**APPIUM**

1. **For Android**

* We should have JDK
* *Download /check if you have Microsoft Visual C++ framework*
* *Download node.js*
* Download Android SDK not Android Studio
  + Set the path of **tools** and **platform-tools** in **Environment Variable** as **ANDROID\_HOME**
* We should download from Android SDK manager
  + Tools
  + Android 4.4.2 (API 19) (as per you mobile version)
  + Extra

1. **For Eclipse**

* Download ANDROID PLUGIN for eclipse (<https://dl-ssl.google.com/android/eclipse/>)
  + Now DDMS will be available
  + SDK manager icon will also be available

1. **Connect Mobile**

* DDMS to check if the device is visible
* So first get the Developer Mode ON
  + USB Debugging mode ON under Developer Option
* Incase mobile is not getting detected or Android Driver set up is not available then go for **PDA net+** on both PC and Mobile
* The device should be visible in DDMS

1. **Connect mobile using WIFI**

* Go to place where “adb.exe” is available i.e. platform tools under Android
* Open CMD
* Write **adb tcpip 5555** (connection gets restart)
* Write **adb connect 192.168.0.1** (your mobile IP address)
* Write **adb devices**
  + Two devices will be visible, one from USB and one from WIFI
* Remove the USB and check **adb devices,** only one device with wifi should be visible

1. **For Appium**

* Download Appium.exe for windows and extract it to see ***Appium.exe***
* ***Two ways to start***
* 1) Using Appium.exe

Run ***Appium.exe*** , server should start at 127.0.0.1: 4723

* 2) Using below commands

Go to path having the appium.js e.g. "E:\Appium\node\_modules\appium\bin"

Write:

E:\\Appium\\node E:\\Appium\\node\_modules\\appium\\bin\\appium -p 4723 -U 192.168.0.4:5555 --chromedriver-port 9515

(4723 is the default port can be changed)

(192.168.0.4:5555 is the ip address of the mobile)

1. **Coding**

Use **remote webdriver** and **desire capability** to configure your code with Appium

* To get element’s locator use **chrome://inspect/#devices**
* To detect elements of APK, go to DDMS and Dump view hierarchy for UI Automator

Extra

1. **Connect two Mobile with Wifi**

Connect mobile 1 with USB

* adb tcpip 5555
* Remove the mobile 1

Connect mobile 2 with USB

* adb tcpip 5555
* Remove the mobile 1
* Now write
* adb connect 192.168.0.4
* adb connect 192.168.0.100

1. **Run code on two devices**

Start server A with IP address 127.0.0.1 and port 1234 (for example)

* + Initialize your driver and connect it to Server A.
  + Make sure to add the following capability to your driver A:capabilities.setCapability("udid", "Device A UDID");

Start server B with IP address 127.0.0.2 and port 4321 (for example)

* + Initialize your driver and connect it to Server B.
  + Make sure to add the following capability to your driver B:capabilities.setCapability("udid", "Device B UDID");

Note:

* + Your two server instances should have different IP addresses and ports in order for them to be able to work simultaneously.
  + Make sure to set the device UDID in the session capabilities for the test to run on the correct device.
  + To get the device udid, from the command line type:
    - adb devices
    - **Example: set the UDI as below**

DesiredCapabilities cap= DesiredCapabilities.android();

cap.setCapability("udid", "42038373d07fc000");

Extra Note:

@BeforeTest

**public** **void** setup() **throws** MalformedURLException

{

File application= **new** File("Resource/TheMallApp.apk");

DesiredCapabilities cap = DesiredCapabilities.*android*();

//cap.setCapability("browserName", "chrome");

cap.setCapability("deviceName", "Lenevo");

cap.setCapability("platformVersion", "5.1.1");

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

cap.setCapability("app", application.getAbsolutePath());

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

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

driver.manage().timeouts().implicitlyWait(30, TimeUnit.***SECONDS***);

}