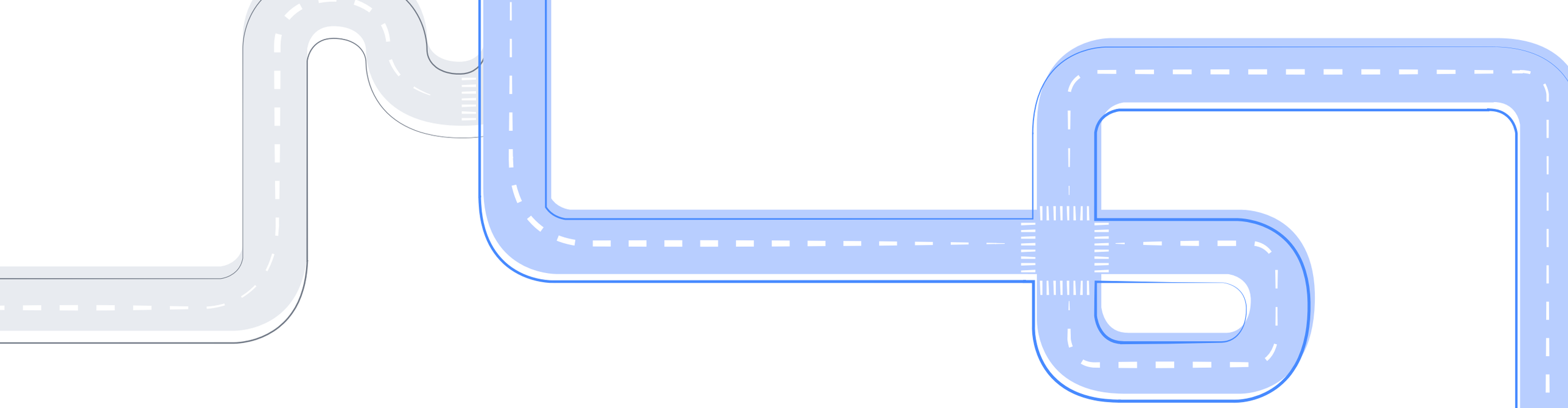
IONIC

Hybrid vs. Native

An introduction to cross-platform hybrid development for architects and app development leaders



App Development at a Crossroads

Back in the early days of mobile, there was really only one way to give users the performance and features they expected. You had to use a native SDK.

Of course, that came with tradeoffs. Building in parallel for each mobile platform. Managing multiple codebases. And hiring and retaining highly specialized native developers.

With demand for mobile experiences increasing at an exponential rate, and talent, time, and dollars remaining relatively constant, it's no surprise that most app dev teams are struggling to keep up with demand.

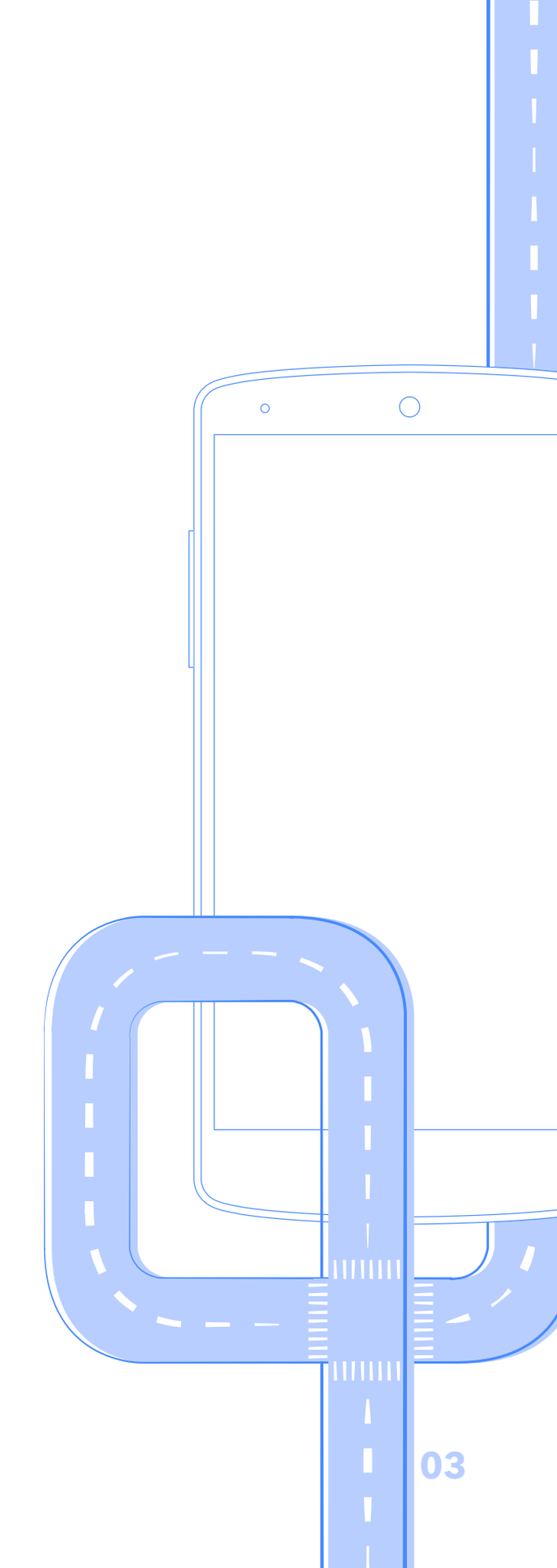
Many are now looking at cross-platform hybrid frameworks as a way to simplify and speed up development. With popular brands like MarketWatch, Sworkit, and Untappd choosing hybrid over native, they've proven that you now have a choice. A choice that will affect future development costs and flexibility well beyond mobile applications, as you'll see later in the ebook. Let's take a look.

