*Mac Setup (for IOS Testing)*

*note: only applicable for V&V Tx Team (not for building app)*

Contents

[Requirements 1](#_Toc524596377)

[Short Intro to Using Mac 1](#_Toc524596378)

[Installing Xcode 1](#_Toc524596379)

[Installing Appium 3](#_Toc524596380)

[Installing App on Device 6](#_Toc524596381)

[Running Tests 8](#_Toc524596382)

[Common Issues 8](#_Toc524596383)

[Resources 9](#_Toc524596384)

# Requirements

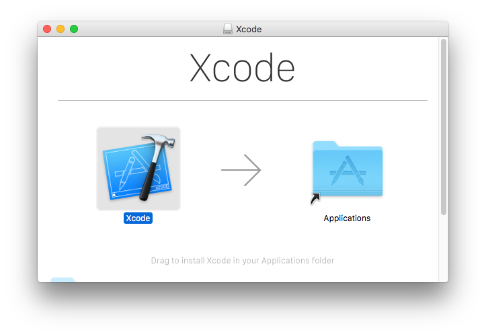
1. Mac Mini
2. iOS Device
3. Apple ID

# Short Intro to Using Mac

1. Opening terminal
   1. Finder  > Utilities > Terminal
2. Accessing Z drive
   1. Finder > Go > Connect to Server > smb://<username>@dexcom.dexcominc.com/dexcomdata/eng6340/softwaretesting
   2. Enter your Dexcom credentials

# Installing Xcode

1. Check the OS version of the Mac
   1. Menu bar > Apple icon > About this Mac
2. Check the iOS version of the device
   1. Settings > General > About > Version
3. Check the compatibility table
   1. <https://en.wikipedia.org/wiki/Xcode#Xcode_7.0_-_10.x_(since_Free_On-Device_Development)>
   2. Note these columns:
      1. Version
      2. Min macOS to run
      3. iOS SDK included
   3. Example: OS (10.13.4) and iOS (10.3.1)
      1. OS version 10.13.4 means that Xcode version should be 9.4.1 –
      2. iOS version 10.3.1 means that Xcode version should be 9.0 +
      3. So, Xcode versions 9.0 to 9.4.1 are compatible for testing
      4. Note: try to install the latest version that the mac mini can support
   4. If there are no Xcode versions compatible (choose one)
      1. Get another device with a lower iOS version
      2. Upgrade the mac mini’s OS (ask IT)
      3. Get another mac mini with a higher OS version
4. Download the Xcode installer with the correct version
   1. <https://developer.apple.com/download/more>
5. Install XCode (choose one)
   1. GUI method
      1. DMG (disc image) file
         1. Mount the disc by double clicking the file
         2. In the new window, drag the Xcode app to the Applications folder (see picture below)

