.widget.RelativeLayout")));

}

@Test

public void swipingHorizontal() throws InterruptedException {

**//Get the size of screen.**

size = driver.manage().window().getSize();

System.out.println(size);

**//Find swipe x points from screen's with and height.**

**//Find x1 point which is at right side of screen.**

int x1 = (int) (size.width \* 0.20);

**//Find x2 point which is at left side of screen.**

int x2 = (int) (size.width \* 0.80);

**//Create object of TouchAction class.**

TouchAction action = new TouchAction((MobileDriver)driver);

**//Find element to swipe from right to left.**

WebElement ele1 = (WebElement) driver.findElementsById("com.fortysevendeg.android.swipelistview:id/front").get(3);

**//Create swipe action chain and perform horizontal(right to left) swipe.**

**//Here swipe to point x1 Is at left side of screen. So It will swipe element from right to left.**

action.longPress(ele1).moveTo(x1,580).release().perform();

**//Find element to swipe from left to right.**

WebElement ele2 = (WebElement) driver.findElementsById("com.fortysevendeg.android.swipelistview:id/back").get(3);

**//Create swipe action chain and perform horizontal(left to right) swipe.**

**//Here swipe to point x2 Is at right side of screen. So It will swipe element from left to right.**

action.longPress(ele2).moveTo(x2,580).release().perform();

}

@AfterTest

public void End() {

driver.quit();

}

}

**Test Explanation**

* **ele1** Is an element located to swipe from right side to left side.
* **ele2** Is an element located to swipe from left side to right side.
* **longPress()**method will press and hold given element.
* **moveTo(X, Y)**method will move element to given X and Y coordinates.
* **release()**method will remove the current touching implement from the screen.
* **perform()**will perform this chain of actions on the driver.

Now start appium server and run test script In eclipse and observe element swipe operation in your android mobile device. This way you can swipe specific element of android app using action chain.

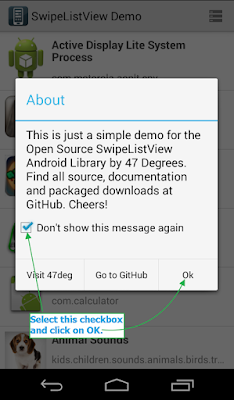
**74. Is there any other way to perform swipe in appium?**

Yes, We can use swipe method to perform swipe action.

**Download And Install SwipeListView Demo App**

We will use SwipeListView Demo App for **android swipe test using appium**. You need to download and install SwipeListView Demo App in your android mobile device.

* You can [**Download it from Google Play Store**](https://play.google.com/store/apps/details?id=com.fortysevendeg.android.swipelistview) or [**THIS PAGE**](http://www.software-testing-tutorials-automation.com/2015/11/test-apps-to-use-in-appium-automation.html).
* Install it in your android mobile device.
* Open SwipeListView Demo App In your mobile device. It will show you alert message on screen.
* Select "**Don't show this message again**" check box and click on OK button as shown in bellow image. Now this message will not display again when we run our test through appium.

[](http://1.bp.blogspot.com/-z43O4y10zvs/Vfd10K47NgI/AAAAAAAACRM/3qd7fxSPLrY/s1600/don't+show+this+message.png)

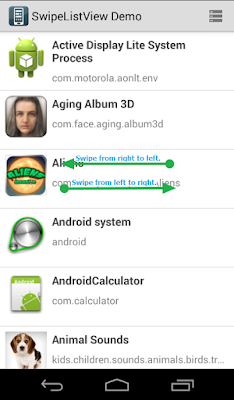
* It will show you list of apps in your android device. Using this app, You can swipe horizontal and vertical.

**Aim To Achieve In This Appium Test**

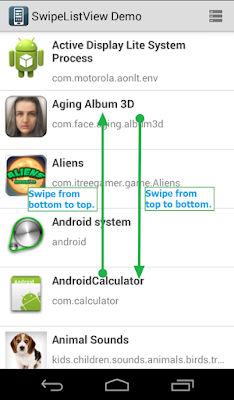
I have a 2 goals to achieve from this post in **appium**. 1. **Horizontal swiping** and 2. **Vertical swiping in android app**.

**1. Horizontal Swiping In Android App**

Using appium, We wants to swipe right to left and left to right horizontally as shown in bellow Image.

[](http://2.bp.blogspot.com/-IAB3Jt9M6F8/Vfd9ZE4SK0I/AAAAAAAACRc/NT24eOEyR6k/s1600/horizontal+swiping.png)

**2. Vertical Swiping In Android App**  
Also we wants to swipe bottom to top and top to bottom vertically as shown in bellow Image.

[](http://4.bp.blogspot.com/-Et5xlsdjqLs/Vfd9jhcvRRI/AAAAAAAACRk/UWLQq2GKouQ/s1600/vertical+swiping.png)

This is our goal to achieve from this post.

**Create And Run Appium Android Test Script To Swipe**

I have prepared simple test to perform swipe on SwipeListView Demo android app as shown bellow. Create new class file driverSwipe.java and paste bellow given test script code in it.

