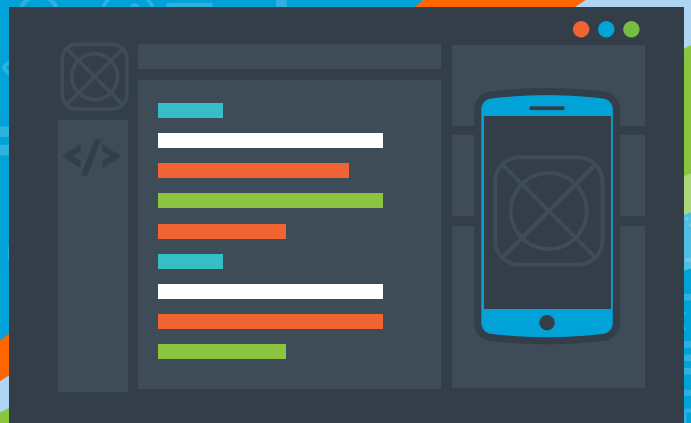
Forward-Looking Mobile App Development

Maximize the power you deliver
today with flexibility
for the future.



kony™

Mobilize the Enterprise

Table of Contents

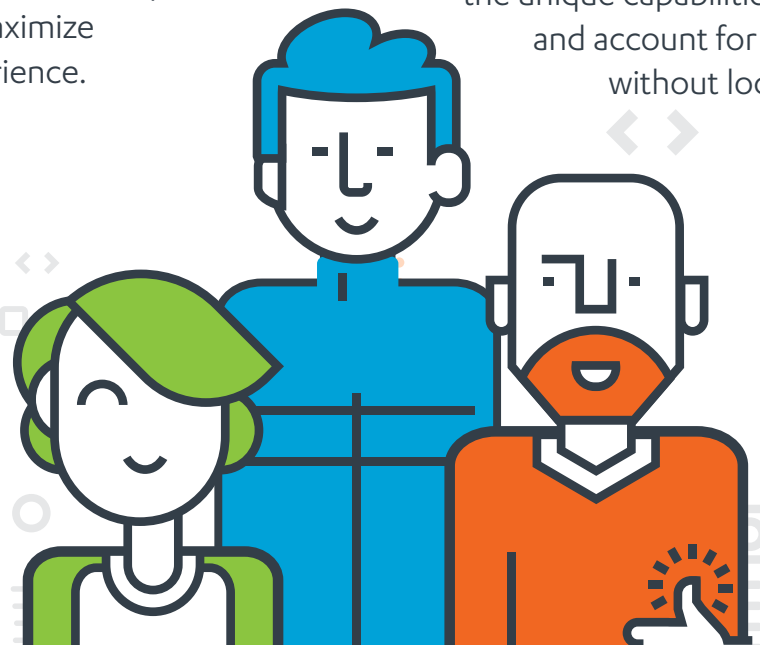
Creating a great mobile app means making choices; not just which device or devices to target but what functions to include and what to leave out. At the same time, some choices are worth avoiding such as short-term tactical compromises that lock you into an approach that makes sense today but limits your flexibility in the future.

