* **Appium IOS Setup**
* Build the IOS App in local machine for practice.
* Download Appium Desktop
* Download Eclipse and Install selenium and appium jars
* Creating Desired capabilities for IOS
* Starting session with Appium Desktop to Inspect Objects
* Understand how Appium IOS Inspector works

1. **Build the IOS App in local machine for practice:**

For real time project developer will give you the Github link for the app. You have to download the zip folder then extract it. Then go to the folder and click on xcode (example: UI Catalog Xcode) and open it. If you don’t’ have Xocde then download from app store.

In this project for testing purpose go to browser and then search UI CATALOG APP GITHUB DOWNLOAD to tdownload the app. Go to this url : <https://github.com/appium/ios-uicatalog>

and download the zip file.

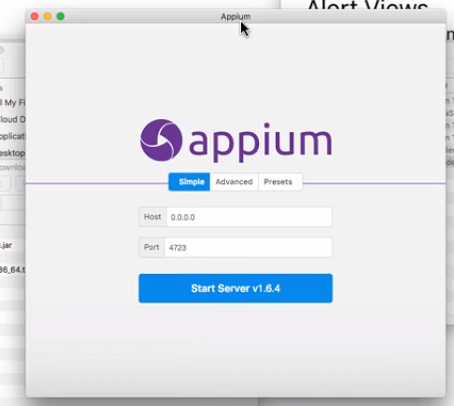
In Xcode the simulator devices are already available, you just have to select it. But in Android sdk you have to create your own emulator and then open it from terminal.

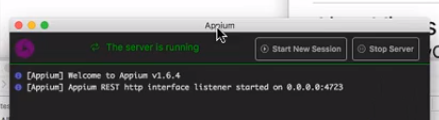
Then run the app buy clicking the play button from top menu then it willbuild and the IOS simulator will be opened automatically. When the build will be done then go to the products folder where you will see the real ios app. IOS app are .app but Android apps are .apk format.

1. **Download Appium Desktop**

Browser->search Download Appium Desktop-> <http://appium.io/downloads.html>-> [Appium-Desktop for OSX, Windows and Linux](https://github.com/appium/appium-desktop/releases/latest)-> then download the mac version.

Then start the server.





* **Pre-requsites:**

Open the Appium Sever->Open the inspect element screen for Appium->Write the code->Run

* **How to configure ios DesiredCapabilities configuration in eclipse?**

**STEP: 1**

**Note**: To get the path of the IOS app open xcode and then right click on the app file and then click on show in finder and then copy the path.

**public** **class** iOSBaseConfig {

**public** **static** IOSDriver<IOSElement> ios\_base\_Config() **throws** MalformedURLException {

IOSDriver<IOSElement>driver;

DesiredCapabilities d=**new** DesiredCapabilities();

d.setCapability(MobileCapabilityType.***DEVICE\_NAME***, "iPhone 6"); // d.setCapability(String, Objectvalue);

d.setCapability(MobileCapabilityType.***PLATFORM\_NAME***, "iOS");

d.setCapability(MobileCapabilityType.***PLATFORM\_VERSION***, "12.1");

d.setCapability(MobileCapabilityType.***APP***, "/Users/mosudahmed/Desktop/UICatalog.app");

/\*-----Found error while running the below code – from OS 10 IOS works with scui test which need to add must-------\*/

d.setCapability(MobileCapabilityType.***AUTOMATION\_NAME***, AutomationName.***IOS\_XCUI\_TEST***);



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

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

**return** driver;

}

}

/\* -------------- standard format for iOS setup---------------

DesiredCapabilities capabilities = new DesiredCapabilities();

capabilities.setCapability(MobileCapabilityType.PLATFORM\_NAME, "iOS");

