**Tools Available For Both Android and IOS.**

1. Squish By FrogLogic

Solve your Mobile test automation challenges with Squish for iOS and Squish for Android. Squish features dedicated support for automated testing of native Mobile Apps, mobile Web Apps as well as a mixture of both.

Due to Squish’s unique and stable object identification methods, Squish tests can run on mobile device emulators and different real devices without any changes. Unlike many other test tools, Squish does not require you to jailbreak or root the device. Instead, you can get started with automated GUI testing of your mobile Apps right away!

**Benefits**

* Advanced gesture support
* Ready for Testing in the Cloud
* Support for embedded web content
* CI and source control integration
* End-to-End and IoT testing

**Disadvantages**

* No support for testing native dialogues; this includes file upload fields.
* HTTPS protocol is not supported\
* Cannot be used with websites loaded with local hosts.

1. **KMAX**

Use KMAX to test your mobile app or device to make sure it performs under adverse network conditions, ranging from the routine to the extreme. KMAX provides pre-defined network scenarios, including 3G/4G, LTE, low-earth orbit satellite, and more. Emulate adverse network conditions in your lab including packet loss, delay, duplication, corruption, and more. Precisely recreate specific scenarios.

### Appium (iOS/Android Testing Tool)

Appium is an open source, and a cross platform test automation tool for the hybrid and native iOS, it supports Android versions from 2.3 onwards. Appium works like a server running in the background like selenium server.

It supports many programming languages, such as [Java](https://www.guru99.com/java-tutorial.html), Ruby,[C#](https://www.guru99.com/c-tutorial.html)and other which are in the WebDriver library. Appium utilizes WebDriver interface for tests running

Appium automates Android using the UIAutomator library, which is given by Google as part of the Android SDK. On mobile devices, it can control Safari and Chrome. It can be synchronized with testing framework TestNG. In this case, UI Automator can produce informative and detailed reports, similar to reports generated by Ranorex

**BENIFITS**

* Due to the use of standard automation APIs on all platforms, you don't have to modify or recompile your app in any way
* You can use any web-driver compatible language (Java, Objective-C, JavaScript) to write test case
* You can use any testing framework
* Easy to setup on a different platform
* Supports various languages like Ruby, Java, PHP, Node, Python
* It does not require anything to be installed on the device
* You can still use Selenium Webdriver JSON wire protocol
* You don't have to re-compile mobile app on a different platform
* With the help of Java, it can be integrated with other tools.
* Provide a record and playback tool.

### ****DOES NOT FACILITATE SCRIPT EXECUTION ON MULTIPLE IOS SIMULATORS****

Appium accelerates iOS app testing by allowing developers to use simulators. The simulators make it easier for testers to mimic internal behaviour of the underlying iOS devices. But it does not allow users to run multiple test scripts on multiple simulators at a time. Hence, the developers have to execute test scripts individually to produce more reliable test results. The shortcoming further increases iOS app testing time significantly

1. TESTDROID (PAID)

Testdroid is a cloud-based mobile application testing tool that helps developers to save costs behind app development, speed up time to market the product and reduce operational and unpredictable costs. It is the fastest way to test your application against various real Android and iOS devices with different HW platforms, screen resolutions, and OS versions. The price range varies from $499- $4999/ month as per requirement. It is one of the solid mobile game testing platforms for Android and iOS games. It allows remote manual access to over 300 real devices running Android before an app is launched.

**Benefits of Testdroid**

* It saves in App development costs
* Minimizes the risks with real devices and agile testing
* Reduce operational and unpredictable costs
* Improve app rating and daily active users
* Cloud based testing

1. **APPIUM STUDIO**

Appium Studio is an enhanced version of open-source Appium developed by Experitest to make Appium testing even better.

As mentioned before, Windows and MAC machines can be used to test on both Android and iOS devices. Simulators and Emulators are supported as well.



1. **JEST**

When we transitioned our mobile app from native code to React Native, testing was the last thing on our mind. Priority one was recreating our application from scratch using React Native — and once we released it, more and more features required implementing, and before we knew it, we had thousands of lines of code …and zero tests. We knew it’d be good practice to write tests but finding the motivation to write them was an entirely different beast.

Fortunately, once we got started, we found testing with Jest to be painless. My goal is to make it even more painless (that’s a good thing) for you by explaining everything I’ve learned so far.

Adding Jest to our project

In React Native versions 0.38 and greater, Jest is included when creating a project via react-native in it, so the following setup may be already done for you. Same goes if you’ve read the documentation from Jest on setting up testing with React Native

1. Installing packages

For our initial setup, we installed three libraries: jest, babel-jest, and react-test-renderer. Be sure to install these as devDependencies using either npm i --save-dev or yarn add -dev.

2. Just a few more lines before we can get testing…

// Add this to your package.json

**"scripts": {**

**"test": "jest"**

**},**

**"jest": {**

**"preset": "react-native"**

**}**

**// Add this to your .babelrc**

**{**

**"presets": ["react-native"]**

**}**

Now to run our tests, we simply type npm test on our command line.