DZone

THE DZONE GUIDE TO

MOBILE DEVELOPMENT

2015 EDITION

BROUGHT TO YOU IN PARTNERSHIP WITH

ALTOVA backendless Couchbase perfecto mobile SAUCE LABS Telerik

Dear Reader,

You've heard the mantra: more mobile, more of the time. You know the numbers. Active mobile broadband subscriptions in developed countries have more than quadrupled over the past seven years. There are now more mobile users than desktop users. Users spend 60% of their digital media time on smartphones on tablets, and 89% of that time is spent in mobile apps, not web browsers. None of these trends are new—but mobile adoption is only just reaching that truly critical mass.

This means both excitement and headaches for mobile developers. Until this year, "mobile first" was a goal for the farsighted but a dream for those of us who just needed to release now. But now "mobile first" is an absolute requirement for any app that wants to cross all verticals—and more and more users are expecting a seamless experience across all devices as well.

Dev-level mobile challenges haven't changed radically over the past year. Neither of the two dominant platforms is eclipsing the other. Native apps are still a lot more performant than web apps. Devices are getting more powerful, but connectivity is still a serious issue. The list goes on—check out our Key Research Findings for more details.

But the mobile development toolchain is getting better. Android Studio 1.0 was released last December, to great love and fanfare, and is already on a stable v1.2. Swift is enjoying blindingly fast adoption, and Apple just promised to make Swift open-source by the end of this year. Use of cross-platform tools is also growing quickly—up 25% from last year, among our survey respondents.

And the promise of IoT creates new possibilities for virtually every mobile application. What notifications would be super-cool to send to someone's wrist? Car? Home lighting system? When do I need to start thinking "wearable first," just as I begin to embrace the "mobile first" ideal?

We can't answer all these questions for you. The mobile market is too mobile for that. But we can give you enough information about the mobile development landscape to simplify your decisions and make your mobile-first coding easier. The **2015 DZone Guide to Mobile Development** includes views from developer, user, and infrastructure perspectives; a sweet listing of platforms and frameworks to facilitate mobile development; plus a fun glance at pain points encountered by mobile developers.

Check it out, make some apps, and let us know what you think.

JOHN ESPOSITO
EDITOR-IN-CHIEF, DZONE RESEARCH
RESEARCH@DZONE.COM

Table of Contents

Credits

EDITORIAL
John Esposito
research@dzone.com
EDITOR-IN-CHIEF
Jayashree Gopal
DIRECTOR OF RESEARCH
Mitch Pronschinske
SR. RESEARCH ANALYST
Benjamin Ball
RESEARCH ANALYST
Matt Werner
MARKET RESEARCHER
Moe Long
MARKET RESEARCHER
John Walter
CONTENT CURATOR
Lauren Clapper
CONTENT CURATOR
G. Ryan Spain
REFCARDZ EDITOR

BUSINESS
Rick Ross
CEO
Matt Schmidt
PRESIDENT & CTO
Kellet Atkinson
GENERAL MANAGER
Matt O'Brian
DIRECTOR OF BUSINESS DEVELOPMENT
Alex Crafts
sales@dzone.com
VP OF SALES
Chelsea Bosworth
MARKETING ASSOCIATE
Chris Smith
PRODUCTION ADVISOR
Brandon Rosser
CUSTOMER SUCCESS ADVOCATE
Jillian Poore
SALES ASSOCIATE
Jim Howard
SALES ASSOCIATE

ART
Ashley Slate
DESIGN DIRECTOR

Special thanks to our topic experts Alexander Podelko, Alex Rosemblat, David Farley, Sergey Chernyshev, Colin Scott, and our trusted DZone Most Valuable Bloggers for all their help and feedback in making this report a great success.

WANT YOUR SOLUTION TO BE FEATURED IN THIS OR COMING GUIDES?
Please contact research@dzone.com for submission information.

LIKE TO CONTRIBUTE CONTENT TO COMING GUIDES?
Please contact research@dzone.com for consideration.

INTERESTED IN BECOMING A DZONE RESEARCH PARTNER?
Please contact sales@dzone.com for information.

Executive Summary

The second DZone Guide to Mobile Development addresses the questions: *How are developers building mobile applications*, and *how can developers build better mobile applications with less pain and greater user satisfaction?*

This guide answers these questions in three ways: (1) key research findings, based on a survey of more than 500 developers; (2) four articles containing recommendations from expert mobile developers and solutions providers; and (3) a solutions directory presenting summary data on almost 50 mobile application development platforms and frameworks.

Key Takeaways

To learn what tools and techniques mobile developers are using and where problems in mobile development arise, we surveyed more than 500 developers (95% confidence level, 5% confidence interval).

Key takeaways include:

- Use of cross-platform tools (e.g. Cordova/PhoneGap, Xamarin) is growing rapidly. Over the past year, usage has jumped 10% (from 41% of respondents last year to 51% this year).
- Fragmentation is still the biggest headache for mobile developers, despite increased use of cross-platform development tools. Among our survey respondents, the top two pain points during mobile development are testing on different hardware and screen sizes (56%) and building native apps for multiple platforms (52%).
- More and more companies are building mobile applications in-house. 70% of survey respondents work for organizations that develop mobile apps/sites, up from 51% last year.

For more on these and over twenty other data points, see Key Research Findings on page 4.

Recommendations

To learn how to build mobile apps with less pain and better results, we interviewed a wide range of mobile developers, on the one hand, and product owners and engineers who create solutions to simplify mobile development, on the other. Recommendations from both perspectives are included in the four articles below, which cover:

- The relation between end-user satisfaction and pain points encountered during mobile development
- Advantages and disadvantages of native, web, and hybrid mobile app development and deployment
- The convergence of mobile devices and the Internet of Things
- The impact of Mobile-Backend-as-a-Service (MBaaS) on mobile app development

MADPs and Mobile Web Frameworks

To help mobile developers find the right tools for simplifying mobile development and deployment, we gathered multiple data points on 33 Mobile Application Development Platforms (MADPs) and 12 popular mobile web frameworks. Results are listed in the solutions directory on page 31.

Key Research Findings

DZone surveyed over 500 IT professionals with some involvement in mobile technology for the 2015 *Guide to Mobile Development*, providing key information on mobile monetization, tool usage, production levels, and platform prioritization. The largest demographic segments are developers (30%) and development team leads (26%). 44% of respondents come from small organizations (under 100 employees) and 56% come from large organizations (100 or more employees). The majority of respondents are headquartered in the US (41%) or Europe (34%).

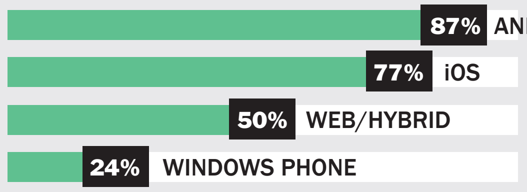
MORE MOBILE HOBBYISTS AND EMPLOYEES THAN FREELANCERS

An overwhelming majority (70%, up from 51% last year) of those surveyed work in an organization that develops mobile apps, and 69% (up from 48% last year) of all respondents have participated in their organization's mobile development. Freelancing is still not as popular (35% of respondents develop mobile apps on their own, down from 37% last year) as those developing as hobbyists (42%), which have also seen a decrease from last year's results (down from 56%). 19% of respondents don't expect to see a return on investment from their apps, which is down from 30% last year.

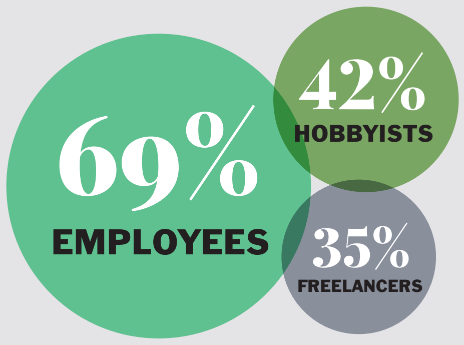
ANDROID STILL MORE POPULAR THAN IOS, BUT NOT AS MUCH; NATIVE DEVELOPMENT UP OVERALL

Android continues to be the most popular platform for organizations and individual developers, with 87% of respondents saying they are targeting the Android platform. iOS has closed the margin, with 77% of respondents saying they are targeting iOS—the difference last year was 14%. Interestingly, non-native development seems to be decreasing slightly: 50% of respondents say they are developing for web or hybrid (down from 56% last year). Windows Phone remains a distant third, targeted by 24% of respondents.

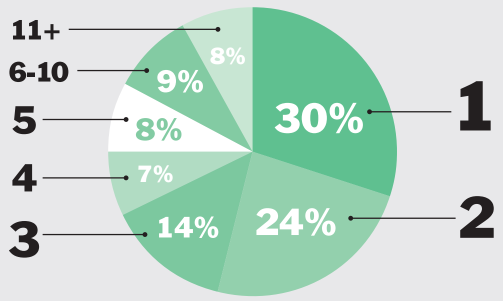
02. **WHICH DEVICE PLATFORMS DO YOU OR YOUR ORGANIZATION TARGET FOR MOBILE APPLICATION DEVELOPMENT?**



01. **TYPES OF MOBILE DEVELOPERS**



03. **IN THE PAST YEAR, HOW MANY MOBILE APPLICATIONS HAS YOUR ORGANIZATION RELEASED?**



INDIVIDUAL DEVELOPERS BECOMING MORE EFFICIENT

App development time can vary widely based on a host of factors, but finding out how much time it generally takes organizations and individuals to develop apps is useful for discovering industry opinions about how long an app should take to complete or how large a project should be. The top three answers for organizations were 12 weeks (15%), 8 weeks (12%), and 4 weeks (12%), which is exactly the same as last year. This suggests consistency for app turnaround time within organizations. For individuals working by themselves, timetables gravitate toward 8 weeks (15%), with 4 weeks (13%) and 6 weeks (11%) close behind. These timetables are all smaller than last year's results, which could be explained by a number of factors—more legacy code, more APIs, or more developer knowledge. As for the number of apps completed per year, chart #3 shows statistics for the number of apps organizations are churning out.

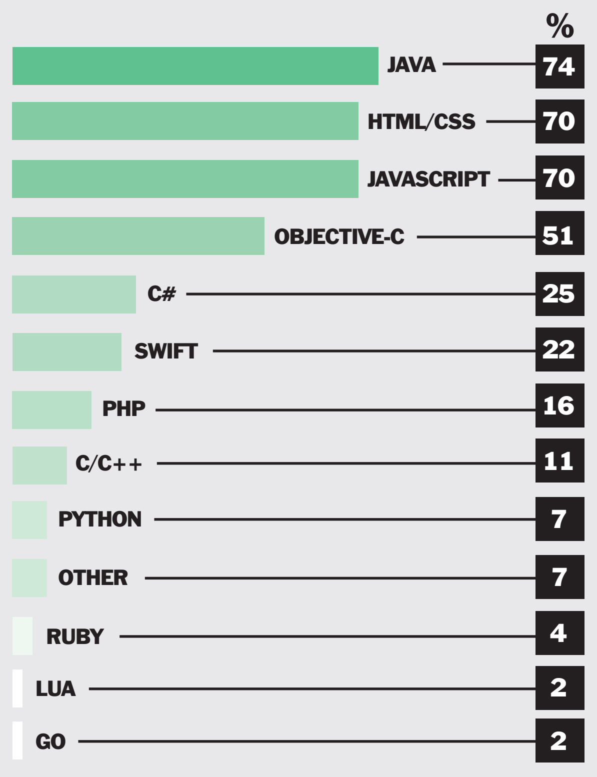
CROSS-PLATFORM TOOLS USED BY NEARLY HALF OF MOBILE DEVS

51% of respondents say they or their organizations are using cross-platform tools such as Apache Cordova/PhoneGap, up from 41% last year. This is the most popular type of mobile tool, with IaaS/PaaS as the second most crucial development utility at 29% usage (up from 20% last year). For programming languages, Java is very popular in this group of enterprise developers (74%), more popular, in fact than HTML/CSS (70%) or JavaScript (70%). Objective-C for iOS is fourth with 51%, while Swift is being adopted quickly (already at 22%). C#, the language of Windows Phone, is fifth with 25%. The mobile usage percentages for the rest of the programming languages can be seen in the chart below.

