# Automation setup environment

        Before starting to use the Automation framework, a setup needs to be done. Please take into consideration that running the automation tests on an iOS device is not possible from the Windows machine, since the WebDriverAgent is necessary. WebDriverAgent is a WebDriver server implementation for iOS that can be used to remote control iOS devices. WebDriverAgent needs to be installed and setup to allow Appium to automate iOS devices.

# ****Setup prerequisites for running mobile automation on Android devices****

Software required:

* 1. Java JDK
  2. Android SDK (Android Studio)
  3. Node.js
  4. Appium Server
  5. Intellij

**1.    Java JDK**

-        Go to <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

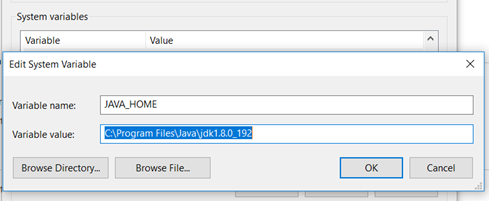
-        Download the suitable version for your machine (ex: for 64-bit )

**On Windows:**

-        Install it in **C:\Java** if possible

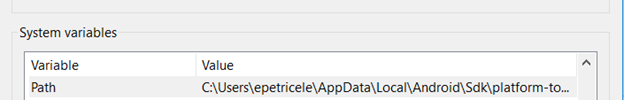
-        **Add JAVA as System Variable:**

* Right click on This PC and select Properties.
* Click on Advanced system setting
* Click on Environment Variables
* In the system variables create a new system variable named JAVA\_HOME and give the path to your Java folder (ex: C:\Program Files\Java\jdk1.8.0\_192)

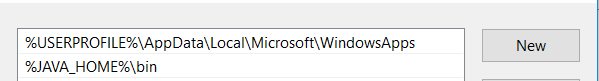


-        **Add JAVA bin to Path:**

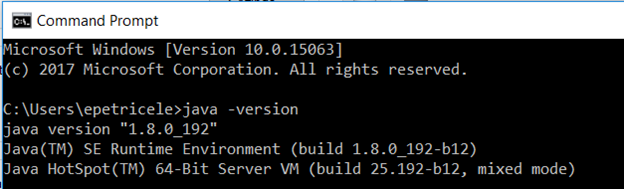
* Click on Path from the System variables



* Select Edit button
* Add a new line with %JAVA\_HOME%\bin



-        Open Command Prompt and type java -version or java -v



**On Mac:**

-        Create a .bash\_profile with JAVA\_HOME:

-        $ cd ~   -> this puts you in the home directory

-        Create the file with the following command $ nano .bash\_profile which opens the file in editing mode

-        Type in the file JAVA\_HOME variable &  edit the PATH variable

• export JAVA\_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0\_71.jdk/Contents/Home

• export PATH=$JAVA\_HOME:$PATH

-        Press CTRL+X. Confirm by typing the Y   key and hit Enter which will save and exit the editor

-        Make changes to take effect by running the following command: $ source~/.bash\_profile

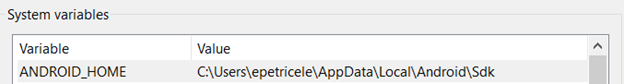
**2.  Android SDK**

-        Download full Android Studio from: <https://developer.android.com/studio/>, install it and download all mandatory files (will be prompted on starting Android Studio)

-        To manipulate SDK manager from UI, start Android Studio à create a simple project and from IDEE click on SDK Manager button (upper right blue arrow next to emulator icon) – from there you can download latest tools and platform tools

**On Windows:**

-        **Add Android SDK as System Variable:** ANDROID\_HOME (C:\AndroidStudio\SDK) à this is an example; you need to put path to your SDK (by default it is installed in C**:\Users\yourUser\AppData\.....)**



 -        **Add Android tools and platform tools to Path:**

* Click on Path from the System variables
* Select Edit button
* Add 2 new lines as the below example:

C:\ae8bbde23a49d84fddfe3df827d7dd8c

**On Mac:**

-        Add in the .bash\_profile ANDROID\_HOME:

