netguru

React Native - New Mobile Applications Standard

Quick Guide for CEOs

Date: 5th Oct 2017

Table of contents

React Native is a framework developed by Facebook which enables building mobile apps using JavaScript. It uses the same design as React, letting you compose a rich mobile UI from declarative components. It allows developers to reuse code across the web and on mobile and through different operating systems. With React Native you don't have to build the same app for iOS and for Android separately from scratch. On top of that, thanks to instant reloading, instead of recompiling you can build apps much faster.

React Native is becoming extremely popular in mobile development among small companies and well-established enterprises. We've also seen a big interest in React Native coming from our clients, so we decided to conduct a small study to see what's behind the framework's success.

Why Has React Native Become So Popular?

React Native enables you to transfer the codebase or its part between different mobile platforms. Once you write an app for iOS devices, you can compile it onto Android in a short period of time (and vice versa). We've noticed that in our latest React Native project, the development took us over 30% less time than it would have taken, if we had been developing for iOS and Android natively. Here are the main reasons why so many companies jump into developing applications with RN.

Additional savings when having a React web app and a React native mobile app

Most companies need both a website and a mobile app. React code for Web relies on features available in Web browsers and React Native code is based on the features available on a given mobile platform. However, you can still reuse large chunks of the code between the mobile and web apps. The future is also bright and we might be able to copy even more code from React to React Native and vice versa.

Smaller team

At the moment, you will still need native developers to develop a React Native app – they need to be able to create custom modules that expose a native feature to Javascript and consult configuration, setup, and other specifications. Depending on the number of native modules, their input will differ, but the total number of developers required in the React Native project will be smaller. It was well explained in the talk by Tal Kol during ReactNext 2016. For a product owner or a project manager, it means a smaller team to manage and lower costs. It also allows more flexibility in building the team and mixing seniority, because you do not need a separate expert for each platform.

Support from Facebook

React Native is an open-source framework, yet, there is a big player behind it who controls its quality. Facebook puts a lot of resources to ensure React Native's reliability and also to guarantee a constant

development of the platform. The company leverages React Native in their own applications, and it's been here with us for 4 years, so we can expect that Facebook won't kill it unexpectedly.

Social proof (Skype, AirBnB and Instagram already use it)

Many well-known brands have embraced React Native in mobile app development. It provides a social proof for other companies that the framework is reliable and can be used in various types of applications.

Apps performance

Most apps are not pure React Native apps, because they have some parts written in native. If we encounter some performance issues with React Native, we can transfer some code to a native module, and the performance is no longer an issue. We should always use the tools for the tasks that they are good at. React Native is great for UI, and whenever we need some advanced features in the app, we can still use a native module for that.

Simplified UI

React Native is solidly based on creating a mobile UI. In native development, it is necessary to create a sequence of actions in the application, whereas RN employs declarative programming in which such an order of implementing an action is obsolete. As a result, it is much easier to detect bugs on the paths a user can take.

4 Kinds of Applications to Build with React Native

Every time you want to create an application, you have to ask yourself what kind of OS APIs it needs to utilise and how to do that. Defining a problem-solution fit from the very beginning helps in delivering a top-notch product. Having known what needs to be done, you can decide which tech stack meets your requirements and what are the limitations of chosen technology. Even in programming, there is no rose without a thorn, so you need to be aware what will bud and what can prick.

React Native framework was presented to developers as the next step towards creating mobile applications. Before choosing it for your project, you can check whether your app matches one of the types below. If so, the idea of using RN is worth considering.

Clickable Prototypes

If you have to build a simple app that needs to be verified by a group of test users quickly, RN might be a good choice. In the process of rapid prototyping, it is important to create a prototype that represents a working version of the app quickly. Although it is hard to achieve total fidelity to the mocks at this stage, it is worth trying to validate your business idea this way.

Apps with Simplified User Interfaces

In case you have to implement User Interfaces that consist of a small number of views and interactions, consider using this framework. It might be harder to implement sophisticated animations with RN, so please keep it in mind during the design stage.

See also: We've rewritten a Swift app in React Native. Was it worth it?

