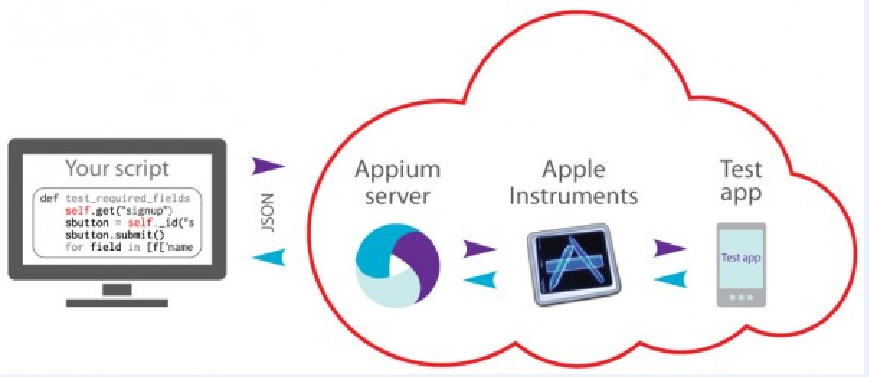
**Appium**

Appium is an open-source tool for automating native, mobile web, and hybrid applications on iOS and Android platforms. Native apps are those written using the iOS or Android SDKs. Mobile web apps are web apps accessed using a mobile browser (Appium supports Safari on iOS and Chrome or the built-in ‘Browser’ app on Android). Hybrid apps have a wrapper around a “webview” – a native control that enables interaction with web content.

Appium is “cross-platform”: it allows you to write tests against multiple platforms (iOS, Android), using the same API. This enables code reuse between iOS and Android testsuites.

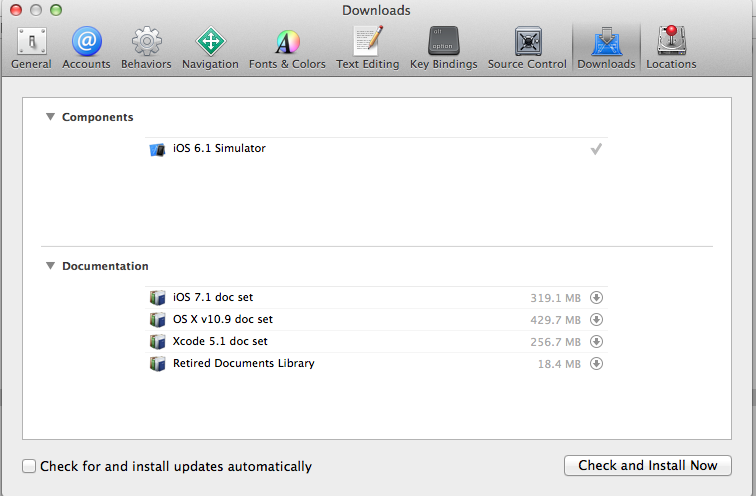


**Requirements:**

1. Mac OS X
2. Xcode
3. Appium
4. Android SDK
5. Eclipse IDE
6. Selenium and Appium Java Client JAR files

**IOS – Setup Steps:**

1. Download Xcode from Apple App Store. After installing Xcode, download all its simulators
2. Click on Xcode button at top left side of the screen
3. Click the Preferences button
4. Download all simulators from Downloads window



1. Also Download Command lines tools from the downloads(If you don’t see the command line tools in the downloads, you can get it from here <https://developer.apple.com/downloads/index.action>)

**Android Automation**

Download Android SDK from **[http://developer.android.com/](http://developer.android.com/" \t "_blank)**Go to Develop => Tools => Download => Download ADT for Mac OS X.  
Install the ADT to Application folder. Install all emulators with SDK 17+

1. Open the ADT folder from Finder
2. Open Eclipse folder
3. Open the Eclipse.app program
4. Click the Window button from top bar
5. Click the **Android SDK Manager** button
6. Check all API’s 17+ and the Tools and Extras folders and install them

**Setup Bash-Profile**

1. Open Terminal
2. Enter $ vim .bash\_profile
3. Export ANDROID\_HOME = (path to SDK folder)
4. Export JAVA\_HOME = <SDK Path>

**Appium App Setup** (Either use Appium APP or Source code to start the Appium server)**:**

Download Appium program from: [http://appium.io/](http://appium.io/" \t "_blank) => **Downloads** => **Appium.app for OS X** =>**appium-x.x.x.dmg**  
Unpack Appium-x.x.x.dmg to Application folder.

