* **What is appium?**

Appium is a mobile automation testing tool to test android and ios apps. For both handling native and hybrid mobile apps.

* **What are the platform?**

1. IOS
2. Android
3. Firefox OS

* **What are the languages supported by appium?**

Java, c#, Javascript, Python, Ruby

* **How appium works?**

CODE

FROM

CLIENT

MOBILE

APP

APPIUM

SERVER

Code follows Json wire protocol while sending the client script to the and Appium server receives code as the form of session object or headless browser. The server interpret and execute commands on the mobile app. Appium server work as media.

* **Full Installation:**

1. Java 1.8
2. Android Studio (To setup an android emulator follow this steps: open->create project->sample project->tools->advm->setup device android emulator)
3. Node.js

* Appium comes through node modules.
* Npm->to download all node modules.

1. Download Appium Server

* npm install –g appium (install appium)
* appium –v (to check appium verison)
* appium (to start the server)

1. Download Appium-client jar file
2. Download appium-doctor

install appium-doctor (install appium doctor

appium-doctor (to check all the configuration are okay or not for appium?)

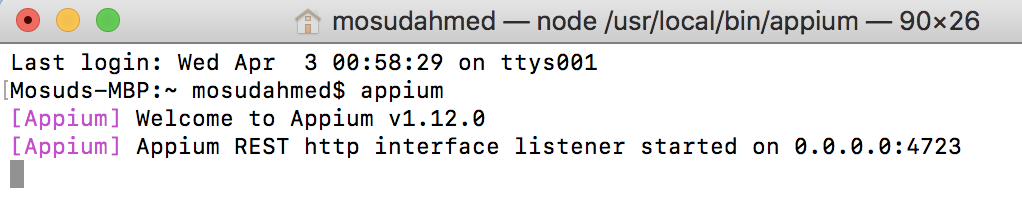
Based on the information setup all the path:

Open the bash\_profile (open .bash\_profile in terminal)-> set all the path based on your locations.