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

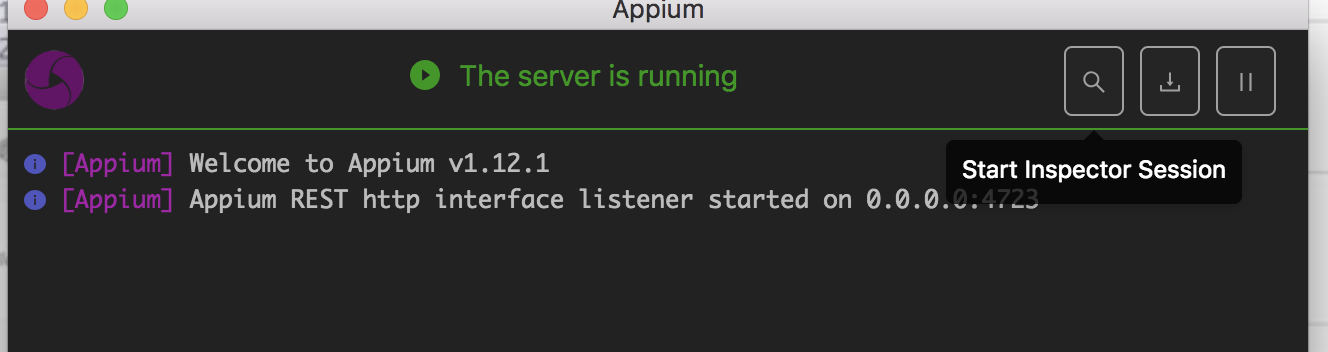
capabilities.setCapability(MobileCapabilityType.DEVICE\_NAME, "iPhone Simulator");

capabilities.setCapability(MobileCapabilityType.APP, myApp);

---------------------------------------------------------------------------------

**STEP: 2**

Open Appium->Click on start inspector session->then configure the session based on your desired capabilities.

****

**STEP: 3**

In the below window you have to pass the following information->then click on start session:

//browserName: "safari",

**platformName: "iOS",**

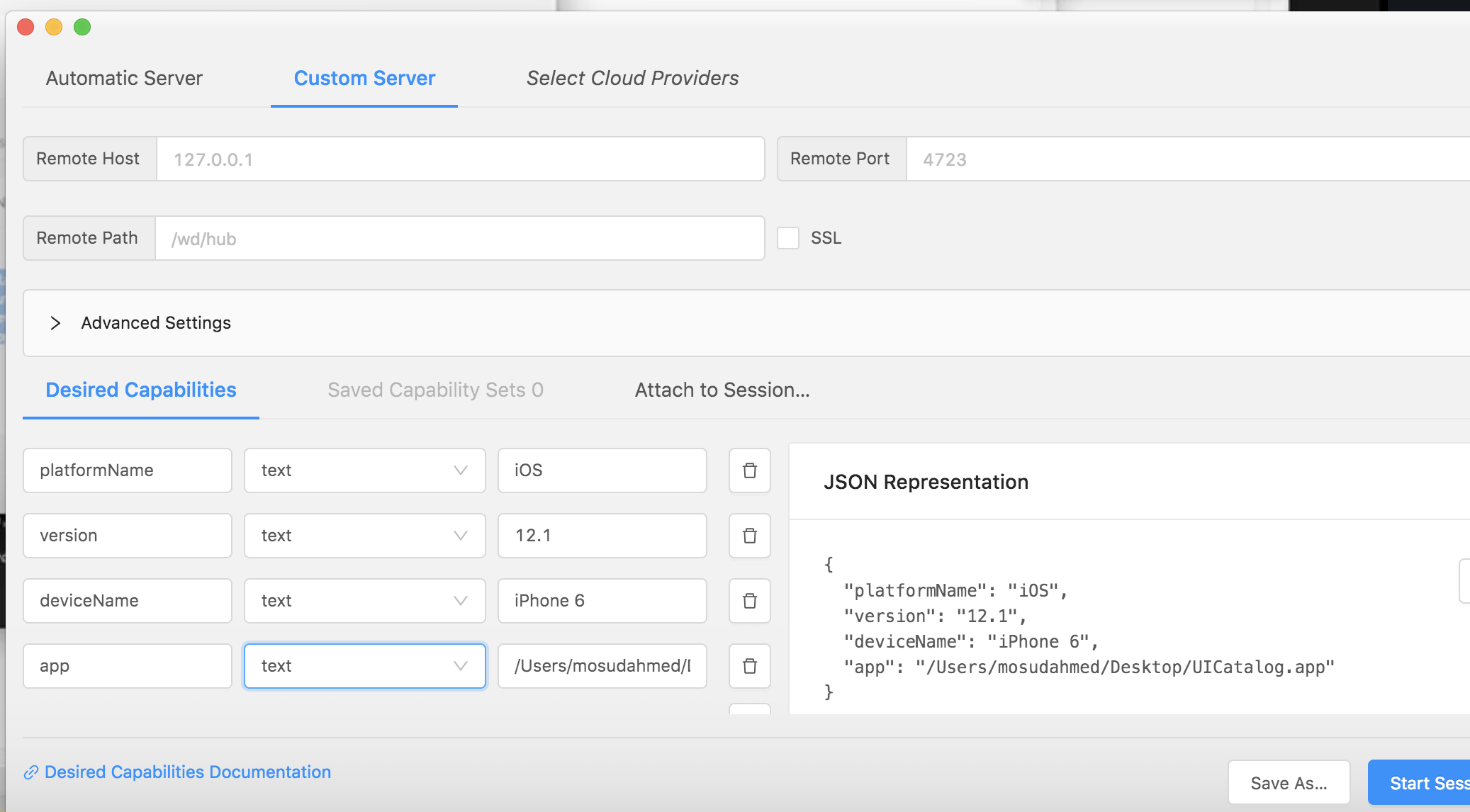
**platformVersion: "10.2",**

**deviceName: "iPhone 6",** (you can get the device name when you open the app in xcode)

**app:path of the app** (when chose click on dropdown then change it to file then go back to text);

//automationName: "XCUITest",

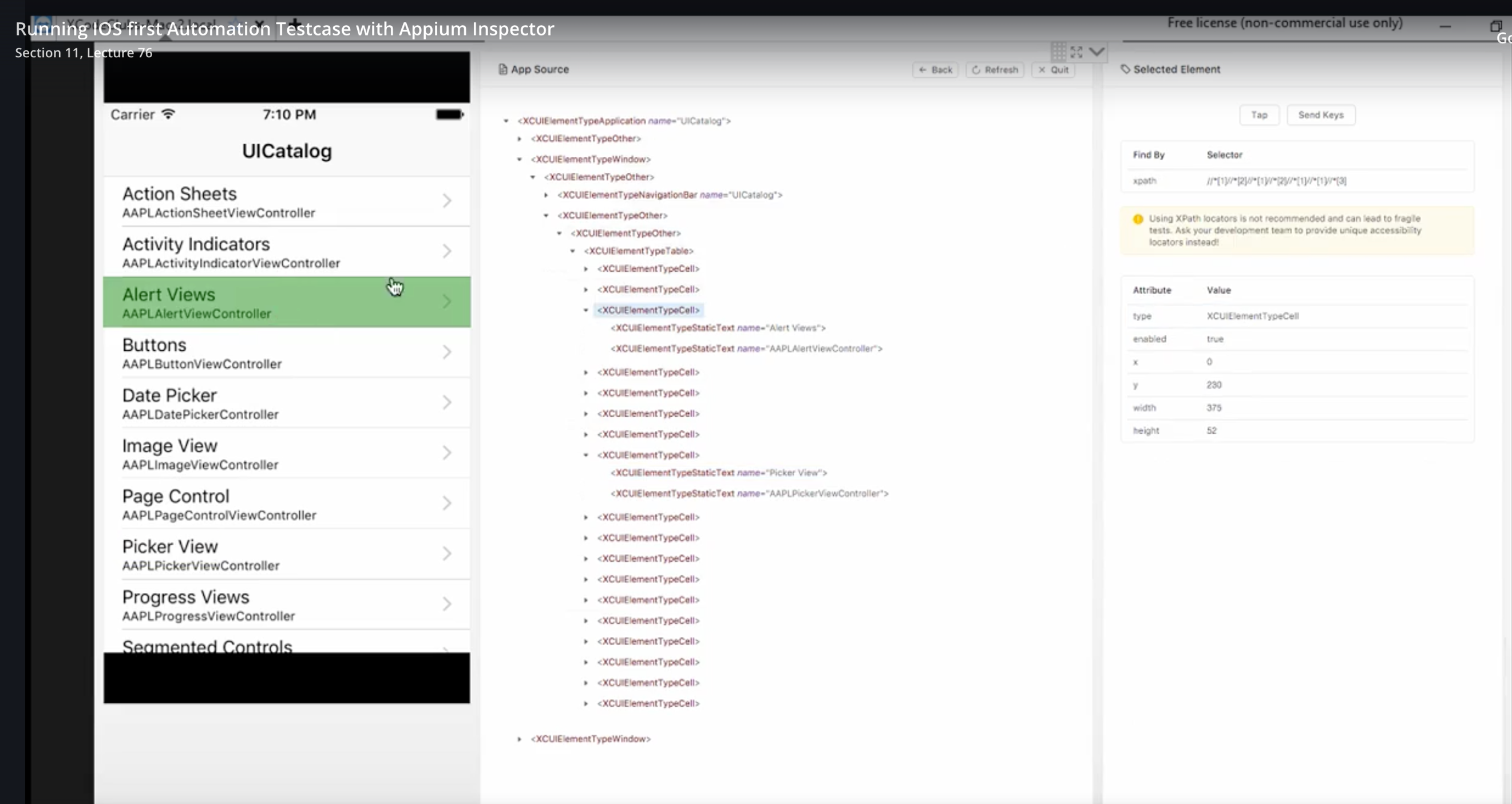
// autoDismissAlerts: "true"

****

Once you click on the start new session based on the app the simulator will open automatically and invoke the app. Why you have to setup the config and invoke app from Appium? Because you have to inspect element from here using Appium.

