WHITE PAPER

REAL MOBILE DEVICE TESTING - BUILD VS. BUY

Modern day development tools have made building a mobile application nearly as easy as building a web application. But there is one major difference: testing. When it comes time to test a mobile application there are considerably more complications. Your tests should include emulators, simulators and physical devices to ensure your app best functions in the real world. Add to this the need to balance your test coverage and release cadence with the need to keep your distributed testing team always in sync. This is why the natural path of building a mobile device lab may not be the right solution. Do you know the impact of your device lab on delivering consistent results? Do you understand the long term risk and opportunity cost of taking it all on yourself? This paper explores the benefits of using a cloud based testing service versus your own local device lab.

SAUCELABS

TABLE OF CONTENTS

1 Executive Summary

1 Automated testing as part of the broader CI/CD pipeline

2 Manual Testing Cripples CI

2 Automated Testing is Necessary for CI

3 Automated Testing Requires Both Virtual & Real Devices

5 Appium Drives Test Automation on Virtual & Real Devices

6 Building & Maintaining a Private Testing Grid Will Slow You Down

7 A Cloud Testing Vendor Delivers the Most Value

8 Sauce Labs - Instantly Available Virtual & Real Devices in the Cloud

9 Key Takeaways

10 About Sauce Labs

EXECUTIVE SUMMARY

Web searches on mobile devices have already surpassed those on traditional computers. There are now numerous companies whose entire business is a mobile app - you can't contact them by phone and they don't have a physical location. As mobile apps become more and more synonymous with modern business, devs and QA professionals alike have to respond by rapidly developing and testing mobile apps. Since the business is totally dependent on these apps, the key to remaining competitive is updating them frequently and adding new features, all the while retaining high levels of quality and a great user experience. In this world of agile development, testing is the primary phase that can slow your mobile app dev cycle to a crawl. Choosing between on-premise mobile device testing or a cloud testing service is critical, because it affects how apps are built and shipped, and therefore has direct consequences for the company's bottom line.

AUTOMATED TESTING AS PART OF THE BROADER CI/CD PIPELINE

Web application testing has undergone a major shift over the past decade. Previously, web app testing was done manually by in-house software testing teams. But as the complexity of testing grew, QA responsibilities were outsourced to vendors who could scale manual testing at lower costs than maintaining an internal team. Later, with the advent of Agile development processes, even outsourcing to vendors did not have quick enough turnaround to integrate into the increasingly short development cycles companies were adopting. As a result, automated testing became a standard practice, and enabled a whole new level of quality via Continuous Integration (CI), where tests could be run as frequently as desired, even on every single proposed code change. Today, we're seeing the same progression with mobile app testing, from manual testing to automated testing to continuously automated testing.

Businesses today compete in a mobile economy. From the large enterprise to the single-developer startup, mobile is a key factor in the success of modern apps and businesses. The top mobile apps are updated every 10 days on average. This is much faster than the quarterly and sometimes yearly updates that have been the norm for the software industry in previous eras. Users have come to expect a frantic pace of updates in their apps; those apps that tolerate bugs for more than a couple of weeks risk losing users to competitors. As a result of this shift in the mobile ecosystem, modern developers increasingly reach for cloud solutions as the only way to quickly scale QA efforts as part of the CI pipeline.

SAUCELABS
Learn more at saucelabs.com

For development to reach the pace required to stay competitive in today's market, the traditional waterfall methodology was not sufficient. It resulted in too many bottlenecks in the pipeline and communication problems between siloed teams. With the advent of Agile, those bottlenecks were reduced through the practices of forming complete self-organizing teams, and leveraging smaller cycles of complete work known as "sprints". To enable testing to keep pace with the demand to produce a full unit of value in each sprint, software teams have come to embrace Continuous Integration.

According to ThoughtWorks, 'Continuous Integration is a development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.' CI is not "faster Waterfall". Instead, its focus is on making development cycles smaller and more iterative, and turning releases into non-events.

