Getting started with Appium

A complete step by step guide

What is Appium

Starting from the very basic question, Appium is a mobile Automation tool. React Native provides a mobile app development experience without sacrificing user experience or visual performance. And when it comes to mobile app UI testing, Appium is a great way to test indigenous React Native apps out of the box. Creating native apps from the same code and being able to do it using JavaScript has made Appium popular. It works for native apps as well.

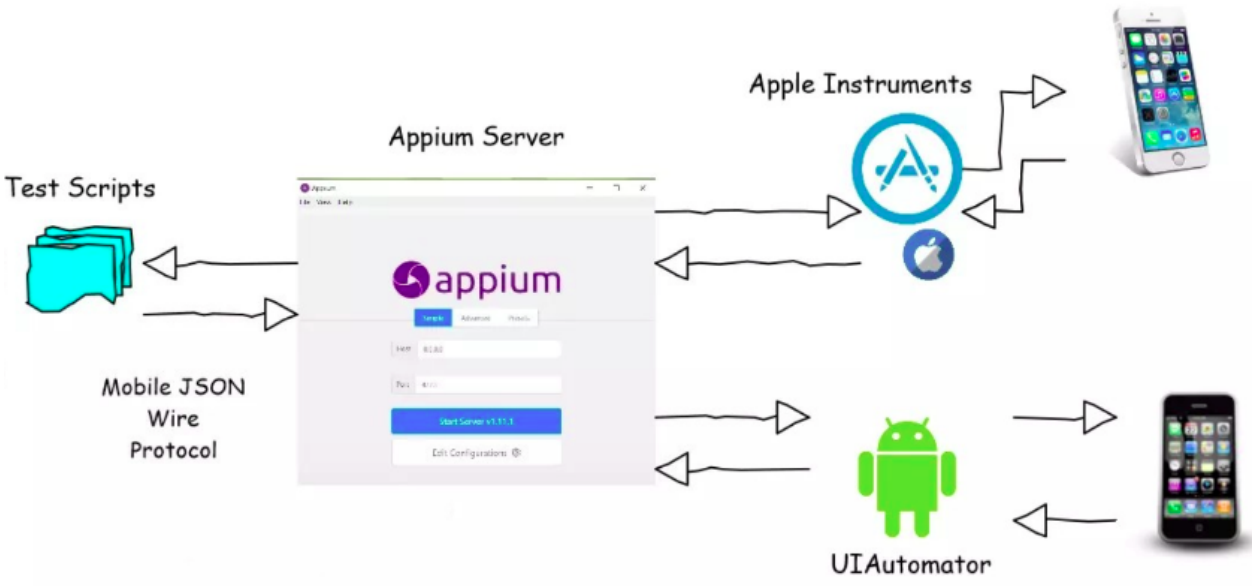


Fig:- Appium Architecture

Importantly, Appium is "cross-platform", allowing you to write tests against multiple platforms (iOS, Android), using the same API. This enables code usage between iOS, Android, and Windows test suites. It runs on iOS and Android applications using the WebDriver protocol.

What is WebDriverIO

WebdriverIO is a widely used test automation framework in JavaScript. It has various features like it supports many reports and services, Test Frameworks, and WDIO CLI Test Runners

The following are examples of supported services:

- Appium Service

- Devtools Service
- Firefox Profile Service
- Selenium Standalone Service
- Shared Store Service
- Static Server Service
- ChromeDriver Service
- Report Portal Service
- Docker Service

Pre requisites

- Install the latest stable version of Android Studio
- Install android-platform-tools from CLI
- Install JDK latest stable version
- Download the latest stable version of VS Code
- Download and install the latest LTS Node.js

Stepwise guide

1. Open Terminal and move to your workspace

2. Create Project Directory

```
$ mkdir Demo_Appium_Project
```

3. Create a sample Appium Project

```
$ npm init
$ package name: (demo_appium_project) demo_appium_test
$ version: (1.0.0) 1.0.0
$ description: demo_appium_practice
$ entry point: (index.js) index.js
$ test command: "./node_modules/.bin/wdio wdio.conf.js"
$ git repository:
$ keywords:
```

This will also create a package.json file for test settings and project dependencies.

