Research on Hybrid Automation Framework for Mobile Application Testing Based on Page Object Model and Appium

Mohamed Abul Hissam A,Karthikeyan J,Nishanth R,Dr.Latha Maheswari,M.E.,Ph.D.
Sri Krishna College of Engineering and Technology, Coimbatore

Abstract : *The project is aimed at providing automation of user interface testing and business function and process testing. The proposed system is to develop highly consistent and reusable test scripts which will run the test cases on both iOS and Android devices simultaneously. It is achieved by using Appium which is a cross-platform testing tool which, supports mobile application testing on both iOS and Android platform. Page Object Model is a design pattern which has become popular in test automation for enhancing test maintenance and reducing code duplication. In the proposed system a page object is created which is an object-oriented class that serves as an interface to a page of the application under test. Further this project can be enhanced by synchronizing with cloud by using Jenkins and a job in a Jenkins can be scheduled periodically to maintain the consistency of the framework and test results can be published to group of subscribers who subscribes to a particular domain.*

Keywords : *Appium, Automation, Framework, Page Object Model, Testing*

I. Introduction

In recent years, with the rapid development of wireless communication network and the gradual evolution of smart phone applications from light-weight applications to more complex applications, the automation testing of applications has become increasingly important . The application of smartphone brings about a new revolution in software testing technology. Compared with traditional testing, mobile phone testing breaks away from the traditional architecture of PC in hardware, and is quite different in the way users interact. Mobile software has unique operation modes in use, such as user's sliding, tapping and clicking operations. How to achieve the automated testing execution on mobile clients has become the difficulty in this kind of testing. Due to the coexistence of iOS and Android operating systems, most mobile applications need to support multiple platforms, and their infrastructure is different, which makes the mobile automation testing framework have higher requirements for compatibility. How to select suitable automation testing tools to test mobile efficiently under limited human and material resources has become a major problem faced by enterprise R&D departments. This paper discusses the mobile application automation testing technology based on Appium, and shows its flexibility and effectiveness.

II. The mobile testing based on appium

A. The mobile application testing process

Because most enterprises need to put mobile applications on the market quickly, mobile applications are usually developed in the agile development model. Agile development is characterized by its development speed and embracing change. Agile development takes the evolution of users' needs as its core, and adopts an iterative and step-by- step approach to software development. That is to say, the whole development process is divided into several iteration cycles. Each iteration cycle usually lasts for a short period of time, usually 1 to 6 weeks. In agile development, software projects are divided into several sub-projects at the beginning of construction. The results of each sub-project have been tested, and have the characteristics of visibility, integration and operational use. In other words, a large project is divided into several interrelated, but also independent small projects, and completed separately, in the process of which the software has been in usable state. In agile development, every iteration has requirements to be developed. In each iteration, developers design and code for the requirements of the iteration, and testers do test the software according to the requirements.

So what is the testing process in agile development? First, in each iteration, testers need to review the requirements of the iteration to fully understand the requirements and find errors or omissions in the requirements during the review process, then design and write test cases according to the requirements. After the completion of test case writing, other members of the project team need to be invited to participate in the test case review to ensure the correctness of test cases and requirements coverage. After the test case is completed, if the coding of the tested module has been completed, the test environment can be built and the test can be executed. During the execution of the test, if defects are found, the defect report

needs to be submitted. After the defect is modified, the defect needs to be retested. When all the test cases of the iteration are executed and passed, the testing task for this iteration is completed, and then the requirement review for the next iteration canbe started. When the development and testing of all iterations are completed, all functions of the software version are developed, system testing and acceptance testing of the software can be carried out.

B. The mobile application testing environment

At the beginning of the test design, testers first need to consider the test environment. In the reality of mobile devices and platform fragmentation, it is impossible for testers to exhaust all the versions of devices and operating systems to achieve full testing coverage. With limited time and effort, we can achieve as much testing coverage as possible from the perspective of input-output ratio. Because most mobile applications support iOS and Android operating systems, it is necessary to test mobile applications in both iOS and Android systems. The version of the operating system to be tested should cover the version of the operating system used by the vast majority of current users. The mobile devices tested should choose the tablets and mobile phones with high market share.

C. Test contents of mobile applications

The following testings need to be done on mobile applications.

1) user interface testing
2) business function and process testing
3) performance testing
4) installation/uninstallation testing
5) software upgrade testing
6) security testing
7) horizontal and vertical screen testing

Appium is mainly aimed at the automation of user interface testing and business function and process testing.

