15 steps Installation Guide download

Section 2, Lecture 11

15 Step by Step Instructions to Configure and run Mobile Automation Testcases using Appium

* Download Java and set Java\_Home in environmental variables
* Download Android STUDIO from below link
* <https://developer.android.com/studio/index.html>
* 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 if it is working.
* Download Node.js
* <https://nodejs.org/en/download/>
* Set Node\_Home Environmental variables path
* Set npm Environmental variables path
* Download Appium Server from Node
* Download Appium and Selenium Java client library
* Install Eclipse – Create a Project in Eclipse - configure Appium libraries
* Start Appium Server-

//Client  - Appium java  client

//Server  - Appium Server

ALL SET GOOOOO

//You can download any node module only using npm

1.1

npm install – g appium –  Latest Stable release version 1.6.9 – Android 7 version

npm uninstall -g appium

//npm install -g appium@beta --no-shrinkwrap

Appium – starts the server

rahulonlinetutor@gmail.com

* -2.1 – Android 11

appium

**Android:**

**Appium: (extended version of webdriver with mobile features) –** webdriver plugin is already there in it

It is open source and cross platform test automation tool for

* **Native** (only for mobile apps)
* **Hybrid** (mobile and web)
* **Mobile** **web apps** (example – chrome)

Tested on **simulators** (iOS, FirefoxOS)

**Emulators** (Android)

**Real devices** (iOS, Andriod, FirefoxOS)

**Why Appium?**

1.As soon as we take it from developer we can test it without recompiling and making Appium compatability.

2.We can use any testing framework – junit, testNG (independent of code and framework)

3.It supports all the languages supported by webdriver

**Important:**

Before for **Apple** they use UIAutomation library

**Andriod** -UiAutomator library

So, by using Appium we can test both regardless of platform and language.

**Challenging part:** It is the set up

**Dom elements or xpath in mobile application:**

**UIAutomatorViewer** – to find the dom element for android application

**Adb** **commands** are responsible for executing Emulator commands -> if it works then only we can open emulator

**Design of Appium:**

It is a HTTP server written in Node.js platform and drives Android and Ios session using JSON protocol. So , before initializing Appium we should have node.js

**Node.js –** in this npm is a command line installer andwith the help of this we can download any node modules (like Appium, protractor, angularjs)

**Limitations of Appium:**

We cannot use inspector in Microsoft windows

Doesn’t support image comparision

Since it is an open source, so difficult to find answers

It doesn’t support the android version <4.1

**Components and internal structure:**

We can test normally by plugin using any android device but if you want to run it in windows (laptop) we will be having a virtual machine that is **mobile Emulator** (like browser)for android in **Android SDK** that will come along with **Android studio** It is just like running in your android device which supports android platform.

Now, we can configure Emulator with android studio. After that we need Appium to test the android application.

We should have 3 things – **Appium jar files**, virtual machine (**emulator** for testing) and Editor (**Eclipse** to write the test cases),

**Step -1:**

Download java and set the path - > download android studio and install -> now android will not come to program files as java we have to find out where it goes

**Step-2:**

**Path setting of android:**

It comes like 2 parts – android **SDK** and android studio

Now go to windows c ->users -> sowmy -> here we cannot find Appdata so we can search by writing **C:\Users\sowmy\AppData - >** now open **local** ->open **Android** ->open sdk **->** copy the path and go to advance settings -> environmental variables - > then create new system variable and then give name and paste the above copied path.