**STEP: 4**

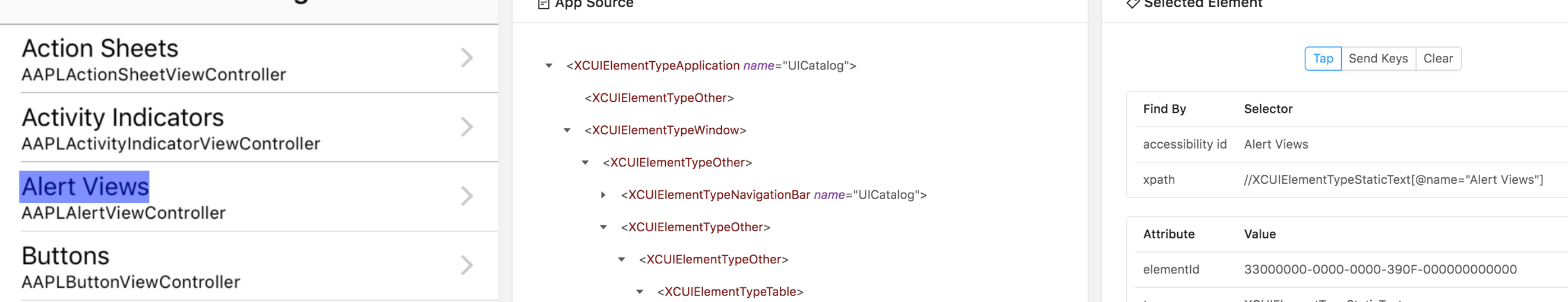
Once the app is successfully invoked from Appium simulator then you can see the inspect element screen of the app in your Appium server ui; on the right side you can see the attribute name and value. The most powerful locator in iOS is Accessibility\_id. To go with xpath you can directly enter the class name;



**STEP: 5**

If you want to click on button and then you want to see the next step then see the below image. Click on Alert Views and the click **TAP** icon on the right top corner. The highlighted part we can consider as class or tag name in the xpath.

