Visual Studio Project Setup

Contents

[Utilities and Versions 3](#_Toc524100848)

[Steps to Execute using Command Prompt 4](#_Toc524100849)

[New Project Setup 5](#_Toc524100850)

[Steps to create new project 5](#_Toc524100851)

[Screenshots 5](#_Toc524100852)

[Extensions 6](#_Toc524100853)

[Steps to Install 6](#_Toc524100854)

[Screenshots 7](#_Toc524100855)

[NuGet Packages 8](#_Toc524100856)

[Steps to Install 8](#_Toc524100857)

[Screenshots 8](#_Toc524100858)

[Appium Setup 10](#_Toc524100859)

[Android automation using Appium 10](#_Toc524100860)

[Android SDK Setup 10](#_Toc524100861)

[Appium Desktop App Setup 14](#_Toc524100862)

[Appium Non-Gui Setup 16](#_Toc524100863)

[iOS Automation using Appium 18](#_Toc524100864)

[Appium Setup 18](#_Toc524100865)

[Test Execution Process 19](#_Toc524100866)

# Utilities and Versions

|  |  |
| --- | --- |
| **Name** | **Version** |
| Visual Studio | Pro 2017 v15.6.6 |
| .Net Framework | v4.7.02556 |
| Appium | v1.7.2 |
| **NuGet Packages** | |
| NUnit Framework | v3.10.1 |
| NUnit Console | v3.8.0 |
| NUnit2TestAdapter | v3.10.0 |
| Selenium Webdriver | v3.11.2 |
| Selenium Support | v3.11.2 |
| Appium Webdriver | v3.0.0.2 |
| Chrome Driver | v2.38.0 |
| Firefox Driver | v0.20.1 |
| Extent Report | v3.1.3 |
| Log4Net | v2.0.8 |
| **Extensions** | |
| NUnit VS Templates | v1.3 |

# Steps to Execute using Command Prompt

1. **Check out the latest code from GitHub repository**

(<https://github.com/OnlifeHealth/qa_automated_tests.git>)

1. **Build the solution from command prompt, following command can be used:**

*%WINDIR%\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe E:\OnlifeSeAutomationFrameWork\AutomationFramework\AutomationFramework.sln /t:Rebuild /p:Configuration=Release /p:Platform="any cpu"*

1. **Start the execution of sanity test script through command:**
2. *“E:\OnlifeSeAutomationFrameWork\AutomationFramework\packages\NUnit.ConsoleRunner.3.8.0\tools\nunit3-consolenunit3-console.exe"*
3. *“E:\OnlifeSeAutomationFrameWork\AutomationFramework\AutomationFramework\bin\Debug \AutomationFramework.dll" --where "cat==BuildSanity” --params=baseurl=varBaseurl; env=varEnv; clientname=var Clientname --result=*[*\\10.244.8.28\Reports\SeReports\LiveOn\_QB%date:~-10,2%%date:~-7,2%%date:~-4,2%\_%time:~3,2%\_%time:~6,2%.xml*](file:///\\10.244.8.28\Reports\SeReports\LiveOn_QB%25date:~-10,2%25%25date:~-7,2%25%25date:~-4,2%25_%25time:~3,2%25_%25time:~6,2%25.xml)

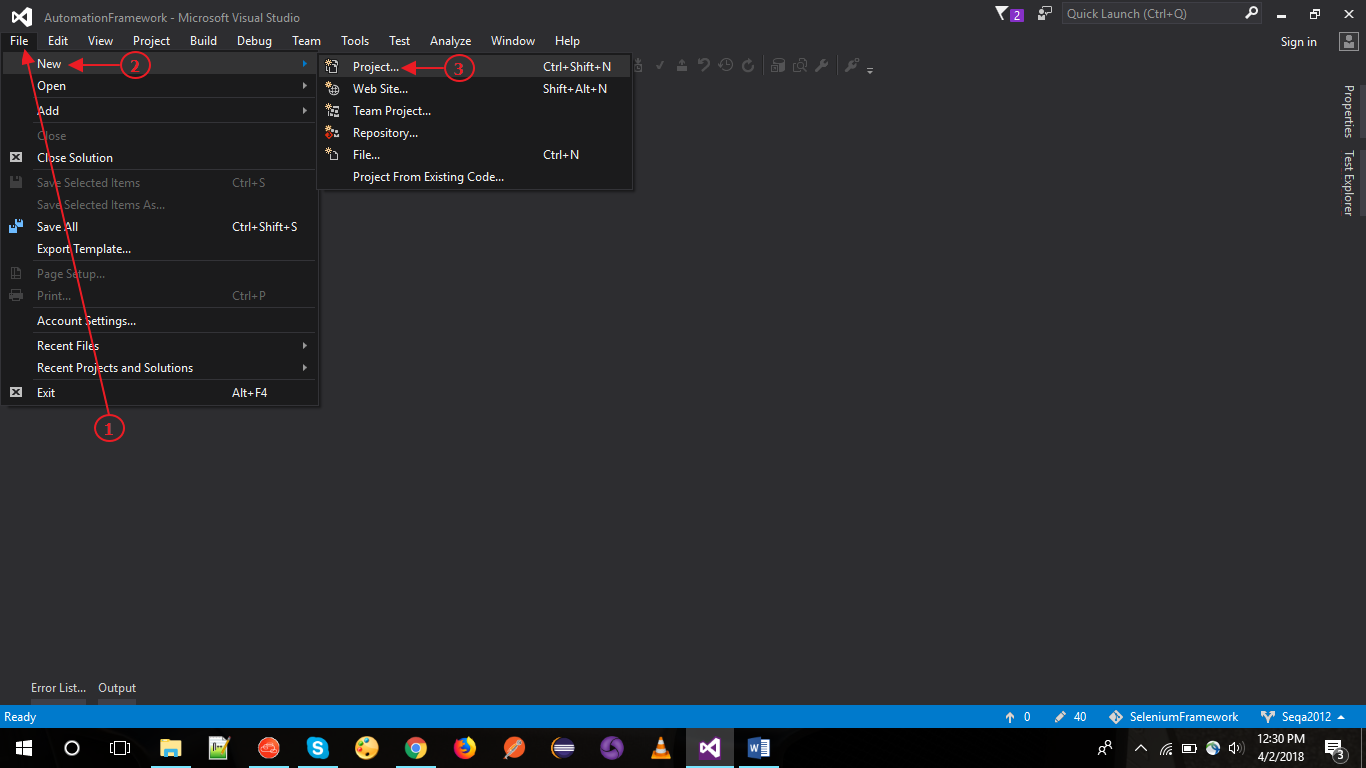
***Note: varBaseurl, varEnv and varClientname are variables***