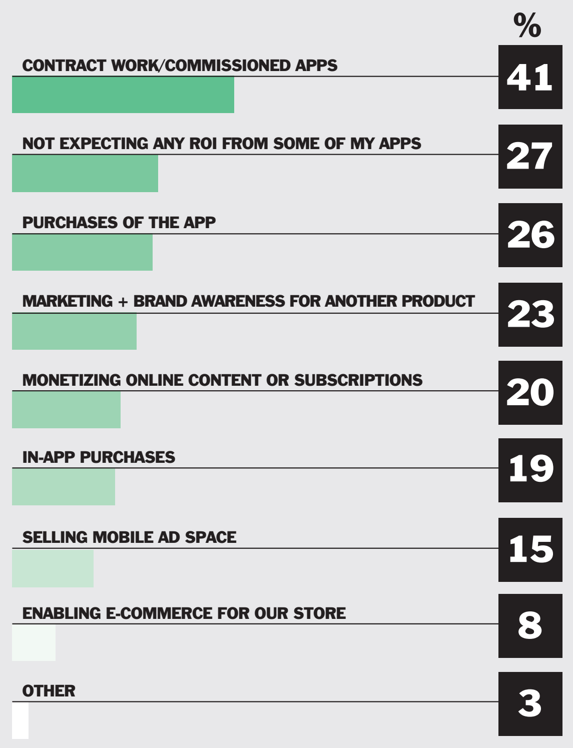
DEVELOPERS STILL RELY ON COMMISSIONED APPS FOR REVENUE

Commissioned apps (41%) continue to be the number one way respondents expect to make money as a mobile developer. The actual purchase of the app (26%) is the second most popular way developers expect to make money from their app. Marketing/brand awareness of another service (23%) and monetizing online content (20%) are other strong options. 19% of respondents said in-app purchases are the way they expect to see a return on investment. Unfortunately, 27% of respondents said they do not expect to see a return on their investment.

04. **WHICH PROGRAMMING LANGUAGES DO YOU OR YOUR ORGANIZATION USE FOR DEVELOPING MOBILE APPS?**



05. **WHERE DO YOU OR YOUR ORGANIZATION EXPECT TO GAIN MONETARY VALUE FROM YOUR APPS?**



Of drivers who own family cars, 86% would rather be ***driving one of these.***

Move into the Fast Lane with MobileTogether®

In a little over two days, with one developer, we created a full-featured survey application and asked Android™, iOS®, and Windows® mobile users about their dream cars. 86% wanted to go faster.

ALTOVA® MobileTogether®

Ditch the Minivan of Mobile Development

It's no longer practical to wait weeks or months for a cutting-edge mobile business solution. Your team needs to create, test, and deploy to all devices at the speed of business. With MobileTogether, we've removed the roadblocks.

- Drag and drop UI design process
- Powerful visual & functional programming paradigm
- Native apps for all major mobile platforms
- Connectivity to SQL databases, XML, HTML, and more
- Deploy full-featured business app solutions in record time
- Pricing so affordable you might just have money left to start saving for that sports car

www.altova.com/MobileTogether

Enterprise Mobile Development at the Speed of Business

Enterprise productivity is the last mile of mobile. Mobile apps in the enterprise fail so often because they don't fulfill end user requirements and they take too long to complete. Development teams are challenged to deliver enterprise apps on time and under budget in an ever-changing landscape of mobile technologies, mobile OS version upgrades, and end user preferences.

THE USER IS MOBILE; THE DEVICE IS SECONDARY

A fatal misstep occurs when enterprises adopt a mobile-first approach by assuming that users only require access to in-house apps on smartphones. While native smartphone or tablet access is indeed an absolute requirement, users also spend significant time on a laptop or desktop computer each day. Device use varies on a per-user basis, as well. Any mobile-first approach must take this into account, delivering seamless, in-the-moment productivity to employees in any scenario, on their device of choice at that time.

THE NEED FOR SPEED

The second factor compounding mobile-first in the enterprise is development speed. With most apps taking several months—or longer—to build, dev teams face a backlog of projects, with requirements that have likely changed in the interim.

A paradigm shift is required to get enterprise apps to end users —regardless of the device they're holding – in a time closer to the pace of business today. Competitive companies simply cannot afford to invest in mobile development approaches that prevent quick turnaround and agility.

A paradigm shift is required to get enterprise apps in end users' hands—on any device—in a time closer to the pace of business

MOBILETOGETHER IS THE SHIFT

With Altova MobileTogether, the mobile-first focus is on the user. Your developers build a single solution that users access via native apps for iOS, Android, Windows Phone, and even Windows 8. MobileTogether takes care of rendering the solution properly for each device, form factor, and screen orientation. Its powerful visual and functional programming approach means in-house developers can draw on existing expertise to develop cutting edge mobile solutions that are simultaneously available on all platforms—in hours or days, not weeks or months.

WRITTEN BY ALEXANDER FALK
PRESIDENT AND CEO, **ALTOVA**

MobileTogether by Altova

ALTOVA®

MobileTogether is an easy-to-use framework that enables companies of any size to quickly design and deploy custom enterprise mobile solutions that are accessible on any mobile device.

CASE STUDY

Common use cases include in-house business workflow processes, BI dashboards, forms-based data gathering, image capture, geo-information for outside sales people or delivery tracking solutions, and more. MobileTogether excels at interfacing backend data with mobile devices, with support for all popular relational databases and access to data from remote servers via XML, HTML, or HTTP request. MobileTogether includes beautiful design elements, text and table formatting options, and more than a dozen highly customizable chart types to create a nuanced business intelligence solution that performs elegantly on iOS, Android, Windows Phone 8, Windows 8, or even in an HTML5 browser-based client for other mobile devices or desktop workstation.

APP CATEGORY

- Web
- Native

MBAAS INCLUDED

Yes

LANGUAGES USED

HTML/CSS
SQL XML
XPATH
XQUERY

TOOLS INCLUDED

- IDE
- Version Management
- Distribution Management

CUSTOMERS

Altova's 4.8 million customers include 90% of the Fortune 500

BLOG blog.altova.com | TWITTER @altova | WEBSITE altova.com

Concerns and Complaints:

Mobile Development from Dev and User Perspectives

BY JOHN WALTER

QUICK VIEW

01
Testing applications efficiently and developing for multiple platforms are major concerns for developers.

02
Functional errors and privacy are the top complaints by application users.

03
When testing, developers should rely on real devices over emulators when possible.

04
Securing applications is a struggle for developers, and a major concern for users.

Mobile development can be both daunting and exhausting. The technology is constantly evolving—new versions of iOS and Android come out every year, and the constant barrage of new physical devices comes with a plethora of unique form factors and other hardware considerations. Mobile developers deal with issues other devs don't—and yet mobile releases must be quick.

This perfect storm of continuous platform-level change plus device-level variation can cause serious headaches for mobile developers, both during and after development. And issues during development can (and often will) lead to major headaches for users. But how exactly do dev-level and user-level issues relate, and what can developers do to keep users happy?

This article will identify major concerns during development and major complaints from users after release. We'll look at multiple data sources, including some original to this Guide, to draw a correlation between pain points reported by mobile developers and frustrations reported by users. Finally, we'll suggest ways for developers to address both sets of concerns.

MOBILE DEVELOPERS' CONCERNS

It isn't easy, developing for mobile. By the time an app is ready to release, a new OS version comes out, or a new handset that everyone just has to own. And if an app is successful in the App Store, there's immediate pressure to release it in the Google Play store.

Mobile-specific hurdles arise before coding even starts. First, developers have to come up with a scalding idea (not as easy as it sounds); then they have to translate the idea into a workable design at multiple levels. Enterprise mobile coders have more hoops to jump through ("What data can I access?" or "What systems do users need access to?") before they can actually begin development. After coding is finished, developers must make sure it's secure, and then test it rigorously—on every platform they can.

Each one of these steps presents a potential problem. According to DZone's Mobile Development Survey, the number one pain point during mobile development is testing efficiently on all of the different hardware sets and screen sizes (56% of our audience). Some developers don't have access to all of the devices they'd want to test on, so they have to use emulators, which can be unreliable and ineffective [1].

Another concern? Developing native apps for multiple platforms. Almost 52% of our audience said this is a major issue during development. The more platforms they develop for, the more testing is required. The more testing is required, the more likely it is that bugs will make it to market. While plenty of tools exist to help ease the process for coding a cross-platform application, these tools may not always be reliable and can present their own usability challenges.

A recent study of mobile-related questions on Stack Overflow [2] revealed more specific issues. According to Rosen and Shihab, two features of APIs particularly stood out as majors concern for developers: limited control and constant change. Often, the change causes problems because documentation lags behind releases. Sometimes developers are led to believe that a function is still present in one version of an API that was actually phased out in a previous update.

After an app is released, it's important to take updates into consideration. Business Insider observed that regularly updated apps were more successful and favorably reviewed than apps that were not updated regularly [3]. In 2014, companies like Amazon and Geico updated their apps around 20 times and were some of the highest rated apps in the App Store or Google Play store. Obviously this is a higher update rate than that of (say) most enterprise desktop apps.

MOBILE USERS' COMPLAINTS

Ultimately, developers create applications to be used mostly by non-developers. And now, more than ever, users have a direct line to developers who build those apps. Users have, and feel that they have, a responsibility to provide informative, candid feedback to developers in order to improve mobile apps.

"Sometimes, developers are led to believe that a function is still present in one version of an API that was actually phased out in a previous update."

Sometimes, this doesn't work out perfectly. Even the top apps in the App Store have some pretty scathing reviews that offer little useful information to the developer.

We can get a lot more granular. McIlroy et al. [4] recently conducted a study identifying the top user complaints once an app has made it into an app store. Their results were pretty interesting in light of the various issues developers face when creating these apps.

According to this study, the most frequent user complaint is functional error, by which is meant that the app wasn't doing something the user expected it to do. And really, this is a bug (even if a unit or integration test would catch). Imagine an email application that didn't notify you of a new email until you opened the application. How could such a glaring oversight make it all the way through the development process and into the App Store?

Interestingly enough, the second most frequent user complaint is lack of features. This isn't necessarily a fault of the developer—people can have unrealistic expectations of what an application should be able to do. Or maybe the design needs to be more ambitious, or based on a better understanding of user needs.

Also interesting is the difference between frequency and impact of complaints While functional error was the most frequent complaint, the most impactful complaint—measured by the ratio of one-star to two-star ratings—regarded privacy. Certainly, an email client accessing your information without your knowledge is more of a concern than it not sending notifications, and leads to a very severe rating.

CONNECTING THE DOTS

Now that we know some of the biggest issues for mobile developers and mobile users, it's time for some specific recommendations. Obviously, a major issue for developers is the sheer amount of devices for which they must develop. This issue isn't going away—if anything, it'll only get worse as wearables become more and more prevalent. And clearly, keeping a pulse on user feedback is a great way to address concerns after the fact. But how can developers best avoid these issues in the first place?

1. Testing. Check every use case on every device (or at least as many as possible). According to DZone's Mobile Development Survey, only 13% of developers use automated testing tools. 20% of respondents claimed they don't test before deployment—their audience acts as guinea pigs for their applications.