1. **Go back previous page:** click n go back button.
2. **Move on next:** select the link and click on tap button.

****

**STEP: 6**

Then you can write the code in eclipse. For example like this:

**public** **class** IosBasics **extends** iOSBaseConfig{

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

IOSDriver<IOSElement>driver=*ios\_base\_Config*();

driver.findElementByAccessibilityId("Alert Views").click(); // Click on alert using powerful findElementByAccessibilityId;

driver.findElementByXPath("//\*[@name='Text Entry']").click(); // Click on text using xpath;

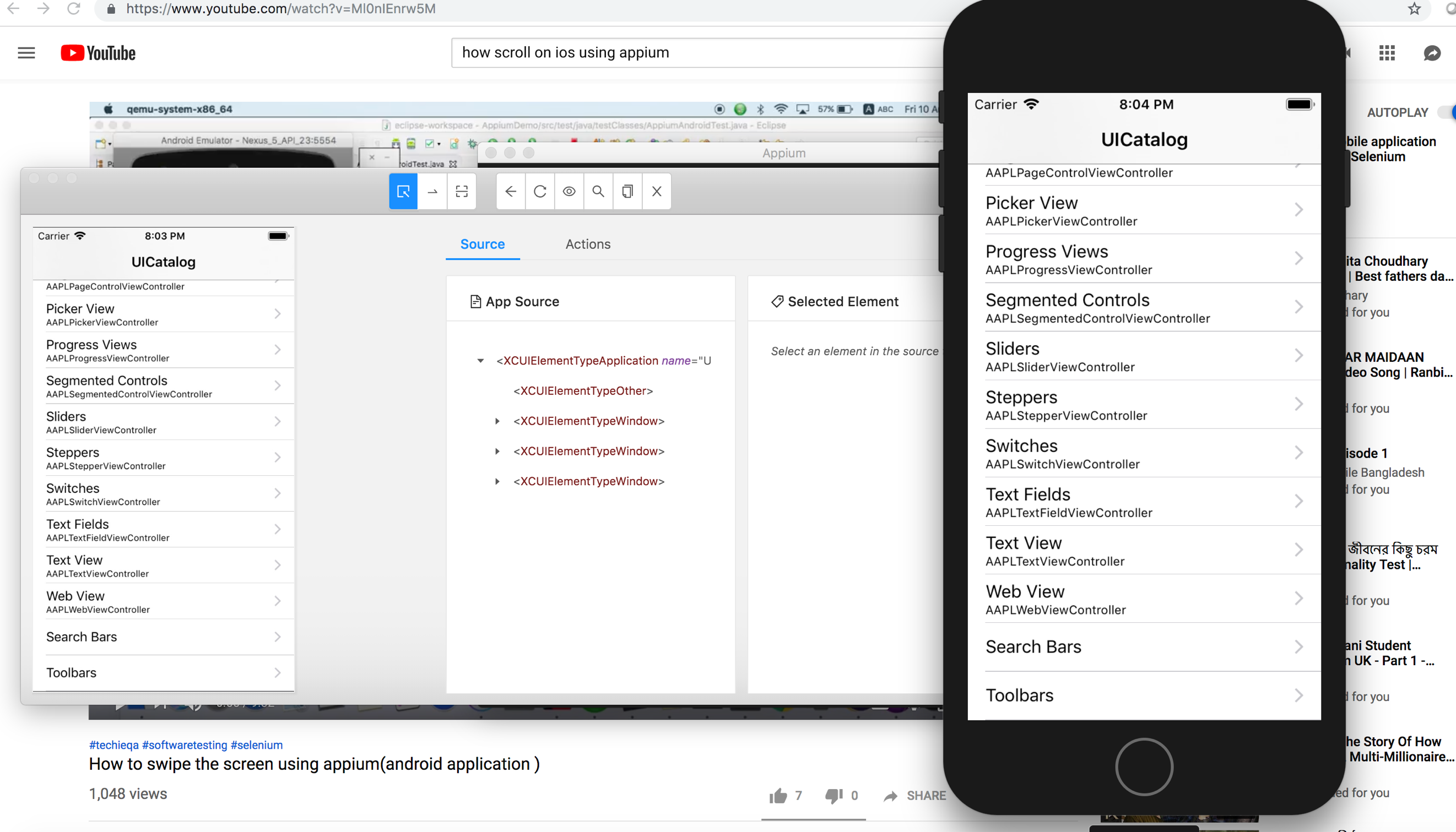
driver.findElementByClassName("XCUIElementTypeOther").sendKeys("hello"); //using class name;