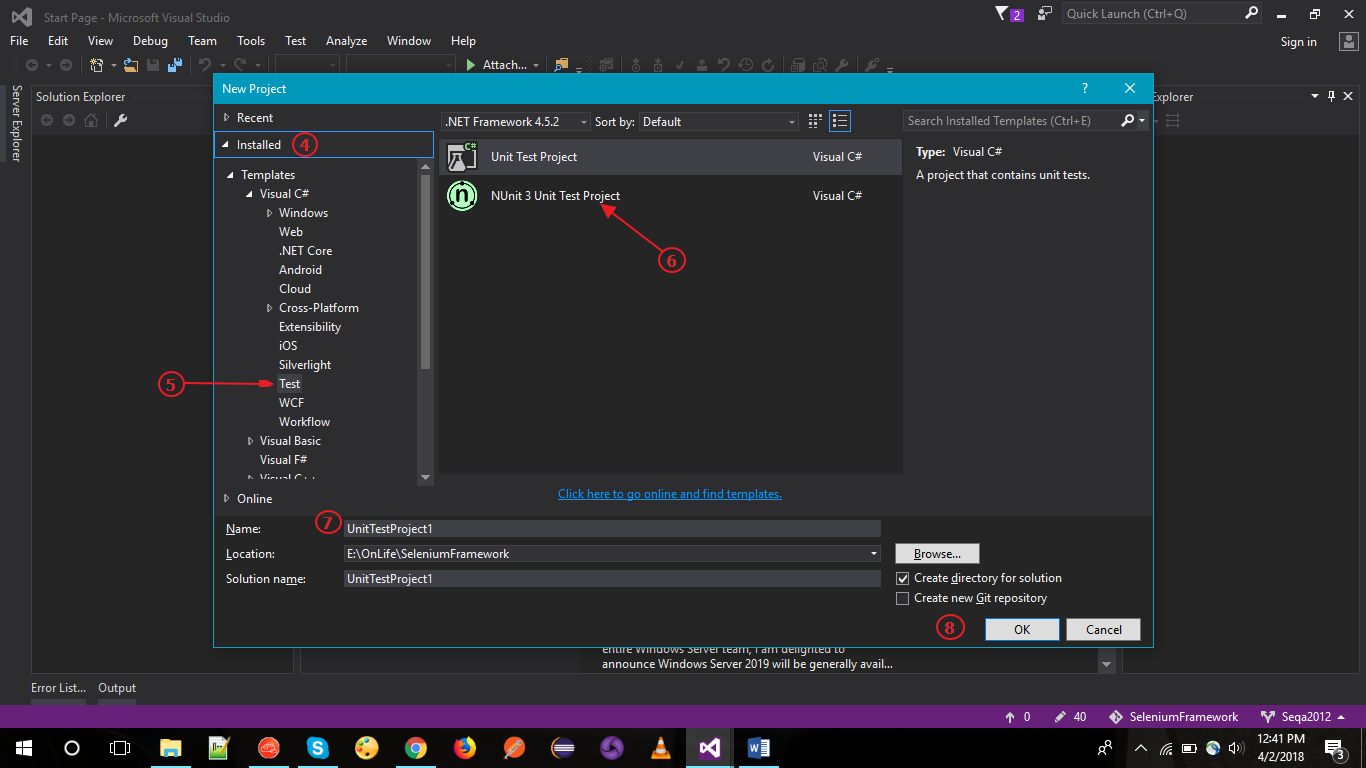
# New Project Setup

## Steps to create new project

1. Install [NUnitVSTemplates extension](#_Extensions)
2. Create New Project: File > New > Project (ctrl+shift+N)
3. Under *“Installed”* menu, select *“Tests”*
4. Select *“NUnit 3 Unit test project”*
5. Name the project and click *Ok*
6. Install required [*NuGet Packages*](#_NuGet_Packages)