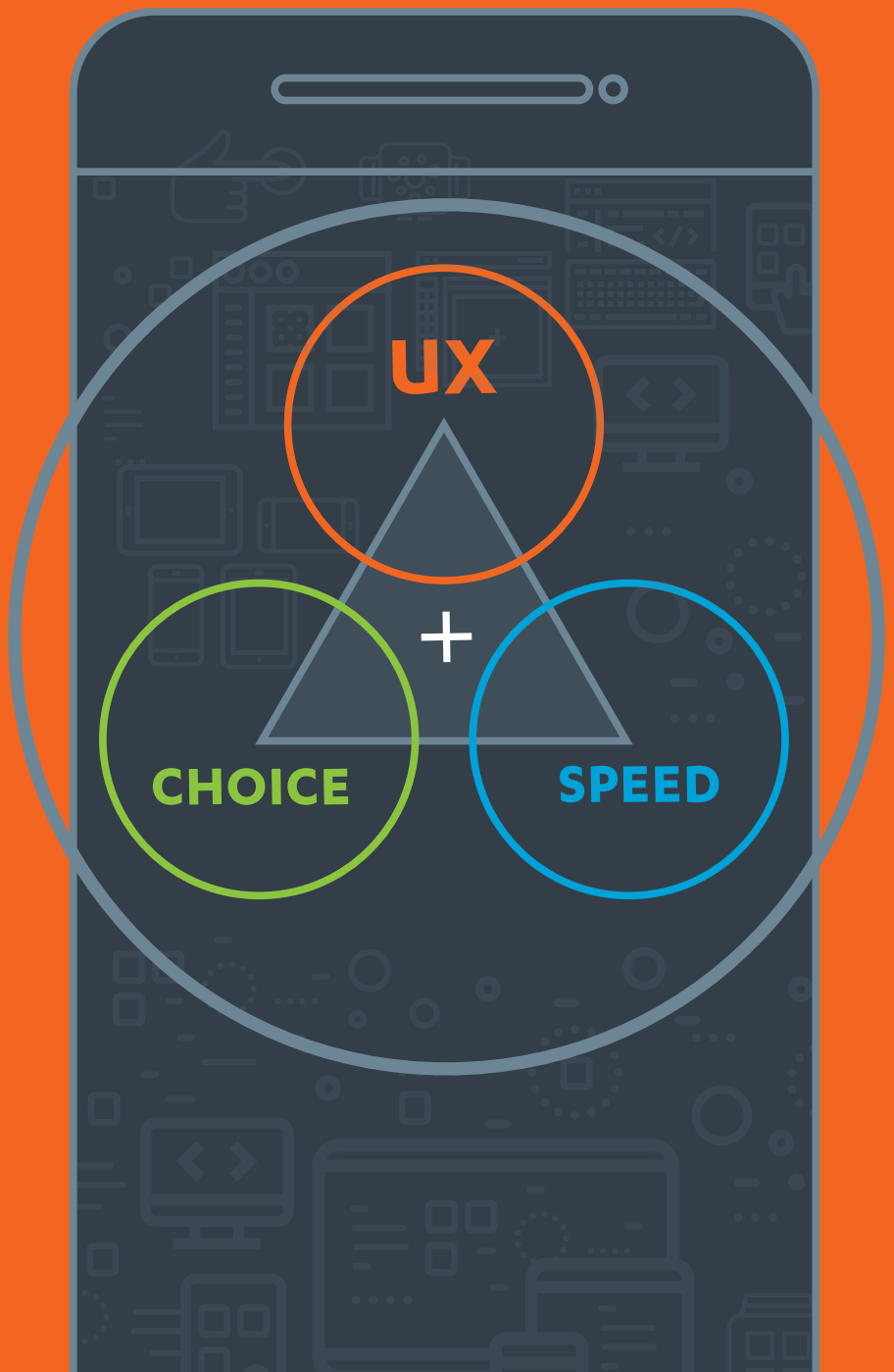
Of course, life and business are full of tradeoffs. Project managers revere the project triangle ("fast, good, or cheap–pick any two") as something close to a law of nature. That is, if you want to deliver a high-quality result quickly, you should be prepared to spend more; if you want to keep costs down, you're likely to have to compromise on speed or quality.

The commonly accepted "unavoidable" tradeoff in mobile app development is optimizing for native app performance versus making apps available on many devices and mobile operating systems. Either you deliver the best native user experience at the cost of app portability, or you maximize portability at the expense of user experience.

Today, that thinking is outdated, and the choices are no longer that stark. For most of the past decade, mobile app developers have sought better ways to deliver code that works across iOS, Android, Windows, and the web. Kony has been part of that, both as a consulting firm helping clients create mobile apps and as the architect of a platform for app development, management, and enterprise integration. Yet we know we are not the only innovators, and our customers want the freedom to incorporate open source libraries as well.

