**Appium Mobile Test Automation Setup (Android + iOS)**

**Steps to be followed:**

This YouTube channel has great project setup tutorials:

https://www.youtube.com/channel/UC1lZyaCk\_seAQfpPxfHm8NA/videos

1. Install JDK 8 or above

Set up path variable

vi ~/.bash\_profile [If the paths file doesn’t exist, then create the file and put the details.]

export JAVA\_HOME=$(/usr/libexec/java\_home)

2. Install Eclipse IDE or IntelliJ IDEA CE

3. Install Homebrew

RUN 🡪 /usr/bin/ruby -e "$(curl -fsSL [https://raw.githubusercontent.com/Homebrew/install/master/install](about:blank))"

4. Install Maven

🡪 brew install maven

Set up path variable

open .bash\_profile

export M2\_HOME=/usr/local/Cellar/maven/3.8.1

export M2=$M2\_HOME/bin

export PATH=$M2:$PATH

4a. When Importing/copying a project in Android

Go to the pom.xml file in IntelliJ

Right click pom.xml -> Maven -> Reload Project

That should rebuild and import all the Maven dependicies

5. Install Android Studio

Set up path variable

open .bash\_profile

export ANDROID\_HOME=/Users/{username}/Library/Android/sdk

export PATH=$PATH:~/Library/Android/sdk/platform-tools

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

6. XCode

Download & Install latest version of XCode in Application directory.

RUN below commands. This will install the necessary dependencies after XCode is successfully installed.

a. brew install ideviceinstaller

b. brew install carthage

c. brew install node

d. npm install -g appium@1.18.3 [If you get error, try -> sudo npm install -g appium@1.18.3 --unsafe-perm=true --allow-root]

RUN npm install -g npm@6, if any issue is observed while appium installation. Thereafter, run command a to d.

e. npm install -g ios-deploy

If you get error while executing npm install -g ios-deploy, then run -> sudo xcode-select -s /Applications/Xcode.app/Contents/Developer

Then run below command

f. npm install -g ios-deploy

g. npm install -g deviceconsole

h. gem install xcpretty

If you get error while executing gem install xcpretty, then run -> sudo gem install xcpretty

i. brew install libimobiledevice

7. Add device support files for XCode

a. Close XCode

b. Download the device support files for required iOS from https://github.com/filsv/iPhoneOSDeviceSupport

c. Unzip it & copy it to /Applications/Xcode.app/Contents/Developer/Platforms/iPhoneOS.platform/DeviceSupport/

**NOTE – OpenCV and Configure OCR and not required for this project**

8. Install OpenCV (only needed if you want to make use of image comparison)

a. brew install cmake

b. npm i -g opencv4nodejs [If you get error, try -> sudo npm i -g opencv4nodejs --unsafe-perm=true --allow-root]

brew update

brew install opencv@4

npm link opencv4nodejs

/usr/local/lib/node\_modules/appium/node\_modules/opencv4nodejs -> /opt/homebrew/lib/node\_modules/opencv4nodejs

c. Now you will find the opencv4nodejs in /usr/local/lib/node\_modules

d. Navigate to /usr/local/lib/node\_modules/appium/node\_modules and RUN npm link opencv4nodejs [This way opencv4nodejs is linked inside appium/node\_module]

**NOTE – OpenCV and Configure OCR and not required for this project**

9. Configure OCR (only needed if you need to make use of google cloud vision api such as getting text out of an image)

a. Create new OCR project in GCP, download the created ocr.json file

1. Open url https://cloud.google.com/vision/docs/setup -> scroll down to ‘Create a project’ section

2. Click on ‘Go to project selector page’. A new window will open which will have all your project details.

3. Click on ‘CREATE PROJECT’ button present on top right corner.

4. Enter the ‘Project name’ & Organization details if any and click on CREATE

5. Now open the created project by selecting it from the list present on top left

6. Go to ‘APIs & Services’ by clicking on the option list present in the left and select ‘Dashboard’

7. Click on ‘ENABLE APIS AND SERVICES” present on top & search for ‘Cloud Vision API’ and select it

8. Click on ‘ENABLE’ button. Now you will get redirected to the Cloud Vision API page.

9. Click on ‘CREATE CREDENTIALS’ button present on top left

10. Select an API -> Cloud Vision API.

i. Select radio button for ‘Application data’.

ii. Select radio button for ‘Yes, I’m using one or more’

iii. Click on NEXT button

iv. Click on DONE button, or, optionally you can create a ‘service account’ by clicking on a required link present.

v. Enter valid ‘Service account name’ and ‘Service account description’ -> click on ‘CREATE AND CONTINUE’ button

vi. Select a role, in this case I’m selecting role as ‘Owner’. (select role according to requirement) -> click on ‘CONTINUE’ button.