1. Download Eclipse
2. Always start the server and emulator/simulator before running your script.

* **Steps to do before writing and executing code on Emulator**

1. Open Terminal and write appium-doctor to check the configuration is ok.
2. Start Appium Server

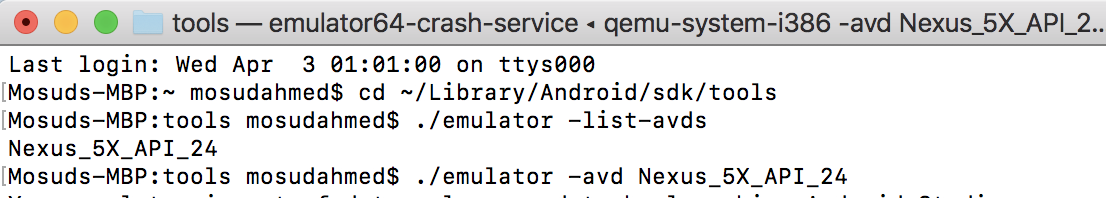


1. Open Emulator (To run your script on Emulator)

*From terminal: cd ~/Library/Android/sdk/tools*

*./emulator –list-avds (to see the available emulator devices)*

*./emulator –avd nameoftheemulatordevice (Copy from top line)*



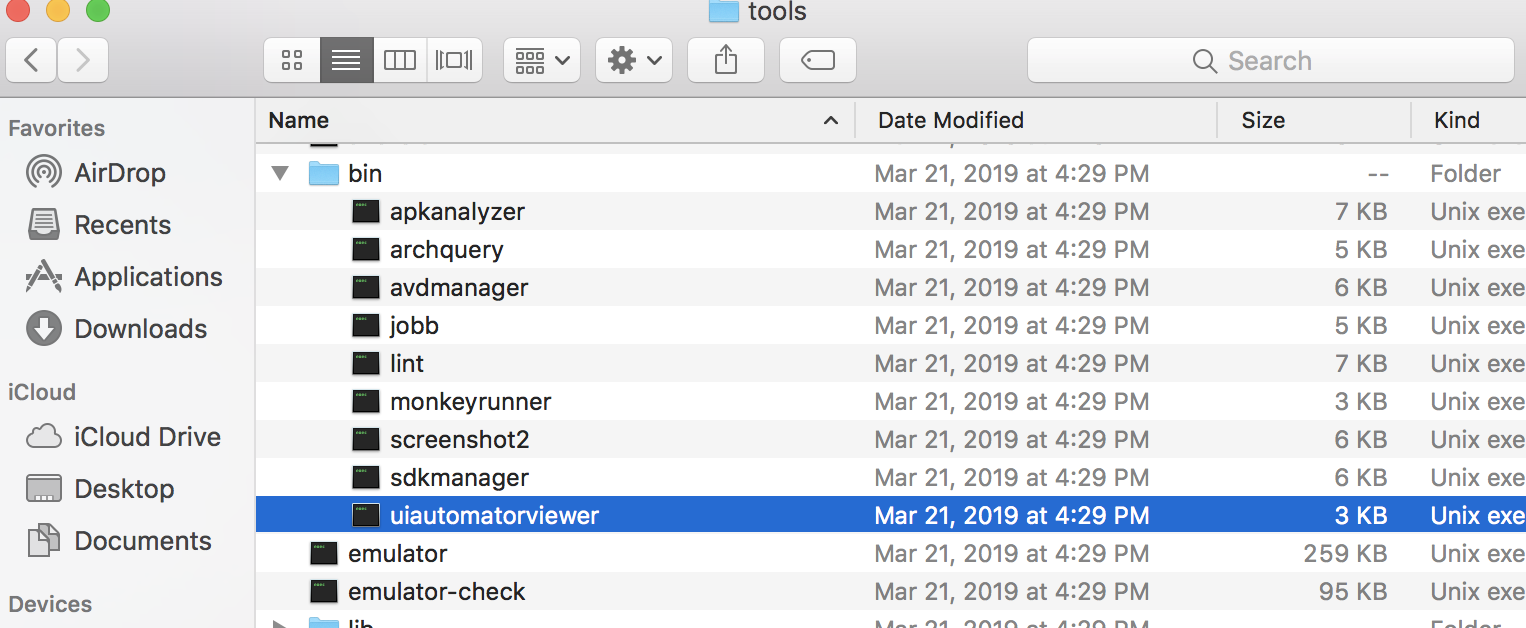
1. Open terminal and start adv server

adv kill-server;/adb kill-server;