### Screenshots





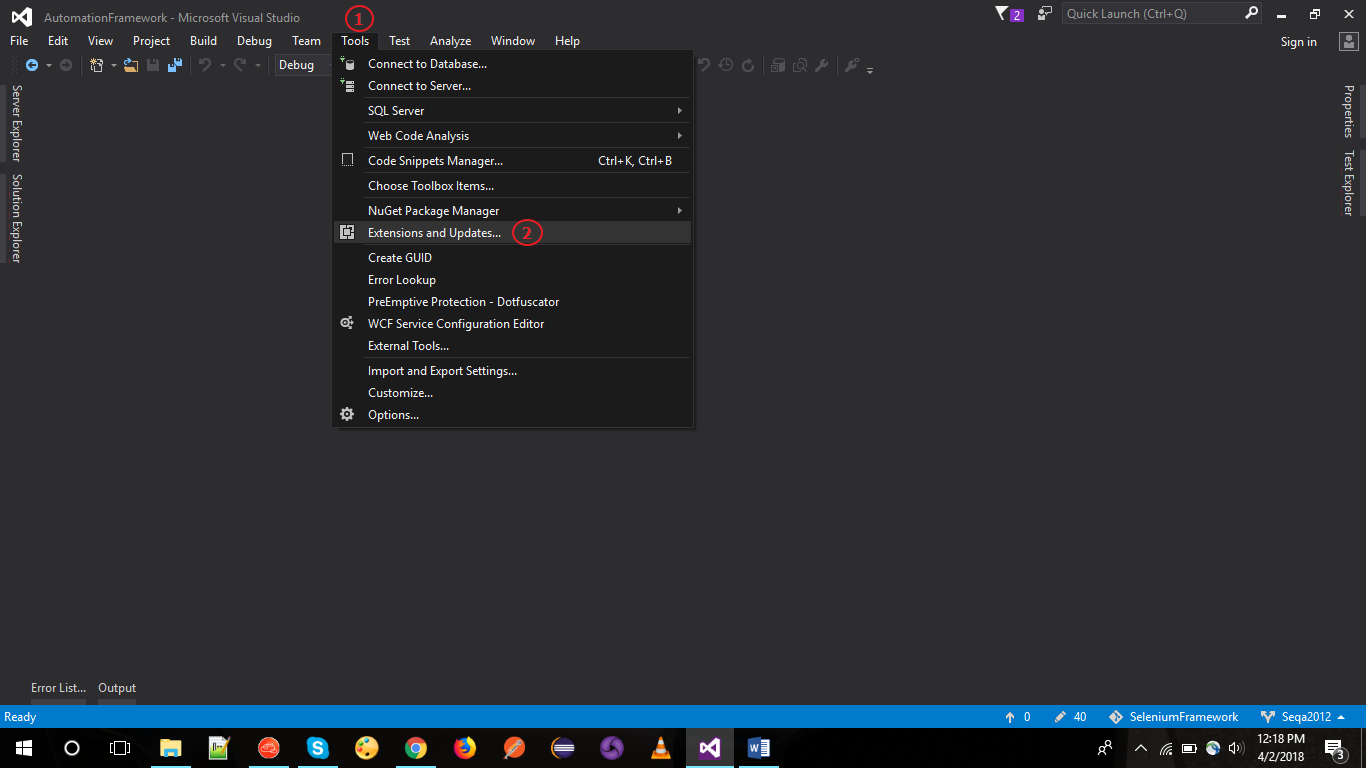
# Extensions

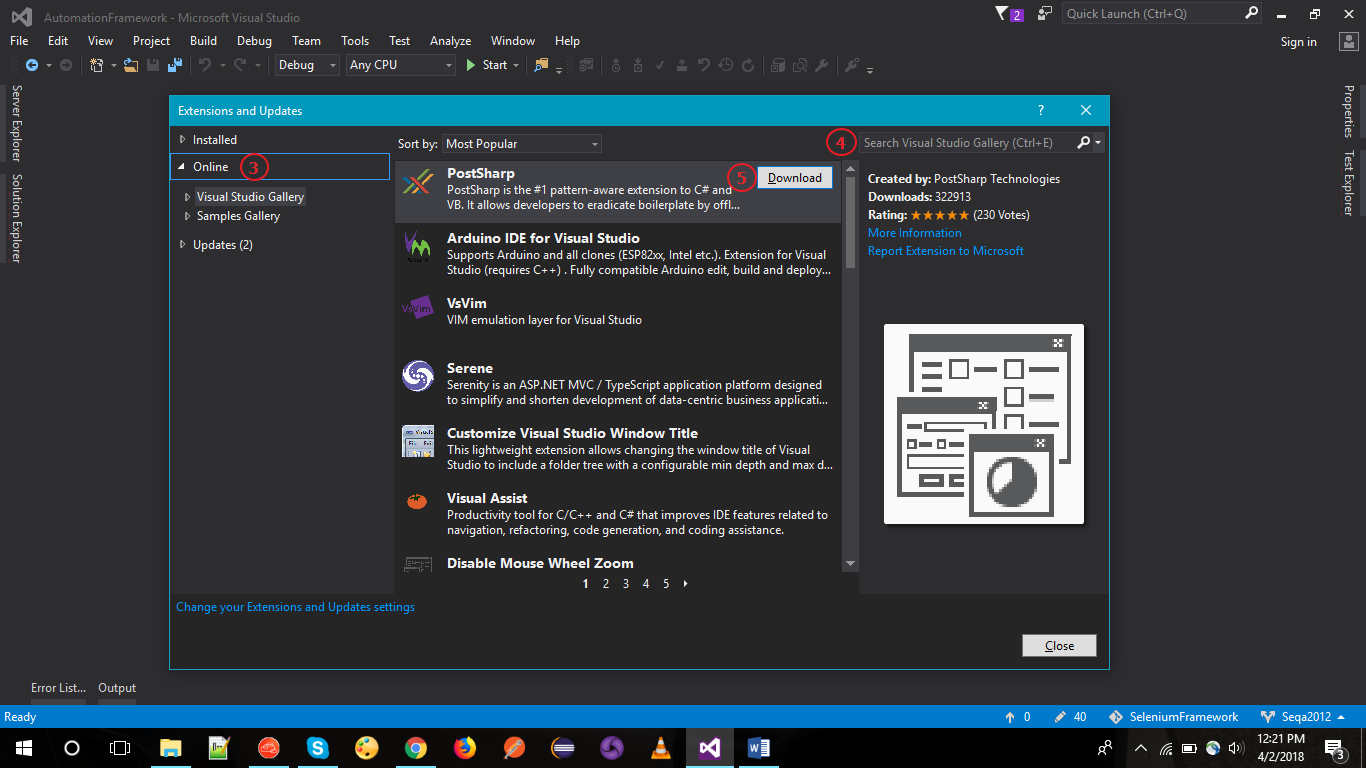
Extensions installed are bound with Visual Studio and needs to be installed by each user in their own system.