**Setup Appium for Testing - IOS**

1. If you use a simulator:
   1. Select a device and check the checkbox in front of “Force Device”
   2. Select a “Platform Version”
   3. Check the checkbox in from of “Show Simulator Log”
2. If you use a Real Device:

1. Select a checkbox in front of UDID

1. In “UDID” field enter devices UDID (device UDID number could be opened in  
   iTunes/device page/Tap the “Serial number” field)
2. Uncheck the “Force Device” checkbox if it is checked
3. Launch the “Appium” app
4. Verify if all settings are corrected by taping the “Doctor” button
5. If all is correct tap the “Inspector” button to start recording the test.

**Setup Appium for Android:**

1. Check checkboxes and select a package in fron of:
   1. Package
   2. Wait for Package
   3. Launch Activity
   4. Wait for Activity
2. Capabilities:
   1. Select “Platform Name” – “Android”
   2. Select “Automation Name” – “Appium”
   3. Select “Platform Version” API 17+
3. Launch Device:
   1. Check the checkbox in from of “Device Ready Timeout”
   2. Set up “60” seconds to “Device Ready Timeout”
4. If you use emulator:
   1. Check the checkbox in front of “Launch AVD” (the emulator device should already be created)
   2. In capabilities  – “Platform Version” select the AVD selected API (Android version)

If you use real device:

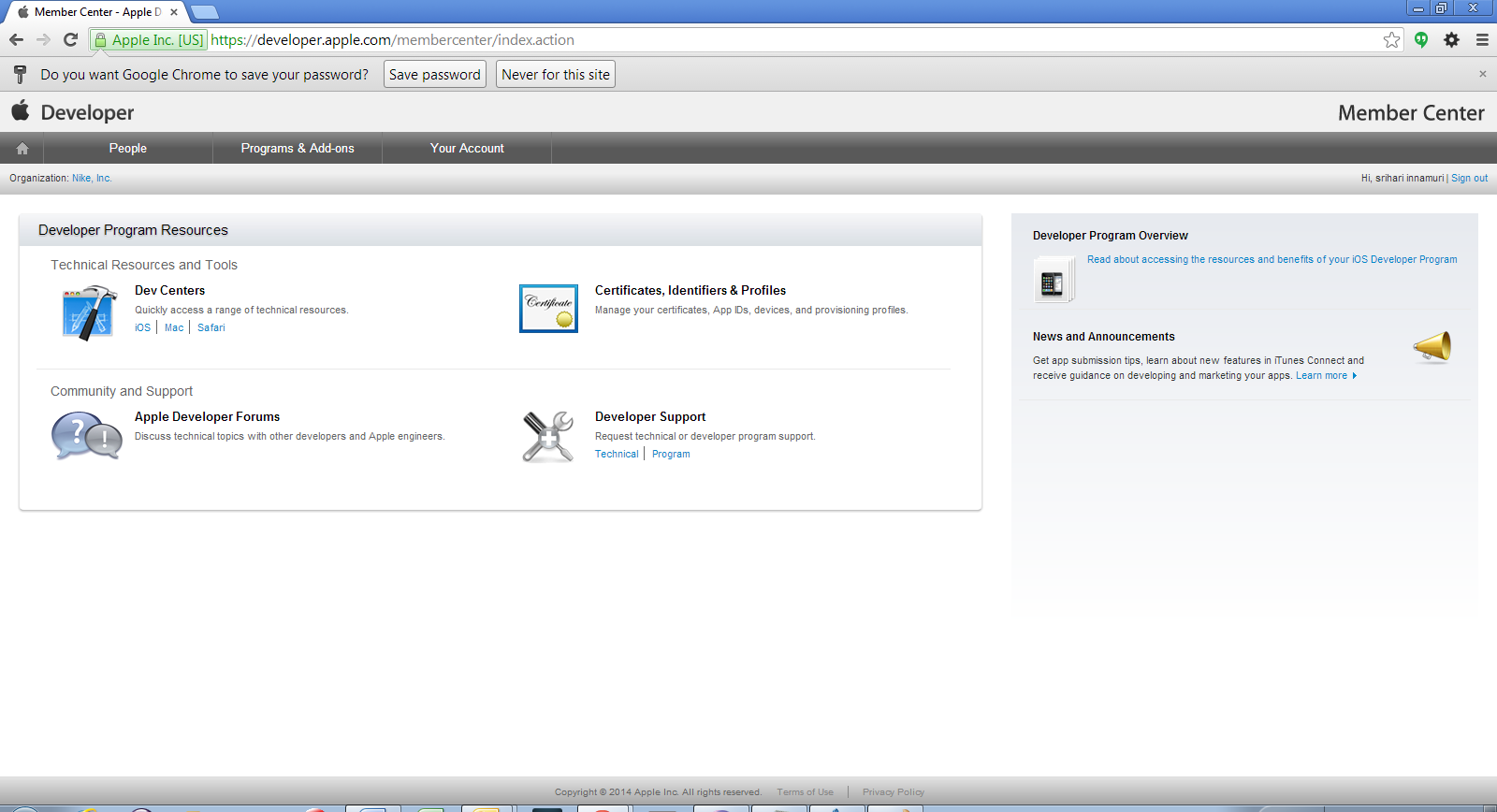
* 1. Check the checkbox in front of “Device Name”
  2. Fill the “Device Name” field with device name which can be get from “Android File Transfer” application
  3. Check the checkbox in front of “Language”
  4. Select devices language