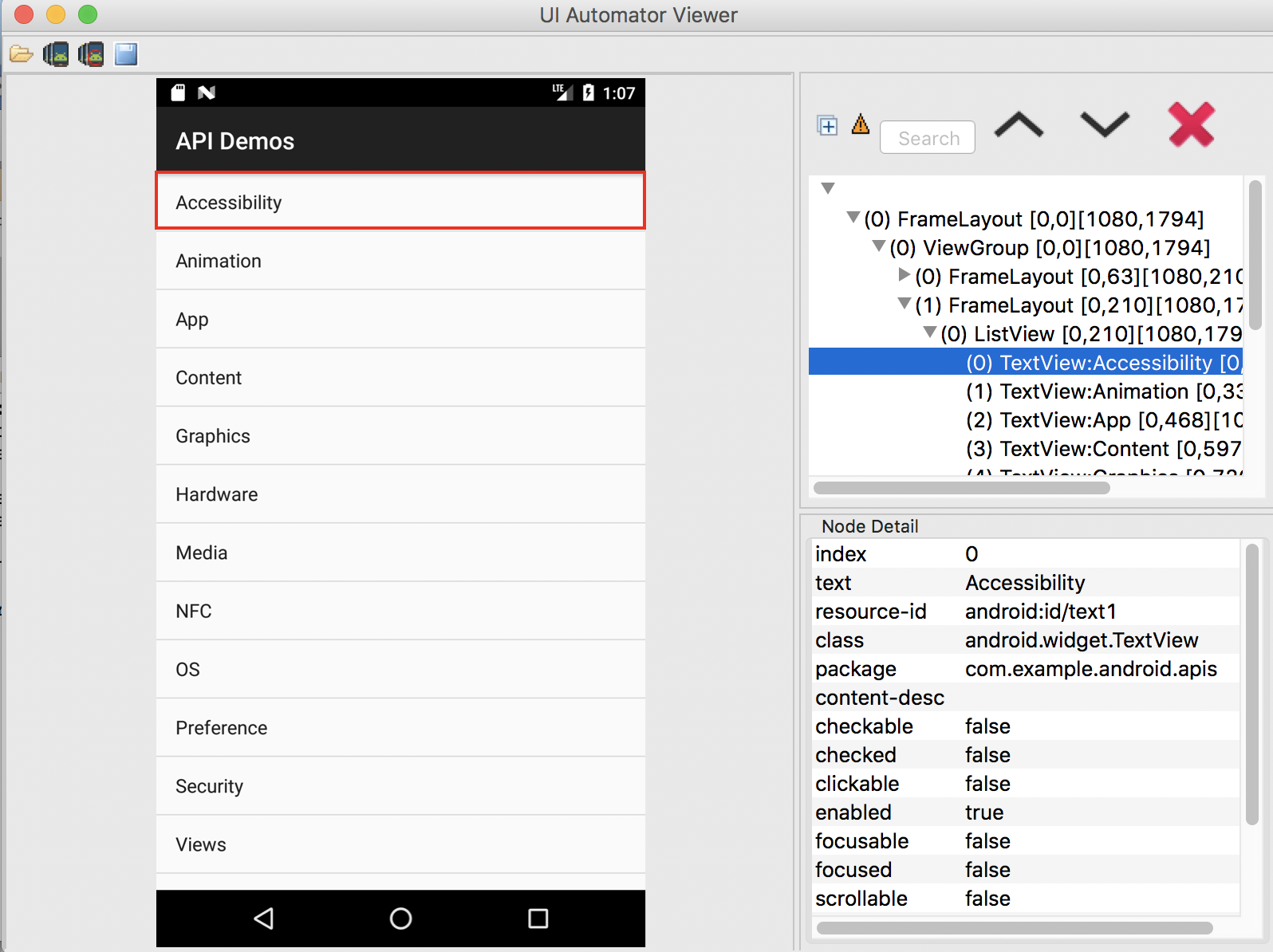
adb start-server/adb start-server;

1. Open UIAutomatorViewer for inspect android element (get locators of mobile app);

Click on Library -> Android -> sdk -> tools -> bin -> click on ui automator to ope the ui screen; Make sure your emulator is open before opening ui automator viewer.



Open the app in the emulator and then click device screenshot just infront of the folder icon to get the same app page on your ui automator viewer. Then move your cursor on the elements to get the icons. If you want the next app page then change another page on the emulator and then click on device screenshot. See below image.



1. Set up your desiredCapabilities configuration and then run as java app;

* **How to open UIAutomatorViewer, get your xpath/id/css locators of your app?**

1. Java 1.8 Must!!!!!!!!!!!!
2. Click on Library -> Android -> SDK -> Tools -> bin -> UI Automator;
3. Click on UI Automator and wait for loading the screen.
4. Automator is used to identify the element from the mobile app.
5. Make sure your emulator is open before opening the ui automator viewer.
6. Click device screenshot ui automator to get and inspect the element.

* **How to invoke an app using .apk and absolute path?**

1. Which emulator/simulator the script to use?
2. What is the app or app info?
3. Connection step to connect to the server.

* **How to stop and restart appium server?**

Ctrl+c to stop the server;

Open terminal type: appium to restart the server.

* **What are the 4 locators android mostly used?**

Xpath/Id/ClassName/Android ui automator;

For Xpath className is considered as tage name;

* **How to use ANDROID UI AUTOMATOR locator?**

***Syntax for attribute:***

driver.findElement(MobileBy.AndroidUIAutomator(“attribute(\”value\”)”)).click;

This is android locator; we used backslash because double quotation inside double quotation.