-        $ cd ~   -> this puts you in the home directory

-        Open the file with the following command $ open -e .bash\_profile

-        Type in the file ANDROID \_HOME variable & edit the PATH variable

• export ANDROID\_HOME=/Users/current\_user/Library/Android/sdk

• export PATH=$ANDROID\_HOME/tools:$ANDROID\_HOME/platform-tools:$PATH

-        Press CTRL+X. Confirm by typing the Y   key and hit Enter which will save and exit the editor

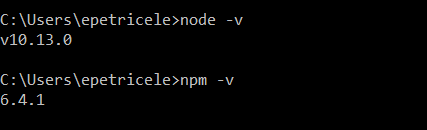
-        Make changes to take effect by running the following command: $ source~/.bash\_profile

**3.  Node JS**

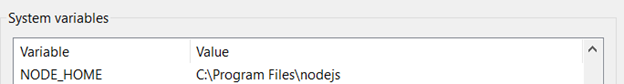
-        Installation of Node and NPM can be found here: <http://blog.teamtreehouse.com/install-node-js-npm-windows> (please verify in cmd that it is installed correctly and it is running)

**On Windows:**

-        Download location: C:\nodejs\node.exe



 -        Add node as System Variable: NODE\_HOME -> C:\nodejs

****

**On Mac:**

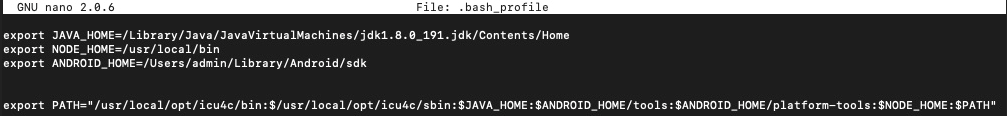
-        Installation of Node and NPM can be found here: <https://blog.teamtreehouse.com/install-node-js-npm-mac>

-        Verify that the installation was succeeded by entering $ npm -v and $ node -v commands in the terminal and it will display the version.

-        Add Node JS in the .bash\_profile as NODE\_HOME:

-        Type in the file NODE \_HOME variable & edit the PATH variable

After adding all the variables, your .bash\_profile should look something like this:



**4.  Appium server**

-        Before starting, type in cmd/terminal : **npm info Appium** and hit enter

-        If you scroll up you will see Appium server versions

-        Latest version is always the one which is in development (new fixes are added but also might introduce other bugs) – follow Appium on GitHub for more info: <https://github.com/appium/appium> mostly on issues to see latest bugs

-        When you decided which Appium server you want to use open again cmd and type: **npm install -g** [**appium@1.9.0**](mailto:appium@1.9.0)(to download a specific version) or **npm install -g appium@beta** (to download latest beta version) and hit enter.

-        Download location will be by default: **C:\Users\yourUser\AppData\Roaming\npm\node\_modules\appium\build\lib\main.js** on Windows

**5.  Intellij**

-        Download Intellij community edition: <https://www.jetbrains.com/idea/download>

-        This IDE it is used for writing automated scripts so it is highly recommended to be the one used for running the scripts.

# ****Setup prerequisites for running mobile automation on iOS devices****

Software required:

* 1. XCode
  2. Homebrew
  3. Dependencies to make appium-xcuitest-driver to work:
  4. WebDriverAgent

**1.  Xcode**

-      Install XCode: <https://developer.apple.com/download/>

**2.  Homebrew**

-      If you haven’t installed it with Node JS, then you should install it now

-      Go to: <https://brew.sh/> and follow the instructions on how to install it

**3.   Install dependencies to make appium-xcuitest-driver to work:**

-        Open terminal and type:

* **brew install ideviceinstaller**  (to add libmobiledevice)

-        Libimobiledevice

* **brew uninstall --force libimobiledevice**
* **brew install --HEAD libimobiledevice**

-        Ios-webkit-debug-proxy

* **brew uninstall --force ios-webkit-debug-proxy**
* **brew install ios-webkit-debug-proxy**

-        In order to make WebDriverAgent to work we need to have Carthage:

* **brew install carthage**

-        For iOS 10 ideviceinstaller does not work so we need ios-deploy:

* **npm install –g ios-deploy** - To be able to install applications on your phone like on android with ADB,  
  here on iOS we have ios-deploy commands. Basically you need ios-deploy installed and after that you only need that device is connected via USB to your mac and just pass the commands.