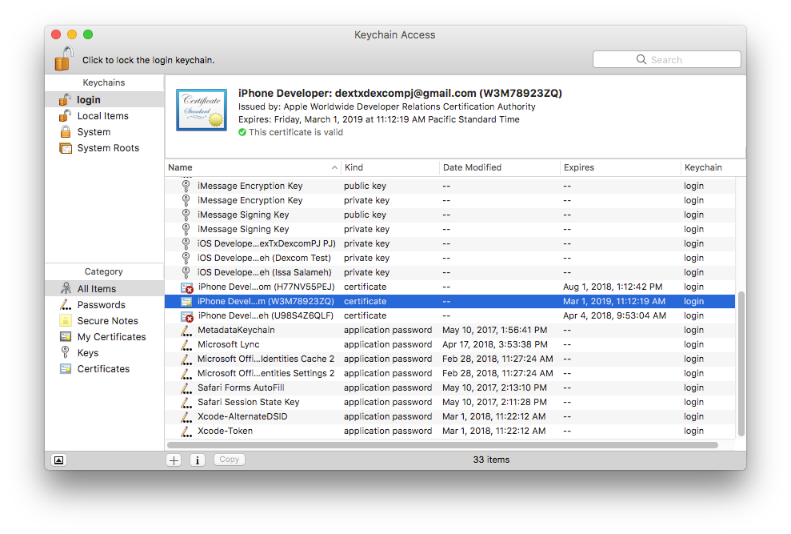


* + - 1. Unmount the disc by left clicking the icon on the desktop and selecting Eject
    1. XIP (signed zip) file
       1. Unzip the package by double clicking the file
       2. Copy the file to the Applications folder (using Finder)
       3. Optional: delete the file
  1. Terminal method
     1. DMG file
        1. hdiutil attach <path\_to\_dmg\_file>
        2. sudo cp -r /Volumes/Xcode/Xcode.app /Applications
           1. Note: there should be no forward slash after Applications
        3. hdiutil unmount /Volumes/Xcode/
     2. XIP file
        1. xip -x <path\_to\_xip\_file>
        2. sudo mv XCode.app /Applications

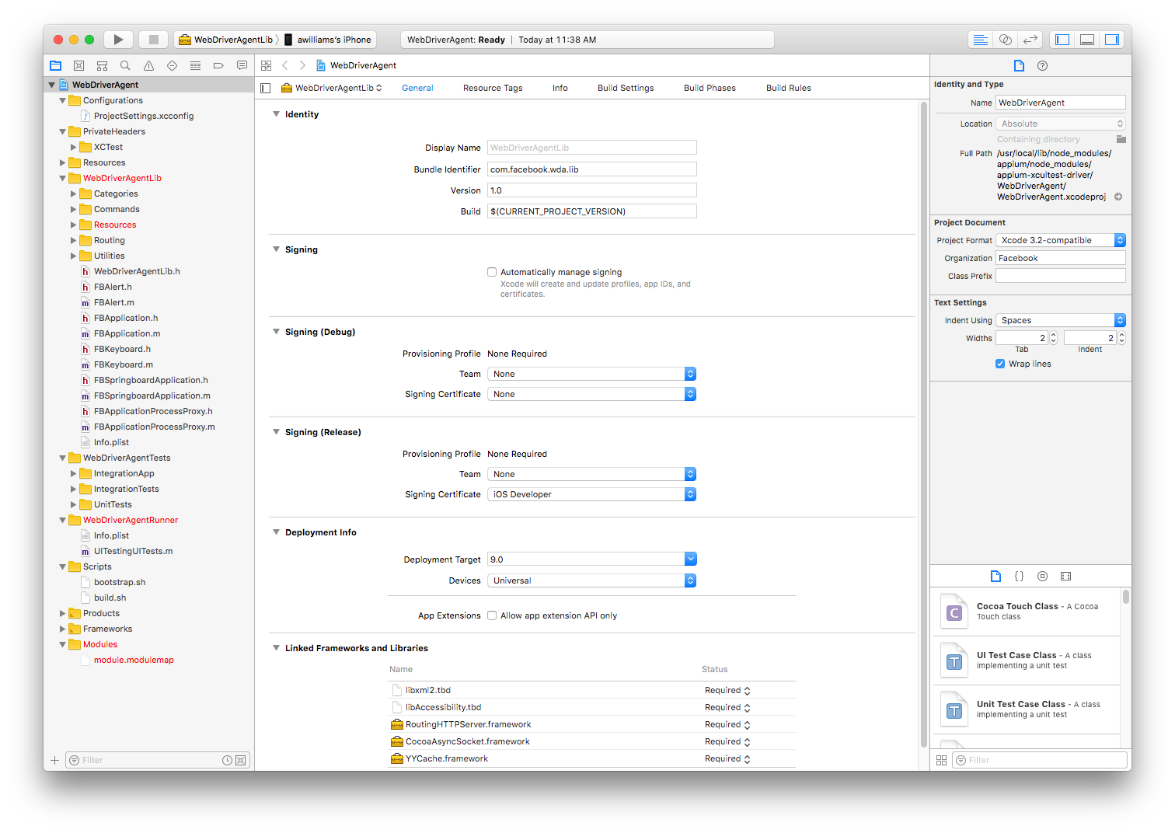
1. Configure Xcode
   1. Open Terminal
   2. Run the commands:
      1. sudo xcode-select -s /Applications/Xcode.app/Contents/Developer
         1. This points the executable to the installer Xcode (similar to adding to the environment path variable in Windows)
      2. sudo xcodebuild -license accept

