WHITEPAPER

CHOOSING A MOBILE QA AUTOMATION PLATFORM

FIRST FACTORY

Mobile applications dominate our daily use of technology and, as such, companies face growing demands to have a visible mobile presence. Those mobile experiences come with different user expectations, where faster performance and more interactive interfaces are required. An additional challenge for companies building mobile applications is the sheer variety of hardware manufacturers, device types, screen resolution sizes, and the number of operating system versions customers use. All of these factors, combined with the selection of almost nine million apps available across the globe, mean that companies need to have clearly defined app strategies and the technical expertise on hand to build and maintain their apps in the wild. This includes having up-to-date QA automation scripts to ensure the user experience consistently meets expectations with each new feature release.

Several tools are available for mobile QA automation. We have evaluated several options for testing React Native apps. Below, we weigh in on the strengths and weaknesses of cross-platform and native platform automation tools.

CROSS-PLATFORM TOOLS

Appium

Appium is an open-source tool for automating native, hybrid, and mobile web applications on iOS and Android platforms. It supports React Native apps and can be used to write automated tests in various programming languages like Java, Ruby, Python, and JavaScript.

Appium's cross-platform support for testing both iOS and Android platforms makes it a versatile option for mobile automation testing. As an open-source tool that hosts a large contributor base, the platform evolves regularly, offers custom extensions, and is tested by a large swath of users. Multiple programming languages are supported, such as Java, Python, Ruby, JavaScript, and more, thus making it easier for testers to write test scripts in their preferred language. A big benefit is that Appium allows users to choose from various automation frameworks such as Selenium, Espresso, and XCUITest to write their test scripts, as well as easy Continuous Integration and Continuous Deployment (CI/CD) tools like Jenkins, TeamCity, and others.

Appium is not for everyone, mainly due to its steep learning curve for those who are new to mobile automation testing, complex setup, and slow execution speeds. It also requires special hardware; for example, a MacBook is required to automate iOS. Support for native app testing is limited, especially when compared to other tools that offer more advanced features. There is also limited support for hybrid app testing as the platform struggles with testing hybrid applications due to the limitations of the webviews. Another consideration is that, with a robust community of contributors, Appium requires regular updates to stay current with the latest changes in the platform.

Cypress

Cypress is a modern end-to-end testing framework well-suited to testing React Native apps. It provides an easy-to-use API for writing tests and offers features like time-travel debugging, network throttling, and automatic waiting.

One of Cypress's biggest benefits is its ease of installation and setup. This ease of use extends to its intuitiveness for writing and running tests. Cypress is fast and efficient: it runs tests in real-time and it provides instant feedback, making it a leader in this category. Error messages that it serves up are detailed and provide stack traces, which makes it easier to debug and fix issues. Cypress supports multiple types of tests, including unit tests, integration tests, and end-to-end tests. As a popular platform for many years, Cypress hosts extensive documentation..

Cypress is designed primarily for testing web applications, and it may not be the best choice for testing desktop and other non-browser applications. Plus, it is not as robust as other mobile testing frameworks, such as Appium or Detox, and is not designed for cross-browser testing. The platform fails to offer parallel testing out of the box, which may limit its scalability for larger projects. Cypress performs well as designed but offers less customization than other testing frameworks, and its configuration options are limited.

Detox

Detox is a mobile testing framework for React Native apps with end-to-end testing capabilities. It enables developers to write automated tests that simulate user interactions with the app and verify the app's behavior.

Detox provides reliable and repeatable end-to-end tests, making it ideal for regression testing and continuous integration workflows. It is also faster than other mobile testing frameworks due to its ability to run tests concurrently on a single device or across multiple devices. Detox is refreshingly user-friendly and provides easy-to-use APIs for developers to write and maintain tests. The platform integrates well with popular CI/CD tools, such as Jenkins, Travis CI, and CircleCI, and generates detailed logs and error messages, resulting in quicker diagnosis and fix times.

As a relatively new platform. Detox has amassed a smaller support community, and support documentation and resources may be limited. This, coupled with a healthy learning curve, may make the barrier to entry harder for some—though once that initial hurdle is cleared, the speed and performance are impressive. Detox requires regular maintenance and updates to stay up-to-date with the latest changes in the mobile platform and has some platform-specific limitations.