2. Security. Given the amount of information people store on and access from their mobile devices, security is a major concern from a user's perspective. And it's one of the biggest concerns for developers as well. 33% of DZone's audience said that securing data within their mobile apps is a struggle, whether it's integrating their app into an existing infrastructure, or making sure users were authenticated securely. Addressing these concerns before release is imperative to app success.

3. Design. Ensure that your design works correctly on multiple platforms. The constantly changing landscape (both hardware and software) will always be an issue for developers. Emulators, unfortunately, just aren't as good as the real thing [1]. It may be worth it to invest in physical hardware and test on multiple OS versions.

4. Audience Monitoring. Gather as much data as you can once your application is deployed. How is your application being used? Does your audience think your app lacks certain functionality—maybe something they expect, maybe something they really, really want? Is your app being used in a way you didn't intend? Feedback is your best friend.

Users, of course, are never going to fully understand the headache developing for mobile can be—nor should they. But some user complaints can be avoided. And are developers going to continue being frustrated by mobile development? Yes, of course—mobile devices are both highly constrained and also changing incredibly quickly. But knowing how users are feeling can help you decide what to prioritize during mobile development and testing.

[1] http://testdroid.com/news/rely-only-on-real-emulators-vs-devices

[2] Rosen, Christoffer, and Emad Shihab. "What are mobile developers asking about? A large scale study using stack overflow." Empirical Software Engineering (2015): 1-32.

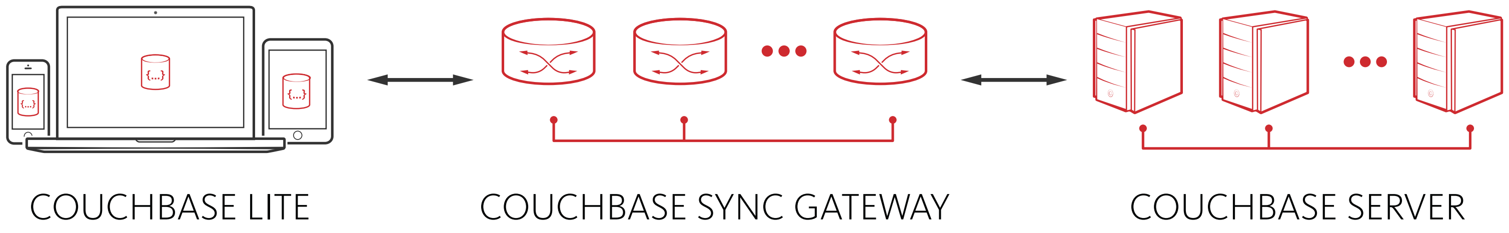
[3] http://www.businessinsider.com/app-update-strategy-and-statistics-2015-1

[4] McIlroy, Stuart, et al. "Analyzing and automatically labelling the types of user issues that are raised in mobile app reviews." Empirical Software Engineering (2015): 1-40.

JOHN WALTER is a Topic Owner and Producer of content for DZone's Mobile and Internet of Things portals. He has a passion for community engagement. In his spare time, John writes and directs short films.

COUCHBASE MOBILE

BUILD BETTER APPS FASTER



Couchbase Mobile is a NoSQL database solution that delivers the full power and flexibility of NoSQL to mobile. It's engineered to provide fast and consistent access to your data, with or without a network connection, removing the network dependency that traditional service-based approaches require.

Get started at http://developer.couchbase.com/mobile/.

Couchbase

The Offline Challenge: Building Mobile Apps That Always Work

As more enterprises focus on delivering great user experiences (UX) with a mobile-first strategy, it's becoming increasingly important to actually deliver that experience. It's also extremely important to not underestimate and then over-simplify mobile offline needs. Your apps always have to work, period.

While app design and ease of use are key factors in UX, even more important is the ability to deliver an application that always works and is always fast, both online and offline. It's actually quite simple: for a great UX, build apps that look great, behave as expected, always work, and are always fast—all the time.

One of the key UX problems that most apps run into is availability. Sounds like a network problem, right? Sort of. Actually, while the network contributes to an app not being available, it is not the root cause. The real problem is where the app's data is located. Data location dictates app reliance on the network. If an app has to access the cloud every time it needs to read or write any piece of data, then it's going to have problems. If the network isn't available, the app can't get to the cloud, can't access its data, and isn't going to work. If the network is slow then getting data from the cloud will be slow and the app will be slow.

Offline-First is the new Mobile-First

What this means is you need data local, you need data in the cloud, and you need something that will securely keep those two data-stores in sync. Couchbase Mobile does this for you.

Learn more at developer.couchbase.com/mobile.

WRITTEN BY WAYNE CARTER
CHIEF ARCHITECT OF MOBILE, **COUCHBASE**

Couchbase Mobile by Couchbase

Couchbase

Couchbase Mobile is a NoSQL database solution that delivers the full power and flexibility of NoSQL to mobile. It's engineered to provide fast and consistent access to your data, with or without a network connection, removing the network dependency that traditional service-based approaches require.

CASE STUDY

Ryanair is the largest European airline by scheduled passengers carried and the busiest international airline by passenger numbers. With Couchbase Mobile, Ryanair solved the performance and scaling challenges for their mobile app and decreased travel-booking times from 5 minutes to 2 minutes. Bookings under the original architecture took over 5 min to complete with a REST architecture backed by a relational database. Booking information had to be retrieved from the server during the booking request, causing latency and performance issues. Bookings with Couchbase Mobile architecture now take under 2 min to complete. Booking information is stored locally on the device and updates are pushed as needed, resulting in less latency and higher performance.

STORAGE MODEL

Document, Key-Value, Distributed Cache

LANGUAGE DRIVERS

JAVA .NET NODE.JS GO PHP C

PRODUCT FEATURES

- User management
- Push/pull notifications
- Object storage
- Cloud storage
- Caching
- Social network integrations

TEXT SEARCH

Via ElasticSearch and Solr

APP CATEGORY

- Web
- Native
- Hybrid

CUSTOMERS

- Ryanair
- General Electric
- PVH
- Crowd Comfort
- Coyote
- AOL
- AT&T
- eBay

BLOG blog.couchbase.com | TWITTER @Couchbase | WEBSITE couchbase.com

The 3 Types of Mobile Experiences

BY JEREMY WILKEN

QUICK VIEW

01
Because no translation process occurs, native mobile applications offer the highest level of performance.

02
Web applications are easiest to maintain as application code is on an accessible server.

03
The abstraction layers in hybrid applications often prevent native-like performance, but they can be deployed to multiple platforms with minimal effort.

There are several ways to build applications for mobile devices, each with strengths and weaknesses. There are three basic types of mobile apps: native apps, mobile websites, and hybrid apps.

The debate continues—is it better to build an application written directly for a native platform or to build an application for the web? And with the proliferation of hybrid apps, mobile experiences can be developed in three unique ways. This article will present the strengths and weaknesses of these basic types of mobile apps: native apps, web apps, and hybrid apps. Given this information, you'll be able to draw your own conclusions for which type is the best fit for you.

In figure 1, you can see how the three types compare in design and architecture. The diagram also shows also how each type of app would access a database or web service API to load data.

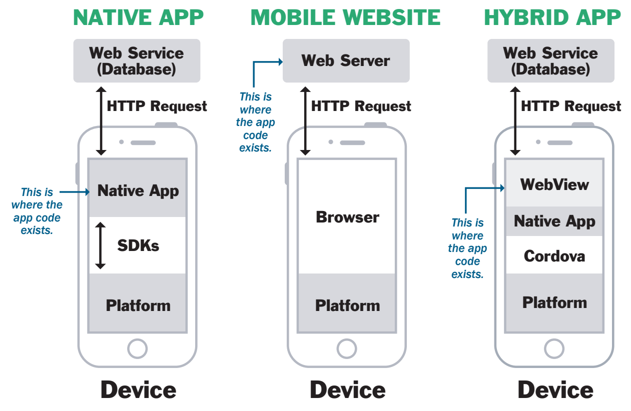


Figure 1 *- Native app, mobile website, and hybrid app architectures compared side by side.*

NATIVE MOBILE APPS

Native apps are written using the default language for the mobile platform: Objective C or Swift for iOS, and Java for Android. Native apps are compiled and executed directly on the device. Using the platform SDK (or API), the app can communicate with the platform to access device data or load data from an external website using HTTP requests.

Both iOS and Android provide a set of tools to enable developers to leverage the platform features in a controlled manner through predefined APIs. There are tools, both official and unofficial, which can also aid in the development of native apps. It is also common for developers to use frameworks in their native app to make development easier.

NATIVE APP ADVANTAGES

The native app comes with a number of benefits over the other types. The benefits revolve around being tightly integrated with the device platform.

- **Native APIs.** Native apps can use native APIs directly in the app, making the tightest connection to the platform.
- **Performance.** Native apps boast the highest levels of performance, since no translation processing takes place.
- **Same environment.** Native apps are written with native APIs, which is helpful for developers familiar with the languages used.

HYBRID APPS PROVIDE A ROBUST BASE FOR MOBILE APP DEVELOPMENT WHILE STILL BEING ABLE TO USE THE WEB PLATFORM.

NATIVE APP DISADVANTAGES

The disadvantages of native apps are generally the level of difficulty in developing and maintaining them.

- **Language requirements.** Requires proficiency in the platform language (for example Java for Android apps) and knowledge of platform-specific APIs.
- **Not cross platform.** Native apps can only be developed for one platform at a time.
- **High level of effort.** Typically native apps require more work and overhead to build, increasing costs.

Native apps may be best suited for developers who have a command of Java and Objective C, or for teams with extensive resources and a strong need for the benefits offered only by native apps.

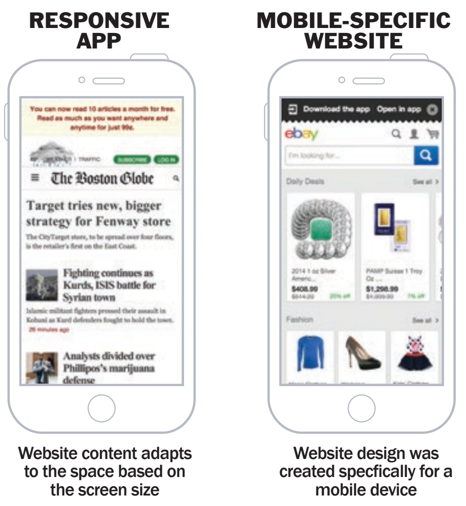

RESPONSIVE APP
MOBILE-SPECIFIC WEBSITE
You can now read 10 articles a month for free. Read as much as you want anywhere and anytime for just 99¢.
The Boston Globe
Target tries new, bigger strategy for Fenway store
Fighting continues as Kurds, ISIS battle for Syrian town
Analysts divided over Phillipos's marijuana defense
Download the app
Open in app
ebay
Daily Deals
Fashion
Website content adapts to the space based on the screen size
Website design was created specfically for a mobile device

Figure 2 - *Example mobile websites, a responsive site from Boston Globe (left) and a mobile specific website from eBay (right)*

MOBILE WEBSITES (WEB APPS)

Mobile websites are applications that work well on a mobile device, but are accessed through the mobile browser. Sometimes they are called Web Apps. Most simply, they are websites viewed on a mobile device in a mobile browser, which are designed to fit a mobile device screen and interact via touch (see next column, top).