## Steps to Install

1. Menu Bar > Tools > Extensions and Updates
2. Select “Online” tab
3. Search for Extensions and click Download

### Screenshots





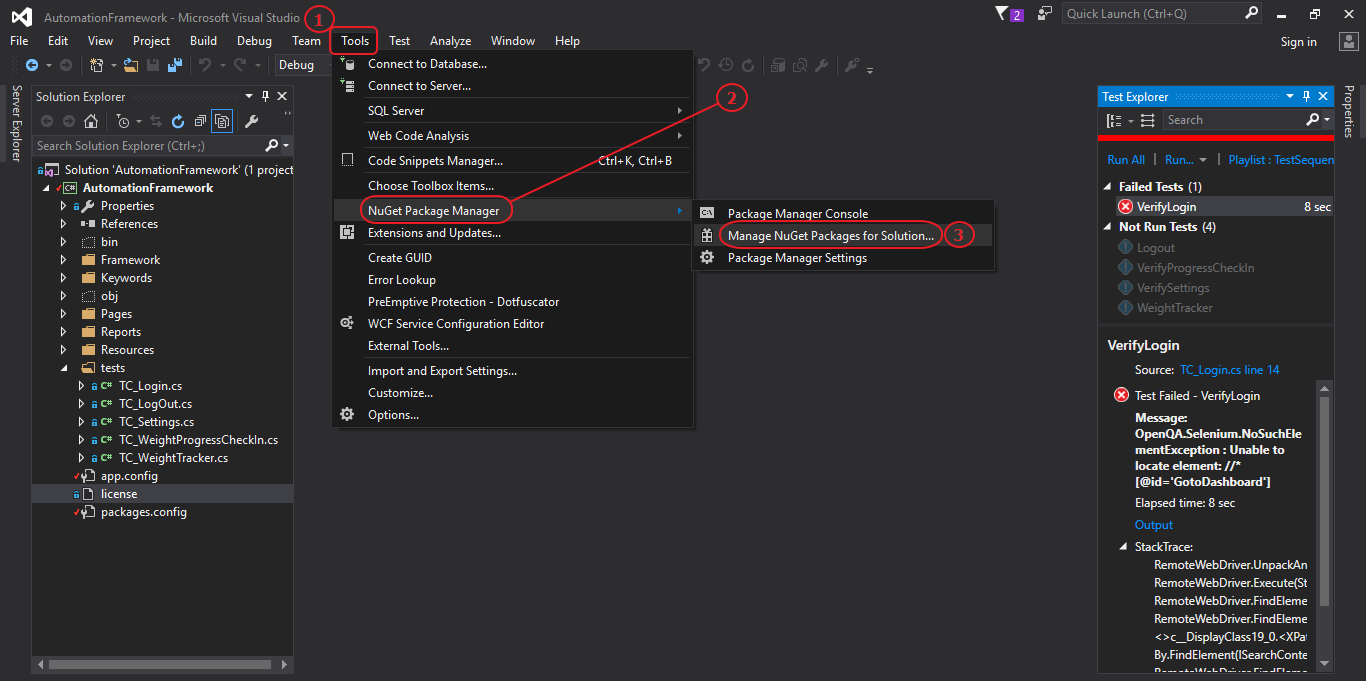
# NuGet Packages

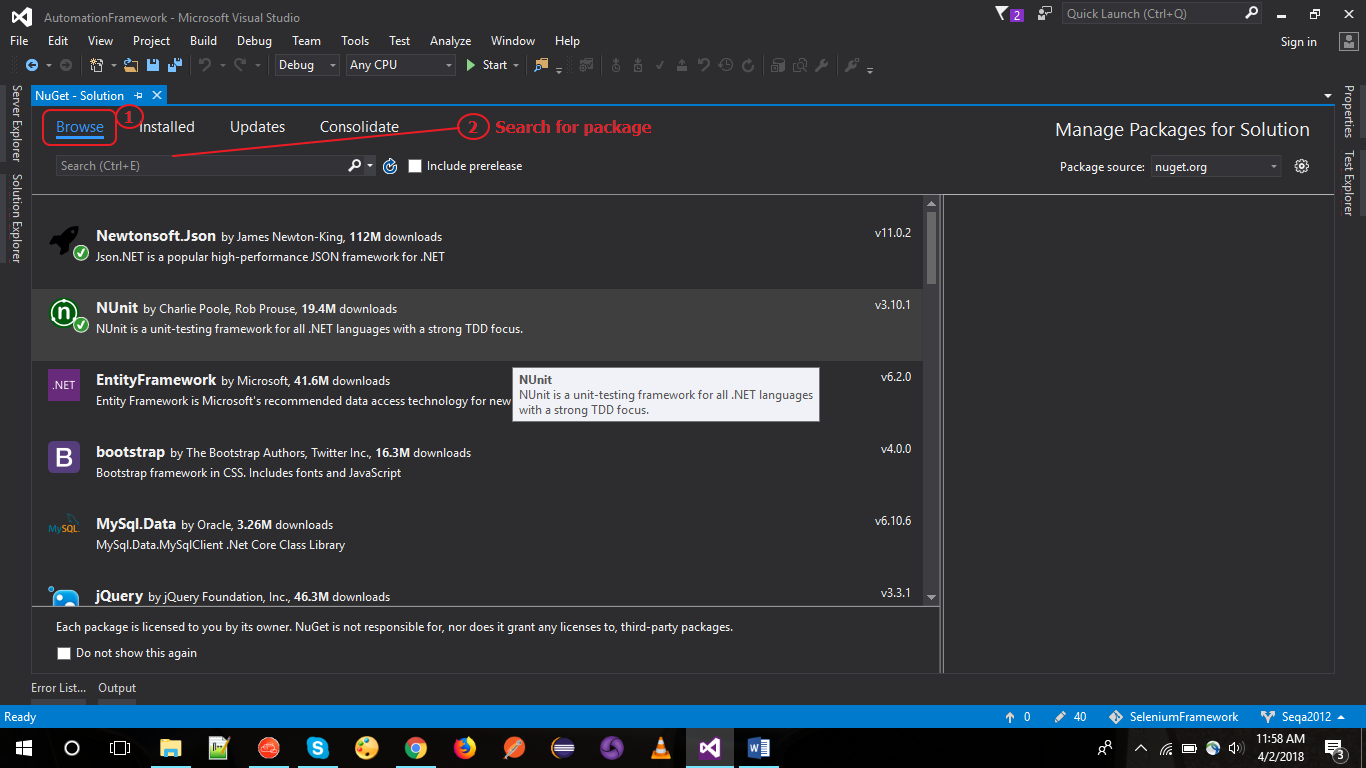
NuGet packages once installed are bound with the Solution itself and are not required to be installed by users in their solution again.