# Installing Appium

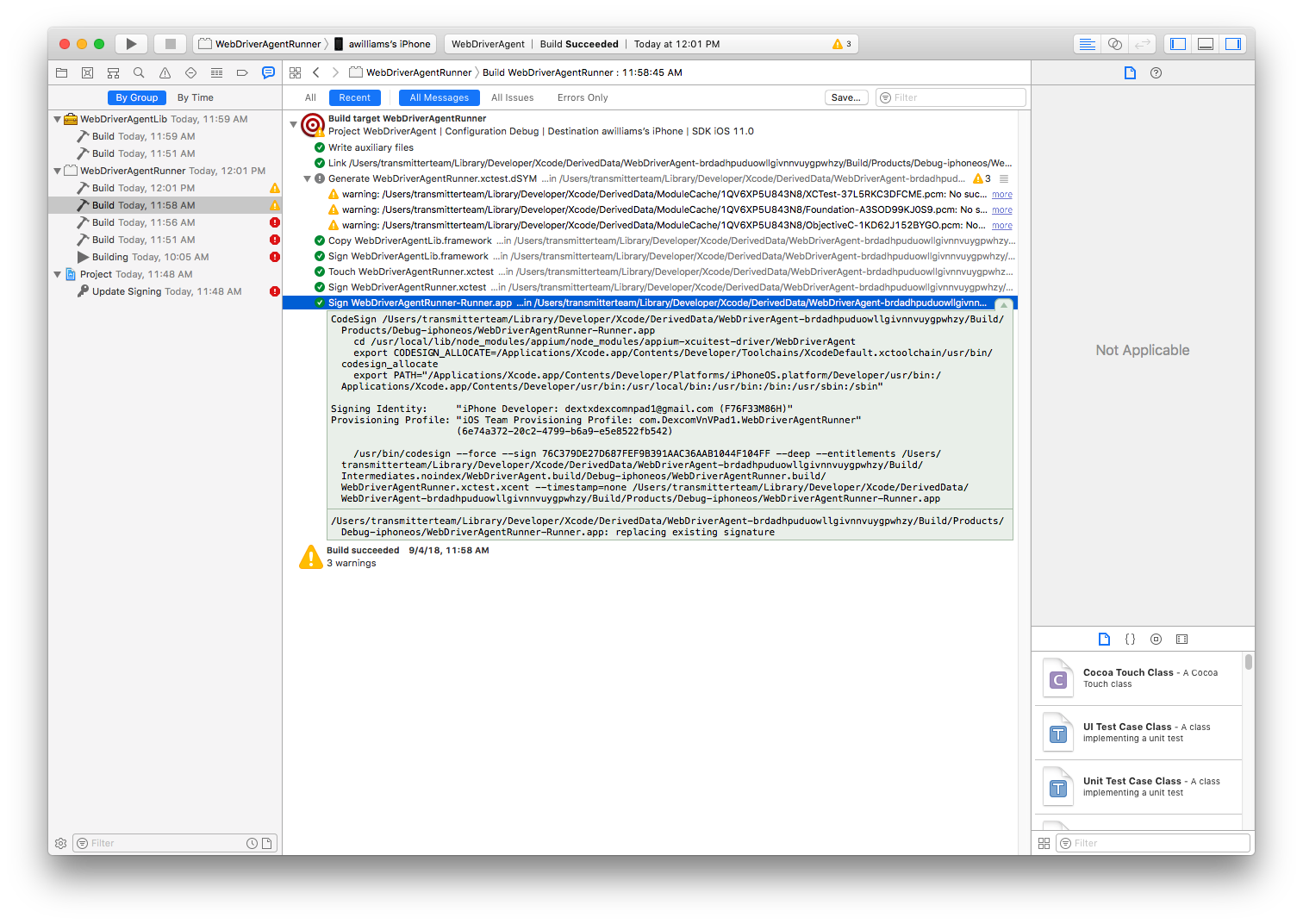
1. Open Terminal
2. Install Appium and its dependencies (run the following commands)
   1. echo | ruby -e “$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/master/install>)”
      1. This installs Homebrew, an installation manager (used to install other dependencies easily)
   2. brew cask install java
   3. brew install node
   4. npm install -g appium
   5. brew install carthage
   6. npm install --global --unsafe-perm=true ios-deploy
      1. Note: if the OS version is below 10.11, the unsafe-perm flag is optional
   7. brew install ios-webkit-debug-proxy
   8. ln -s /usr/local/Cellar/ios-webkit-debug-proxy/1.8.3/bin/ios\_webkit\_debug\_proxy /usr/local/bin/ios-webkit-debug-proxy
      1. This links the command ios-webkit-debug-proxy to the installed package (uses hyphens instead of underscores)
3. Configure Appium’s WebDriverAgent (WDA) project (run the following commands)
   1. cd /usr/local/lib/node\_modules/appium/node\_modules/appium-xcuitest-driver/WebDriverAgent
   2. ./Scripts/bootstrap.sh -d
   3. mkdir –p Resources/WebDriverAgent.bundle
4. Add the Apple ID to XCode
   1. Open Xcode
   2. Sign into the Apple ID account
      1. Menu bar > Xcode > Preferences > Accounts > Plus sign
      2. Note: an Apple ID can only be used up to a max of 5 mac minis
5. Build the WDA project (choose one)
   1. Terminal method (only available for Xcode version 9.0+)
      1. Open the Terminal
      2. cd /usr/local/lib/node\_modules/appium/node\_modules/appium-xcuitest-driver/WebDriverAgent
      3. xcodebuild -project WebDriverAgent.xcodeproj -scheme WebDriverAgentRunner -configuration Debug -allowProvisioningUpdates PRODUCT\_BUNDLE\_IDENTIFIER=<bundle\_identifier> DEVELOPMENT\_TEAM=<10\_character\_dev\_team\_id> CODE\_SIGN\_STYLE=”automatic”
         1. bundle\_identifier = “com.<apple\_id>.WebDriverAgentRunner”
         2. 10\_character\_dev\_team\_id can be found in Finder > Applications > Utilities > Keychain Access > iPhone Developer Certificate (see picture below)