Avoiding unnecessary compromises means keeping our options open and looking for the win-win where we can achieve multiple goals simultaneously. We want to design for performance and omni-channel availability, not one or the other. The more basic hard choices (cost versus schedule versus quality) do not go away, but we can avoid making app development harder than it absolutely has to be. We can take advantage of the unique capabilities of one mobile OS/device combination and account for the constraints imposed by another–without locking ourselves into any one choice.





The Mobile Priorities Triangle

Mobile developers have three competing priorities:

- **User experience** – making the app rich and engaging, taking full advantage of the native capabilities of specific mobile devices and operating systems.
- **Choice** – incorporating popular frameworks that let the same app run on the web, phones, tablets, and other devices with different operating systems.
- **Speed of development** – taking advantage of visual, low-code tools to create applications more quickly.

Consider the dimension of user experience versus choice of platform. Do you maximize the user experience by writing code that runs really well on the iPhone but must be rewritten from scratch for Android or Windows devices? According to Forrester Research, rewriting an app to port it to another mobile OS adds 50% to 70% to the cost of development. Or do you use a web-based framework that maximizes portability at the cost of sub-optimal performance on all platforms?

According to Forrester Research, rewriting an app to port it to another mobile OS adds 50% to 70% to the cost of development.

Competing approaches to mobile software development are another example. If you use a development tool that promises drag-and-drop ease of application assembly, do you give up the ability to innovate outside the boundaries of a proprietary framework? Will you wind up building generic apps that look the same on all mobile platforms but make a strong impression on none of them? On the other hand, can you afford to ignore the low-code alternatives if it means hiring for scarce and expensive skills specific to a given mobile OS?

These issues particularly weigh on enterprise app development projects. A well-funded startup might be able to put parallel app development teams to work creating lovingly hand-coded iPhone and Android versions of the same app and have both ready to go on launch day. Enterprises rarely have the resources to pursue multiple paths at once. Yet enterprise apps face the same expectations as every other app, regardless of whether those expectations are realistic.

Project leaders tend to see themselves as having three choices:

- **Mobile web** – standard HTML and JavaScript delivered on a smaller screen, perhaps using responsive design techniques to reformat the same web pages delivered to mobile clients.
- **Hybrid** – HTML5 and JavaScript packaged using a wrapper such as Apache Cordova so the mobile web experience can be delivered as an installable app that can access native resources such as a phone's camera or accelerometer.
- **Native** – code compiled for a specific mobile OS and device type.

Our advice to mobile application architects is to avoid letting themselves be forced into irrevocable decisions. By combining the best of several approaches, we can deliver the same app via mobile web for occasional users but also as a crisp, natively installed experience. Further, we can take advantage of widely available JavaScript coding skills but still deliver high performance and a rich native user interface.

Enterprise IT organizations often start with the assumption that they must make compromises to deliver an app within their limited resources, and as a result they don't impress anyone.

Breaking down these barriers is important because it lets us be more ambitious with mobile app development.

Investments in mobile apps are often justified as a response to rising employee expectations that work tools should be as mobile and responsive as consumer computing experiences. Yet enterprise IT organizations often start with the assumption that they must make compromises to deliver an app within their limited resources, and as a result they don't impress anyone. They would love to deliver an app everyone would recognize as "awesome" but instead find themselves settling for "good enough." But even in those cases where employees can be required to use a workplace app, they will use it more enthusiastically and effectively if the experience is top-notch.

That means shifting focus from base technologies to making good application design choices. For example, a good smartphone app cannot duplicate all the functions of a desktop application without becoming unusable. An app designed to run on an Apple Watch or similar wearable device must be even more streamlined. A recent Gartner report on *"The Mobile App Dilemma: One App Versus Many,"* suggests that as a rule of thumb smartphone apps should be "no more than two screens deep and four to six screens in total." That means more than one app may be required for a given business process. For example, it might be better to create a separate app for approving employee expense reports rather than trying to cram that functionality into the same app employees use to submit their expenses. On the other hand, enterprises that pursue a multi-app strategy must take care not to overdo it–and to make sure all their apps work well together.