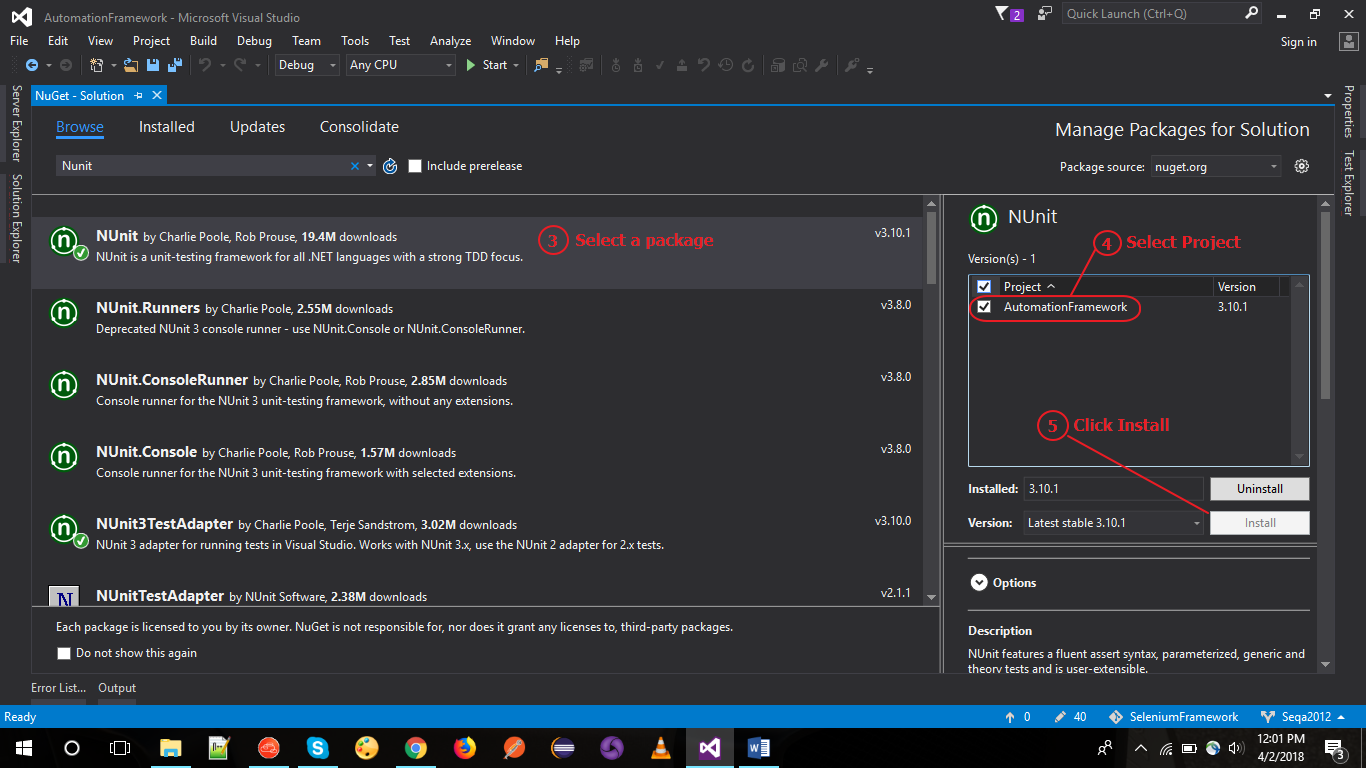
## Steps to Install

1. Menu Bar > Tools > NuGet Package manager
2. Click on “Manage NuGet packages for Solution”
3. Search for required package under the Browse Section
4. Select required package and latest stable version
5. Select Project in which Package needs to be installed
6. Click Install