Jest

Jest is a popular testing framework created by Meta and is widely used to test React Native (also created by Meta) apps. It is primarily used for testing JavaScript code, including unit tests and integration tests. It provides a simple API for writing tests and supports features such as snapshot testing, mocking, and code coverage.

Meta builds these tools first for themselves, so they undergo extensive testing and are often easy to set up and use. Jest comes with a built-in test runner and assertion library. It runs tests in parallel—making it faster than other testing frameworks—and its snapshot feature speeds up the testing process. A highly used feature, especially early in app development, is the ability to mock objects and test complex interactions between components. While Jest does have some capabilities for end-to-end testing, such as testing the functionality of an entire application from the user's perspective, it is not specifically designed for this purpose.

Considering that Jest was built in-house to serve the needs of the Facebook development team, it is understandable that it has limited support for non-React applications. To the same point, Jest is not very customizable, and its configuration options are limited. While Jest does run tests in parallel, it may not be as scalable as some other testing frameworks.

IOS-ONLY AUTOMATION TOOLS

EarlGrey 2.0

EarlGrey is a testing framework for iOS apps that was developed by Alphabet. It is a powerful and flexible tool that provides developers with a range of features for **UI testing** of their iOS apps.

EarlGrey has a powerful set of features for testing the user interface of iOS apps. It can interact with any UI element on the screen, including complex views and animations. EarlGrey provides a simple and expressive syntax for writing tests. The framework's syntax is easy to read and write, allowing developers to create tests that are easy to understand and maintain. The platform supports both Swift and Objective-C, which makes it easier for developers to integrate the framework into their existing iOS app development process. From the start, it was designed to be highly scalable and can be used to test apps of any size or complexity—consider the use cases for testing some of the world's most popular iOS apps built by Alphabet: Google, Gmail, Google Maps, and Google Photos. As a result, EarlGrey has excellent documentation and a strong community of users, which makes it easy for developers to learn and use the framework.

EarlGrey is a UI testing framework, which means that it does not provide support for testing the backend or server-side components of an app. If you are new to iOS app testing, expect some minor challenges setting up and configuring EarlGrey. Similar to Meta's Jest platform, EarlGrey may be less flexible than other testing frameworks, particularly when it comes to customizing test scenarios or integrating with third-party tools, due to the fact that this was initially built by Alphabet for their own needs. And, as with all open-source frameworks, support and ongoing maintenance may not be as reliable or predictable as closed-source products.

XCTest/XCITest

XCTest is a testing framework provided by Apple for testing iOS, macOS, watchOS, and tvOS applications. It is part of the Xcode development toolset and is designed to enable developers to write automated tests for their applications. It provides a number of features for testing, including unit testing, performance testing, and UI testing. It includes a wide range of APIs for making assertions about the behavior of your code, as well as APIs for mocking and stubbing objects.

In addition to XCTest, Apple also provides XCUItest, which is a UI testing framework specifically designed for testing the user interface of iOS, macOS, watchOS, and tvOS applications. XCUItest allows developers to write automated tests that simulate user interactions with their applications, such as tapping buttons, swiping screens, and entering text.

XCTest and XCUItest are integrated into Xcode, which makes them easily accessible to developers who are already using Xcode for their development work. Both frameworks are designed to work seamlessly with Apple's development tools and APIs—which means that developers can test their applications in the same environment in which they are built. XCTest provides a wide range of assertion APIs, making it easy for developers to write tests that verify the behavior of their code. XCUItest is specifically designed for testing the user interface of applications and provides a number of APIs for simulating user interactions, such as tapping buttons and scrolling screens. Both tools can be used for CI/CD workflows, allowing developers to catch bugs early in the development process.

As you might expect, XCTest and XCUItest are designed specifically for testing Apple platform applications and cannot be used to test applications on other platforms. XCTest provides a wide range of assertion APIs that require a strong understanding of the underlying concepts. XCUItest can be slow to run, especially when testing applications with complex user interfaces or a large number of screens. While XCTest and XCUItest are powerful tools, they cannot replace manual testing entirely, and developers should still plan to test their applications manually as well.

ANDROID-ONLY AUTOMATION TOOLS

Espresso

Espresso is a testing framework for automating functional and UI testing for Android applications. It provides a simple API for writing tests and supports different programming languages, such as Java and Kotlin.