III. Page Object Model

Page Object Model is a design pattern to create Object Repository for UI elements. Under this model, for each web page in the application, there should be corresponding page class. This Page class will find the Mobile Elements of that web page and also contains Page methods which perform operations on those Mobile Elements.The chief problem with script maintenance is that if 10 different scripts are using the same page element, with any change in that element, you need to change all 10 scripts. This is time consuming and error prone.

A better approach to script maintenance is to create a separate class file which would find web elements, fill them or verify them. This class can be reused in all the scripts using that element. In future, if there is a change in the web element, we need to make the change in just 1 class file and not 10 different scripts.

This approach is called Page Object Model(POM). It helps make the code more readable, maintainable, and reusable.

IV. Implementation

4.1 Prerequisite to use Appium

1) Install JDK (Java Development Kit).
2) Install Eclipse.
3) Install TestNg for Eclipse.
4) Appium Client Library.
5) Install Appium server.
6) JavaScript (Not Required - Whenever Appium server is installed, it by default comes with "Node.exe" & NPM. It's included in Current version of Appium.)
7) Install Appium Desktop.

4.2. Appium Setup

To test the application using Appium, the steps to set up the testing environment are as follows:

1.Install Appium Server.

Appium Server is a core service of Appium. It can receive requests and automate test execution by manipulating mobile devices. The tool itself can be installed independently. After the installation, the command "appium" can be entered in the terminal and the port of 4723 is opened locally. This port is a small service that receives requests from test scripts and automates testing. If Appium Desktop is installed, the Appium Server is already embedded. When Appium Desktop is started, Appium Server is actually started in the background.

3.Install Appium Client

Install Appium Client. Appium Client is a client encapsulation library in various languages for connecting to Appium Server. To run the test script successfully, Appium Client needs to be installed, for example, to run the Python script, a Python client needs to be installed to drive it locally. When installing a Python client, you can go to https://github.com/Appium/python-client to follow the installation instructions.

4.Install Appium Desktop

Install Appium Desktop. Appium Desktop is a comprehensive tool embedded with Appium Server and inspector. It can be downloaded and installed directly from https://github.com/appium/appium-desktop/releases.

4.3 Appium Inspector

Similar to Selenium IDE record and playback tool, Appium has an 'Inspector' to record and Playback. It records and plays native application behavior by inspecting DOM and generates the test scripts in any desired language. However, currently, there is no support for Appium Inspector for Microsoft Windows. In Windows, it launches the Appium Server but fails to inspect elements. However, UIAutomator viewer can be used as an option for Inspecting elements.

Steps to start with Appium Inspector on Mac machine:-

1.Download and start your Appium server with the default IP Address 0.0.0.0 and the port 4723.

Figure 4.1 Appium server

2.Now, click on Start Inspector session button which will give configurations window to setup the capabilities for the device under test and start the inspector session.

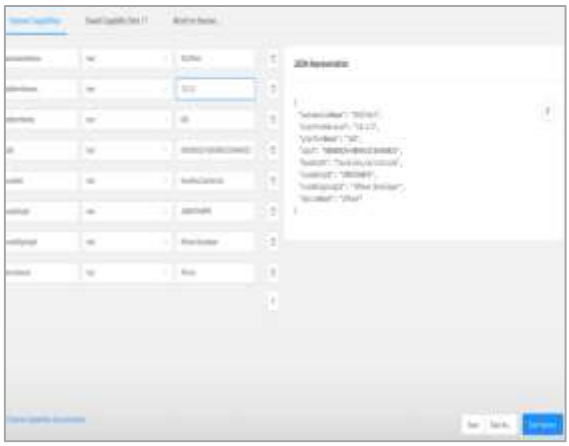

Figure 4.2 Device capabilities setup

3.Launching your Appium Inspector will show the elements on the AUT from where using required locator strategy elements can be captured.

4.4 .Test Environment Setup-Android

1.Install Android SDK

- Install the latest version of Android studio.
- To open SDK Manager, do any of the these: On Android Studio landing page, select Configure > SDK Manager.
- In the Default Settings dialog box, click these tabs to install Android SDK platform. packages and developer tools. SDK Platforms: Select the latest Android SDK package.

4.5 Test Environment Setup-iOS

1.Install Xcode

Xcode is the development and debug environment on Mac.It provides required tools, files for dev/automation for Mac and iOS apps.