* + 1. The full path to the WDARunner app should be /usr/local/lib/node\_modules/appium/node\_modules/appium-xcuitest-driver/WebDriverAgent/build/Release-iphoneos/WebDriverAgentRunner-Runner.app
       1. If not, check the logs (should be listed in the last few line)
  1. GUI method
     1. Open the WDA project
        1. Open another project… > Favorites > macOS HD
        2. Show hidden files by pressing Command + Shift + .
        3. Navigate to /usr/local/lib/node\_modules/appium/node\_modules/appium-xcuitest-driver/WebDriverAgent
        4. Open the project WebDriverAgent.xcodeproj
     2. In the left panel, select the WebDriverAgent project (first item) (see picture below)



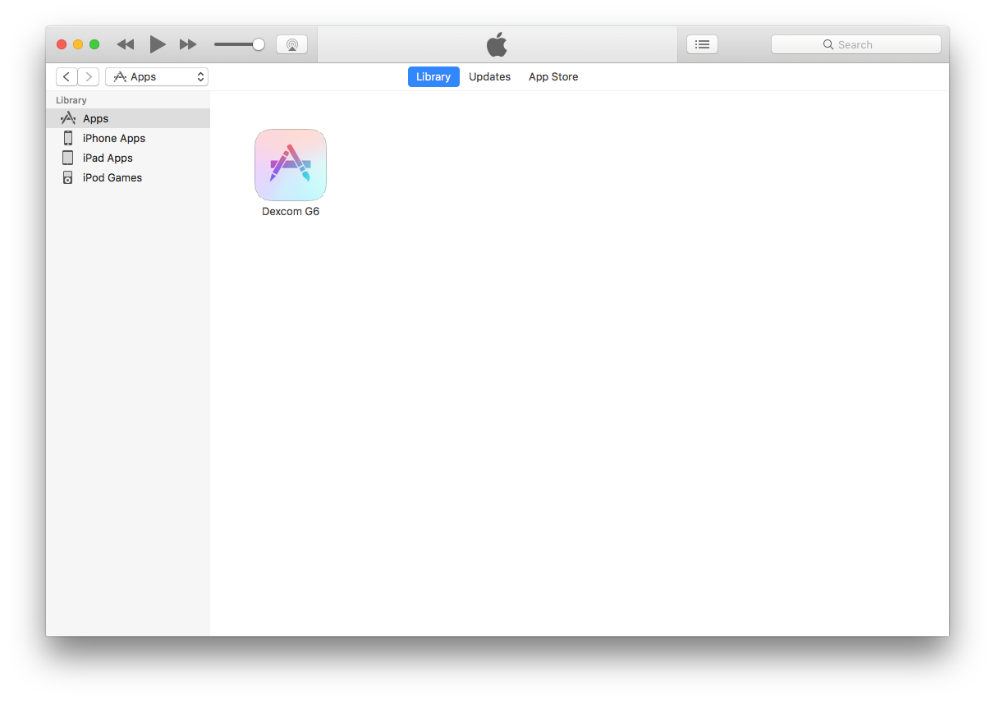
* + 1. Change the following:
       1. Bundle Identifier – replace facebook with anything else (like the apple ID)
       2. Select Automatically manage signing