MANUAL TESTING CRIPPLES CI

Some development shops that build mobile apps may rely exclusively on manual testing for their mobile apps. They assume the few devices in their lab, and the few QA Analysts on their team can catch and resolve the worst bugs, and users would tolerate the remaining bugs whose resolutions are inevitably postponed for a future release. However, in today's competitive mobile ecosystem, users demand more. Apps that are blind to real-world issues that users face will quickly lose out to the competition.

Manual testing can't keep pace with CI. It is prone to human error, making the results for performance tests highly inaccurate. Because they depend on a human to run and monitor each test, they can't be run in parallel, and thus slow down the pace of testing a mobile app. Test automation is the only viable option for testing mobile apps that need to be competitive in today's marketplace.

AUTOMATED TESTING IS NECESSARY FOR CI

According to Jez Humble and Dave Farley, the CI pipeline is "an automated implementation of your application's build, deploy, test, and release process." Thus, if you want your testing to be part of your CI process, you need to automate it. Automation will not only speed up your testing process, but bring other benefits like increased testing coverage, more accurate and shareable results, and testing across a wider range of devices.

CRITERIA	MANUAL TEST	AUTOMATED TESTING
Faster time to market		✓
Ideal for cross-platform tests		✓
Accurate		✓
Performs complex tests		✓
Increases frequency of feedback		✓
Supports agile methodology		✓
Frees up testers for their best work		✓
Scales to larger projects		✓
Is repeatable		✓
Enables after-hours testing		✓
Improves documentation & traceability		✓
Cost effective		✓
Improves over time		✓
Ideal for exploratory tests	✓	
Idealfor one-off tests	✓	
Easy to adopt	✓	
Easy to maintain after adoption		✓
Allows (massive) parallelization		✓

[To learn more about the benefits of test automation, read our white paper "How to Get the Most out of Your CI/CD Workflow Using Automated Testing"]

AUTOMATED TESTING REQUIRES BOTH VIRTUAL & REAL DEVICES

When automating mobile tests, it is a good practice to use both virtual and real devices to optimize costs and get your apps to market faster. It's obvious what a "real device" is: the physical phone or tablet of a certain make, with a certain mobile OS installed. What, then, is a "virtual device"? This is the umbrella term for the iOS simulator and the Android emulator. One is called "simulator" and

Emulator (left) & real mobile devices (right)

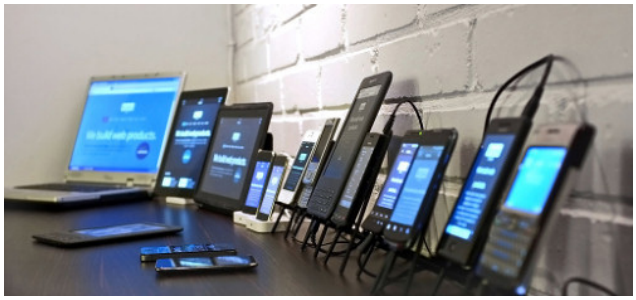


Image Source: Left - Google.com, Right - SmashingMagazine.com

SAUCELABS

Learn more at saucelabs.com

the other "emulator" for the technical reason that iOS simulators execute mobile apps within macOS by simulating the iOS environment, whereas an Android emulator actually implements the Android OS as a VM, regardless of the host environment. Whether we're talking simulators or emulators, the theoretically unlimited availability of virtual devices make them easy to spin up and use to test your app logic. On the other hand, the accuracy of real devices makes them ideal for functional testing later in the cycle. In a perfect world free of resource constraints, the best practice would be to use real devices for all testing. In practice, the requirement to run a significant number of tests on every commit means that virtual devices will always have a place in mobile CI. Spinning up a new virtual machine with a virtual device attached is much simpler than going to the store and buying a new physical device! In sum, going exclusively with real devices, or exclusively with virtual devices, will mean you lose out on the benefits of the other.

Typically, for builds with a large number of tests or which are run frequently, virtual devices are used for a majority of tests. They are lightweight, and provide great flexibility to the testing process. Virtual devices facilitate the reliability of a CI setup not just because they can be spun up easily, and at scale: they also guarantee a consistent environment, which is a must for CI. Reproducibility is key for understanding whether test failures are due to some fluke of the environment or are instead a real issue in your app. Unlike real devices that can be expensive, virtual are inexpensive, making them the ideal choice for the bulk of mobile app testing.