What is a “Hybrid App”?

The term “hybrid app” is a bit of a misnomer. Hybrid apps are essentially native apps. They’re downloaded from a platform’s app store or marketplace, and access the same native features and hardware-based performance acceleration as any app built with a native SDK.

The key difference is that hybrid apps are built using open web technologies like JavaScript, HTML, and CSS, rather than the proprietary or specialized languages used by iOS, Android, and others. That means anyone with a basic web developer skill-set can build an app using the hybrid approach.

Hybrid apps run in a lightweight web container that is invisible to the user. Through customizable native plugins, they can access the native features of specific mobile devices (such as the camera or touch ID), without the core code being tied to that device. **That means hybrid apps can run on any platform or device, all from a single codebase, while still delivering native features and performance.**



It’s Not 2007 Anymore

In the early days, hybrid got a bad rep.

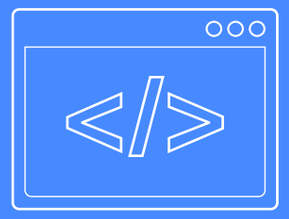
At that time (circa 2007 to 2010), none of the JS frameworks were optimized for mobile. And the devices weren’t quite ready for hybrid either. On top of that, while native developers got a complete SDK with a rich library of UI components, hybrid developers got a few Cordova plugins and a Webview.

What’s changed? A lot.

Hybrid mobile frameworks like Ionic give developers a complete SDK in a lightweight package that’s optimized for mobile devices.

At the same time, the latest smartphones are now more powerful than most laptops. **As a result, today’s hybrid apps look and feel just as smooth and responsive as any native-built app.**

Why Hybrid?



Write once,
run anywhere

Use the talent you
already have

Deliver a great
user experience
across platforms

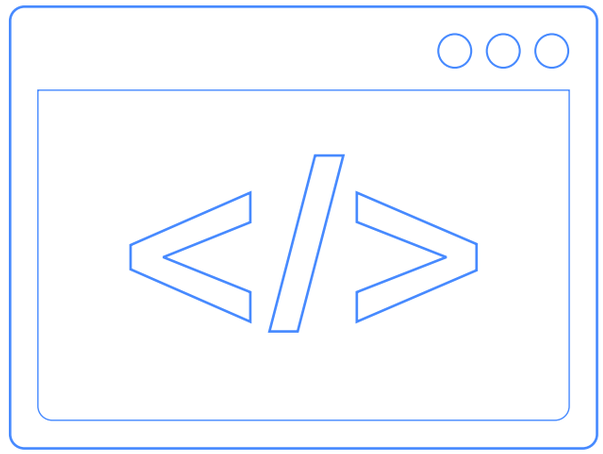
Build for
the future

Let’s take a look at each of these.

Until recently, it was quite common to program Android apps in Java and iOS apps in Swift/Objective-C. This is quite cumbersome because you have to maintain two code bases that are doing almost the same [thing].

Multiplatform frameworks get rid of having to maintain two code bases, and they are thus becoming very popular.

InfoWorld - October 2017
Apple's Swift is losing developers to multiplatform frameworks



Write Once, Run Anywhere

Rarely is a mobile app only designed for a single platform. Consumers, partners, and employees all have a choice of platforms and devices.

Following the native approach, that meant you needed to build separate apps for each mobile platform, and sometimes specific apps for tablets and smartphones.