Some websites have a unique version of the normal website that has been developed specifically for use on a mobile device. Perhaps you've visited websites that redirect you on a mobile device to a limited-feature application, often on a subdomain such as m.ebay.com. In other cases, the design adjusts to the form factor and screen size using a technique called responsive design—for example, www.bostonglobe.com. Depending on the size of the browser window, the website reflows the page to fit better on a smaller screen and perhaps even hides content.

MOBILE WEBSITES ADVANTAGES

Mobile websites enjoy a number of benefits, primarily in the level of effort and compatibility on devices.

- **Maintainability.** Mobile sites are easy to update and maintain without the need to go through approval processes or update installations on devices, because application code is located on the server.
- **No installation.** Since the app exists on the internet, it doesn't require installation on mobile devices.
- **Cross platform.** The app works with any mobile device browser, allowing your application to be accessible from any device.

MOBILE WEBSITES DISADVANTAGES

Mobile websites run inside of a mobile browser, which is the major cause of limitations and disadvantages.

- `**No native access.** Since the app is run in the browser, it has no access to the native APIs or the platform, just the APIs provided by the browser.
- **Requires keyboard to load.** The user has to type an address in a browser to find or use your mobile website, which is more difficult than tapping an icon. (Users can also create custom icons for mobile web apps, but this too is another additional step.)
- **Limited user interface.** It is difficult to create touch friendly applications, especially if you have a responsive site that has to work well on desktops as well as mobile devices.
- **Mobile browsing is declining.** The amount of time users browse the web on a mobile device is declining, while app usage increases.

Mobile websites can be important even if you have a mobile app, depending on your product or service. Research shows users spend much more time using apps compared to the mobile browser, so mobile websites tend to have a lower engagement. However you would need to test your own use cases.

BOTH IOS AND ANDROID PROVIDE A SET OF TOOLS TO LEVERAGE THE PLATFORM FEATURES IN A CONTROLLED MANNER THROUGH PREDEFINED APIS

HYBRID APPS

A hybrid app is a mobile app that contains a web view (essentially an isolated browser instance) to run a web application inside of a native app, using a native app wrapper that can communicate with the native device platform and the web view. This means web applications can run on a mobile device and have access to the device, such as the camera or GPS features.

Hybrid apps are possible because of tools that facilitate the communication between the web view and the native platform. These tools are not part of the official iOS or Android platforms, but are third party tools such as Apache Cordova. When a hybrid app is built, it must be compiled, transforming your web application into a native app.

HYBRID APP ADVANTAGES

- **Cross platform.** You can build your app once and deploy it to multiple platforms with minimal effort.
- **Same skills as web development.** Hybrid apps allow you to build mobile apps using the same skills already used to develop websites and web applications.
- **Access to device.** Since the web view is wrapped in a native app, your app has access to all of the device features available to a native app.
- **Ease of development.** Hybrid apps are easy and fast to develop, without the need to constantly rebuild to preview. You also have access to the same development tools used for building websites.

Hybrid apps provide a robust base for mobile app development while still being able to use the web platform. You can build the majority of your app as a website, but anytime you need access to a native API, the hybrid app framework can provide a bridge to access that API by using JavaScript. You can detect swipes, pinches, and other gestures like you can detect clicks or keyboard events.

HYBRID APP DISADVANTAGES

- **Web view limitations.** The application can only run as well as the web view instance, which means performance is tied to the quality of the platform's browser.
- **Native via plugins.** Access to the native APIs you need may not be currently available, and may require additional development to make a plugin to support it.
- **No native user interface controls.** Without a tool (such as Ionic or PhoneGap), developers would have to create all of the user interface elements from scratch.

To reiterate, the point of this exercise is not to determine which experience is best. All three app types have clear strengths and weaknesses. The choice is dependent on a number of factors, such as a developer's skill, the purpose of the app, and the development timeline. Each option should be explored and weighed heavily to come to a prosperous conclusion.

JEREMY WILKEN is a senior software engineer, where he develops mobile apps with Ionic, crafts user interfaces with AngularJS applications for big data systems, and builds web service layers with NodeJS and Go. He lives in Texas with his wife and daughter, and when he isn't coding you can find him brewing his own beer.

diving deeper

INTO MOBILE DEVELOPMENT

top 10 #mobile twitter feeds

TO FOLLOW RIGHT AWAY

@SUNDARPICHAI	@ASYMCO	@GRUBER	@ANDROIDDEV	@FLEXMONKEY
@JOEBELFIORE	@RCASTANONM	@HELLOJACKSON	@RWENDERLICH	@MATIASDUARTE

mobile zones

LEARN MORE & ENGAGE YOUR PEERS IN OUR MOBILE-RELATED TOPIC PORTALS

Mobile Zone

dzone.com/mz/mobile

The Mobile Zone features the most current content for mobile developers. Here you'll find expert opinions on the latest mobile platforms, including Android, iOS, and Windows Phone. You can find in-depth code tutorials, editorials spotlighting the latest development trends, and insight on upcoming OS releases. The Mobile Zone delivers unparalleled information to developers using any framework or platform.

IoT Zone

dzone.com/mz/iot

The Internet of Things (IoT) Zone features all aspects of this multifaceted technology movement. Here you'll find information related to IoT, including Machine to Machine (M2M), real-time data, fog computing, haptics, open distributed computing, and other hot topics. The IoT Zone goes beyond home automation to include wearables, business-oriented technology, and more.

Web Dev Zone

dzone.com/mz/html5

Web professionals make up one of the largest sections of IT audiences; we are collecting content that helps web professionals navigate in a world of quickly changing language protocols, trending frameworks, and new standards for user experience. The Web Dev Zone is devoted to all things web development—and that includes everything from front-end user experience to back-end optimization, JavaScript frameworks, and web design. Popular web technology news and releases will be covered alongside mainstay web languages.

top mobile refcardz

Getting Started with PhoneGap
bit.ly/DZ-PhoneGap

Code Gems for Android Developers
bit.ly/DZ-Android

HTML5 Mobile Development
bit.ly/DZ-HTML5

Mobile Web Application Testing
bit.ly/DZ-MobileWeb

top mobile websites

Android Developer Blog
developer.android.com/develop

iOS Developer Blog
developer.apple.com/news

NSHipster Blog
nshipster.com

top mobile newsletters

Android Weekly
androidweekly.net

iOS Weekly
iosdevweekly.com

Mobile Web Weekly
mobilewebweekly.co

MARTIAL ARTS
HAS BRUCE LEE.

AUTOMATED MOBILE
TESTING HAS APPIUM.

Maybe you can't do a one-fingered push-up, but you can master automated testing for mobile applications with Appium running on Sauce Labs. Optimized for continuous integration workflows, our scalable, reliable, and secure platform enables you to run your builds in parallel, so you can get your native or hybrid iOS and Android apps to market faster.

Download Your Free Appium Bootcamp Guide.

See why these companies trust Sauce Labs.

Dropbox salesforce YAHOO! PayPal intuit VISA

SAUCE LABS

Mobile Testing Automation for a CI World

As the mobile economy grows, so does the pressure to provide excellent products. Mobile users demand high quality apps, ruthlessly abandoning or deleting those with bugs. The highly competitive market also demands that mobile teams release apps faster and more often. Add to that the challenge of supporting different device types across a fragmented market, and mobile teams are left with a daunting amount of work to ensure quality, consistency, and velocity.

Automated mobile testing done in a CI workflow can help development teams identify errors quickly and early on in the development cycle to ensure their app works as expected on the devices and platforms their customers are using. It also reduces the need for time-consuming manual testing by traditional QA teams, leaving time for QA professionals to run more valuable exploratory testing programs. Taking advantage of the ability to run automated tests in parallel is key to reducing build time and shortening dev cycles.

I am a big fan of open source, so the mobile testing platform I recommend for running functional tests is Appium, the world's leading cross-platform mobile automated testing platform (supported by Sauce Labs and a thriving community of open source developers). Appium, which could be described as 'Selenium for mobile apps,' is designed to support testing of native, hybrid, and mobile web apps across many mobile device and platform combinations, and comes with many other benefits; for example, it's easy to integrate into CI and allows teams to use any programming language or test framework.

WRITTEN BY JONATHAN LIPPS
DIRECTOR OF ENGINEERING, ECOSYSTEMS, **SAUCE LABS**

Mobile Testing Platform by Sauce Labs

sauce LABS

Sauce Labs provides a cloud-based testing platform for native, hybrid and mobile web apps. Users run Selenium, Appium, and JS unit tests written in any language on over 500 browser/OS combinations, 80 mobile emulators, and hundreds of the newest, most popular devices with no wait time.

CASE STUDY

Bleacher Report, a leading sports news site, has a monthly audience of more than 16 million unique users with more than 50% of site traffic originating from mobile devices. Previously the QA team tested their native applications manually, but without an automated testing process, their mobile apps had bugs after deployment. Unsatisfied with the process, they searched for a better solution. Recently, they started using Appium with Sauce because they could use it with their existing framework and didn't have to learn a new language or set of commands. They now regularly run 12 native application tests and 7 mobile web tests, each checking specific end points in various scenarios. Because they're run in parallel, the test suite only takes approximately 2 minutes. The mobile web tests are run at 4 different times with different parameters to cover iPhone, iPad, and in-app experiences for both.

CI TOOL SUPPORT

- Jenkins
- Travis CI
- CircleCI
- Bamboo
- TeamCity

PRICING

By number of VMs and test run minutes

LANGUAGE SUPPORT

JAVA · JAVASCRIPT · NODE.JS · PERL · PHP · C# · RUBY · PYTHON

APP CATEGORY

- Web
- Native
- Hybrid

TESTING SUPPORT

- Cross-browser testing
- Real mobile devices
- Mobile emulators
- Mobile simulators
- Appium
- Selenium

OPEN SOURCE

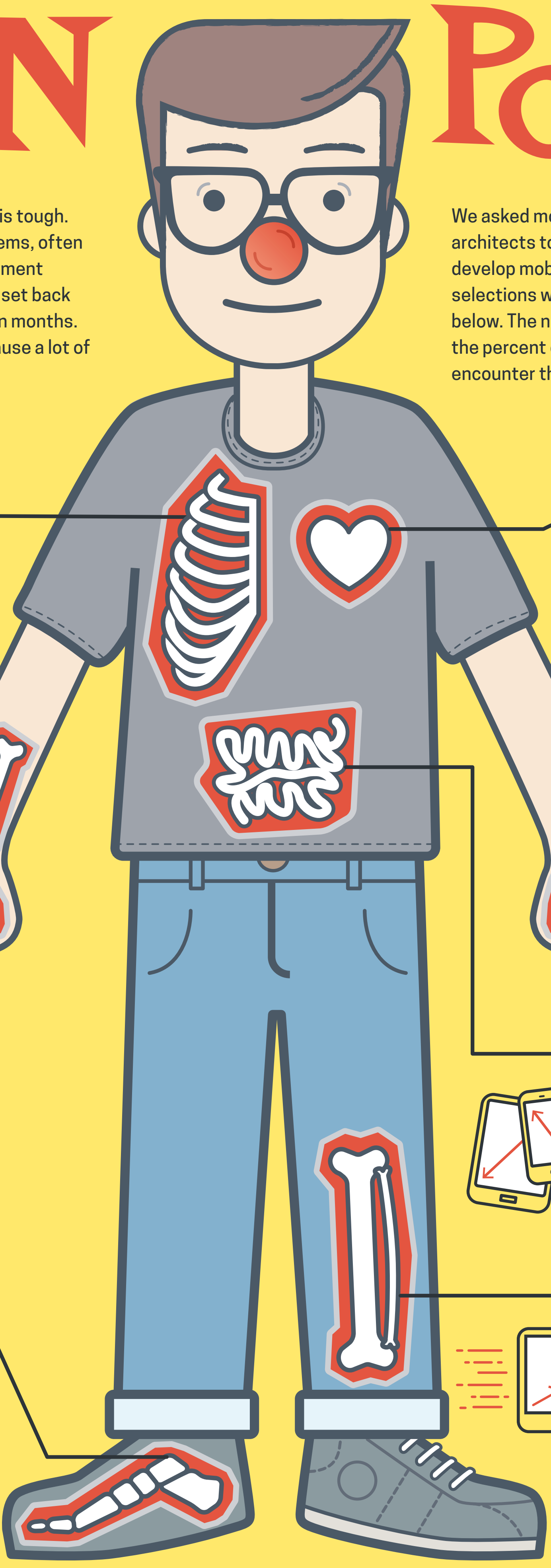
Yes

CUSTOMERS

- Yahoo!
- Capital One
- Twitter
- Travelocity
- Mozilla
- Zendesk
- Salesforce
- Puppet Labs

BLOG sauce.io | TWITTER @saucelabs | WEBSITE saucelabs.com

MOBILE DEVELOPMENT PAIN POINTS



Let's face it — mobile development is tough. Developers encounter several problems, often unforeseen, throughout the development process. Facing these problems can set back development by days, weeks, or even months. Not anticipating these issues can cause a lot of frustration and can lose you money.