Virtual devices also offer certain functional advantages, for example the ability to emulate a variety of geolocations or network bandwidths. It's simpler to convince a virtual device that it is in Copenhagen than it is to find a phone physically present there available to your CI rig.

Despite the foregoing, real devices can't be left out of the equation. At the end of the day, your users are running real devices, after all. Real devices provide crucial real-world feedback that gives you confidence an app is ready for production. Certain third-party apps can only be tested on real devices, since they cannot always be installed on virtual devices due to mobile vendor security requirements. In-depth functional testing of specific device manufacturers or configurations requires real devices, and should be a part of your release process as at least a final gate before publishing a new version.

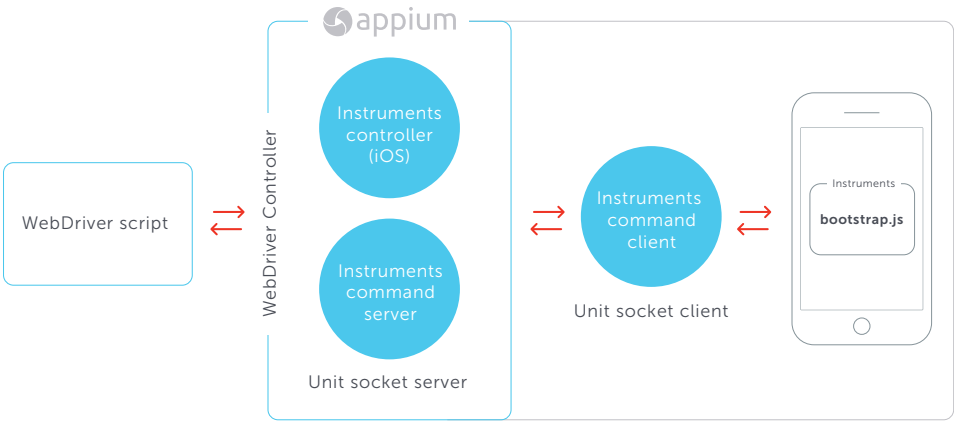
[To learn more about the balance between emulators and real devices, read our white paper "Automated Mobile Testing Requires Both Real Devices and Emulators"]

SAUCELABS
Learn more at saucelabs.com

APPIUM DRIVES TEST AUTOMATION ON VIRTUAL & REAL DEVICES

Appium is the leading mobile test automation platform, and can be thought of as the counterpart of Selenium for mobile testing. It is used to automate UI behaviors on both virtual and real devices, and is architected to allow massive parallelization of tests on cloud providers.

Appium works on both Android and iOS, which is a claim that few mobile testing frameworks can make.



Appium architecture for iOS app testing.

Tim Converse, Head of Search Science Engineering at eBay, highlights this in his article 'Reviewing open source Appium,' where he says, "Appium encourages multiple-platform testing by using test tools and modes of operation that are well known to developers. Also, Appium's familiar tool sets eliminate extra work by making sure that the tester doesn't have to do anything special to the code in order to make testing possible". Because Appium builds on top of vendor-provided automation frameworks, you won't need to compile any Appium-specific or third-party code or frameworks to your app — you can test the same app you're shipping. This goes a long way in ensuring quality.

Further, Appium allows tests to be created and run in any language and environment. Because Appium exposes the standard WebDriver protocol, it can take advantage of the massive Selenium ecosystem, including the client libraries that already exist for Selenium. (Appium does offer its own client libraries as extensions to the Selenium libraries that give access to Appium-specific mobile commands). The upshot of all this is that you can write automated test scripts in just about any modern language, not just Java/Kotlin for Android or Objective-C/Swift for iOS.

Once you've decided that automated mobile testing is the way to go, and you've decided to use Appium to power your virtual and real devices, the next obvious question is where you should execute your tests - in a private grid, or with a cloud testing vendor? This is a critical decision that will likely determine the success of your mobile testing strategy.