### Screenshots





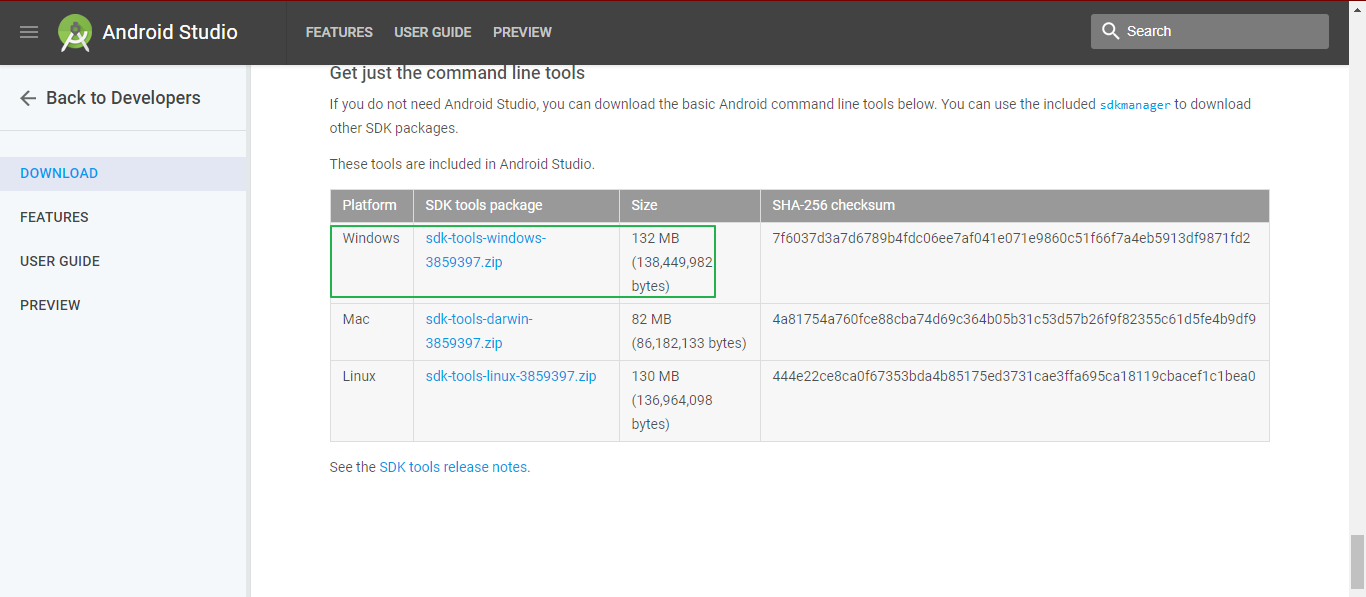


# Appium Setup

## Android automation using Appium

### Android SDK Setup

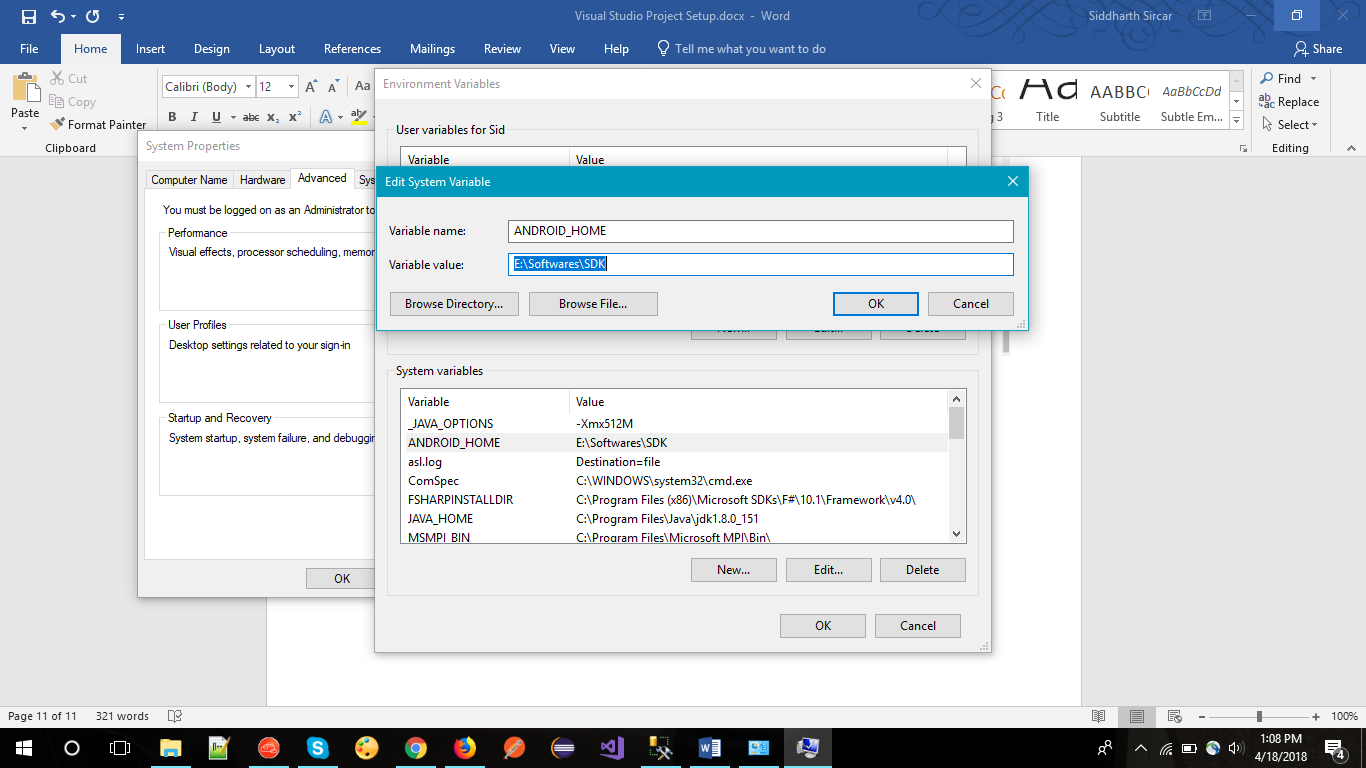
1. Install SDK Tools from <https://developer.android.com/studio/index.html>



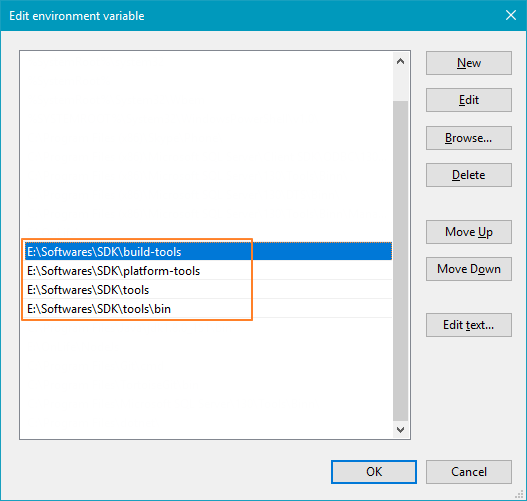
1. Setup required Environment variables
2. Create a new System Variable:

Name: “ANDROID\_HOME”

Value: <SDK Path> *eg.* *E:\Softwares\SDK*



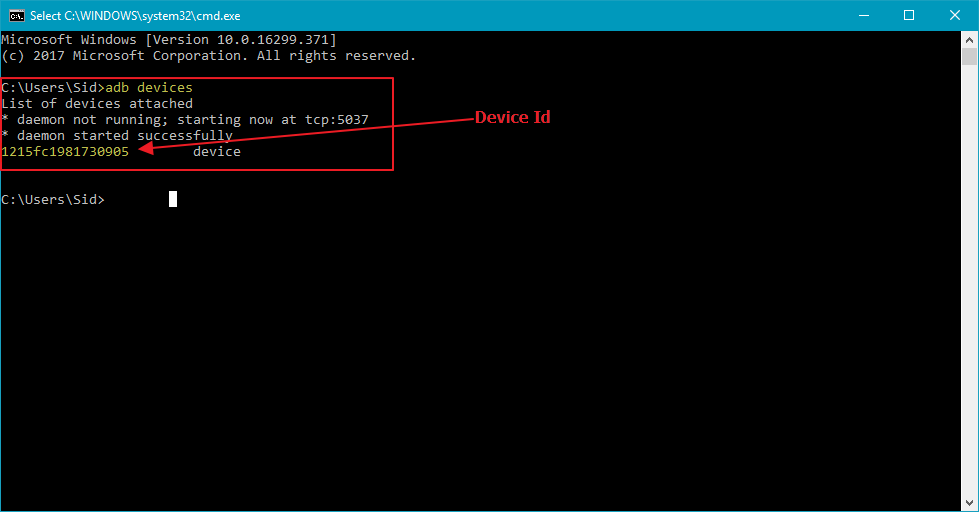
1. In System Variable add following paths to the *‘Path’* values
2. Localpath\SDK\Platform-Tools
3. Localpath\SDK\build-tools
4. Localpath\SDK\tools
5. Localpath\SDK\tools\bin



#### ADB Commands used for device properties

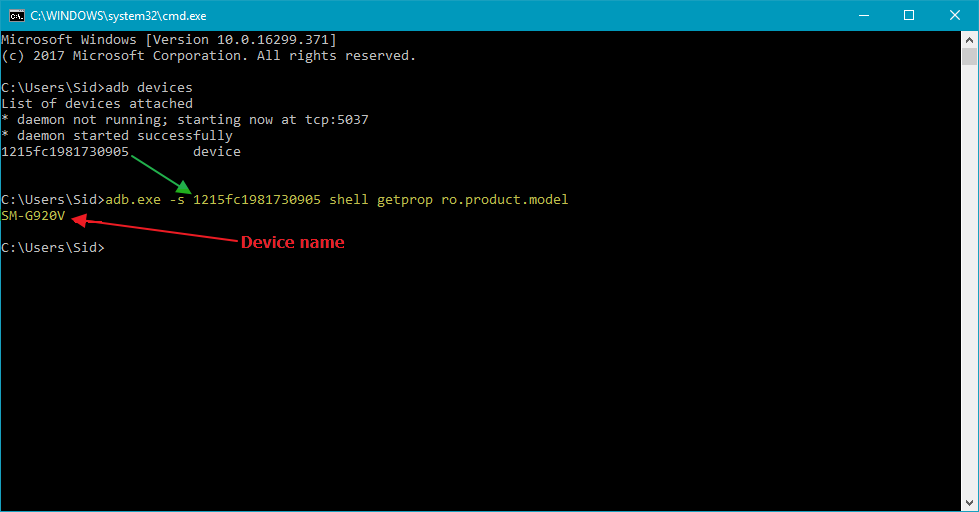
Use command prompt for the following commands.