2.Install XCode command line tools

- Open terminal and run
- xcode-select --install

3.Install Carthage brew install Carthage

In order to launch WebDriverAgent, your macOS will need to have Carthage installed
(Not needed for automation on android)

4.Initialize WebDriverAgent project

- On terminal cd to WebDriverAgent folder

- WebDriverAgent driver will be found at /usr/local/lib/node_modules/appium/node_modules/appium-xcuitest-driver/WedDriverAgent
- Run the following commands in terminal
    - mkdir -p Resources/WebDriverAgent.bundle
    - ./Scripts/bootstrap.sh -d

5.Open WebDriverAgent.xcodeproj in Xcode.
6.Under the project in Xcode for WebDriverAgentLib and WebDriverAgentRunner targets, go to general tab and select "Automatically manage signing", and then select your Development Team
7.Clean - Build - Run

Build the project:
xcodebuild -project WebDriverAgent.xcodeproj -scheme WebDriverAgentRunner -destination 'id=udid' test

4.6 Running The Test Scripts

1.Create the device config json

This json file contains the desired capabilities which are required to establish a session with the server.

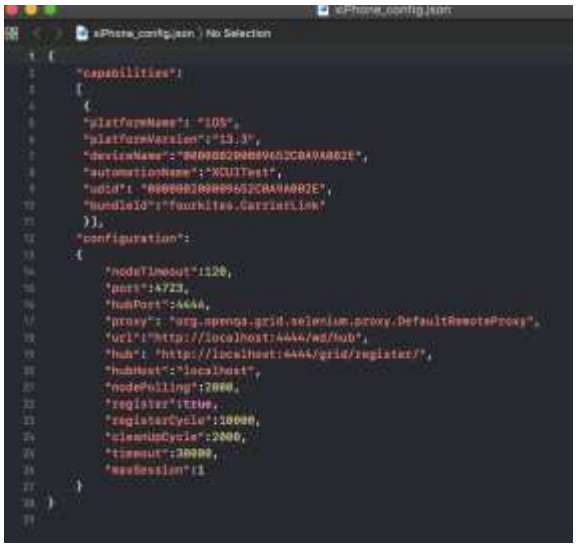

Figure 4.3 Device configuration json

2.Create a script to start the server

This script will start the Appium server in the machine with specified configurations.

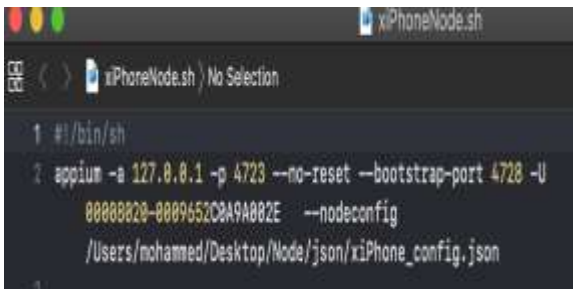

Figure 4.4 Appium server start-up

3.Download the latest version of selenium standalone jar.
4.Start the hub using the following command

- java -jar selenium-server-standalone-3.141.59.jar -role hub

5.Check whether the node (device) is registered to the hub.
6.Run the tests from the eclipse.

V. Conclusion

The project is mainly about to reduce the time consuming activity of manual regression testing in the application development cycle and to provide an alternative solution. Which The main objective of this project is to reduce the man effort on testing and to provide a effective automation framework which will run the test scripts on both the Android and iOS, which will considerably reduce the need for native coding for two different platforms. This project provides the design which has the advantages of low cost, less maintenance and readily configurable according to the need. This system can be effectively implemented in the organisations for better results and for better quality of products.

VI. Refrences

[1] Research on Mobile Application Automation Testing Technology Based on Appium by Wang Junmei, Wu Jihong, 2019 International Conference on Virtual Reality and Intelligent Systems

[2] Novel Framework for Automation Testing of Mobile Applications using Appium by Ashwaq A. Alotaibi, Rizwan J. Qureshi, Published Online February 2017 in MECS

[3] Research on Automated Testing Framework For Multi-Platform Mobile Applications by Da Zun, Tao Qi, Liping Chen, Proceedings of CCIS 2016

[4] Survey on Mobile Automation Testing Tools by Sarafarazahmad Momin, International Journal of Application or Innovation in Engineering & Management (IJAIEM), ISSN 2319 – 4847, Volume 4, Issue 1, January 2015

[5] An Integrated Test Automation Framework for Testing on Heterogeneous Mobile Platforms Hyungkeun Song, Seokmoon Ryoo , Jin Hyung Kim , 2011 First ACIS International Symposium.