With that goal in mind, let's review the barriers to no-compromise app development and why they are no longer insurmountable.

Developers can maximize portability by writing apps for the mobile web; however, truly impressive and immersive mobile web experiences are rare.

Two Extremes for User Experience

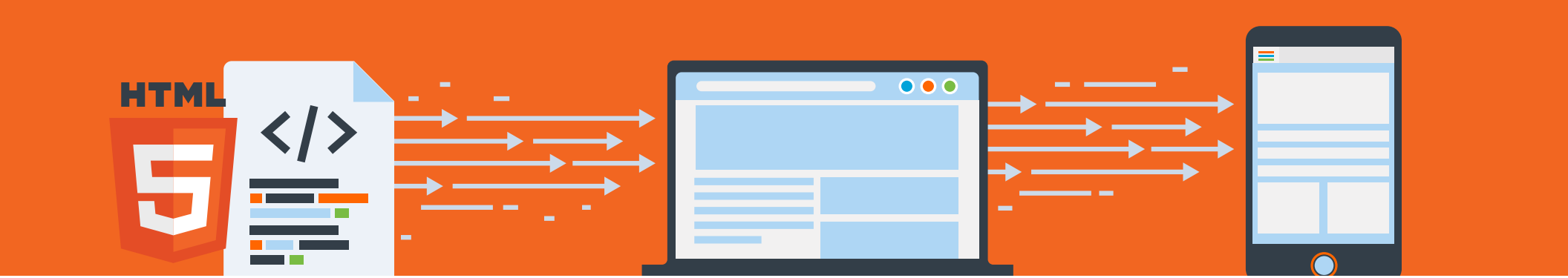
Following the release of the iPhone in 2007, and the introduction of the App Store the next year, developers scrambled to learn the Objective C programming language Apple favored for development on the platform. Those who mastered this previously somewhat obscure object-oriented programming language could produce apps that enjoyed maximum performance and the best operating system integration on iOS. The only catch: producing an equivalent app for Google's Android, which was also growing in popularity, meant rewriting the software in Java.

Using the native programming framework associated with each mobile operating system was the best way to access user interface widgets and device features, leading to the smoothest, fastest, best integrated user experience.

Other options have proliferated since then, including rapid application development tools that generate much of the native code. But deciding whether to target the native platform or use a cross-platform framework is still a common tradeoff.

At the other extreme, developers can maximize portability by writing apps for the mobile web. Using a combination of HTML, JavaScript, and CSS formatting rules that sense and adapt to various screen sizes, developers can create web apps that work well within mobile browsers. Mobile web apps cannot take advantage of all the features of a mobile OS, but neither are they tied to it. This makes perfect sense for publications and marketing websites, allowing visitors to view and interact with content immediately, without the need to download and install an app. Mobile web experiences can also present simple web forms and AJAX-style JavaScript interactivity.

However, truly impressive and immersive mobile web experiences are rare. A September 2016 Forrester Research report titled *"A Billion Mobile Sites Spark No Joy"* concluded that one reason mobile web experiences disappoint is they are too often "responsive retrofits to 20-year-old desktop web designs." Forrester predicts that by 2019 a majority of web traffic will be coming from mobile devices, meaning that businesses need to put more effort into mobile-first web design. Still, the most frequent complaint reported by the mobile users surveyed by Forrester is that websites load too slowly–an almost inevitable consequence of downloading content from a remote server, as opposed to having it instantly available within an app.



The Hybrid Solution

Hybrid apps are created with advanced HTML5 web technologies but can be installed locally and take advantage of native resources such as the storage, camera, GPS, and other features of the device and operating system. Usually, when mobile developers talk hybrid they are thinking of the Apache Cordova open source framework (the foundation technology for Adobe's PhoneGap) and complementary resources such as the user interface components of the Ionic Framework.

In addition to allowing HTML5 code to be packaged into an app, Cordova provides plugins for access to device-specific features. For example, the Cordova camera plugin gives developers a common API for accessing the camera on iOS, Android, Windows Phone, and Windows desktop apps.