We asked more than 500 software developers and architects to tell us what causes them pain when they develop mobile applications. Multiple answer selections were allowed. The results are summarized below. The number next to each icon/label indicates the percent of respondents who indicated that they encounter this pain point during mobile development.

33% 4
SECURITY

42% 3
LACK OF SKILLED MOBILE DEVELOPERS

15% 6
INTEGRATION WITH EXISTING THIRD-PARTY APPS

52% 2
BUILDING NATIVE APPS FOR MULTIPLE PLATFORMS

14% 7
COLLECTING AND UNDERSTANDING USER FLOW DATA

21% 5
INTEGRATION WITH EXISTING IN-HOUSE APPS

56% 1
TESTING ON DIFFERENT HARDWARE SETS AND SCREEN SIZES

33% 4
MAINTAINING GOOD PERFORMANCE WHEN CONVERTING DESKTOP APPS TO MOBILE

The Internet of Things and Its Impact on All Things Mobile

BY ANDREW TRICE

QUICK VIEW

01

Network infrastructures are already evolving to meet the data demands to support such complex systems.

02

More sensors will be integrated into everyday devices to aggregate more data.

03

Mobile devices will become the way we interact with and control these emerging complex systems.

04

As IoT becomes more ubiquitous, processes will be more streamlined and automated.

The emergence of mobile devices has fundamentally changed computing forever. We are now able to engage with data in new ways, nearly anywhere, and in nearly any context. From this paradigm shift emerged a platform war, new dominant tech companies, new developer ecosystems, and entirely new workflows. We are now in the midst of another such change - perhaps less noticeable to the average consumer, but with the potential to change our lives in ways that are just as dramatic—the emergence of the Internet of Things.

IoT is a buzzword that needs to be defined carefully. An IoT device can be anything that sends information to or receives information from the the Internet. These devices range from smart vehicles to medical equipment; from wearables to beacons that enable context sensitive apps on our phones; and much more. We're just now starting to crack the shell of what the Internet of Things will bring, but one thing we know for sure is that it means that data—lots of it—will be delivered right to our pockets (or wrists, or glasses), in real time, with the potential to change how we make real-time decisions. In this article we will explore what the Internet of Things can do; how it will impact our lives, our computing needs, and our interactions with the world; and, specifically, what all this means for our mobile technology infrastructures.

SO, WHAT MAKES AN IOT DEVICE?

Before we dig deeper into how the Internet of Things is going to change our lives, let's elaborate on what we mean when we talk about IoT devices. These generally fall into one of two categories: 1) sensors that distribute information across the Internet or 2) devices that receive information from the Internet and affects the real world with that data. This is not to be confused with a peripheral device—one that can be used in conjunction with your phone/computer to augment the experience, but may not rely on communications or automation.

Let's consider the automobile as a possible example of an IoT device... Several new vehicles are now being created with Internet connections. This enables those vehicles to send information from the vehicle in real time without any interaction from the user. This also makes them capable of receiving and responding to information. If your car experiences mechanical issues, it can automatically communicate with the manufacturer or service center to transmit the issues and even schedule an appointment. Next thing you know, you get a reminder on your phone that an appointment has been scheduled for service. This is an experience that is only achieved by the networked connectivity between devices.

This is just one example; don't limit your thoughts to just this singular scenario. A la Moore's Law, technology keeps getting smaller and faster with every passing year. In fact, University of Michigan researchers have recently created a functional computer that is roughly the size of a grain of rice. We already tiny, ubiquitous connected devices that send information all over the world. As these devices become even smaller, cheaper, easier to produce, and more powerful, you can expect their abundance and applicability to surge.

WHAT ARE THE IMPLICATIONS OF THE IOT?

For starters, it means that more information can be captured, analyzed, and delivered to your pocket in real time, in any context. It also means that you can use your mobile devices to interact with the objects around you in new ways.

Consider a few hypothetical scenarios:

- What if you could remotely unlock your doors using your phone so that your neighbor could walk your dog while you're away?
- What if every FedEx package could automatically update its location, temperature, and force/shock/acceleration in real time and notify business process and consumers seamlessly? How would that affect business logistics? How would that change the customer experience?
- What if every vehicle on the road communicated its location, speed, and bearing to traffic monitoring/management/routing

systems in real time to eliminate traffic? Now what if this system automatically sends messages to your phone, or directly to your car, to inform you of faster routes?

- What if manufacturing facilities could monitor the location, supply, and use of every part in their products? What if an assembler could get notifications in real time about where the next part in their workflow is.
- What if your home could automatically know when you're approaching and set the air conditioning, turn off the alarm, and start heating up the oven for dinner?
- What if every home was able to capture weather conditions and send that information to weather prediction centers automatically?
- What if you wanted to know the quality of the air in your child's bedroom at any time, regardless of where you are?
- What if a farmer wanted to monitor all temperature and moisture level variations across multiple farms and be able to accommodate conditions for target crops?
- What if you could use your phone to monitor your health and vital conditions? Or better yet, what if your physician could monitor your vitals from their phone, even when you both are away hundreds of miles apart?

While these may sound farfetched, they are not. Much of the technology is already here, and some of these solutions have already made it to their market. For example, many people can already control their home alarm or automation systems directly from their smartphone devices.

Information from sensors distributed throughout the world will be able to be analyzed, processed, and responded to in ways never before possible—triggering actions in other systems, sending notifications to personal devices, even manipulating physical objects or conditions without any human interaction.

Sensors may be built as standalone devices that push information to pertinent systems. Or sensors may be built into devices that we use every day—perhaps in your phone, perhaps in your home, perhaps in your watch, perhaps in a device implanted within your body.

WHAT DOES THIS MEAN FOR MOBILE DEVICES?

Let's first start with the most fundamental technology behind mobile devices: the communications network infrastructure. Without a network connection your phone will make a great paperweight, but not much more. Having millions, if not billions, of additional devices sending data across the network would cause network congestion, and potentially create significant load on data processing and storage systems.

Network infrastructures have already started to evolve to meet the data demands as the number of connected devices increases. Without these changes, the infrastructure would never be able to meet its potential for connected systems.

If you have a sensor that sends updates every second, it will send 60 updates per minute, 3,600 updates per hour, and 86,400 updates per day. If there were only 1,000,000 such devices in the world, these devices would send a staggering 8,640,000,000,000 updates per day. Let's say that each of those updates is only 1 kilobyte in size. That adds up to 8,640,000 gigabytes, or 8.64 petabytes, of additional traffic every day. Though such devices likely will not be updating every second, in reality there will probably be many, many more updates than just 1 million per day, and these updates will vary in payload size.

Of this massive amount of data, how much has really changed in each update? IoT monitoring systems are already evolving to distribute computing logic from a core server/cluster to a distributed computing system where information is processed closer to where it is collected. This way not every update has to go out across the network, just the important information will be sent, minimizing the load of the overall network, and ensuring that your devices will remain connected.

Next, let's consider your mobile device as a way of collecting information. More sensors will be built into your mobile devices or integrated into device peripherals. Consider the recently released Apple Watch, which includes sensors to monitor your general activity and heart rate. Your mobile device will increasingly become a way to get your data into the system.

Your mobile device will also increasingly become your portal for receiving information from distributed systems. You can expect the push notification paradigm to evolve far beyond just engagement on social media. As devices throughout the network gather information, that information needs to be consumed. Mobile notifications will be able to inform users of updated conditions, identify critical situations, provide proactive suggestions, or notify you to take action based upon a never ending stream of information.

Your mobile device will also become the way you control and interact with other systems. Need to adjust your home security system? Need to adjust your air conditioning? Need to adjust your home brewing system? Need to interact with your car? Need to control your aerial drone that monitors your power line or pipeline system? Use your phone.

SO, WHAT DOES THIS MEAN FOR THE FUTURE?

The world will continue to transform as more devices connect to the Internet and companies drive innovation and technical evolution. There will be more information that can be analyzed for actionable outcomes. There will be more endpoints available from which to consume information, and there will be more ways that processes can be streamlined, automated, and/or synchronized. Perhaps most important of all: these updates will be more personal than ever. Updates can be delivered based upon an individual context directly to the device that lives inside of your pocket. As the Internet of Things increasingly leads users to expect this kind of context awareness, mobile app developers will need to treat mobile devices as increasingly more than pocket-size computers with mediocre connectivity.

ANDREW TRICE is a Technical Evangelist with Adobe Systems. He has more than a decade of experience designing, implementing, and delivering rich applications for the web, desktop, and mobile devices. He is an experienced architect, team leader, accomplished speaker, and published author, specializing in object-oriented principles, mobile development, real-time data systems, GIS, and data visualization.

Telerik® Platform

Mobilize and Modernize Your Apps. Fast.

Cross-Platform Mobile Development
for iOS, Android and Windows Phone

Learn more at
telerik.com/platform

Telerik®
A PROGRESS COMPANY

Why Isn't Your Mobile Strategy Modular?

When you kick off a new mobile app project, you typically ask yourself two questions:

- What are the requirements?
- What skills will I need to complete the project?

Even after you have decided what you're going to build and how you're going to build it, you're still going to need to ensure that the mobile solution you chose is modular in nature. Why? You need to give your development team the power to pick and choose which services to include in the design and construction of your mobile app.

IN THE PAST

It hasn't always been this way. Development teams have often been hamstrung by the limited choices available to build mobile apps. Want to tie in other services such as analytics, backend services and testing? Bolting on a new tool often presented serious logistical challenges. And what guaranteed that the service would be around in six months?

THE PARTS WE'LL FORGET

Mobile app analytics, backend services, and mobile testing are certainly three parts of the mobile app strategy that we keep front and center. But what about the other parts—services that could have made our app better from the first version and increased adoption? Such services include prototyping early, leveraging client feedback often, deploying your app to the app stores without spending hours reading guidelines for each mobile platform, and controlling who has access to your app using a centrally managed dashboard.

Give your development team the power to pick and choose which services to include in your next mobile app.

THE PARTS WE'LL REMEMBER

Whichever mobile platform you choose, make sure it has the services that prove their worth throughout the app lifecycle. Modules that allow you to prototype a mobile app with drag and drop and send a link to a customer for review. Modules that supercharge your IDE and let you add a backend service and analytics by just including a few lines of code. Modules that remind you every day that you had a choice in the matter: the choice to use the best services to make your mobile app a success.

