Appium Intro

https://youtu.be/H1Z4BO37_v4

Practice

We need to apply the basic Appium concepts in order to correctly install the tool, along with the UI Automator driver, and start identifying and connecting with Android devices.

To start, we should have Node JS's latest LTS version installed, and then we have to follow this guide to install Appium **http://appium.io/docs/en/2.0/quickstart/install/**

After Appium is installed, follow this guide to install the UI Automator driver: **http://appium.io/docs/en/2.0/quickstart/uiauto2-driver/**

Right after we have our software installed, we have to connect a device to it and make sure the Appium server is correctly detecting it. For this, install the latest Android Studio Version, and create and launch any Android device of your choice using the included Android emulator (most recent devices work better). When the emulator is configured and running, Launch Appium and connect to the emulator using the required capabilities. Once the Appium session is started, we will be ready to develop test automation frameworks for mobile devices.

Additional Resources

Intro to Appium - Appium Documentation https://docs.google.com/document/d/14Ntd_umzmAeVi7AbWweD44QrCahAxCBschElMcN7M_8/edit

Node, NPM intro and Appium setup

https://youtu.be/f24OeqqAV0M

Practice

learnyounode:

1. Open a command prompt (on any location) and install globally learn you node (npm install -g learnyounode).
2. Run command learnyounode to see all available workshop exercises.
3. Start by running the command learnyounode select 0 and solve the Node exercise specified there.
4. Once the exercise is complete and successfully working, continue with the command learnyounode select and the exercise of the following available exercise (as you can see on the example commands when running the learnyounode command). Solve all the available exercises.
5. More information about learnyou node may be found at **https://www.npmjs.com/package/learnyounode**

how-to-npm:

1. Open a command prompt (on any location) and install globally learn you node (npm i -g how-to-npm).
2. Run command how-to-npm to see all available workshop exercises.
3. If an error is displayed specifying "Cannot find module X" install the required module globally using npm i -g moduleName. This may be required before first running how-to-npm or when trying to select any of the exercises listed there.
4. Select the first available npm exercise and solve it.
5. Once the exercise is complete and successfully working, continue solving all the available challenges (may require running again the command how-to-npm and installing additional missing dependencies).

6. More information about how-to-npm may be found at **https://www.npmjs.com/package/how-to-npm**

Additional Resources

Introduction to Node.js

Android Locators

https://youtu.be/EwNlOlfmH74

Practice

We are almost ready to start performing test automation over Android apps. Before that, we must first make sure to understand how to properly identify elements using locators on Android apps.

For this practice first, we need to install the WDIO dummy app on an Android device, you can use an emulator created using Android Studio, or use your own Android device if you have one. The APK is available for download here **https://github.com/webdriverio/native-demo-app/releases** (Make sure to use the .apk file).

After installing the app, we are ready to start identifying the elements. Create the different Page Objects required, including only the class definition and the instantiation of the Mobile Elements or Android Elements needed for the following scenarios:

- Check the navigation on all the categories of the bottom menu bar (Home, Webview, Login, Forms, Swipe).
- Check the successful login.
- Check the correct swipe of the cards on the swipe section.

Additional Resources

https://developer.android.com/training/testing/instrumented-tests/ui-tests https://www.browserstack.com/guide/android-app-automation-testing-tools https://www.browserstack.com/guide/locators-in-appium

IOS locators

https://youtu.be/I5YoenGOwbM

Practice

Follow **these instructions** for this practice.

Mobile Sequence Diagram

https://youtu.be/JkkhhJhp-Ig

Practice

In order to start performing test automation on Android apps, we need to fully understand how to design a scalable, maintainable, and usable framework for our automated tests.