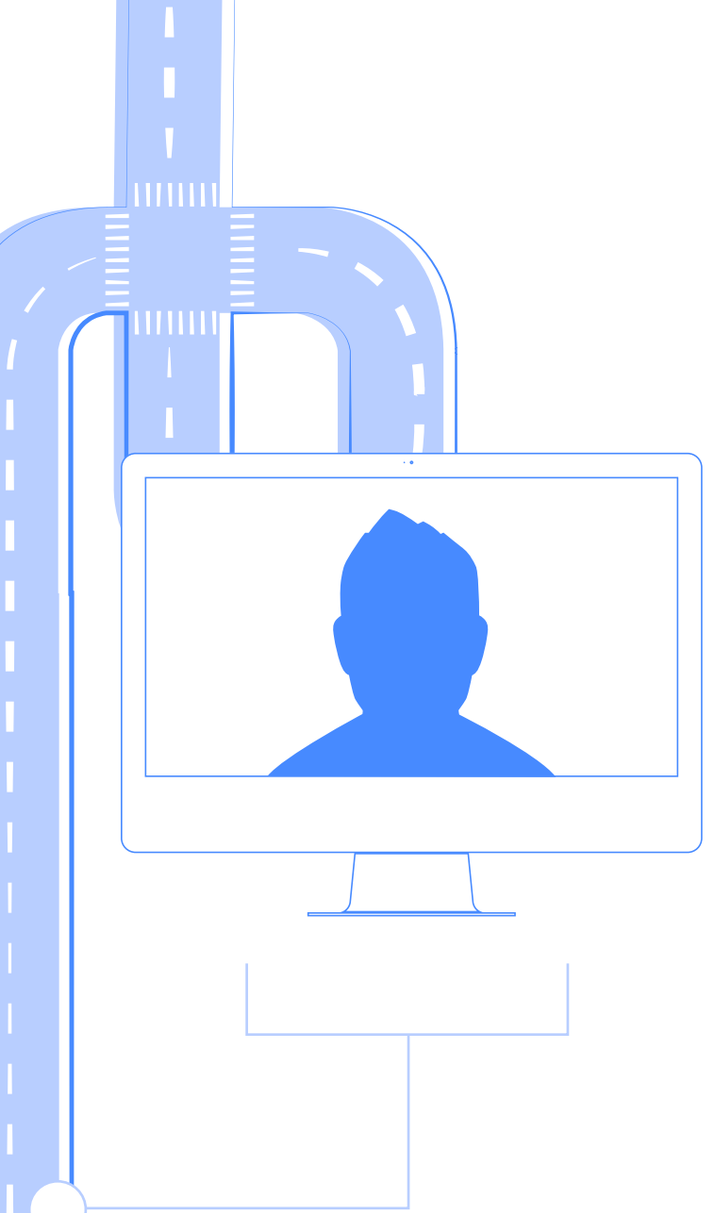
This is where hybrid development shines.

With a hybrid framework like Ionic, you can run your app on any platform or device, all from a single codebase. Ionic also provides platform continuity, so the look and feel of your app isn't one-size-fits-all. It automatically adapts to each platform. And because hybrid technology is all web-based, you can even run your app in a desktop browser as a progressive web app, or PWA.

That means your users get a great experience across platforms and devices, and you only have one codebase to worry about.

According to the 2017 Stack Overflow Survey, only 23% of respondents classified themselves as mobile developers. And only 6.5% of all developers cited Swift and Objective-C as familiar languages. In contrast, web developers made up 72.6% of respondents, and JavaScript appeared as the most commonly used programming language in the survey.

Ionic Blog - July 2017
In-House Teams & Talent Rising to the Challenge
Stack Overflow Developer Survey Results



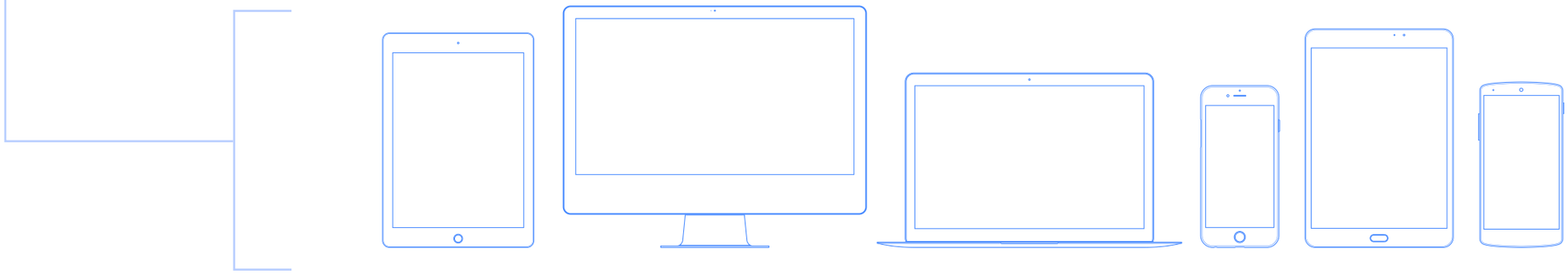
Use the Talent You Already Have

The web developer community is about 30x greater in size than the number of native mobile app developers. Many development teams already have a deep bench of programmers who understand HTML, CSS and JavaScript.