WRITTEN BY MICHAEL CRUMP
SENIOR DEVELOPER ADVOCATE, **TELERIK**

Telerik Platform by Telerik

Telerik A PROGRESS COMPANY

The Telerik Platform is a modular platform for web, hybrid, and native mobile development that integrates a rich set of UI tools with powerful cloud services.

CASE STUDY

Bhagwan Marine chose Telerik as its technology platform to rise to the occasion and demonstrate that it is possible to successfully harness mobile in off-shore energy production industry. Bhagwan Marine first centralized fleet vessel data within a custom CMS module, leveraging Telerik Sitefinity CMS. It then built the cross-platform app called "LiveFleet" using the full Telerik Platform solution, giving it a native look and feel for both iOS and Android devices. Telerik gave Bhagwan Marine the ability to provide one central point in the cloud for editing and distributing all content, saving the company time and money. The company incorporated app Analytics, provided by Telerik, to access the insights it needs to competitively move forward.

APP CATEGORY

- Web
- Hybrid
- Native

TOOLS INCLUDED

- IDE
- Bug tracking
- Usage analytics
- Performance analytics
- Version management
- Distribution management
- Prototyping
- Device testing

MBAAS INCLUDED

Yes

LANGUAGES USED

HTML/CSS JAVASCRIPT

CUSTOMERS

- Paylocity
- University of Wisconsin Madison
- Bhagwan Marine
- Crown Prince Court - Abu Dhabi
- KiZAN Technologies
- Apex Revenue

BLOG blogs.telerik.com | TWITTER @telerik | WEBSITE www.telerik.com

Mobile Backend-as-a-Service (MBaaS):

Doing the Heavy Lifting on Backend Integration

QUICK VIEW

01

MBaaS platforms reduce the complexity of backend integration.

02

Controlling and managing multiple apps for a variety of platforms becomes easier with MBaaS.

03

Utilizing MBaaS drives team-based development and promotes a flexible delivery model.

04

Backend integration platform solutions will enable developers to quickly deploy secure, cloud-based applications for the Internet of Things.

BY CATHAL MCGLOIN

The appetite for enterprise mobile apps continues to intensify, but satisfying that hunger is not always as easy as it appears. Despite the hype, the reality is that many organizations are still on the journey to mobile maturity—from building one-off apps for specific use cases to creating compelling mobile solutions that underpin an organization-wide digital strategy. As they take their mobility initiatives to a more strategic level, they look to develop business productivity or revenue-generating apps where backend integration can be a real and complex issue. Couple this with the need to churn out apps at increasing speed and with tip-top user experience, the pressure is on to create these backend integrations in the most efficient, effective, and secure manner.

To alleviate the burden on developers to create apps that integrate with backend systems and services, Mobile Backend-as-a-Service (MBaaS) and Mobile Application Platforms emerged, taking a fresh cloud-based approach to the new demands that mobile app developers face. Initial MBaaS solutions were focused solely on consumer apps, solving issues like remote data storage, social integration, and push notification. However, as enterprise mobility gained momentum, another breed of MBaaS platforms emerged to fill the gaps between core enterprise data sources, systems of record, and mobile apps. Often referred to as "mobile middleware," MBaaS exposes APIs and services to promote agile mobile app development and solve more complex tasks such as offline data sync, single sign-on (SSO), and back-end integration to legacy systems. This new server-side architecture provides an ideal way to hide complex logic from the app developer and mobile-enable legacy systems that were never designed to be mobilized.

However, despite the emergence of mobile app platforms and MBaaS to ease the burden of integrating apps with multiple backend systems, it appears that many businesses are still tackling backend integration in an ad hoc manner. A recent survey by Red Hat Mobile indicates that 55% of companies building mobile apps were either custom coding backend integration from scratch or using external sources such as libraries, marketplaces, and external vendor services. To put this in context, consider that creating custom integration can take between 50% and 70% of the costs and time associated with mobile app development[1]. Scaling this model can only result in escalating app development costs and rapidly increasing mobile app backlogs.

So how does an enterprise-grade Mobile Backend-as-a-Service (MBaaS) promote a more strategic approach to the mobile app development lifecycle? Here are seven ways in which MBaaS platforms can support strategic mobile enablement.

The complexity involved in integrating different backend systems, many of which are proprietary, is the nemesis of simple, high-performing mobile design.

1. Reducing Complexity of Backend Integration

MBaaS enables full access to cloud services such as storage, security, caching, and business logic that are indispensable to developing enterprise-class mobile experiences. Back-end server-side complexity shifts to the cloud, enabling a more modularized technology stack as well as more efficient data-level management and scaling capabilities. The complexity involved in integrating different backend systems, many of which are proprietary, is the nemesis of simple, high-performing mobile design. Rather than forcing organizations to completely re-architect those systems, an enterprise-grade MBaaS enables them to mobilize in a lightweight, component-based, agile fashion. MBaaS excels at integrating disparate legacy systems by acting as a hub between mobile clients and the systems they need to access.

2. Scaling Mobile App Development

Consider, when building a mobile app, coding the connections to the required backend systems and data sources, either from scratch or by using third party libraries, etc. As app development projects multiply in your organization, this model is difficult and costly to scale. With much of the complexity and effort of app development consumed by backend integration, the need to centralize developer components, services, and APIs, and make them accessible and reusable across multiple app projects, is an effective model in maximizing re-usability of code, and reducing development time and cost. As the number of mobile apps being developed reaches into the tens and possibly hundreds of apps, organizations often find themselves at the tipping point for needing a platform to help them control and manage multiple apps for a variety of device platforms in multiple iterative lifecycles.

3. Supporting Iterative App Development

Mobile app development is a continuous, highly iterative, dynamic process that goes beyond pure development to encompass a cycle of continuous design-build-integrate-test-deploy. To support this iterative develop-and-deploy nature, operational flexibility and agility is essential. Rather than rigid monolithic architectures, mobile lends itself to loosely-coupled component- or microservices-based approaches that can be composed and decomposed as needed in making changes to backend services. A cloud platform, either MBaaS or Mobile Application Platform, which supports the discovery of these backend developer components—and controls access to them—supports agile development. Mobile development can be greatly enhanced, for example, by easily creating reusable Node.js-based services in the Red Hat Mobile platform, which can be shared across all apps.

4. Centralizing Control & Visibility for IT

Enterprise IT professionals need to be able to maintain control and visibility over sensitive corporate assets. Extending traditional enterprise systems and applications to mobile devices can greatly expose the organization to security breaches as well as an overload on backend systems. Rather than integrating mobile apps directly to backend systems, the MBaaS acts as a middleware whereby data can be transmitted over a secure channel to the cloud which then sends the data on to the app on device. This takes the load off the enterprise's backend systems and acts a hub for security, user access management and integration. An enterprise-grade MBaaS handles tasks like authentication/authorization, data storage (caching and persistence), data sync, security & encryption, push notifications, and also offers extended services specific to the integration of the app with different enterprise systems and cloud services. Some enterprise MBaaS offer a broad range of cloud or on-premise hosting options to meet the required infrastructure and SLA requirements of the organization, further hardening control over the security of their data. An MBaaS that allows IT to centralize backend integration components and control access to these is a more secure development approach than ad hoc app development.

5. Driving Team-Based Development

Mobile app development for enterprise is increasingly team-based, relying on multiple roles and skillsets to get apps to market in a fast and effective manner. A typical mobile app project is likely to involve several different developer skills, from UI/UX design to frontend coding to backend services development, analytics, administration, DevOps, and more. Apps may be developed by teams of internal and external developers. For multi-disciplinary and outsourced teams, it is important to support collaboration and code sharing, and a platform can provide this environment where access to code, services, and developer components is centralized and managed.

MBaaS exposes APIs and services to promote agile mobile app development and solve more complex tasks such as offline data sync, single sign-on (SSO), and back-end integration to legacy systems.

6. Flexibility of Deployment

With mobile projects characterized by a continuous delivery model, development and operations need to work in closer harmony and in more agile ways. How the MBaaS is deployed is a critical, and sometimes, overlooked aspect of mobile application development. Not all deployments are created equal, so whether server-side code is deployed to a private, public, or hybrid cloud or deployed fully on-premise, it's important to maintain flexibility and portability. Enterprises also need to think about how their backend data is managed, whether in the cloud or on-premise. Enterprise-grade MBaaS serves to simplify the configuration, deployment, and maintenance of code in a DevOps-friendly fashion and frees the developer to focus on frontend user experience. Enterprises cannot afford to be locked in to a particular deployment model; instead, they need to be able to respond and adapt as needs evolve.

7. Mobile Backend Integration Meets Internet of Things

Gartner anticipates that a remarkable 4.9 billion devices will be connected via the Internet of Things (IoT) in 2015, up 30 percent from 2014. Organizations will look to manage and enable non-traditional IT assets such as vehicles, buildings, and machinery, incorporating them into the IoT ecosystem just as they have done for mobile devices.

Backend integration platform solutions that enable developers to quickly create and deploy applications that are secure and operational within the cloud, are well positioned to help translate key backend systems to these new IoT frontends.

Conclusion

As the volume and sophistication of mobile apps that companies develop increases, so also does the development complexity. Mobile projects look to integrate data from multiple backend systems, applications, and services, some of which are based on legacy technologies and architectures. Much of this integration goes beyond basic backend functionality to encompass more complex integrations to core systems of record, many of which were implemented over the past few decades. This means extending, not only common CRM, ERP, and database applications to mobile, but runs the gamut from industry-specific applications (e.g. a flight management systems, electronic medical records systems, energy management systems, etc.) to bespoke enterprise applications developed and implemented to meet an organization's specific business IT needs.

Tackling backend integration on a tactical per-app basis is neither scalable nor economical as mobile app projects multiply and as mobile becomes a key strategic initiative for business success. Rather than focusing agility and innovation on frontend development, organizations are now also embracing agility for backend development as well. Enterprise MBaaS offers a flexible, interoperable, cloud-based service delivery model that provides better control and manageability to IT and more agility to app developers.

[1] Gartner, Market Guide for Rapid Mobile Application Development Tools, Nov 14

CATHAL MCGLOIN is ex-CEO of FeedHenry, the enterprise mobile application platform provider acquired by Red Hat, the leading open-source solutions company, in October 2014. Cathal is based in Boston and develops the enterprise mobility business for Red Hat Mobile.

5 Questions to Find Out if MBaaS Fits Into Your Technology Stack

The MBaaS promise is rather grand, quite often touted as a silver bullet against the enterprise integration complexities or a magic, which can elevate developer productivity to the next level. This is all true to a degree, but there are no two identical deployment scenarios and every case must be reviewed individually. So could MBaaS work for you? The questions below can be used as a guideline for narrowing down your choices of platforms or perhaps ruling out MBaaS from the technology stack:

1. Do you have an existing infrastructure, database(s), enterprise system?
Some MBaaS systems operate as closed silos and (if you're not starting with a clean slate) you'd be required to migrate data from your database into MBaaS. More mature solutions provide integration connectors. These may include database connectivity, identity management server connectors, and integration with ERP, SCM, and CRM systems.

2. Can you tolerate dependency on the cloud?
Most of the available MBaaS solutions run exclusively in the cloud. That means all of your data, files, user accounts, and app-related business processes would be residing outside of your control. Some MBaaS products are packaged to run both in the cloud and on-premise. It is important to evaluate packaging options and determine your level of tolerance for relying on the cloud.

3. Are you prepared to embrace vendor lock in?
An MBaaS-based integration is not governed by a standards body and there is no formally standardized client-server API. As a result, vendor lock-in is absolutely inevitable with the MBaaS approach. Some vendors try to make an argument about "no vendor lock-in" and justify it by pointing out the open-sourced client-side libraries. This is still not true. If you code your app to use an API and decide to switch the backend, you will have to spend time to recode the app to use the API of another provider.