**driverSwipe.java**

package Android;

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

import java.net.URL;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.Dimension;

import org.openqa.selenium.remote.DesiredCapabilities;

import org.openqa.selenium.support.ui.ExpectedConditions;

import org.openqa.selenium.support.ui.WebDriverWait;

import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.Test;

public class driverSwipe {

AndroidDriver driver;

Dimension size;

@BeforeTest

public void setUp() throws Exception {

DesiredCapabilities capabilities = new DesiredCapabilities();

capabilities.setCapability("deviceName", "ZX1B32FFXF");

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

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

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

capabilities.setCapability("appPackage", "com.fortysevendeg.android.swipelistview");

capabilities.setCapability("appActivity","com.fortysevendeg.android.swipelistview.sample.activities.SwipeListViewExampleActivity");

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

driver.manage().timeouts().implicitlyWait(15, TimeUnit.SECONDS);

WebDriverWait wait = new WebDriverWait(driver, 300);

wait.until(ExpectedConditions.elementToBeClickable(By.className("android.widget.RelativeLayout")));

}

@Test

public void swipingHorizontal() throws InterruptedException {

**//Get the size of screen.**

size = driver.manage().window().getSize();

System.out.println(size);

**//Find swipe start and end point from screen's with and height.**

**//Find startx point which is at right side of screen.**

int startx = (int) (size.width \* 0.70);

**//Find endx point which is at left side of screen.**

int endx = (int) (size.width \* 0.30);

**//Find vertical point where you wants to swipe. It is in middle of screen height.**

int starty = size.height / 2;

System.out.println("startx = " + startx + " ,endx = " + endx + " , starty = " + starty);

**//Swipe from Right to Left.**

driver.swipe(startx, starty, endx, starty, 3000);

Thread.sleep(2000);

**//Swipe from Left to Right.**

driver.swipe(endx, starty, startx, starty, 3000);

Thread.sleep(2000);

}

@Test

public void swipingVertical() throws InterruptedException {

**//Get the size of screen.**

size = driver.manage().window().getSize();

System.out.println(size);

**//Find swipe start and end point from screen's with and height.**

**//Find starty point which is at bottom side of screen.**

int starty = (int) (size.height \* 0.80);

**//Find endy point which is at top side of screen.**

int endy = (int) (size.height \* 0.20);

**//Find horizontal point where you wants to swipe. It is in middle of screen width.**

int startx = size.width / 2;

System.out.println("starty = " + starty + " ,endy = " + endy + " , startx = " + startx);

**//Swipe from Bottom to Top.**

driver.swipe(startx, starty, startx, endy, 3000);

Thread.sleep(2000);

**//Swipe from Top to Bottom.**

driver.swipe(startx, endy, startx, starty, 3000);

Thread.sleep(2000);

}

@AfterTest

public void End() {

driver.quit();

}

}

**swipingHorizontal() Method Description**

In above test script, swipingHorizontal() method is responsible for horizontal swipe. Here,

* **driver.manage().window().getSize();** will find your device's screen size(Width X Height).
* **startx** Is located at 70% (From left) of your device's screen width.
* **endx**Is located at 30%  (From left) of your device's screen width.
* **starty**Is located at the vertical middle of the screen.
* First**driver.swipe** method will swipe from right side to left side as swipe start point(startx) is located at right side of the screen and end point(endx) is locate at left side of the screen. Here 3000 Is time in milliseconds to perform swipe operation.
* Second**driver.swipe** method will swipe from left side to right side as swipe start point(endx) is located at left side of the screen and end point(startx) is locate at right side of the screen.
* Vertical point **starty**will remain steady as we are performing horizontal swipe.

**swipingVertical() Method Description**

swipingHorizontal() method is responsible for vertical swipe In above android automation script . Here,

* **starty** Is located at 80% (From top) of your device's screen height.
* **endy**Is located at 20%  (From top) of your device's screen height.
* **startx**Is located at the horizontal middle of the screen.
* First**driver.swipe** method will swipe from bottom to top as swipe start point(starty) is located at bottom side of the screen and end point(endy) is locate at top side of the screen. Here 3000 Is time in milliseconds to perform swipe operation.
* Second**driver.swipe** method will swipe from top side to bottom as swipe start point(endy) is located at top side of the screen and end point(starty) is locate at bottom side of the screen.
* Horizontal point **startx**will remain steady as we are performing horizontal swipe.

What is the difference between Android SDK and Android Studio ?

As any development requires an SDK (*Software Development Kit),*Android SDK is required for developing apps for Android. It contains APIs which you can use or reference to develop your application.

Now being humans we do make mistakes and hence came the IDE (*Integrated Development Environment*).An IDE is basically an enhanced editor which makes development easier. Nowadays you get lot more features in an IDE than just an enhanced editorbut you get the idea right?

In a nutshell you need an Android SDK to develop apps and an IDE to help you with coding, spell checking, warnings/errors, developing UI etc.