1. **adb devices:** lists the *deviceids* of all the connected android devices

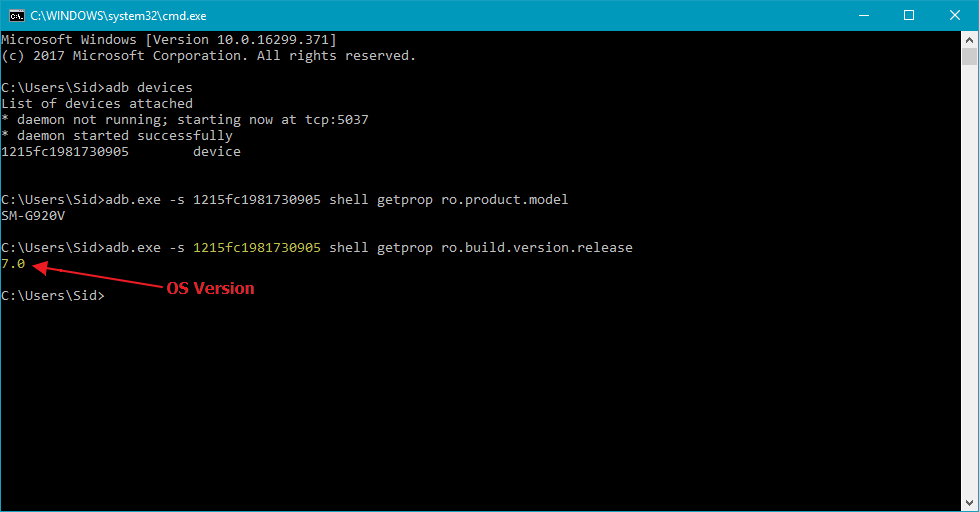


1. **adb.exe -s <deviceid> shell getprop ro.product.model:** Returns the device name using the *deviceid* from the above command.

Device name passed as a Capability for Appium to specify on which device automation is to be executed.



1. **adb.exe -s <deviceid> shell getprop ro.build.version.release:** Returns the OS version of the connected device

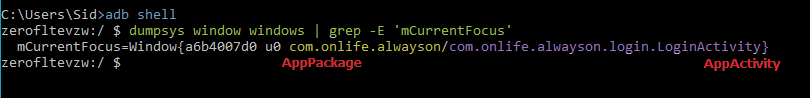


1. **To find the App Activity and App Package of the app in focus:**

***Command:***

adb shell

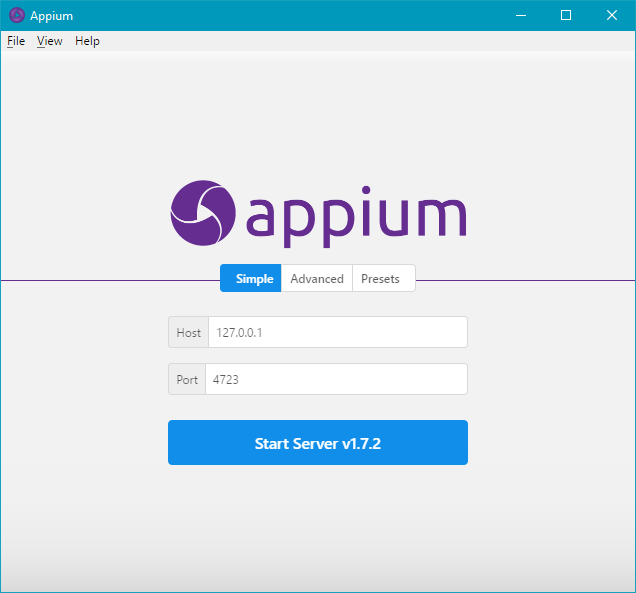
dumpsys window windows | grep -E ‘mCurrentFocus’



### Appium Desktop App Setup

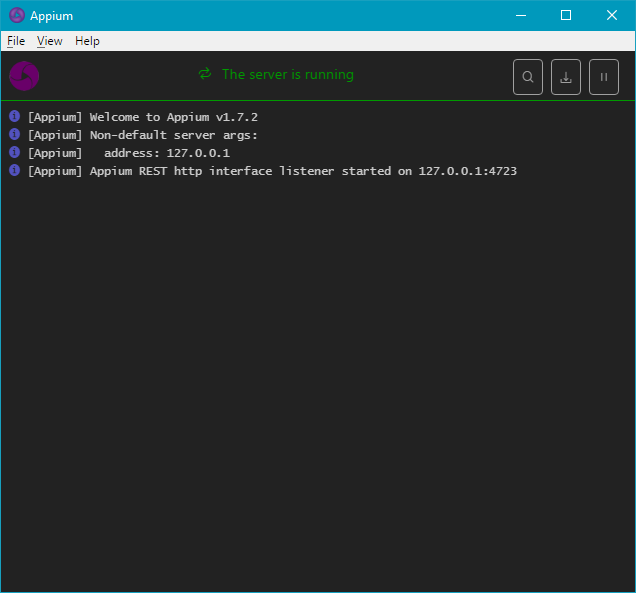
1. Install Appium Desktop client from <http://appium.io/downloads.html>
2. Launch application from location:

*C:\Users\<USER FOLDER>\AppData\Local\appium-desktop*



1. Specify Host Name and Port at which the automation script will communicate with Appium server and click Start Server.

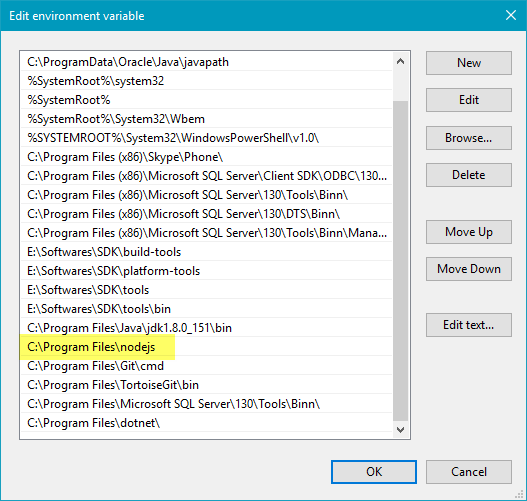
*The mentioned ip and port is configured in the config file in the project. Sample URI:* <http://127.0.0.1:4723/wd/hub>