4. Do you value extensibility? Explore ways to customize client or server-side.
An MBaaS is a black box guaranteeing specific behavior for an API call. It is important to know that you can modify the behavior both on the client-side (look for open source SDKs) and on the server (look for custom server-side logic).

5. How mobile-friendly is the mobile backend platform?
Calling a backend 'mobile' does not make it one. Be cautious of immature implementations that will drain device battery, unwisely use network bandwidth, or will not support app-in-the-background mode.

WRITTEN BY MARK PILLER
FOUNDER AND CEO, **BACKENDLESS**

Backendless by Backendless

backendless

Backendless is a general-purpose mobile application development platform. It consists of an mBaaS foundation, a File Hosting service (with node.js support), API management engine, and a Services Marketplace.

CASE STUDY

GMO Internet Group is a major internet services provider in Japan. The conglomerate provides a wide variety of services including cloud hosting, online payment processing, game development, ad management and many more. The company was under a lot of competitive pressure to provide its own mBaaS offering. After an in-depth study of the build vs buy options it decided to license an existing solution. GMO considered several established mBaaS offerings and in the end decided on Backendless. The deciding factors were platform's maturity, flexibility and the breadth of functionality. The company has rolled out a white-label version of the product known as "GMO mBaaS by Backendless" and offers it as a commercial service to the customers.

APP CATEGORY

- Web
- Hybrid
- Native

TOOLS INCLUDED

- IDE
- Support desk
- Usage analytics
- Performance analytics
- Version management
- Distribution management

MBAAS INCLUDED

Yes

LANGUAGES USED

HTML/CSS, JAVASCRIPT, JAVA, OBJECTIVE-C, SWIFT, C#, RUBY, PYTHON, PHP, C/C++

CUSTOMERS

- Bank of New York
- Sapporo
- NEC
- GMO Internet Group (Japan)
- GearStream
- Acrodea
- OCI Beaumont

BLOG backendless.com/blog

TWITTER @backendless

WEBSITE backendless.com

Ensure a great user experience with our **Continuous Quality Lab**

perfecto mobile

Redefining Mobile App Quality

Many organizations use agile to deliver mobile apps faster, but have challenges fully incorporating the methodology for mobile. They perform nightly builds but aren't aware of the **state of quality**.

3 factors impact visibility of quality:

1. Limited test automation reduces the number of non-unit tests that can be run. Test automation capabilities are like Swiss cheese – it looks good but beware of holes!
2. Unstable development environments cause workarounds like non-scalable scripting, missing major factors that impact a test environment, or weak Wi-Fi connections.
3. Out-of-cycle tests impact quality. Organizations often exclude load testing because of complexity. That means there's no true quality insight throughout the mobile lifecycle.

To figure out where quality stands, take a "no code left behind" approach. Continually embed quality into the lifecycle to assure working code and working apps.

Here's how:

- **Eliminate Swiss cheese:** Combat limited test automation with stable automation that meets requirements. Set the requirements before selecting an automation engine:
    1. Do you need to interact with the device or just the app?
    2. Do you need to analyze what's on the screen?
    3. Do you need self-recovery if the system crashes?
- **Staff Accordingly:** Devices are unstable by definition. Humans are more stable. Make the environment stable by making someone responsible for testing.
- **Leverage a hybrid cloud:** Create an always-on test lab with human oversight. This reduces traps like USB connectivity, breach of security guidelines, Mac support, etc. Staffing an outlier to manage a test lab WILL break your cycle.
- **Bring it In-Cycle:** Out-of-cycle test sections will break your cycle. Build an environment that supports a variety of test cases including outliers: different network conditions, multiple devices connected at the same time, capacity issues, load and security testing.
- **Educate:** Educate your staff on the benefits of automated testing in the cloud.

WRITTEN BY YORAM MIZRACHI
CHIEF TECHNOLOGY OFFICER, **PERFECTO MOBILE**

Continuous Quality Lab by Perfecto Mobile

Perfecto Mobile provides a Continuous Quality Lab that enables mobile app development and testing teams to deliver better apps faster. Users access real mobile devices connected to live networks located in the cloud and leverage them for testing and monitoring throughout the mobile application development lifecycle.

CASE STUDY

Mobile's increasing criticality at an $11B insurance giant drove the need to significantly reduce manual testing with an automated solution capable of scaling to meet the two-week sprint cycles. The company also wanted to use a single test script for both mobile and web testing. Specific requirements included a continuous quality test automation framework for end-to-end testing of mobile and desktop Integration with existing development, CI, and ALM processes as well as timely reporting for more informed decisions. Test automation delivered measurable results and the company was able to streamline application testing across multiple browsers and mobile device types to two days. Before automation it took thirteen people twelve days to execute a complete test plan across all browsers and devices. By replacing manual processes with an automated solution, testing now fits within the company's two-week sprint cycles. All bugs detected in CI level tests are automatically logged in the company's ALM system for full corporate visibility of issues. Because the test scripts are coded by priority, developers and QA managers can easily collaborate, prioritize workloads and determine how to go forward with any outstanding issues. By addressing difficult bottlenecks early in the project, this insurance company was able to boost velocity and quality in all phases of the development process. As the company continues to mature and become more agile, their ability to quickly deliver against customer expectations will increase as well.

CI TOOL SUPPORT

- Jenkins
- TeamCity
- Bamboo
- CloudBees
- Microsoft TFS

LANGUAGE SUPPORT

JAVASCRIPT

JAVA C# RUBY

VB SCRIPT

TESTING SUPPORT

- Seleium
- Appium
- JavaScript testing
- Manual testing
- Automated testing
- Monitoring integrations with HP and SmartBear

PRICING

perfectomobile.com/pricing

APP CATEGORY

- Web
- Native
- Hybrid

OPEN SOURCE

No

CUSTOMERS

- Verizon
- Reuters
- Tesco
- Paychex
- Discover
- Virgin
- Lego
- T Mobile

BLOG blog.perfectomobile.com

TWITTER @PerfectoMobile

WEBSITE perfectomobile.com

Mobile App Testing Checklist

Device Checks

- ☐ Can the app be downloaded?
- ☐ Can the app be uninstalled?
- ☐ Can the app be updated?
- ☐ Can the app be updated when multiple updates exist?
- ☐ Can the app be downloaded on multiple operating systems?
- ☐ Can the app be redownloaded?
- ☐ Does the app behave as intended when the device is connected to power?
- ☐ Does the app behave as intended when the device is not connected to power?

UI / UX

- ☐ Is the user experience as intended?
- ☐ Is the user interface appropriate for every form factor?
- ☐ Is the user able to navigate between various pages within the app?
- ☐ Do all buttons conform to standards (e.g. Share button shares, Trash Can deletes, etc.)?
- ☐ Is the UI consistent with its Desktop equivalent (if applicable)?
- ☐ Does the app allow for touchscreen gestures (e.g. zoom, pinch, swipe, etc.)?

Integrations

- ☐ Does the app access the device's camera?
- ☐ Does the app access the device's sensors (e.g. accelerometer, gyroscope, etc.)?
- ☐ Does the app access any peripheral devices (e.g. Bluetooth headphones, keyboard, etc.)?
- ☐ Does the app integrate with other apps (e.g. Twitter, Facebook, etc.)?

Storage

- ☐ Does the app write to a memory card?
- ☐ Does the app read from a memory card?
- ☐ Does the app write to the cloud?
- ☐ Does the app read from the cloud?
- ☐ Does the app function if storage is full?
- ☐ Is the correct data being stored?

Network

- ☐ Does the app function on Wi-Fi?
- ☐ Does the app function on Cellular data (including 3G, 4G, LTE, EDGE, etc.)?
- ☐ Does the app function on Bluetooth?
- ☐ Does the app function when transitioning between networks?
- ☐ Does the app function in Airplane mode?
- ☐ Does the app function when not connected to any network?

Security

- ☐ Is any personal data stored on the device?
- ☐ Is all app data removed if the app is uninstalled?
- ☐ Can app data be accessed outside of the app?
- ☐ Does the app authenticate?
- ☐ Can the user be locked out of their account (if applicable)?
- ☐ Can the user reset their password (if applicable)?
- ☐ Does the app require any permissions?

General App Functionality

- ☐ Does the app perform the functions it is designed to perform?
- ☐ Does the app perform functions it is not designed to perform?
- ☐ Is the user prompted to turn on services (e.g. location services, Wi-Fi, etc.) if necessary?
- ☐ Does the app run on multiple operating systems?
- ☐ Does the app run on multiple devices?
- ☐ Is navigation clear and intuitive?

Interruptions

- ☐ Does the app function when the user receives a phone call?
- ☐ Does the app function when the user receives a text message?
- ☐ Does the app function when the user receives a push notification?
- ☐ Does the app function when the user receives a battery notification?
- ☐ Does the app function when the user receives a voicemail notification?
- ☐ Does the app function when the device is locked?
- ☐ Does the app function when the device is unlocked?

Accessibility

- ☐ Does the app perform the functions it is designed to perform?
- ☐ Does the app perform functions it is not designed to perform?
- ☐ Is the user prompted to turn on services (e.g. location services, Wi-Fi, etc.) if necessary?
- ☐ Does the app run on multiple operating systems?
- ☐ Does the app run on multiple devices?
- ☐ Is navigation clear and intuitive?

Solutions Directory

This directory of mobile application development platforms, frameworks, and services provides comprehensive, factual comparisons of data gathered from third-party sources and the tool creators' organizations. Solutions are selected for inclusion in the directory based on several impartial criteria, including solution maturity, technical innovativeness, relevance, and data availability.

FRAMEWORKS

NAME	LAYOUT	IDES SUPPORTED	WEBSITE
Angular.js by Google	App-centric	Eclipse, WebStorm	angularjs.org
Bootstrap by Twitter	Content-centric	Eclipse, NetBeans, WebStorm	getbootstrap.com
Dojo Toolkit by Drifty	App-centric	Eclipse, WebStorm	dojotoolkit.org
Ionic	App-centric	Eclipse, Visual Studio, NetBeans, IntelliJ IDEA, WebStorm	ionicframework.com
jQuery Mobile	Content-centric	Eclipse, Visual Studio, NetBeans, IntelliJ IDEA	jquerymobile.com
KendoUI by Telerik	App-centric	Eclipse, Visual Studio, NetBeans, IntelliJ IDEA	telerik.com
Mobile Angular UI	Content-centric	Eclipse, WebStorm	mobileangularui.com
OnsenUI by Monaca	Content-centric	Monaca IDE	onsen.io
React by Facebook	Content-centric	Visual Studio, WebStorm	facebook.github.io/react
Sencha Touch by Sencha	App-centric	Eclipse	sencha.com
Touchkit by Vaadin	App-centric	Eclipse, NetBeans, IntelliJ IDEA	vaadin.com
Wijmo by GrapeCity	Content-centric	Eclipse, Visual Studio, NetBeans, IntelliJ IDEA	wijmo.com