Why not leverage the talent you already have in-house to build your next mobile apps?

With a hybrid framework like Ionic, your existing web teams can build beautiful looking apps that run on any platform or device, using the tools and technology they already know and love. That's a lot easier than recruiting, training, and hiring specialists.

Plus, centralizing on a single skillset makes it much easier to reassign teams when a project is finished - whether that's a desktop web app or another mobile project.



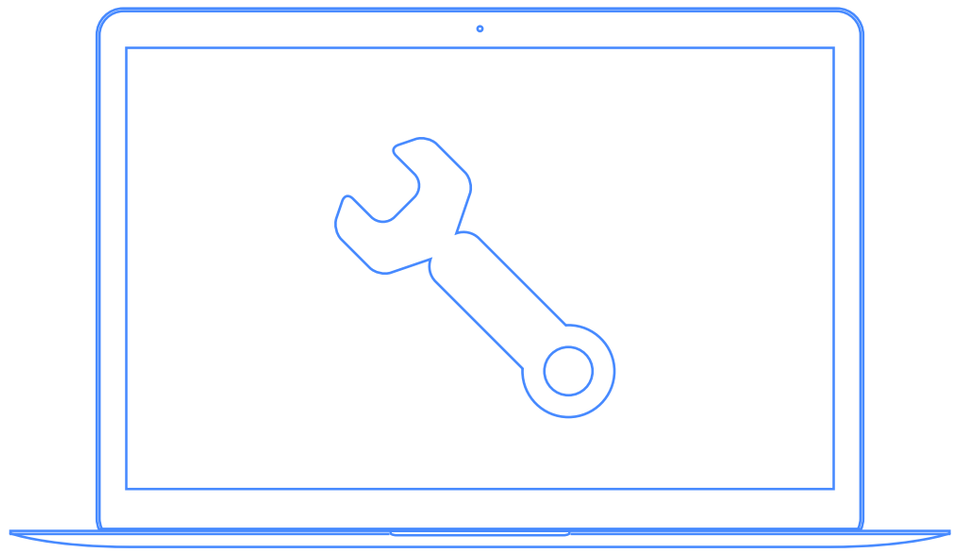
The Best UX Across Platforms

Before Dow Jones MarketWatch made the switch to hybrid, their app store reviews were in the low 2s. Now, they're consistently around 4.5 out of 5.

Native advocates will claim that only a native approach can deliver the speed and performance that you need to create a great UX. Sure, we think that's important as well, and hybrid-built apps offer the same hardware-based performance acceleration as native apps.

But what the MarketWatch team found is that user experience isn't just about performance. Simplifying app dev and consolidating onto a single codebase means more time to add features. Fewer defects. And more time to fix bugs that find their way through.

Most importantly, a great UX means a consistent app experience as your users move between platforms, devices, and modes of interaction, including mobile and desktop browsers. Only hybrid can deliver that kind of consistency.



Building for the Future

Development organizations are tasked to build applications for the future.

Future applications will be run on a growing number and diversity of platforms oriented around the web: wearables, IoT devices and other M2M communications like GPS in cars, asset tracking systems, or portable medical devices.

Thankfully, the web is the most widely used application runtime in the world.

Hybrid development taps into this mainstream. Using a forward-looking development stack lays the groundwork for you to seize opportunities beyond today's devices, like Progressive Web Apps.

Moving now to the web platform offers you richer, more innovative options moving forward. How you capitalize on these opportunities will be the measure of your success.

Final Thoughts

It’s pretty clear the future will require increasing mobile interaction with employees, partners and consumers.

Development organizations can meet this challenge better by moving to hybrid tools and frameworks.

Hybrid development offers time and cost savings from a single code base, with more control over a quality user experience. And joining the mainstream of hybrid development on web offers future cost control and flexibility well beyond mobile applications.

Let Us Help With Your Hybrid Strategy

Ionic makes it easy to build high performance apps that look and feel beautiful on any platform or device. The Ionic Framework is the #1 adopted cross-platform development framework in the world, with a vibrant community of more than 5 million developers in over 200 countries. Connect with one of our App Strategists to see how we can help.

Let’s Talk Strategy