IOS Real Device Automation :

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

brew install libimobiledevice

brew install ios-deploy

Three new Capabilities are needed to run the Tests on real Device :

d.setCapability("xcodeOrgId","xxxxxxxx");

d.setCapability("xcodeSigningId","iPhone Developer");

d.setCapability("udid","xxxxxxxx");

d.setCapability("updateWDABundleId","xxxxxxx");

How to get the xcodeOrgId?

It is TeamID generated by Apple.You can find your Team ID using your developer account. Sign in to developer.apple.com/account, and click Membership in the sidebar.

Your Team ID appears in the Membership Information section under the team name

How to get udid?

https://www.wikihow.com/Obtain-the-Identifier-Number-(UDID)-for-an-iPhone,-iPod-or-iPad

What is updateWDABundleId?

Often we need provisioning profile from Apple to run the apps on Real Devices.And this Provisioning Profile generates Bundle ID

And we need to Sign the App to run on the device using Bundle id.

Connect your Device.

How to Create Provisioning Profile to run APpium Tests?

From Xcode -

Verify Untrusted Developer

Run Build for WebDriverAgent and App under Test with this Provisional Profile.

**Appium (Android & IOS) Framework Development from Scratch**

Appium (Android & IOS) Framework Development from Scratch

Understand the importance of Framework development

• Build Page object Factory pattern to drive the Element locators to the Tests

• Implement Android & IOS Page object repositories for the respective

Platform tests

• Build Actions utility classes for both Android & IOS Apps to store reusable

Mobile Gestures

• Build Generic Utility called Appium Utils Class which have commonly used

methods for Android & IOS.

• Build Test Strategy on how tests should be written in isolation with setup &

Teardown methods

• Data driven testing from json payloads into Test using Jackson Mapper &

TestNG Data provider

• Implement Base Test with Capabilities where all Tests inherit it into their

respective classes

• Methods to dynamically Start & Stop Appium Server before every Test

• Create Properties file to implement Global configuration values into the

Framework

• Build TestNG.xml files to run all tests together with proper class, Suite, Test

conventions inside the xml file

• Introduce Extent reports to get excellent HTML reports of Framework

results

• Implement TestNG listeners to capture screenshots on test failures &

report it back to Extent reports

• Implement TestNG groups to run selected tests based on tags set on Test

level

⚫ Data driven testing from json payloads into Test using Jackson Mapper &

TestNG Data provider

• Implement TestNG groups to run selected tests based on tags set on Test

level

• Implement set of Maven Profiles using TestNG.xml runner files to run tests

with Maven commands

• Implement Run time variable inputs through Maven and adjust the code

accordingly to receive inputs

• Integrate the Full Framework Project with Jenkins CI/CD tool and

parameterize the job with run time variables

• Enhance and integrate the framework with Cloud infrastructure to schedule

tests on cloud devices with Parallel execution

**Appium Framework with Cloud Integration**

**(Browser Stack)**

• Create Account in Browser Stack

• Install Browser Stack plugin in Eclipse

• Convert Project to Browser Stack

compatible

• Understand Yaml File and fill login and App

details

• Run the Tests

• Understand how to switch between local

and cloud execution.