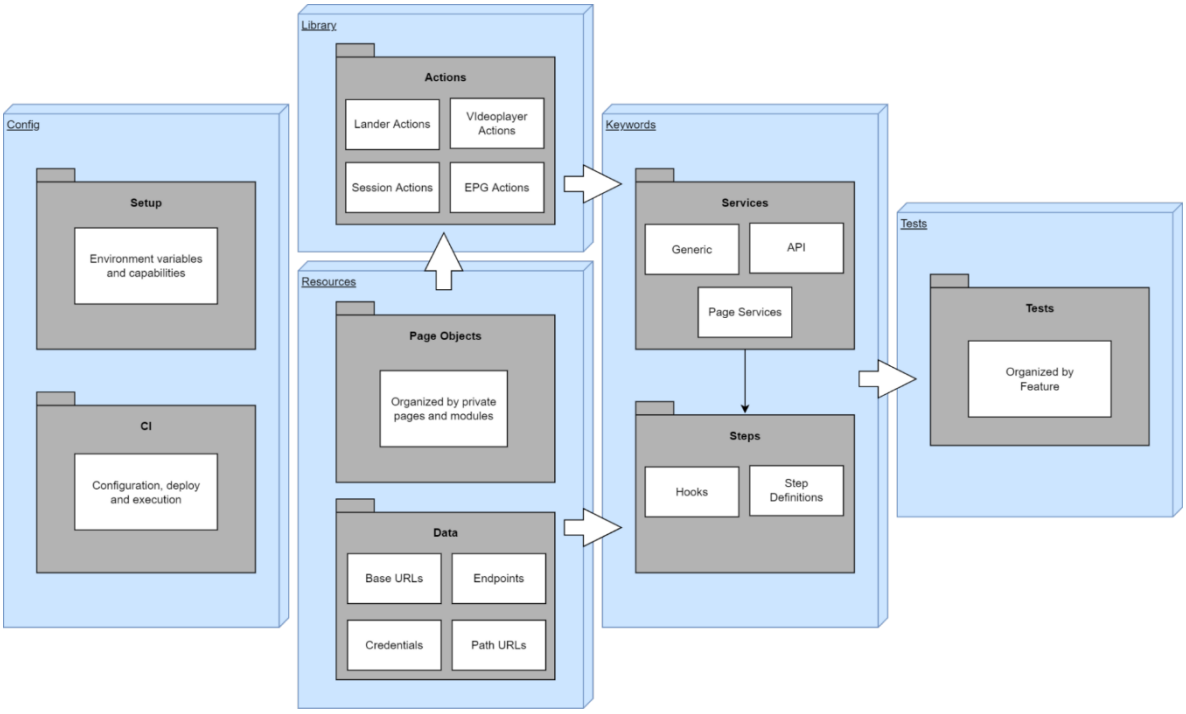
For this practice first, we need to install the WDIO dummy app on an Android device, you can use an emulator created using Android Studio, or use your own Android device if you have one. The APK is available for download here **https://github.com/webdriverio/native-demo-app/releases** (Make sure to use the .apk file). We won't start doing test automation right now, however, we need to use this app in order to understand how it works, how it's organized, and its related business terms.

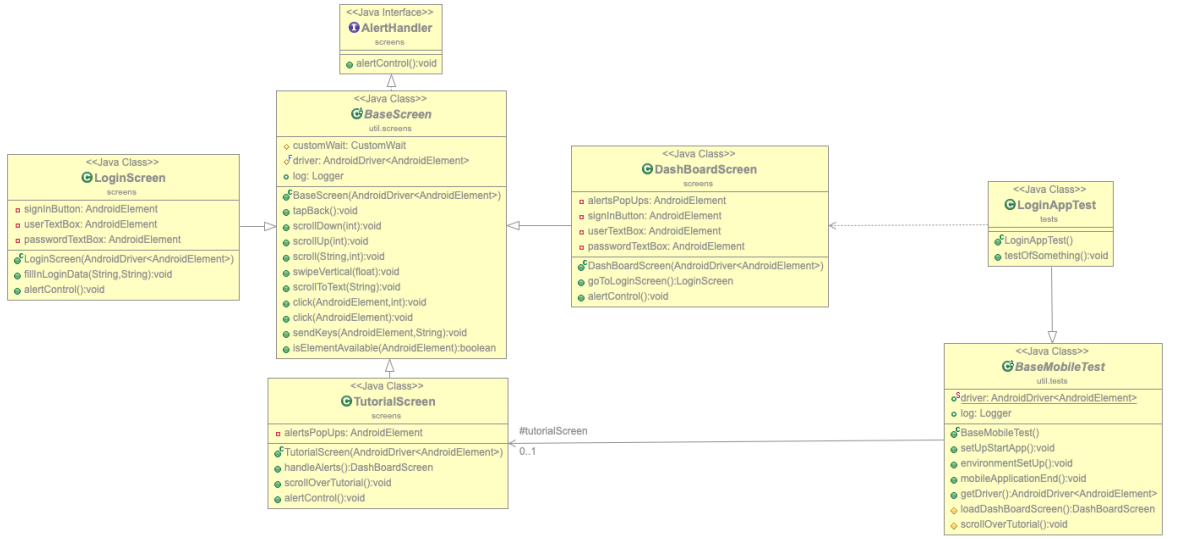
Consider we will automate a regression testing suite on the application, meaning all functionalities should be tested at a not-so-deep level. Create a document describing the approach for test automation on the app, including:

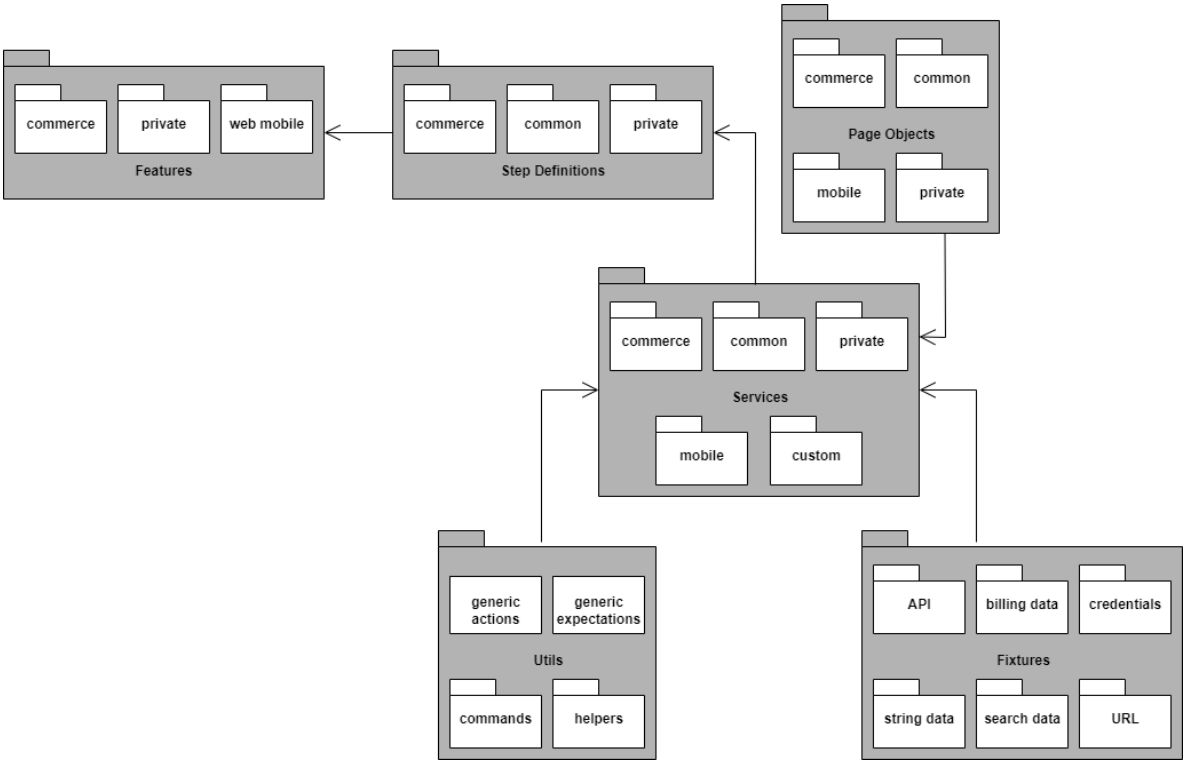
1. Definition of testing approach, specifying how the test cases should be defined and including the scope (What will be automated and what won't be automated)
2. Architectural Class Diagram of the framework architecture, considering screens, test files, external data, configuration, and Utils. Architectural layers should be specified. No specific format is required, however common formats such as UML are recommended. Feel free to make your own architectural decisions, however, they should be properly justified according to software and automation best practices.
3. Explanation of the architecture of the framework, including detail on how different layers behave, what they include, and how they interact with each other.

The implementation of the framework is not required, however, it is useful in case you would like to improve your test automation skills.

Here you can find some examples of architectural diagrams. Take into account that these diagrams are not related to the business or technologies required for this practice.







Additional Resources

[Choosing the Right Mobile Test Automation Framework | Sauce Labs https://medium.com/swlh/appium-architecture-793ae79cc85c]

Framework Setup

https://youtu.be/uunIEOX8kIw

Practice

In this practice, we will join all the knowledge learned from mobile automation in order to automate some scenarios on Android devices.

For this practice first, we need to install the WDIO dummy app on an Android device, you can use an emulator created using Android Studio, or use your own Android device if you have one. The APK is available for download here **https://github.com/webdriverio/native-demo-app/releases** (Make sure to use the .apk file). Optionally, you can also do this practice in IOS, however, please consider you would need a Mac device in order to install the required tools.

After the app is correctly installed and working, you will find here 4 scenarios to be fully automated. First of all you have to **recreate each step manually** in order to understand what is being tested and which steps we have to execute.

This set of tests are oriented to the navigation of the dummy WebdriverIO application.

1. **Navigation on the bottom menu bar:**
    1. As a precondition, the user should be on the Home screen.
    2. Navigate to each section by tapping their respective icon on the bottom menu bar.
    3. After tapping on the respective menu section, assertions should be made in order to check all elements on the respective sections are correctly displayed (Check visibility and properties).
2. **Successful Sign Up:**
    1. As a precondition, the user should be at the login section
    2. Navigate to the Signup section.
    3. Fill in all the form data to sign up (Consider the test must be able to be executed multiple times, so a random email may be generated each time, or an action should be done after the test to delete the created account).
    4. Check if the signup process was successfully completed.

3. **Successful Login:**
    1. As a precondition, the user should be in the Login section and have a previously created user.
    2. Navigate to the Login View.
    3. Fill in all the form data to log in.
    4. Check if the login process was successfully completed.
4. **Swipe cards on the Swipe section:**
    1. As a precondition, the user should be in the Swipe section.
    2. Perform a right swipe on the cards and check the old card is hidden, the new card is visible, and the respective card dot is selected.
    3. Perform a left swipe on the cards and check the old card is hidden, the new card is visible, and the respective card dot is selected.

Consider the test cases should be fully independent one from another, so, for example, the login test may reuse code from the signup test, but it may not rely on the signup test to be done earlier in order to properly work.

Additional Resources

Learning Paths | Applitools