          1. This will combine the debug and release signing sections
       3. Team – select your Apple ID
    2. Do step iii. for WebDriverAgentRunner (select in upper left red box in picture)
       1. Note: Bundle Identifier will be in Build Settings instead of General and will be named Product Bundle Identifier
    3. Build the WDA Runner
       1. Set the target (in the upper left corner of the window, beside the phone icon) as WebDriverAgentRunner
       2. Click the play button
    4. Verify the build is successful (see picture below)



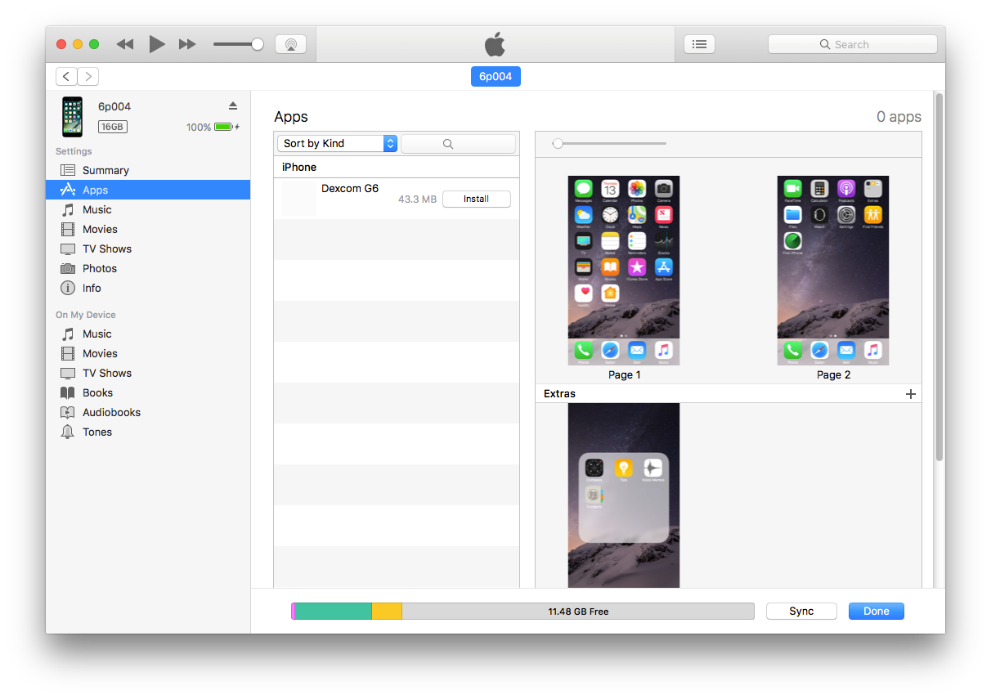
* + - 1. Open the logs (chat bubble icon)
      2. Select All Messages
      3. Select WebDriverAgentRunner > Build in the left panel
      4. Verify that the last line “Sign WebDriverAgentRunner.app …“ was successful
      5. Expand the log and find the full path to the app (will be used in the next step)