vii. Click on DONE button.

viii. Now click and open the created account. Click to open ‘KEYS’ section.

ix. Click on ‘ADD KEY’ and select ‘Create new key’

x. Select the file type as ‘JSON’ and click on ‘CREATE’ button

xi. Private key (rename it to ocr.json) will get download to the machine. Use the same JSON file to authenticate as and when required.

b. RUN below command in the terminal.

launchctl setenv GOOGLE\_APPLICATION\_CREDENTIALS <Absolute path to the attached JSON>

example -> launchctl setenv GOOGLE\_APPLICATION\_CREDENTIALS /Users/{username}/Documents/OCR/ocr.json

c. Set environment variable

export GOOGLE\_APPLICATION\_CREDENTIALS=/Users/ username}/Documents/OCR/ocr.json

10. iPhone settings to be done before running automation

a. Settings > Developer > Enable UI Automation (If Developer option does not appear, open XCode and connect the iPhone to the Mac)

b. Settings > Safari > Advanced > Enable JavaScript, WebInspector

c. Settings > Safari > Advanced > Experimental WebKit Features > Enable VisualViewPort API (In some devices, it is also named as Viewport Fit)

d. Settings > Display & Brightness > Enable Auto-Lock > Set to "Never"

e. Settings > Display & Brightness > Set Display Zoom View to ‘Standard’

f. Disable notification for all other unused apps. Allow notification for only intended app which needs to be tested.

g. iPhone language should be set to English US

h. Check for Wi-Fi and Cellular data

i. Disable auto OS updates

11. Android settings to be done before running automation

a. Settings > Make sure screen is unlocked

b. Settings > Lock Screen and Security > Change Screen Lock type to None

c. Settings > About Device > Keep tapping on Build number section 7 times till it shows Developer mode is turned on.

d. System > Developer Options > Enable USB debugging option

e. When you are using the device for first time on a mac connect the device to mac/windows and select “Always allow from this computer” for USB debugging and click “OK” f. System > Developer Options > Enable Stay Awake option

g. Make sure the default browser on Android phone is set to Chrome (Go to settings > Select Apps, Select Default App (by clicking 3 dots on top right) > Select Browser App as Chrome.

h. Settings > Disable keyboard suggestions on device

i. Disable notification for all other unused apps. Allow notification for only intended app which needs to be tested.

j. In order to clear device cache - Clear Chrome Browser Cache (Settings > App > Chrome )

k. Check for Wi-Fi and Cellular data

l. Disable auto OS updates

12. Make entry of new iOS device in developer account

a. Open url, https://developer.apple.com/account/ios/profile/, login in using valid credentials.

b. Get the UDIDs (idevice\_id -l) of the iPhones for which entry need to be made

c. In order to add new device, navigate to Account > Certificate, IDs & Profiles [Join the Apple Developer Program, if said option is not available in left under ‘Program Resources’] > Click on ‘Devices’ > Click on ‘+’ icon > Under ‘Register a New Device’ section, enter ‘Platform’, ‘Device Name’ & ‘Device ID (UDID)’ > Click on ‘Continue’ button. d. Under ‘Profiles’ > select a profile [create an iOS (Platform) Development (Type) profile, if no profile found] > Click Edit > look for your device under ‘Devices’ > Select your device > Click on ‘SAVE’.

13. Latest developer profile certificate for iOS

a. Download the created provisioning profile certificate. (Generated in Step 12)

b. RUN -> cd /usr/local/lib/node\_modules/appium/node\_modules/appium-webdriveragent/

c. RUN -> mkdir -p Resources/WebDriverAgent.bundle [If observing any permission issue, try -> sudo mkdir -p Resources/WebDriverAgent.bundle]

d. RUN -> ./Scripts/bootstrap.sh -d [If observing any permission issue, try -> sudo ./Scripts/bootstrap.sh -d]

e. RUN -> open WebDriverAgent.xcodeproj

f. For both, WebDriverAgentLib and WebDriverAgentRunner targets, select "Automatically manage signing" in "Signing & Capabilities" tab, then select your Development Team. g. This should also auto select Signing Certificate. The outcome should look as shown below:

i. Follow instruction, https://github.com/appium/appium-xcuitest-driver/blob/master/docs/real-device-config.md

ii. Do XCode Product Clean and Build and make sure there is no error.

Note : Fix error related to 'CocoaAsyncSocket.framework' & YYCache.framework'.