Espresso is known for its fast execution times, thanks to its ability to interact directly with an app's UI components without requiring a full system integration test. Espresso integrates seamlessly with Android Studio and other developer tools, making it easy to set up and use for developers who already use these tools. Its API is designed to ensure that tests are stable—waiting for UI components to become idle before executing any actions and verifying that the correct UI elements have appeared or changed after an action has been performed. The API is easy to understand and use, even for developers who are new to Android UI testing.

Espresso is solely focused on UI testing for Android apps, meaning it cannot be used to test other aspects of an app, such as backend functionality or business logic, and cannot test apps for iOS. While Espresso's API is easy to use, there can still be a learning curve for developers who are new to Android app testing or unfamiliar with the nuances of Espresso's API. Tests can only be run on Android devices running version 4.4 or later, so testing on older devices may require a different testing framework.

Robolectric

Robolectric allows developers to run unit tests in a simulated Android environment without requiring an actual device or emulator. This can save developers time and money, as they do not need to purchase physical devices or wait for emulators to start up.

Robolectric provides a range of tools and features for testing Android apps. It supports various testing frameworks, including JUnit and Mockito, and allows developers to test both Java and Kotlin code. It is fast. Because it simulates the Android environment, tests can be run much faster than if they were run on a physical device or emulator. This is a main benefit of the tool and is particularly useful for developers who want to run tests frequently as they develop their app. Robolectric is also flexible; it can be used with multiple testing frameworks and languages and can be integrated into existing Android app development workflows. As an open-source platform, Robelectric has a strong community of developers contributing to its development and maintenance. This ensures the framework is well-maintained and up-to-date with the latest features and best practices.

Robolectric is primarily designed for unit testing and does not provide support for testing the user interface of Android apps. Developers may need to use additional testing frameworks, such as Espresso, to test the UI of their apps. The framework requires a significant amount of configuration and setup, which can be time-consuming. Robolectric may be less reliable than other testing frameworks, particularly when it comes to testing complex Android features such as multi-threading or hardware integration. Developers may need to use additional tools or techniques to test these features effectively.

UIAutomator

UIAutomator is a testing framework for Android apps that allows developers to write tests that interact with the user interface of their Android apps. It supports testing across multiple apps, provides a range of testing frameworks and languages, and can run on both real devices and emulators.

UIAutomator is another creation of Alphabet that comes with plenty of documentation and support. Developers can rely on the framework to be up-to-date with the latest Android features and best practices. With UIAutomator, you can test the integration between different apps on an Android device. It can interact with UI elements across different apps, allowing developers to test how their app interacts with other apps. You will be able to run tests on real Android devices and emulators. This makes it easier for developers to test their apps on a range of different Android devices, and to identify and fix issues that may be specific to certain devices or Android versions. As you would expect from Google products, UIAutomator is fast. It is also highly efficient thanks to its support for multi-threading and its ability to execute tests in parallel. It provides a robust set of tools and features for testing Android apps, supports a variety of testing frameworks, including JUnit and Espresso, and allows developers to test both Java and Kotlin code.

Like other platforms, UIAutomator can be tricky to set up and configure, particularly for developers who are new to Android app testing. As it is primarily designed for UI testing of Android apps, it does not provide as much support for testing the underlying code or business logic of an app.

- UIAutomator may be less reliable than other testing frameworks, particularly when it comes to testing complex Android features such as multi-threading or hardware integration. Developers may need to use additional tools or techniques to test these features effectively.
- Tests may be slower to run than tests written with other frameworks, particularly when testing complex UI interactions or animations.

GENERAL CONSIDERATIONS: MAKING THE RIGHT DECISION

Each client's needs and situations are different; while Appium is the most popular mobile application testing platform we see, that does not necessarily mean that it is the right choice for you. Consulting with technical experts at First Factory and considering pilot projects can provide valuable insights into how each platform aligns with your business needs. If you are interested in discussing the capabilities of these automation testing platforms with us to help determine the right choice for your organization, we would be happy to have a call with you.

If you have a product development need, consider the nearshore development team at First Factory for design, development, project management, and product ideation.

Contact us at +1.646.688.5070 or firstfactory.com/contact-us.