***Syntax for property:***

driver.findElement(MobileBy.AndroidUIAutomator(“new UiSelector().clickable(true)”)).click;

* **How to invoke an app using .apk and absolute path?**

**public** **static** AndroidDriver<AndroidElement> capabilities() **throws** MalformedURLException {

// Invoke app using .apk and absolute path;

File f=**new** File("src/test/java");

File fs=**new** File(f,"ApiDemos-debug.apk");

DesiredCapabilities cap=**new** DesiredCapabilities();

cap.setCapability(MobileCapabilityType.***PLATFORM\_NAME***, MobilePlatform.***ANDROID***);

cap.setCapability(MobileCapabilityType.***DEVICE\_NAME***, "Nexus 5X API 24");

cap.setCapability(MobileCapabilityType.***NEW\_COMMAND\_TIMEOUT***, "25");

cap.setCapability(MobileCapabilityType.***APP***, fs.getAbsolutePath());

AndroidDriver<AndroidElement> driver=**new** AndroidDriver<>(**new** URL("http://127.0.0.1:4723/wd/hub"),cap);

//System.out.println("server passed");

**return** driver;

}

* **How to invoke an app using package name and activity?**

**public** **static** AndroidDriver<AndroidElement> capabilities() **throws** MalformedURLException {

/\*

\* In this code we invoke our real device app by APP\_PACKAGE and APP\_ACTIVITY;

\* You don't need to use the absolute path or invoke any .apk file in that case;

\*

\* How to get an app package name and activity?

\* Ans: By Installing APK-INFO app which will provide all the package and activity information

\* for all the installed app;

\* \*/

//File f=new File("src/test/java");

DesiredCapabilities cap=**new** DesiredCapabilities();

cap.setCapability(MobileCapabilityType.***PLATFORM\_NAME***, MobilePlatform.***ANDROID***);

cap.setCapability(MobileCapabilityType.***DEVICE\_NAME***, "Nexus 5X API 24");

cap.setCapability(MobileCapabilityType.***NEW\_COMMAND\_TIMEOUT***, "25");

cap.setCapability(AndroidMobileCapabilityType.***APP\_PACKAGE***, "com.whatsapp");

cap.setCapability(AndroidMobileCapabilityType.***APP\_ACTIVITY***, "main.activity");

AndroidDriver<AndroidElement> driver=**new** AndroidDriver<>(**new** URL("http://127.0.0.1:4723/wd/hub"),cap);

//System.out.println("server passed");

**return** driver;

}

* **How to perform Swipe?**

// import this packages first:

//This packagae need to import manually for touchoption();

**import** **static** io.appium.java\_client.touch.TapOptions.*tapOptions*;

// This package needs to be imported manually for element option;

**import** **static** io.appium.java\_client.touch.offset.ElementOption.*element*;

//This package needs to be imported manually for longPressOptions option;

**import** **static** io.appium.java\_client.touch.LongPressOptions.*longPressOptions*;

//This package needs to be imported manually for ofSeconds() option;

**import** **static** java.time.Duration.*ofSeconds*;

LongPress->on element->wait->moveToElement->release->perform;

***t.longPress(longPressOptions().withElement(element(first\_element)).withDuration(ofSeconds(2))).moveTo(element(second\_element)).release().perform();***

* **How to scroll down?**

/\*

\* ScrollDown till the given element;

\* We are using Android codes inside our appium test script;

\* Creating object of UiScrollable class -> UiSelector class - scrollIntoView() method where you have to specify until which

\* element it should scroll down; We are passing properties inside the scrollIntoView method;

\*

\*/

***driver.findElement(MobileBy.AndroidUIAutomator("new UiScrollable(new UiSelector()).scrollIntoView(text(\"WebView\"));"));***

* **How to DragAndDrop?**

TouchAction t=**new** TouchAction(driver);

WebElement source=driver.findElementsByClassName("android.view.View").get(0);

WebElement destination=driver.findElementsByClassName("android.view.View").get(1);

***//t.longPress(longPressOptions().withElement(element(source))).moveTo(element(destination)).release().perform();***

***t.longPress((element(source))).moveTo(element(destination)).release().perform();***

* **Important Appium Methods?**

// To see the current activity of the app;

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

// To see which specific mode/app you are in? Native, Hybrid or WebView?