1. Launch the “Appium” app
2. Verify if all settings are corrected by taping the “Doctor” button
3. If all is correct tap the “Inspector” button to start recording the test.

**Provisioning Profile for IOS**

A valid iOS Development Distribution Certificate and Provisioning Profile are necessary to test on a real device

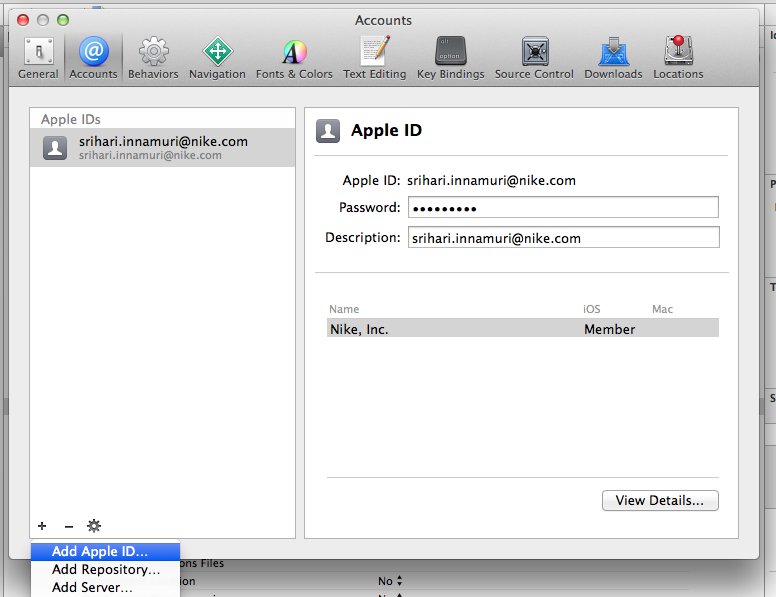
* 1. Create Appleid (<https://appleid.apple.com/account>)
  2. Create a Jira ticket under MDS project with Appleid and device UDID
  3. Once you got the approval, access below link (<https://developer.apple.com/membercenter/index.action>) to see the certificate