MADP

NAME	WEB OR NATIVE HOSTING	HOSTING	FREE TRIAL	WEBSITE
Akula by Verivo	Web	SaaS	Upon request	verivo.com
Alpha Anywhere by Alpha Software	Web and Hybrid	On-Premise	30 days	alphasoftware.com
Appery.io by Exadel	Web and Hybrid	SaaS	Free tier available	appery.io
AppMachine	Native	SaaS	Free until app is published	appmachine.com
AppMethod by Embarcadero	Native	On-Premise	30 days	appmethod.com
Architect by Sencha	Web	On-Premise	30 days	sencha.com
Backbase	Web	On-Premise	Upon request	backbase.com
Backendless	Web, Hybrid, and Native	On-Premise or SaaS	Free tier available	backendless.com
Catavolt	Native	SaaS	Upon request	catavolt.com
Codename One	Native	On-Premise	Free tier available	codenameone.com
Composer by AppGyer	Hybrid	SaaS	Free tier available	appgyver.com
Cordova by Apache	Web	On-Premise	Free tier available	cordova.apache.org
Corona SDK by Corona Labs	Native	On-Premise	Free solution	coronalabs.com
DevExtreme Mobile by DevExpress	Web and Hybrid	On-Premise	60 days	devexpress.com
DSI Mobile Platform	Native	SaaS	Upon request	dsiglobal.com
iFactr Enterprise	Native	On-Premise	Upon request	ifactr.com
Intel XDK	Web, Hybrid	On-Premise	Free tier available	intel.com

MADP CONT'D

NAME	WEB OR NATIVE HOSTING	HOSTING	FREE TRIAL	WEBSITE
KendoUI by Telerik	Web	On-Premise	30 days	telerik.com
Kony Dev Cloud by Kony Solutions	Web, Hybrid, and Native	SaaS	90 days	kony.com
LiveCode	Web and Hybrid	On-Premise	Free tier available	livecode.com
Mendix App Platform	Native	SaaS	Free tier available	mendix.com
MobileFirst by IBM	Web, Hybrid, and Native	SaaS	Free tier available	ibm.com
MobileTogether by Altova	Native	On-Premise	30 days	altova.com
Outsystems	Native	SaaS	30 days	outsystems.com
Parse by Facebook	Web and Native	On-Premise	Free tier available	parse.com
PhoneGap Build by Adobe	Web	SaaS	Free tier available	phonegap.com
Pivotal Cloud Foundry Mobile Services	Web, Hybrid, and Native	SaaS	60 days	pivotal.io
Red Hat Mobile	Web, Hybrid, and Native	SaaS	Upon request	redhat.com/mobile
Rho Elements by Zebra Technologies	Web and Hybrid	SaaS	Free tier available	rhomobile.com
SAP Mobile Platform	Hybrid and Native	SaaS	Free tier available	sap.com
Smartface	Native	On-Premise	Free tier available	smartface.io
Telerik Platform	Web, Hybrid, and Native	On-Premise or SaaS	30 days	telerik.com
TheAppBuilder	Native	SaaS	90 days	theappbuilder.com
Titanium by Appcelerator	Web, Hybrid, and Native	On-Premise	Open Source	appcelerator.org
Xamarin Platform	Native	On-Premise	30 days	xamarin.com

diving deeper

INTO FEATURED MOBILE DEVELOPMENT PRODUCTS

Looking for more information on individual mobile development solutions providers? Nine of our partners have shared additional details about their offerings, and we've summarized this data below.

If you'd like to share data about these or other related solutions, please email us at research@dzone.com.

MOBILE TESTING

Automated Testing Platform

BY SAUCE LABS

API/SDK FOR DEV MANAGEMENT

✗

SECURITY FEATURES

- Authentication
- Authorization
- Offline authentication
- Network encryption

DEVICES SUPPORTED

- iPhone 4S, 5, and 6
- Samsung Galaxy Note 3
- Samsung Galaxy S3, S4, and S5

TEST MANAGEMENT SUPPORT

- Test data management, analytics, and visualization
- Mobile testing suite
- Service virtualization
- Functional testing

OPERATING SYSTEMS

- iOS
- Android

MADP

Backendless

BY BACKENDLESS

SSO ✓

MULTI-FACTOR AUTHENTICATION ✓

RESPONSIVE ✓

FRAMEWORKS

- jQuery Mobile
- Sencha Touch
- Dojo
- Backbone.js
- Angular.js

OPERATING SYSTEMS

- Android
- iOS
- Windows Phone

DATABASES FOR LOCAL STORAGE

- n/a

MOBILE TESTING

Continuous Quality Lab

BY PERFECTO MOBILE

API/SDK FOR DEV MANAGEMENT

✓

SECURITY FEATURES

- Authentication
- Authorization
- Network encryption

DEVICES SUPPORTED

- iPhone 4S, 5, and 6
- Samsung Galaxy Note 3
- Samsung Galaxy S3, S4, and S5
- HTC One M8

TEST MANAGEMENT SUPPORT

- Mobile testing suite
- Service virtualization
- Network virtualization
- Unit testing
- Functional testing
- Code-free test creation
- API testing
- Performance testing

OPERATING SYSTEMS

- Any

DATABASE

Couchbase Mobile

BY COUCHBASE

SECURITY FEATURES

- Authentication
- Authorization
- Network encryption
- SSO
- Multi-Factor

DEVICE SUPPORT

- iPhone 4S, 5, and 6
- Samsung Galaxy Note 3
- Samsung Galaxy S3, S4, and S5
- HTC One M8

TOOLS PROVIDED

- IDE
- Support desk
- Performance analytics

SINGLE POINT OF FAILURE?

No

BEST USE CASE

OLTP

BUILT-IN MAPREDUCE?

Yes

MADP

FeedHenry

BY RED HAT

SSO ✓

MULTI-FACTOR AUTHENTICATION ✗

RESPONSIVE ✓

FRAMEWORKS

- Sencha Touch
- Backbone.js
- Angular.js
- Ember.js

OPERATING SYSTEMS

- iOS
- Android
- Windows Phone
- Blackberry

DATABASES FOR LOCAL STORAGE

- MySQL
- Oracle
- MongoDB

MADP

MobileTogether

BY ALTOVA

SSO ✓

MULTI-FACTOR AUTHENTICATION ✗

RESPONSIVE ✓

FRAMEWORKS

- n/a

OPERATING SYSTEMS

- iOS
- Android
- Windows Phone

DATABASES FOR LOCAL STORAGE

- iOS
- SQL Server
- IBM DB2
- Oracle
- MySQL
- PostgreSQL
- SQLite

MADP

Outsystems Platform

BY OUTSYSTEMS

SSO ✓

MULTI-FACTOR AUTHENTICATION ✓

RESPONSIVE ✓

FRAMEWORKS

- jQuery Mobile

OPERATING SYSTEMS

- PhoneGap/Cordova is used

DATABASES FOR LOCAL STORAGE

- n/a

MADP

PhoneGap Build

BY ADOBE

SSO ✓

MULTI-FACTOR AUTHENTICATION ✓

RESPONSIVE ✓

FRAMEWORKS

- jQuery Mobile
- Sencha Touch
- Dojo
- Ionic
- Angular.js
- Monaca

OPERATING SYSTEMS

- iOS
- Android
- Windows Phone

DATABASES FOR LOCAL STORAGE

- SQLite
- IndexedDB
- WebSQL

MADP

Telerik Platform

BY TELERIK

SSO ✓

MULTI-FACTOR AUTHENTICATION ✓

RESPONSIVE ✓

FRAMEWORKS

- jQuery Mobile
- Sencha Touch
- Dojo
- Ionic
- Angular.js
- Monaca

OPERATING SYSTEMS

- iOS
- Android
- Windows Phone

DATABASES FOR LOCAL STORAGE

- SQLite
- IndexedDB
- WebSQL

glossary

ACCELEROMETER A sensor that measures the acceleration (change in velocity) of an object in order to determine movement of a mobile device.

ADAPTIVE LAYOUT A website layout that uses media queries to change the site's design for specified devices or window sizes. This is a more manual and less granular strategy than responsive design.

APPCACHE An HTML5 standard that allows a web application to be cached and available offline.

BUSINESS-TO-BUSINESS (B2B) A mobile application built specifically for use within another business.

BUSINESS-TO-CUSTOMER (B2C) A mobile application built for the average consumer.

BUSINESS-TO-EMPLOYEE (B2E) A mobile application for internal business use (i.e. an enterprise app).

DEVICE API A mobile platform-specific API that lets applications access specific mobile hardware functionality (e.g. accelerometer, gyroscope).

EMULATOR An application that duplicates the functionality of hardware or operating systems for testing purposes.

GYROSCOPE An instrument that measures the orientation of a mobile device in order to orient display.

HTML5 A specification that defines the fifth major revision of HTML. It is often used as an adjective to describe web apps that use newer HTML, CSS, and JavaScript specifications (e.g. HTML5 apps), or more generally as shorthand for the browser as application platform.

HYBRID APP A mobile application written in HTML, CSS, and JavaScript that uses a web-to-native abstraction layer, allowing the application to access mobile device APIs that pure web applications cannot access.

INTERNET OF THINGS A network of physical objects embedded with electronics, software, sensors, and connectivity to enable it to achieve greater value and service by exchanging data with other connected devices.

MADP (MOBILE APPLICATION DEVELOPMENT PLATFORM) A development tool, sometimes including a mobile middleware server, that builds hybrid or native apps for each mobile platform from a single codebase. Includes MEAPs and MCAPs.

MBAAS (MOBILE BACKEND-AS-A-SERVICE) A service that connects mobile applications to cloud databases while also providing user management, push notifications, and social integrations.

MCAP (MOBILE CONSUMER APPLICATION PLATFORM) A MADP for developing mobile consumer apps.

MEAP (MOBILE ENTERPRISE APPLICATION PLATFORM) A MADP for developing mobile enterprise apps.

MOBILE MIDDLEWARE Technology that provides application management functions that allow applications to communicate securely with on-premise and cloud-based applications.

NATIVE APP A mobile application that is written in a programming language that is directly compatible with the target platform.

NATIVE BRIDGE An abstraction layer that gives a non-native application access to mobile device APIs.

NATIVE PACKAGER A tool that builds a native wrapper and native bridge around a mobile application written in HTML, CSS, and JavaScript, creating a hybrid app.

NATIVE WRAPPER A component that packages a non-native app so that it is viable for native distribution.

NFC (NEAR-FIELD COMMUNICATIONS) A feature that allows devices to establish radio communication with other nearby systems or mobile devices.

OAUTH A common open standard for authorization.

PUSH NOTIFICATIONS Short messages that mobile applications can send to users even if the application isn't open.

RESPONSIVE LAYOUT A website layout based on a fluid grid, allowing the site to have hundreds of dynamically generated states with mostly minor differences based on browser window size. A less manual strategy than adaptive layouts.

SAAS (SOFTWARE-AS-A-SERVICE) A service residing on a layer of abstraction above IaaS and PaaS with all of the software and features already built and provided over the network. The main advantage of this model is that a customer does not need to install or maintain this software on-premises, or store any of its data.

SAML (SECURITY ASSERTION MARKUP LANGUAGE) A common XML-based open data format for authentication.

SDK (SOFTWARE DEVELOPMENT KIT) A set of programming tools and resources built specifically to aid software development on a particular platform or technology.

SMS (SIMPLE MESSAGE SERVICE) A text messaging service component of a phone using standardized communication protocols to send short text messages.

SOA (SERVICE-ORIENTED ARCHITECTURE) An architecture style that uses discrete software services (each with one clearly defined business task) with well-defined, loosely-coupled interfaces that are orchestrated to work as a complete system by sharing functionality.

SSO (SINGLE SIGN-ON) A feature for access control systems that allows users to log in to (or log out of) multiple, independent software systems using one set of credentials.

USER EXPERIENCE (UX) A term to describe all aspects of the end user's interaction with an application.

W3C (WORLD WIDE WEB CONSORTIUM) The main international standards organization for the World Wide Web, developing common web language specifications such as HTML5 and XML, as well as JavaScript APIs that browsers must implement.

WEB APP A mobile application developed using web standards and accessed through a browser.

WEBVIEW In Android, Webview is an extension of the View class that displays web pages within an application.