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

// To see the rotating position of the current app;

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

//driver.isLocked();

// To hide the keyboard if it is open on the screen;

//driver.hideKeyboard();

// To perform Back, Backspace, Home or keyboard specific methods;

driver.findElement(MobileBy.*AndroidUIAutomator*("text(\"Views\")")).click();

***driver.~~pressKeyCode~~(~~AndroidKeyCode~~.~~BACK~~);***

* **How to handle toggle on/off?**

***See github RaagaEnd2EndApp for details.***

Point point= driver.findElementById("com.raaga.android:id/toggle\_notifyrecommend").getLocation();

TouchAction t= **new** TouchAction(driver);

//toggle on

// t.tap(point.x+20, point.y+30).perform(); // Scroll on;

//toggle off

// t.tap(point.x+100, point.y+30).perform(); // Scroll off;

/\* ---------------------------------------------------------

\* Appium has deprecated the t.tap(point.x+20, point.y+30); Instead of this we have to use the below

one option now;

// Using PointOption.point(x, y), which is a static instance of PointOption with those coordinate values

TouchAction touchAction = new TouchAction(driver);

touchAction.tap(PointOption.point(1280, 1013)).perform();

//Using PointOption().withCoordinates(x, y), which returns a reference to the PointOption instance after setting those coordinate values

TouchAction touchAction = new TouchAction(driver);

touchAction.tap(new PointOption().withCoordinates(1280, 1013)).perform();

**Configuring Real (Android) Phone/Device to system for running Appium Tests**

<https://developers.google.com/web/tools/chrome-devtools/remote-debugging/>

* **How to enable developer options in your android phone?**

*Setting->About phone/device->Build Number (Tap 7 times to enable Developer Options)->Back to setting->Developer Options.*

* **How to active USB debugging on your android phone?**

*Developer Options->USB Debugging (active)->verify apps over USB(active)->Ok;*

* **How to check the real active phone in your computer?**

Terminal->

*adb devices (To see all the active real android devices)*

*if you already see the adb devices you do not need to write below two steps. If you don’t see the adb devices try below two steps:*

1. *Restart USB debugging on phone and restart terminal.*
2. *Restart the phone/Reconnect the cable and then open terminal and type adb devices. If its not working then try below two commands;*
3. *Adb kill-server; (kill the server)*

*adb start-server; (start the server)*

*if you see unauthorized then follow the above two steps;*

Note: If you setup everything properly and you still don’t see the device on terminal then yo have the USB drivers.

* **How to run your test in real device?**

Just change the device name to Android Device and connect your android phone through usb cable. Also for terminal device checking go to the platform-tools by using cd and then type adv devices to see how many devices are connected. You can also close the emulator. If you face any kind of errors then use this two commands:

***cap.setCapability(MobileCapabilityType.DEVICE\_NAME, "Android Device");***

* **What is the challenge in Appium Mobile Automation for real device?**

When you write your script based on emulator screen and execute in emulator it usually shows more contents on the screen as the emulator screen is bigger than real device. But when you want to run the same script in real device you may need to scroll for the same content as real device screen size is smaller than emulator. In that case you always have to check and modify your script.

**Mobile Browser Configuration Setup for Android Device**

* **How to configure the base configuration?**

**In the main config class:**

**public** **class** BaseConfigAndroidRealDevice {

**public** **static** AndroidDriver<AndroidElement> capabilities() **throws** MalformedURLException {

/\*

\* In this program we will execute our test script thorugh mobile browser; For that first i have to configure

\* Browser Name in the config page; Then i have establish connection between mobile browser and system browser;

\*

\* Step: go to chrome browser->more tools->New->Developers tools->remote devices (click on 3 dots)->

\* check if any remote devices are connected->click on the device name (nexys)->

\*

\* \*/

AndroidDriver<AndroidElement> driver;

DesiredCapabilities cap=**new** DesiredCapabilities();

cap.setCapability(MobileCapabilityType.***DEVICE\_NAME***, "Android device");

cap.setCapability(MobileCapabilityType.***BROWSER\_NAME***, "Chrome");

driver=**new** AndroidDriver<>(**new** URL("http://127.0.0.1:4723/wd/hub"),cap);

//System.out.println("server passed");

**return** driver;

}

Create another class:

**public** **class** FacebookLoginMobileBrowser **extends** BaseConfigAndroidRealDevice{

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

AndroidDriver<AndroidElement> driver = *capabilities*();

driver.get("http://www.facebook.com");

}

}

Note: If your mobile browser is not running the after this code then please check and update the chrome version in your android device;