driver.findElementByName("OK").click(); // using name locator;

}

* **How to scroll Ios app on your appium ui screen to inspect element?**

First you have to click and hold till go to bottom of the iOS Simulator. Then on the appium screen you need to refresh. After that you will see the same copy on the appium screen just like your simulator. See images below for details:



Appium screen IOS Simulator

* **How will you scroll in Android and ios?**

/\*

\* In this Program we tried scroll down and then click the Toolsbar button

\* and then navigate back; Some of the appium codes are not working in this new appium version;

\* That's why i used Javascript executor to scroll down;

\*

\* \*/

**public** **class** IosScrollDownOnApp **extends** IOSBaseConfig{

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

IOSDriver<IOSElement> driver=*ios\_base\_Config*();

/\*----------------Scroll Down-----------------------\*/

JavascriptExecutor js = (JavascriptExecutor) driver;

HashMap<String, String> scrollObject = **new** HashMap<String, String>(); // Takes key valu pair;

scrollObject.put("direction", "down"); // put() is used to store data in HashMap;

js.executeScript("mobile: scroll", scrollObject);

* **How to inspect element for Safari Browser web apps?**

You have to use user agent and configure for the same browser in your mac.

* **How to work on iPhone Safari Browser?**

\*

\* Safari Browser Automation;

\* Setup the configuration and then write normal website code;

\* To inspect element got your browser->install user agent->

\* Open user agent->select iPhone->select browser->select operating system;

\* Then inspect element from your mac browser and write your script.

\* Finally run the program;

\*/

**public** **class** SafariBrowserTesting{

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

IOSDriver<IOSElement>driver;

DesiredCapabilities d=**new** DesiredCapabilities();

d.setCapability(MobileCapabilityType.***DEVICE\_NAME***, "iPhone 6"); // d.setCapability(String, Objectvalue);

d.setCapability(MobileCapabilityType.***PLATFORM\_NAME***, "iOS");

d.setCapability(MobileCapabilityType.***PLATFORM\_VERSION***, "12.1");

d.setCapability(MobileCapabilityType.***BROWSER\_NAME***, "safari");

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

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

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

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

driver.findElementByName("email").sendKeys("mosud");

driver.findElementByName("pass").sendKeys("1234567");

}

**Cloud Base Testing on Sauce Lab**

* **What is cloud testing?**

**Cloud testing** is a form of software **testing** in which web applications use **cloud** computing environments (a "**cloud**") to simulate real-world user traffic.

We used saucelab cloud to test our script;

* **How to test your native app on Cloud?**

For testing native on cloud we have to send the app to our sauce lab account by curl language. Curl is a language who sends HTTP request to the particular server server or end point. We can send our native app information through curl to Sauce Lab.

Download curl to your system. By the help of curl you can deploy your app into your sauce lab account and this will be installed on your remote iphone device.

See cloud base testing on github for details.

* **How to open simulator in IOS?**

Download the testing app and then open xcode. Select the device from the list and then click on the play button which will open the simulator for you. For Android you have to create the simulator device but for IOS you have to chose the device from the list. You do not have to create one.