BUILDING & MAINTAINING A PRIVATE TESTING GRID WILL SLOW YOU DOWN

Modern development teams are distributed globally, and with this organizational scale comes added complexity. For testing results to be uniform, all teams need access to the same devices for testing. Additionally, the test results should be shareable across teams irrespective of location. Any member of the QA team should be able to debug a failed testcase using video, screenshots, and logs. Tools like Selenium and Appium are powerful automation allies, but running them at scale is a different proposition altogether than running one test at a time on a developer's machine. When you can't see what's going on with a test, or interrogate test details after the fact of its finishing, figuring out what caused a failure is an exercise in frustration. When scaling up your test organization or your test suite, additional test management features are essential.

This kind of scalability is an issue for the QA team, but that's only part of the story. Ops faces major hurdles when maintaining the private cloud. Maintenance includes updating, acquiring, and troubleshooting devices. After every test, the devices need to be cleaned and prepared for the next test on an ongoing basis, not to mention the requirements for maintaining the host machines. Let's face it, this is hard to get right when done by an in-house Ops team without the specialized knowledge needed to manage a test cloud.

You're also susceptible to wildcard consequences due to changes in the Ops team, and when someone who is key to managing the stack leaves, your entire QA process may be affected. Leaving this crucial piece of your CI pipeline to chance is not worth the risk.

Soft costs like scalability and maintenance are strong reasons to avoid going with an in-house grid stack. However, even when you consider hard costs, the story is no different - developing and maintaining a private cloud is expensive. For example, you need a physical, non-virtualized Mac computer for every few iOS devices, so every iOS device in your cloud is something close to double the cost of the device itself. And don't forget to take into account other capital expenses, or the salaries of all the employees who work on the lab.

SAUCELABS
Learn more at saucelabs.com

Test automation requires high scalability, low maintenance, and should be cost effective. For these reasons, a private grid usually ends up hindering rather than enabling test automation. It's tempting to go your own way, and it's easy to say "I can just buy a device to test on," but in reality, mobile testing is not about the devices alone: it's about enabling automated testing at scale. This is best done with a cloud testing vendor.

[For a detailed comparison of building or buying a testing solution, read our white paper "Selenium Grid — Build vs. Buy"]

A CLOUD TESTING VENDOR DELIVERS THE MOST VALUE

Organizations that are committed to mobile test automation need a vendor platform that delivers both real mobile devices and virtual devices in the cloud. A cloud testing platform like Sauce Labs is purpose-built to enable automated testing on virtual and real devices at scale. Sauce allows your QA team to focus on their test scripts rather than optimizing infrastructure to run those scripts on. This not only speeds up your QA process, but delivers a level of quality that can't be matched by a private cloud testing stack.

Maintenance is a breeze with a cloud vendor because maintaining infrastructure becomes the responsibility of the vendor rather than your internal Ops team. This eliminates issues of updating, acquiring, and troubleshooting devices. Adding new devices doesn't require ordering physical devices through a complex budget and procurement process, instead, you can fire up a new real device in the cloud in a matter of seconds.

In today's global workplace, it's imperative for the entire distributed team to have access to the same devices for testing. Additionally, any member of the team should be able to share test results with the rest of the team. This helps troubleshoot and debug failed tests, especially if the cloud platform offers essential tools like videos, screenshots, logs, etc...

When it comes to cost, comparing a single device on premise with a single device in the cloud may tilt the balance in favor of a private cloud at first glance, but looking at both hard costs and soft benefits holistically will show that a cloud vendor offers significant benefits that can't be matched by an in-house grid. This makes a big difference to the pace and quality of your mobile QA efforts. Furthermore, you can optimize costs for a cloud testing vendor by leveraging cheaper virtual devices for the bulk of the mobile testing, and relying on real devices for specific tests or at a final verification stage of the release process. A cloud testing vendor offers you the option to mix and match strategies based on the needs you have at any given time.