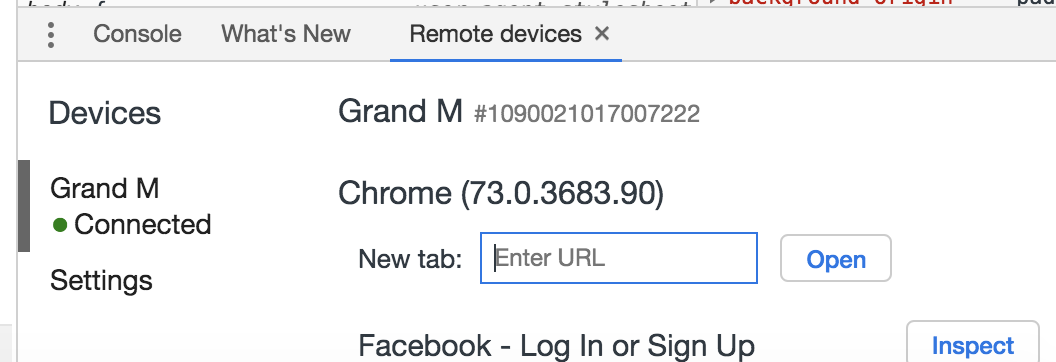
* **How to connect mac chrome browser to your android chrome browser?**

Chrome->new->more tools->developer tools->remote devices->click on device name(Grand M)

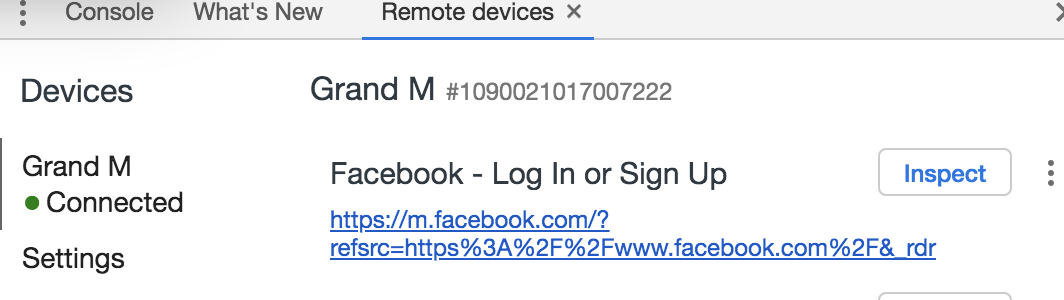
Then open your mobile chrome browser->click device name->new tab and type url;

When you will hit enter then on your mac and android phone both will open the url;

Please update the chrome to above 65 version;



If you want the same url to run on your mac then after giving the url on new tab you will see the link below, just click on it.



* **How to scroll down in mobile browser? Follow below steps:**

JavascriptExecutor js = (JavascriptExecutor)driver;

js.executeScript("window.scrollBy(0,250)", ""); //250 medium //480 highest'

* **How to validate a logo/text is header content or not?**

Driver.findELementByXpath(“”).getAttribute(“class”).contains(“header”);

* **How to check if an object is present in the page or not?**

We wil get the web element and then perform size() which will count how many of them are present on the page. We will run a for loop and set our condition

If(count>0)

{

then it will execute the statement;

break; when condion matches;

we wont use isDisplayed as it is just mention thhe object is visible or not?

**Hybrid App Features and Ways Testing**

**What is Hybrid App?**

A **hybrid** application (**hybrid app**) is one that combines elements of both native and Web applications.

**What is new\_command\_timeout?**

**newCommandTimeout**, How long (in seconds) Appium will wait for a new command from the client before assuming the client quit and ending the session, e.g.

**How to test hybrid app?**

We have to follow two main steps for testing Hybrid Apps:-

1. Ask developer to enable WebView debugging, call the static method setWebContentsDebuggingEnabled on the WebView class. Otherwise, you may see the switching is not working;

1. You need to switch to native app for working on native app object and also switch to web view app for working on web view of the app. You can first use getContext() to see which context view you are working in and then you can use getContextHandles() to get all the context views of this current app. Next we will run for-each loop to get all the context views names or you can ask your developer to give you the names. After iterating you will get nameslike this:

* NATIVE APP AND
* WEBVIEW\_com.example.testapp;

The next thing you have to do is switching to the context view by their name. See example below:

driver.context("WEBVIEW\_com.example.testapp");

// then you may start doing test for web view/ of that apps.