Now, again go to **C:\Users\sowmy\AppData - >** now open **local** ->open **Android** ->open sdk ->

open **lib** ->copy path and go to advance settings -> environmental variables -> then double click on path then click on new -> paste the path

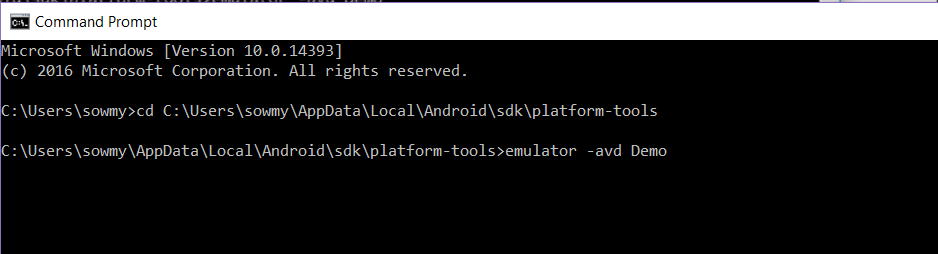
**optional**

Now again go to **C:\Users\sowmy\AppData - >** now open **local** ->open **Android** ->open sdk ->

open **platform tools** ->copy path and go to advance settings -> environmental variables -> then double click on path then click on new -> paste the path

now open windows c -> program files -> open android ->android studio -> open studio64.exe ->

now create project ->go to tools ->android -> avd manager -> choose version and model of the device.



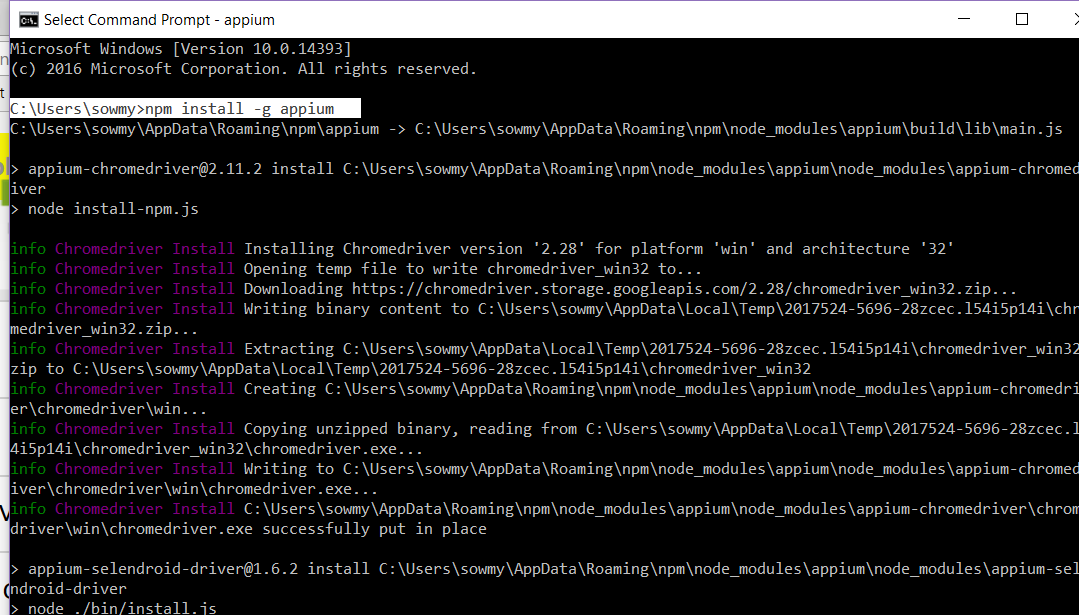
Here in Appium there are 2 things Appium client and Appium Server

First Eclipse (Appium client ) will trigger Appium Server ->then our test case will run in the mobile device which is emulator

Appium Server invocation is very important and make sure your emulator is working

Now go and download node.js -> and set the path in environmental variables just like java and android studio -> here it will come with npm (node package manager)

With the help of npm we can download Appium server



Now, we have to download Appium client for that -> we have to download jar files of Appium

**Implementation:**

Create a project and import selenium and Appium jar files to make the project strong because sometimes css and xpath in web is robust.