Select Project (WebDriverAgent) → Build Settings → Search for ‘Validate’ in search-box → Build Options (iOS) → Set Validate Workspace option to ‘Yes’

14. XCode Settings

Go to WebDriverAgent project.

a. Click on ‘WebDriverAgentRunner’ under Targets

1. Under ‘Signing & Capabilities’ > select ‘Automatically manage signing’

2. Select a ‘Team’

3. Select ‘Signing Certificate’ as ‘Development’

b. Click on ‘UnitTests’ under Targets

1. Under ‘Signing & Capabilities’ > select ‘Automatically manage signing’

2. Select a ‘Team’

3. Select ‘Signing Certificate’ as ‘Development’

c. Click on ‘WebDriverAgentLib’ under Targets

1. Under ‘Signing & Capabilities’ > select ‘Automatically manage signing’

2. Select a ‘Team’

3. Select ‘Signing Certificate’ as ‘Development’

d. Click on ‘IntegrationApp’ under Targets

1. Under ‘Signing & Capabilities’ > select ‘Automatically manage signing’

2. Select a ‘Team’

e. Do XCode Product Clean and Build and make sure there is no error.

15. Xcode 14 and iOS version 16 on the Simulator

Every Xcode release has the possibility to break Appium. The Appium community will release fixes. Xcode 14 broke the Appium WebDriverAgentRunner. Here is documentation of those fixes:

If you encounter this build error: Cannot link directly with dylib/framework

<https://github.com/abdulowork/LinkWithXCTAutomationSupportXcode14>

Run these commands:

npm install -g appium@2.0.0-beta.44

appium plugin install execute-driver

appium -pa /wd/hub --use-plugins execute-driver

appium driver install xcuitest

appium driver install uiautomator2

In Xcode, WebDriverAgent project build settings

Allow Multi-Platform Builds = Yes

Build Active Architecture Only = No

If you encounter Xcode Error 65 watch and follow this video:

https://www.youtube.com/watch?v=4E\_Dcu6Ifc0

**The Framework**

Setup:

Ensure Appium is running (open terminal and enter Appium in Command Line

1. Open up IntelliJ and open the following folder: AppiumTestIK

A screenshot of a cell phone

Description automatically generated with medium confidence

2. Navigate to the following folder: src/test

These are where all the automation tests are developed:

A picture containing text, electronics, screenshot, computer

Description automatically generated

3. Navigate to the following folder: src/pageobject

These are where the page object files are located(all the web locators) :

A screenshot of a cell phone

Description automatically generated with low confidence

4. Right click on the Smoke.xml file and select RUN – this will run all the smoke test automation scripts:

A screenshot of a cell phone

Description automatically generated with low confidence

**Manual Test Cases for Jira are located here:**

<https://jira.mheducation.com/projects/REDI?selectedItem=com.thed.zephyr.je%3Azephyr-tests-page#test-cycles-tab>

**Git Repo for the Framework :**

<https://github.mheducation.com/Redi/Redi-QA>

**JAVA Capabilities Settings:**

You may need to change the path settings for the App within the Java files:

cap2.setCapability(MobileCapabilityType.APP, "/Users/jeremycallahan/Documents/Redi-QA/Redi.app");

This setting is in:

Src/test/java in files: iOSBase, AndroidBase

**Xcode Project and Simulator Notes:**

Make sure all the targets are set to the McGraw Hill Higher Education certificate for the WebDriverAgent & Redi projects:

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

In Xcode make sure the simulator device types are the same in the Redi and WebDriverAgent projects:

A picture containing graphical user interface

Description automatically generated

Build the Redi project: Select Product -> Build For -> Running

Graphical user interface, text, application, chat or text message

Description automatically generated

Then go to Product -> Show Build Folder in Finder

Select Products -> Debug-iphonesimulator -> and locate the Redi app icon: A picture containing shape

Description automatically generated

Move this file to your Appium project. In my case it is documents/appiumauto-main

Graphical user interface, application

Description automatically generated

Open InteliJ

Find the file Smoke.xml

Select Run

This will open an iPhone simulator and the tests will start running

**CHROME DRIVER UPDATE**

Get the latest Chrome Driver

Drop it into the Appium-main directory (replace the old file)

If the file is locked run this command: xattr -d com.apple.quarantine chromedriver

**OTHER DOCUMENTATION**

Webdriver API Documentation: <https://webdriver.io/docs/api>

<https://webdriver.io/docs/api/appium/>

Appium Documentation: <https://appium.io/docs/en/about-appium/intro/>

Appium Desktop Setup Videos: <https://www.youtube.com/channel/UC1lZyaCk_seAQfpPxfHm8NA/videos>

Appium Inspector:

Graphical user interface, application

Description automatically generated

**INTEGRATING** APPIUM WITH FASTLANE

<https://docs.fastlane.tools/actions/appium/>

<https://www.raywenderlich.com/233168-fastlane-tutorial-getting-started>