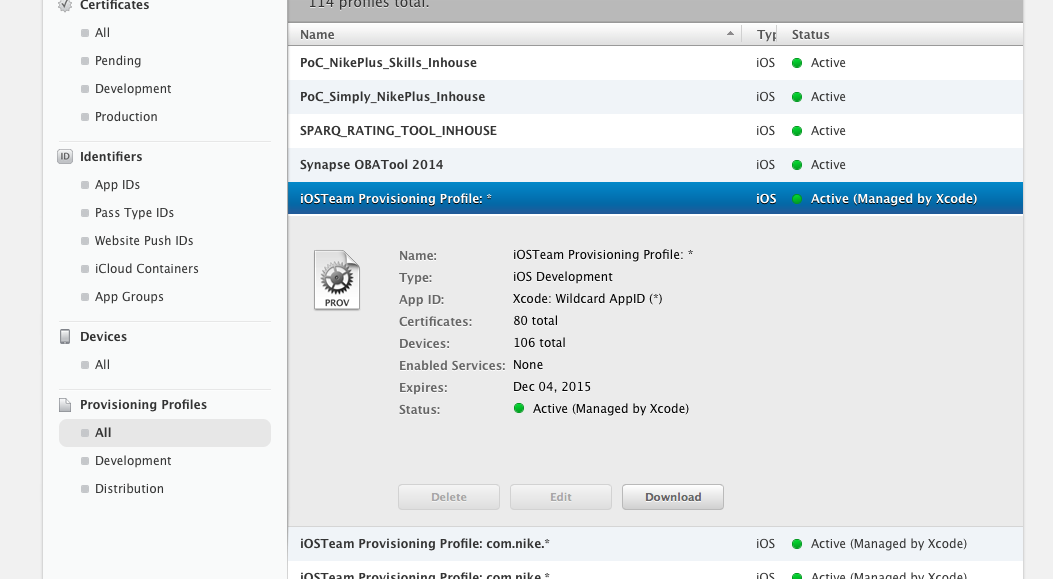
* 1. Click on Certificates, Identifiers & Profiles and then again click on Certificates link. You must see your name in the Certificates list.

**Refreshing Provisioning profiles in the Xcode**

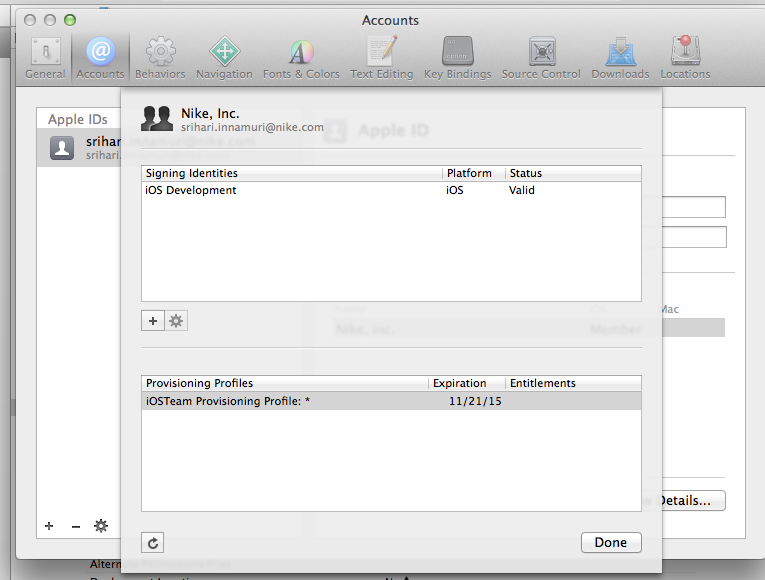
* 1. In the Xcode Preferences window, click Accounts.
  2. **Add Your Apple id by click on ‘Add Apple ID’ link**

****

1. **Go to the devices section in** [**https://developer.apple.com/account/ios/device/deviceList.action**](https://developer.apple.com/account/ios/device/deviceList.action) **and check UDID number**
2. **Expand the ‘Provisioning Profile’ section and click on ‘All’. Download the IOs team provisioning profile**

****

5. In the dialog that appears, click the Refresh button in the lower-left corner under the Provisioning Profiles table. Xcode updates the list of profiles in the Provisioning Profiles table.