**Important things:**

1. **Appium server should start – first type Appium in command prompt and see whether it is started or not**
2. **Your Emulator should be in opened state**
3. **Set the path of the app**

Now, first Appium server should start for that there is one thing called

“DesiredCapabilities” – this will taken info in json structure and gives to server

we have to create object for this

before:

DesiredCapabilities cap = new DesiredCapabilities ;

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

cap.setCapability(MobileCapabilityType.***ORIENTATION***, "landscape");

cap.setCapability(MobileCapabilityType.***APPIUM\_VERSION***, "1.6.5");

After:

**Step-1: setting the path**

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

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

// File fs = new File(f,"path of the app");

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

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

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

}

Giving connection to server link

AndroidDriver driver = new AndroidDriver(connectiontoserverlink,cap);

* **The above line of code is helpful in giving connection to the server.**
* Here the 2 parameters are **“connectiontoserverlink**” and “**cap**”
* In **“cap”** all the capabilities will be stored and as soon as we give the connection it will hit the emulator and invoke the device.

Connectiontoserverlink = <http://127.0.0.1:4723> (local ip address) here **127.0.0.1** is common for all windows based systems

**Code:**

import java.io.File;

import java.net.MalformedURLException;

import java.net.URL;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.remote.DesiredCapabilities;

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

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

import io.appium.java\_client.remote.MobileCapabilityType;

public class base {

public static AndroidDriver<AndroidElement> capabilities() throws MalformedURLException

{

AndroidDriver<AndroidElement> driver;

// TODO Auto-generated method stub

File appDir = new File("src");

File app = new File(appDir, "ApiDemos-debug.apk");

DesiredCapabilities capabilities = new DesiredCapabilities();

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

capabilities.setCapability(MobileCapabilityType.APP, app.getAbsolutePath());

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

return driver;