SAUCELABS
Learn more at saucelabs.com

SAUCE LABS - INSTANTLY AVAILABLE VIRTUAL & REAL DEVICES IN THE CLOUD

Sauce Labs delivers a large volume of real devices alongside more than 80 device platform combinations of iOS and Android simulators and emulators. This enables users to cover the majority of their testing with virtual devices and to augment that with more complete testing on the most popular real devices for maximum coverage at a fraction of what it would cost to use real devices only. Unlike many real-device clouds that require long wait times for devices to be ready for use, the Sauce Labs cloud features a high volume of devices, so they are available on-demand.

Here are the key features of the Sauce Labs Continuous Testing Cloud:

Massive Concurrency: The Sauce Labs Real Device Cloud supports high parallelism, allowing teams to test many functions at the same time. The result is overall reduction of test time by as much as 90%, depending on the degree of parallelism achieved.

Web, Native and Hybrid App Testing: With support for Appium and Selenium, developers can test all their mobile apps including native, mobile web and hybrid apps, across virtual and real devices.

Enterprise Features: The Sauce Labs secure proxy supports testing of pre-production apps, APIs and back-ends. Account provisioning is also available with Team Management and SSO.

Videos and Screenshots: Developers can count on a complete set of analysis tools including video, screenshots, logs and metadata to help quickly identify issues with their apps.

With Sauce Labs, Appium is deeply integrated with the testing platform, enabling cross-platform testing on multiple mobile OS and device types. Not only this, all the Appium servers are set up and maintained by Sauce Labs. Sauce Labs is a primary contributor to the Appium project, so new versions of Appium are guaranteed to work better on Sauce than any other cloud.

Along with instant availability, you also get massive scalability. You won't need to worry about provisioning additional servers and devices for peak times, spinning up a new device takes a few seconds. Because Sauce Labs prepares every new device, you can rest assured your test results won't be affected by previous test run on the same device. All you'll need to do is sign up for an account, and decide the number of virtual and real devices you need to power your mobile QA strategy.

SAUCELABS
Learn more at saucelabs.com

KEY TAKEAWAYS

- Manual testing can't keep pace with automated testing. If you've adopted, or are in the process of adopting CI, your testing needs to be automated.
- Automated mobile testing is best served up as a cocktail of virtual & real devices. Emulators/Simulators are used early and often for a majority of tests, and real devices are used for in-depth mobile app testing where virtual devices alone are unable to do the job, where the highest fidelity is required, or where budgets are more expansive.
- Appium, Espresso and XCUITest are the leading open-source test automation frameworks for mobile. They facilitate cross-platform testing and orchestrates both virtual and real devices for a holistic mobile testing solution.
- Building and maintaining a private mobile testing grid is tedious. It throws a wrench in automation efforts, hinders scalability, involves extensive maintenance, and is expensive in the long run. A cloud testing vendor that provides both emulators and real devices in the cloud is the better option for mobile testing.
- Sauce Labs delivers instantly available virtual and real devices in the cloud. The Sauce Labs cloud is powered by Appium, and allows for massive concurrency, testing of any mobile app type, and includes powerful troubleshooting and collaboration features.
- For any organization that competes in today's fierce mobile economy, a robust mobile testing solution like Sauce Labs can be a game changer.

SAUCELABS

Learn more at saucelabs.com

ABOUT SAUCE LABS

Sauce Labs ensures the world's leading apps and websites work flawlessly on every browser, OS and device. Its award-winning Continuous Testing Cloud provides development and quality teams with instant access to the test coverage, scalability, and analytics they need to deliver a flawless digital experience. Sauce Labs is a privately held company funded by Toba Capital, Salesforce Ventures, Centerview Capital Technology, IVP and Adams Street Partners. For more information, please visit saucelabs.com.

SAUCELABS

SAUCE LABS INC. - HQ
116 NEW MONTGOMERY STREET, 3RD FL
SAN FRANCISCO, CA 94105 USA

SAUCE LABS EUROPE GMBH
NEUENDORFSTR. 18B
16761 HENNIGSDORF GERMANY

SAUCE LABS INC. - CANADA
134 ABBOTT ST #501
VANCOUVER, BC V6B 2K4 CANADA