This means apps can be written in HTML5 and installed on any supported target operating system, delivering comparable functionality on each of them. User interface widgets are defined using web design and programming techniques but styled to resemble the native widgets of the mobile platform.

Although this web content is wrapped in an app, rather than displayed in a browser, apps written this way pay a performance penalty for being delivered as interpreted code rather than natively compiled code. Overall, that penalty has been shrinking thanks to faster processors and improved HTML and JavaScript interpreters.

For many enterprise mobile app projects, hybrid development falls into the category of "good enough"–the fastest, most practical way of writing an app that will work across multiple platforms. Perhaps the greatest advantage of this approach is that it allows developers to repurpose the same skills they use for web development–HTML, JavaScript, and CSS–rather than having to learn Objective C or the Android Java framework.

This compromise may or may not make sense for a given app. Performance characteristics and overall user experience quality can vary, depending on the functionality of an app and the plugins it incorporates. The biggest drawback to be aware of is getting locked into this approach and discovering it does not meet your needs, requiring a rewrite from the ground up when you decide you need a native app after all.

The best way to speed up the coding of an app is to write less code.

Gaining Speed From an App Development Platform

A third option is seeking help from an enterprise mobile app development platform. Tools for developers can simplify programming, design, and integration, while generating code that delivers native performance across multiple platforms. This is the approach we take with Kony Visualizer® and Kony Mobile Fabric®, so naturally we think it has a lot of advantages. But there are also some potential pitfalls for Visualizer and other tools in this category. In the next section, we will explain our approach to minimizing them. But first, the background:

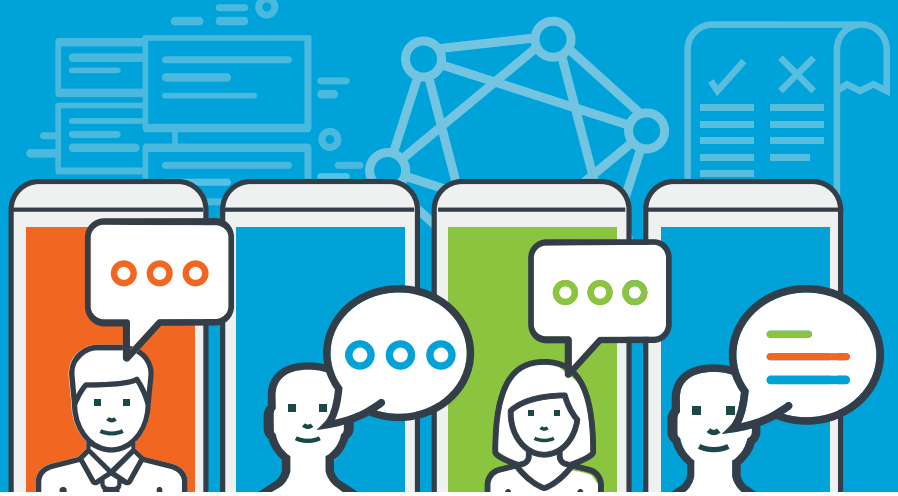
The best way to speed the coding of an app is to write less code. Visual tools simplify mobile app development by providing widgets that can be assembled on a canvas and wired together to specify common interactions such as executing a search when a button is pressed. Custom application logic can then be added as code, but much less of it will be required.

By allowing early prototypes to be shared through an app preview, Kony Visualizer also helps developers gather end-user feedback and focus on coding the features users find most valuable.

What's the catch? Certainly, you could find yourself locked into a development framework specific to your choice of tool. In contrast to the hybrid approach that takes advantage of web standards, a tool-specific strategy may require training people on much less transferable skills.

The biggest worry is that the tool, or the development platform, winds up limiting what you can create, steering your development team in the direction of generic apps rather than giving you the freedom to innovate.

We have heard all these concerns from customers and potential customers. Our answer is to build bridges between several of the approaches we have discussed here.