}

}

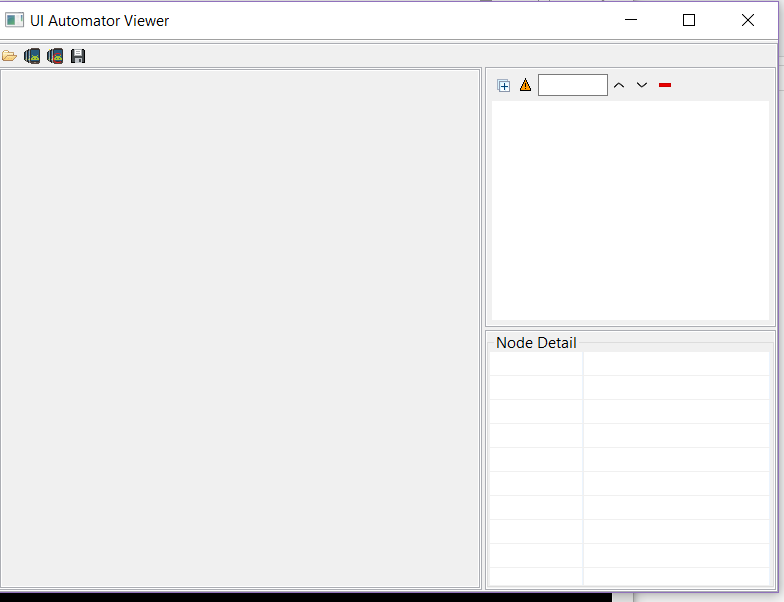
**UIAutomatorViewer:**

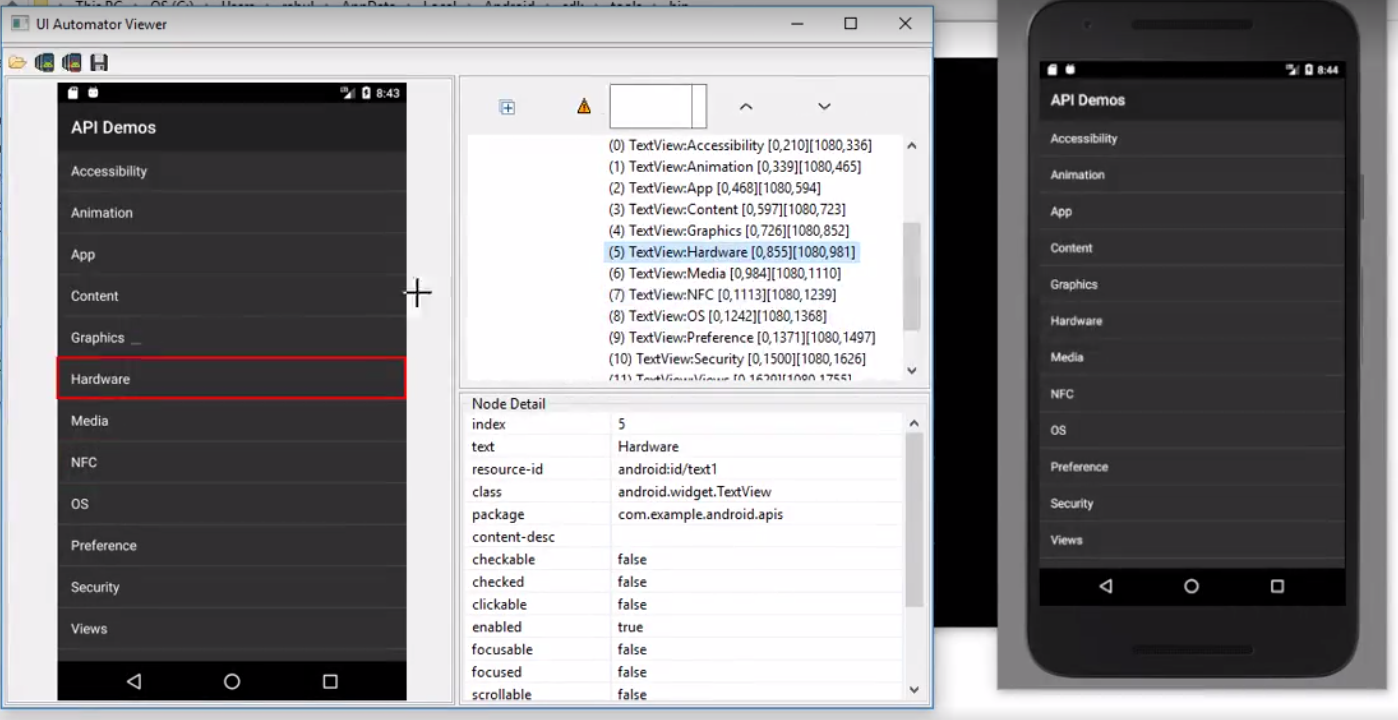
it is used to locate dom (document object model) – a tree of objects

go to **C:\Users\sowmy\AppData - >** now open **local** ->open **Android** ->

open sdk -> open tools -> then open bin -> open “uiautomatorviewer”

now a big window will open and in that if you click on second icon(device screenshot) then it will take screenshot of the emulator -> it will all properties associated with the objects



****

**Code:**

@Test

public void test() throws MalformedURLException {

File appDir = new File("src");

File app = new File(appDir, "bookMyShow-ucb.apk");

DesiredCapabilities cap=new DesiredCapabilities();

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

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

cap.setCapability(MobileCapabilityType.NEW\_COMMAND\_TIMEOUT, "100");

cap.setCapability(MobileCapabilityType.APP, app.getAbsolutePath());

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

//driver.findElementById("com.bt.bms:id/btnLogin").click();

driver.findElementById("com.bt.bms:id/btnSignUp").click();

int s=driver.findElements(By.className("android.widget.EditText")).size();

System.out.println(s);

List<WebElement>a=driver.findElements(By.className("android.widget.EditText"));

a.get(0).sendKeys("Appium");

a.get(1).sendKeys("Mobile");

a.get(2).sendKeys("appium@training");

a.get(3).sendKeys("password");

driver.findElementById("com.bt.bms:id/action\_icon").click();

IOS:

Steps:

Install xcode

Checking IOS simulators

Install java and eclipse

Download selenium/Appium jar files

Xcode- it is like an IDE which have set of softwares which support apple softwares.