-        For real device use, and to make Xcode user friendly we need to install xcpretty:

* **gem instal xcpretty**

**4.   Install WebDriverAgent**

-        Now we have all dependencies downloaded and we need to configure appium-xcuitest-driver in order to be able to run test on real iOS devices. Go to terminal and change directory to the one where WebDriverAgent is  installed and run the script to download all necessary frameworks:

    Ex: In terminal type:

* cd /usr/local/lib/node\_modules/appium/node\_modules/appium\_xcuitest\_driver/WebDriverAgent

-        Then run the command:

* ./Scripts/bootstrap.sh

-         Once all dependencies are downloaded from WebDriverAgent folder open WebDriverAgent.xcodeproj

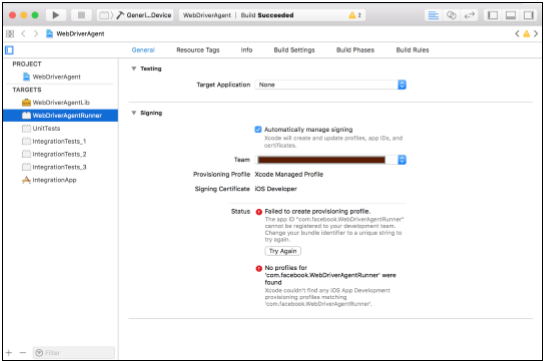
(a small project will appear).

-  **BEFORE MAKING ANY CHANGE** be sure that a valid certificate is installed on our Mac Machine (preferably the one which was used to sign the application that you want to automate).

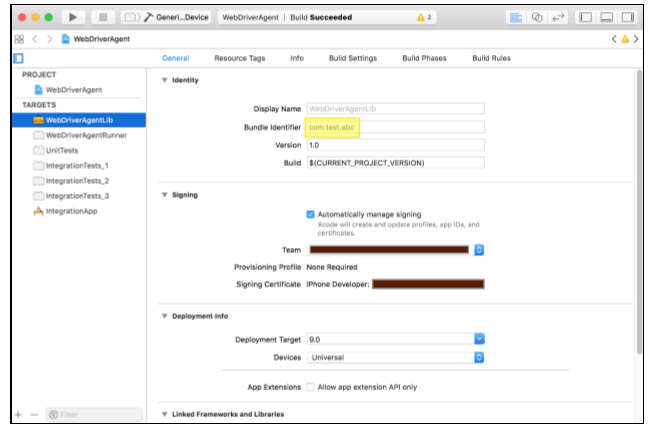
-           On WebDriverAgent Xcode project go to **WebDriverAgentLib** and from **General** tab on **Signing Debug** and **Signing release** select the Team of your certificate from above (hit build and save to save changes and to see that it   is working). After select **WebDriverAgentRunner** and in **General** tab on **Signing debug** and **release** select **Provisioning profile** for your certificate (hit build and save).

**Note:**

-        If an error appears saying that Xcode failed to create provisioning profile:



-          The easiest way to resolve that is 1) Select **WebDriverAgentLib** under **TARGETS**, 2) select **Automatically manage signing**, 3) select valid **Team** and most important 4) change the **Bundle Identifier** and put the Bundle Identifier of your existing valid XCode project the purpose here to put something that Xcode will accept.



-           Also ensure that you should have installed the valid Provisioning profile (Of course compatible with entered Certificate and Bundle Identifier). Now move to **WebDriverAgentRunner** again and 1) Select valid Provisioning Profile under Signing (Debug) and 2)  Select valid Provisioning Profile under Signing (Release).

 -          Connect valid iPhone device to your Mac machine (Please ensure device is included in selected provisioning profile).

 -          Select **WebDriverAgentRunner** under TARGETS and click on Test button to execute build on your connected device.

 -           You can observe that when you click on Test/Run button the WebDriverAgent application will be installed to iOS device and it will open and give you the black screen for a moment and automatically closed. That means Success. **Now you can able to Run Appium script on this device**.(In fact it applies to all the valid devices registered under selected provisioning profile).