We enable people trained as web developers to become mobile developers.

Eliminate Unnecessary Compromises

We do not believe it is necessary to choose between native performance and flexibility or between speed of development and innovation.

For example, Kony Visualizer lets developers create applications that include high performance native widgets, but write their application logic in JavaScript. That is the most basic way we allow people trained as web developers to become mobile developers in short order. As a low-code environment built around a development tool, Kony is essentially delivering its own framework, which we refer to as "Kony JavaScript."

One of the most important recent platform improvements is the ability to import code from other JavaScript frameworks such as Cordova, Ionic, JQuery, AngularJS, and others.

This is significant in a few ways:

- You can enhance a Kony app with components pulled from other frameworks.
- You can enhance an existing hybrid app, such as one created with Cordova, by replacing parts of it with Kony high-performance native user interface components.
- You can mix-and-match, using a Kony-optimized user interface for the home screen and other parts of the app requiring rich interactivity, while using a hybrid user interface for other parts of the app with less exacting requirements. For example, a shopping app can take advantage of Kony native integrations to deliver push notifications of recommended products but use a hybrid approach to display search results pulled from an e-commerce website.

Regardless of which of these approaches you take, developers will always write application logic in JavaScript–meaning you can take advantage of a widely available programming skill rather than a scarce one.

These are some of the benefits of Kony Nitro™, the patented, omni-channel engine at the heart of the Kony Mobility Platform. Kony Nitro makes it easy to build rich native and web apps that work across devices and operating systems using visual design & development tools to accelerate delivery. Kony Nitro's cross-platform Javascript API enables enterprises to use widely available and lower cost web development resources to build 100% native apps without compromises.

Apps created on the Kony platform can also be tweaked for delivery in multiple ways–not just for smartphone and tablet operating systems but also desktop apps, kiosks, wearable devices, and emerging modalities like chatbots. For example, Kony customer QRS Music created an Apple Watch version of its player piano control app, using a subset of the functions from its apps for phones and tablets.

Planning for wearables as part of mobile app development makes sense because wearable apps are often extensions of mobile apps, functioning as an additional screen and input device linked to the code running on a phone. One difference is that with wearable platforms like WatchOS for the Apple Watch, compiling to native code is mandatory. Kony Visualizer gives you that option.

Get Started Now

Our goal is to let you take advantage of everything Kony offers to speed mobile app development without feeling boxed in. In addition to allowing imports from other popular JavaScript frameworks, we give you direct access to the native platform APIs when you need it. You can write code designed to work on any mobile platform, most of the time, but still reserve the option of using platform-specific capabilities that will make your app better. If a mobile OS introduces a new capability that we do not yet support in Visualizer–or if you have an idea about how to exploit a native capability that we never imagined–you can write your own extensions.

We promote Kony Visualizer as a low-code tool, meaning developers can accomplish most tasks by adding short snippets of code to our visual components. But we have enhanced the Visualizer code editor to allow for more ambitious coding where necessary.

Avoiding unnecessary compromises does not mean that you will never make choices or set priorities, but it means you can make bolder choices and set more ambitious goals for your mobile development.

Make bolder choices and set more ambitious goals for your mobile development.

Mobilize the Enterprise

Kony is the fastest-growing, cloud-based enterprise mobility solutions company and an industry leader among mobile application development platform (MADP) providers. Kony empowers today's leading organizations to compete in mobile time by rapidly delivering multi-edge mobile apps across the broadest array of devices and systems, today and in the future. Kony offers pre-built business mobile apps to help organizations better engage with customers and partners, as well as increase employee productivity through mobile device access to company systems and information. Powered by Kony's industry-leading Mobility Platform, enterprises can design, build, configure, and manage mobile apps across the entire software development lifecycle, and get to market faster with a lower total cost of ownership.

To learn more contact us at:

Phone: 1.888.323.9630 | info@kony.com | www.kony.com

How can Kony help you get started? Find out today with a free download of Kony Visualizer Starter Edition.

DOWNLOAD VISUALIZER