4. Install node packages

```
$ npm install
```

5. Install Appium through npm or as a standalone app.

```
$ npm install -g appium or npm install --save appium
```

6. Install WebdriverIO

```
$ npm install -g webdriverio or npm install --save-dev webdriverio @wdio/cli
```

7. Install Chai Assertion library

```
$ npm install -g chai or npm install --save chai
```

8. Make sure you have following versions installed (or above if stable):

```
$ node --version - v.14.17.0
```

```
$ npm --version - 7.17.0
```

```
$ appium --version - 1.21.0
```

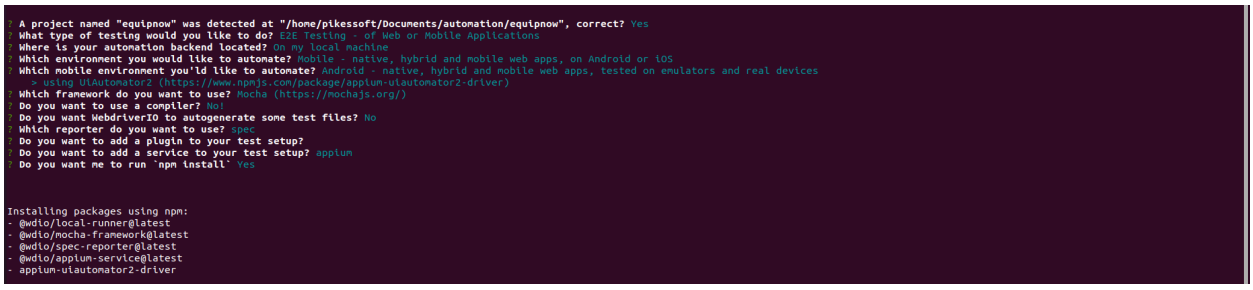
```
$ java --version - java 16.0.1
```

```
$ allure --version - 2.14.0
```

WebdriverIO Configuration

```
$ npx wdio config
```

With the following series of questions, install the required dependencies,



If you encounter the following error

```
ts -node/esm/transpile-only 'resolve'
```

Run the following command

```
$ npm i -D typescript ts-node
```

For iOS Automation, just add the following capabilities in wdio.conf.js & the Appium Configuration:

```
{
 "platformName": "IOS",
 "platformVersion": "14.5",
 "app": "/Your_PATH/wdioNativeDemoApp.app",
 "deviceName": "iPhone 12 Pro Max"
}
```

After this, it is important to install Appium Doctor to check and verify if we have all the dependencies installed for Appium. To get a guide on installing Appium Doctor, click here.

Resultantly

You have everything installed for Appium. Now what we do is see how we can combine everything and start our mobile automation.

In the automation project directory,

1. Open VS Code and under the test/specs folder make a file say test.e2e.js
2. Write your test scripts in this file and save it
3. If you have an emulator attached, open Android studio and run your emulator
4. Run the Appium server from the system terminal
5. Lastly, go back to VS Code, open terminal and type

   ```
   $ npm run wdio
   ```
6. This will execute your test scripts

Pro Tip

While finding the selectors for elements to be used in test scripts, we can manually navigate through the code, but what helps save all that effort is **Appium Inspector.** We can install Appium Inspector and write the desired capabilities of the emulator or real time device whichever we are using. We have to make sure our virtual/physical device is running and the Appium server is active too. After this, we can start the session on Appium Inspector and open our app there. Now whichever element id/selector we need, all we have to do is click the element. The Appium Inspector will tell us the id/selector of that element.

Recommendations

Course for Automation testing using Appium

Guide to install prerequisites

Guide to install Appium Doctor

Guide to install Appium Inspector

Appium documentation