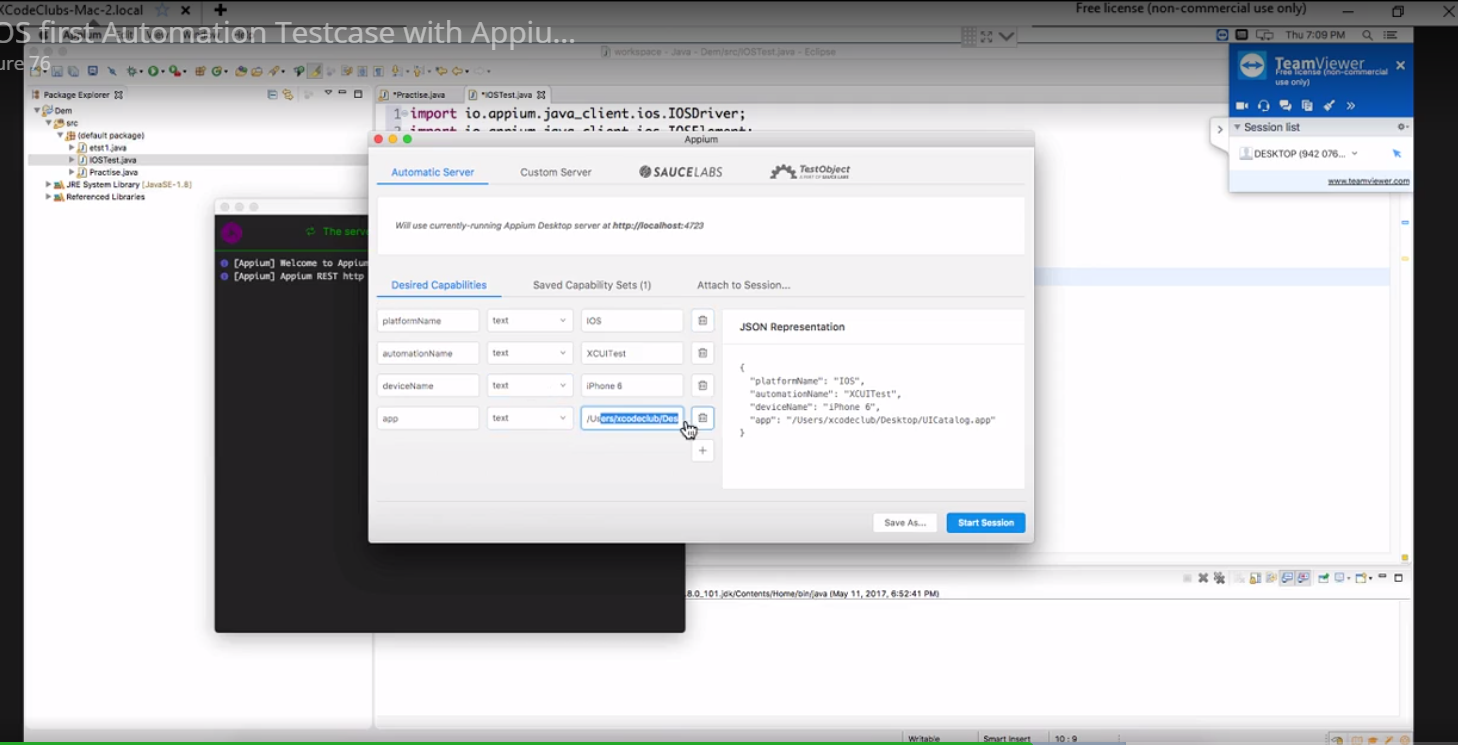
We should install xcode so that we will get an iOS simulator where it acts like a virtual machine (where we can choose our phone type like iphone 5/6/7 or iphone pro or iphone air