****

**Setup environment in Mac using Source Code**

Open terminal and follow the below steps

**Step 1: Install IOS-Webkit-debug-proxy**

1. Install Brew

2. brew install node

3. brew install ios-webkit-debug-proxy

**Step 2: Get Safari Launcher and build – To launch safari browser in real devices**

1. git clone <https://github.com/budhash/SafariLauncher.git>

2. cd SafariLauncher

3. sudo ./build.sh --=[path to your apple developer cert] --profile=[path to your provisioning profile]

4. open SafariLauncher in xcode and develop to device

**Step 3: Trust device and run ios\_webkit\_debug\_proxy**

1. connect device to Mac

2. idevicepair pair

3. idevicepair validate

4. ios\_webkit\_debug\_proxy -d -c <UDID>:27753

**Step 4: To create a profile for the launcher go into the Apple Developers Member Center and get "identity code"**

1) Create a new App Id and select the WildCard App ID option and set it to "\*"

2) Create a new Development Profile and for App Id select the one created in step 1.

3) Select your certificate(s) and device(s) and click next.

4) Set the profile name and generate the profile.

5) Download the profile and open it with a "Text Editor" or any text editor.

6) Search for the UUID and the string for it is your identity code.

**Step 5: Install and run appium server (To execute the scripts on real devices)**

1. git clone <https://github.com/appium/appium.git>

2. cd appium

3. ./reset.sh --code-sign=[path to your apple developer cert] --profile=[path to your provisioning profile]

4) Copy SafariLauncher/build/Debug-iphoneos/SafariLauncher.zip from build in Step2  to

<Path>/appium/build/SafariLauncher/SafariLauncher.zip

5) $ node ./lib/server/main.js -U <UDID>

6) To execute the scripts on simulator, open terminal and follow the below steps

i) cd appium

ii) Appium (this command will start the appium server)

**Step 6: Enable inspector on iOS device**

1) Turn on web inspector on iOS device (settings > safari > advanced)

**Appium Desired Capabilities – Real Devices:**

DesiredCapabilities capabilitiesIOS = new DesiredCapabilities();

capabilitiesIOS.setCapability("platformName", "iOS");

capabilitiesIOS.setCapability("platformVersion", "7.0");

capabilitiesIOS.setCapability("safariAllowPopups", false);

capabilitiesIOS.setCapability("safariIgnoreFraudWarning", "true");

//capabilitiesIOS.setCapability("deviceName", "iPad Air");

//capabilitiesIOS.setCapability("deviceName", "iPhone Simulator");

capabilitiesIOS.setCapability("deviceName", "<UDID>");

capabilitiesIOS.setCapability("UDID", "<UDID>");

capabilitiesIOS.setCapability("browserName", "Safari");

try {

driver = new IOSDriver(new URL("http://localhost:4723/wd/hub"),

capabilitiesIOS);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}

**Appium Desired Capabilities – IOS Simulators:**

DesiredCapabilities capabilitiesIOS = new DesiredCapabilities();

capabilitiesIOS.setCapability("platformName", "iOS");

capabilitiesIOS.setCapability("platformVersion", "7.0");

capabilitiesIOS.setCapability("safariAllowPopups", false);

capabilitiesIOS.setCapability("safariIgnoreFraudWarning", "true");

capabilitiesIOS.setCapability("deviceName", "iPhone Simulator");

capabilitiesIOS.setCapability("browserName", "Safari");

try {

driver = new IOSDriver(new URL("http://localhost:4723/wd/hub"),

capabilitiesIOS);

} catch (Exception e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

}