### Appium Non-Gui Setup

#### Setup Nodejs

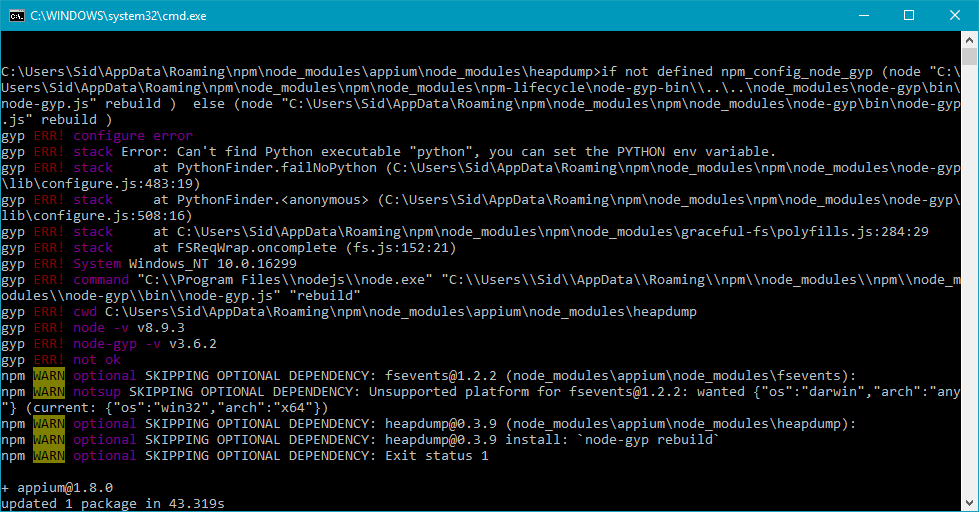
1. Download Node.js from <https://nodejs.org/en/download/>
2. Add NodeJS file path to environment variables



1. Open Command Prompt and execute command ‘*npm’* to verify NodeJS has been setup

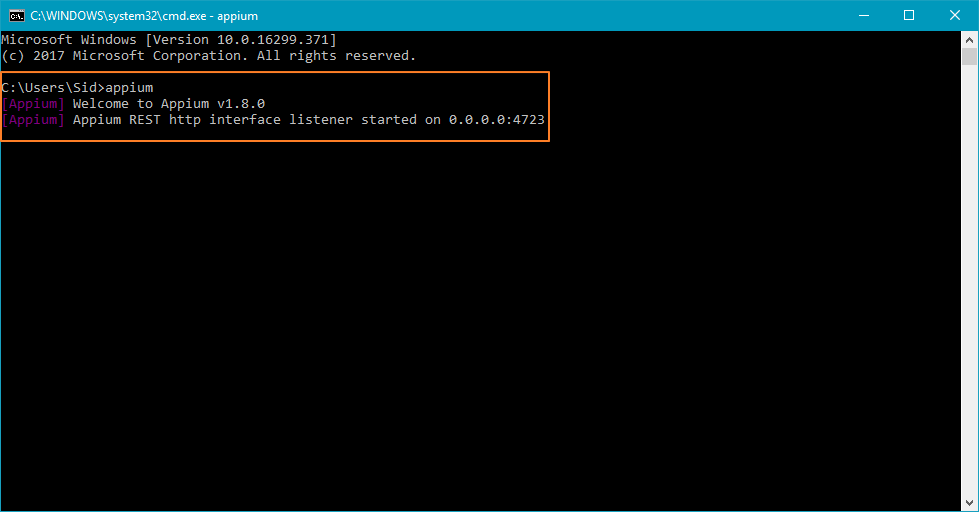
#### Install Appium

1. In Command Prompt, execute: ‘*npm install -g appium’*
2. NodeJS will install the Appium node modules and you will get the following screen after installation is complete



1. In Command Prompt, execute command: *‘appium’*

If successfully installed, appium server would be started with default IP as 0.0.0.0 and be listening at Port 4723



## iOS Automation using Appium

### Appium Setup

1. Install XCODE (Appium uses XCode services to control UI) Version
2. Sign in to XCODE using following credentials:

**Username/Email Id:** [skytesters@icloud.com](mailto:skytesters@icloud.com)

**Password:** Venture7@QA

1. Download Provisioning Profile and Certificate from Apple developer (developer.apple.com) site for the app (Murali can help with the required details)
2. Figure out UDID of the test device from iTunes
3. Download iOS AlwaysOn .ipa file from HockeyApp on MAC (if app to be passed externally)
4. If App present on device, update AppBundle Id key in Config file with Test App's bundle id (Murali can help with Bundle Id)
5. Set Platform to Mobile and OS to iOS in Code config file
6. Make sure Device UDID for Test device is uncommented
7. Get XCodeorgId from App developer Website > Account > Membership > Team ID is XCodeorgId

#### All following packages to be installed using Terminal. Commands written alongside the package names

1. Install Homebrew (To install all missing packages)

**Install command:** */usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"*

1. Install NodeJS and npm using homebrew

**Install command:** *brew install node*

1. Install Appium non-Gui version using npm

**Install Command:** *npm install -g appium*

1. Install Appium doctor using npm

**Install Command:** *npm install appium-doctor -g*

1. Run appium doctor to verify if all required packages for appium are installed

I**nstall Command:** *appium-doctor –ios*

if not, appium doctor will notify

1. Install Carthage if not found on system after appium doctor check, it will be mentioned as manual fixes along with command to install
2. Install libimobiledevice using homebrew if testing on real devices

**Install Command:** *brew install libimobiledevice --HEAD*

1. Install ios-deploy using npm

**Install Command:** *npm install -g ios-deploy*

### Test Execution Process

1. Identify MAC ip address and start Appium on that IP and user defined port

**Start Command:** *Appium -a <MAC Address> -p <Custom Port>*

1. Set Appium Address and Port, used to start appium server in MAC, in config file
2. Start Automation Script from windows.