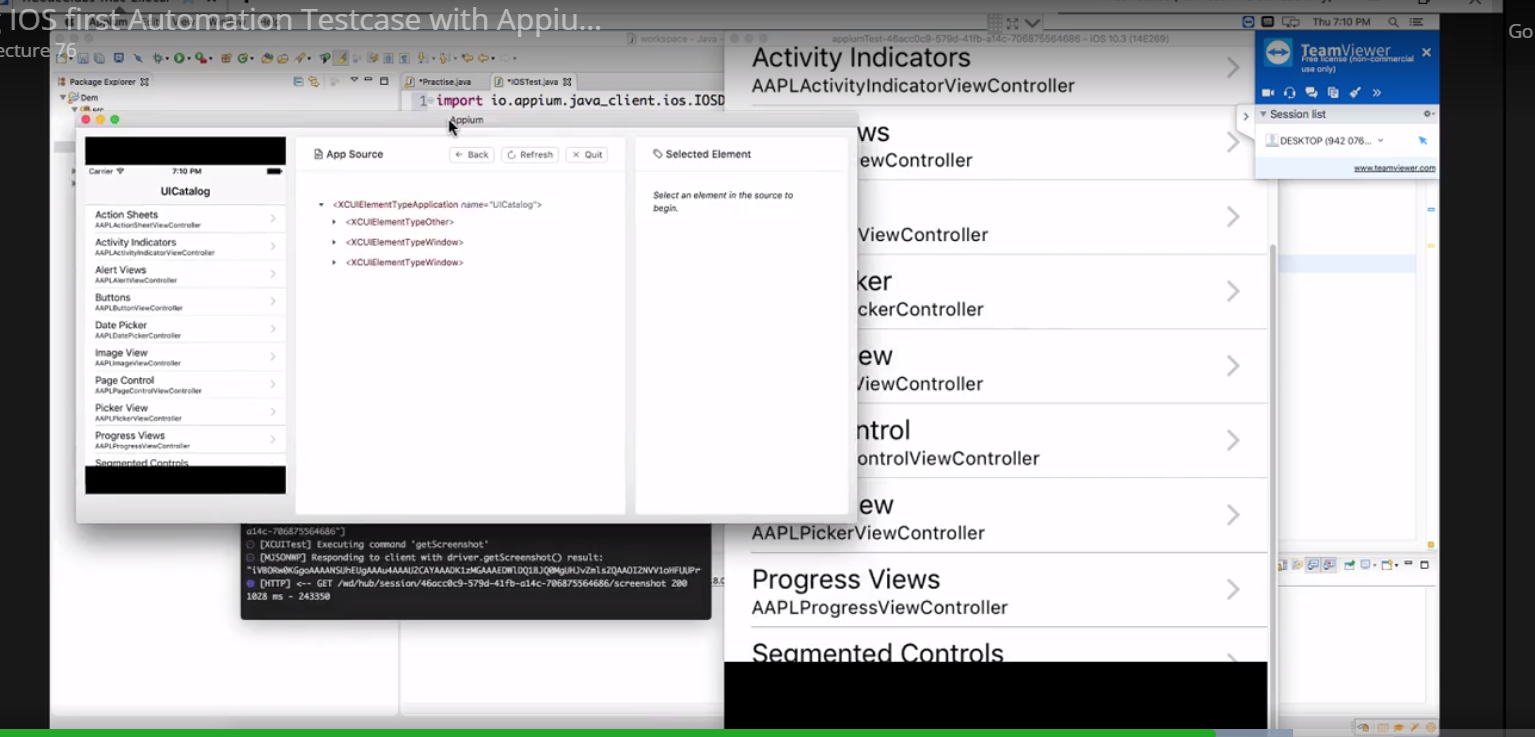
**Appium Desktop (for the connection of server)**

* Now A graphical interface for the Appium Server. You can set options, start/stop the server, see logs, etc... You also don't need to use Node/NPM to install Appium, as the Node runtime comes bundled with Appium Desktop.
* In Background – your logs will be running simultaniously

Inspecting elements:

Important: if you want to inspect the elements we have to start with the Appium desktop

* Once the server connection is done and the ios simulator opened
* Now, as soon as the app is invoked in to the simulator we can see a window which is an inspector
* Information is in json structure
* 
* Appium inspector



**Appium detailed view:**