# Installing App on Device

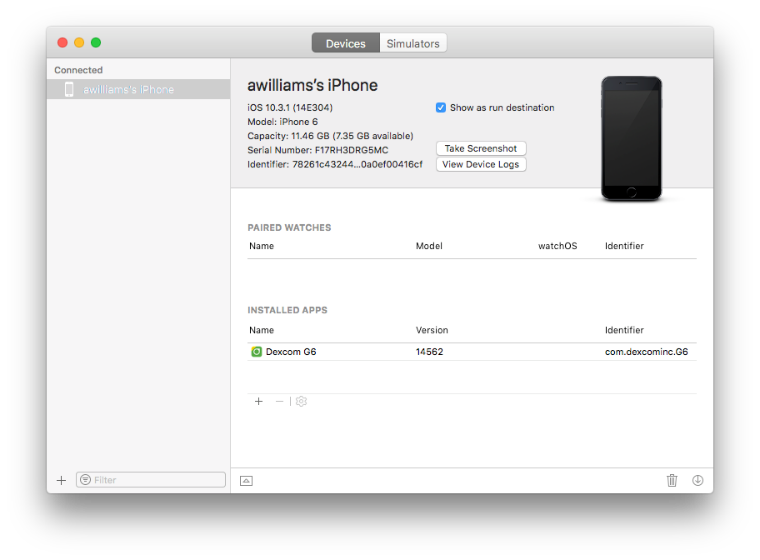
1. Connect the device to the Mac
2. In the phone, trust the mac
3. Install the app (choose one)
   1. Using iTunes
      1. Select apps (see picture below)



* + 1. Find the app using Finder
    2. Drag the app into the iTunes window
    3. Select the phone icon > Apps (you should see the picture below)



* + 1. Select install and sync/apply
  1. Using Xcode
     1. Menu bar > Window > Devices and Simulators (or just Devices for older versions)
     2. Select the device
     3. Find the app using Finder
     4. Drag the app into the Installed Apps section to install the app (see picture below)



1. Trust the apps
   1. Device > Settings > General > Device Management
   2. For both the Dexcom and WDARunner app, verify the app

# Running Tests

* + - 1. Install Dexcom App on the device
         1. Download from: <https://mobileapps.dexcom.com/estore_v3/>
         2. Ask login credentials from Misha
      2. Run Appium on the Mac Mini
         1. Terminal command: /usr/local/bin/node <appium\_folder> --address <ip\_address> --port 4723 --session-override --automation-name XCUITest
         2. Example: /usr/local/bin/node /usr/local/lib/node\_modules/appium --address 10.3.103.33 --port 4723 --session-override --automation-name XCUITest
      3. Run Mock Server on the PC

# Common Issues

1. Test failing with “code 65”
   1. Cause 1: WDARunner app is not trusted in the phone
      1. Manually place the WDARunner app in the phone
      2. See “Installing Appium” step 5 for the path to the app
      3. See “Installing App on Device”
   2. Cause 2: WDARunner was not codesigned properly
      1. Usually happens when using the terminal method when building the app
      2. Try using the GUI method
      3. Also check the “Resources” for Appium documentation

# Resources

1. <http://appium.io/docs/en/drivers/ios-xcuitest-real-devices/#appium-xcuitest-driver-real-device-setup>

How to get udid:

idevice\_id -l