Cross-platform Applications with Basic Functionalities

RN allows validating your business ideas quickly - you can have both Android and iOS prototypes developed concurrently without additional costs. As long as your app is a quasi-static one or a simple external API consumer, consider using RN.

Applications that Do Not Rely on Using Native APIs Heavily

You can always use RN by writing bridges, but - believe us - it is not something you want to do. Doing so requires additional Java/Objective-C/Swift knowledge and complicates the scope of the app. Instead of having one developer, you end up with three. Furthermore, passes over the bridge affect your performance, so - to keep your application performant - these calls have to be reduced to a minimum.

9 Examples of Big Players Using React Native

Facebook

React Native started as Facebook's hackathon project developed in response to the company's needs. Facebook wanted to bring all the benefits of web development - such as fast iterations and having a single team build the whole product - to mobile. That's how React Native was brought to life and leveraged in mobile app development for both iOS and Android apps.

The dev team converted the Events Dashboard feature in the Facebook for iOS app to React Native to test app performance, such as startup time, which is crucial in this kind of applications. It is a significant part of the user's first impression of the app and determines whether they will stay or leave. What they achieved was cutting time-to-market in half. Read about their journey with React Native here.

Skype

Skype has recently announced that it is testing a new Android app written in React Native. It's a pretty good information for all the users, as the fairly well-designed app has suffered from numerous issues. The new version is totally revamped starting from the icons to the whole layout, adding a few extra neat features as well. As Microsoft has also announced, Skype Preview for iOS is already available in limited capacity through Apple's TestFlight for Skype Insiders.

It's also worth noting that the GitHub repo with the React Native plugin for Universal Windows Platforms was recently moved to Microsoft. That means that not only can we expect Skype for iOS but also a desktop version which will be the first the big React Native project for Windows. This is a good sign for the platform. Very soon we might see RN working quite well on Windows too.

Facebook Ads

The social networking platform isn't the only React Native application that was developed under Facebook's roof. Facebook Ads was actually the first React Native app for Android and, the first fully React Native based, cross-platform app built in the company. The framework seemed perfectly suitable for the lot of complex business logic required to accurately handle differences in ad formats, time zones, date formats, currencies, currency conventions, and so on, especially that a big chunk of it was already written in JavaScript. On top of that, implementation of UI surfaces with much data would be much easier with React Native.

Numerous components developed alongside with the Facebook Ads app have been useful for other developers in building their apps.

Read more about the first fully React Native app from Facebook.

Instagram

Instagram took the challenge to integrate React Native into their existing native app starting from the simplest view you can imagine: the Push Notifications view which was originally implemented as the WebView. It didn't require building navigation infrastructure, as UI was quite simple.

The dev team at Instagram faced a few problems on the way, but they substantially improved developer velocity. **85% to 99% of code was shared between Android and iOS apps**, depending on products, thus the team was able to deliver the app much faster than they would have with a native solution.
Read more at Instagram engineering blog.

Walmart

Walmart aims really high, aspiring to become the world's largest online retailer. With such big goals the company needs to take bold moves that involve higher risk in order to gain a competitive advantage. That's why they always seek the ways to improve customer experience by trying new technologies. Walmart has already proved its innovative attitude introducing Node.js into their stack. A few years later they also rewrote their mobile app into React Native.
Walmart managed to improve performance of the app on both iOS and Android by using less resources and within shorter time span. 95% of the codebase was shared between platforms while skills and experience of developers were leveraged across the organization. React Native allowed for great performance, nearly identical to native apps, and extremely smooth animations.
Read more about their success story.

Airbnb

Airbnb has also integrated React Native into their mobile application. What they first noticed when working with the framework was that the cost of integration with existing native apps was high, but it paid off. React Native was very easy to start off, but some challenges popped up along the way. The major issue resulted from the fact that people new to React struggled with some concepts of status management in the context of a React app. The biggest advantage, on the other side, was the ability to reuse the code. Most components were extremely reusable. Moreover, React made the code very easy to refactor and iterate on.
Check out more about React Native implementation in Airbnb on this tech talk.

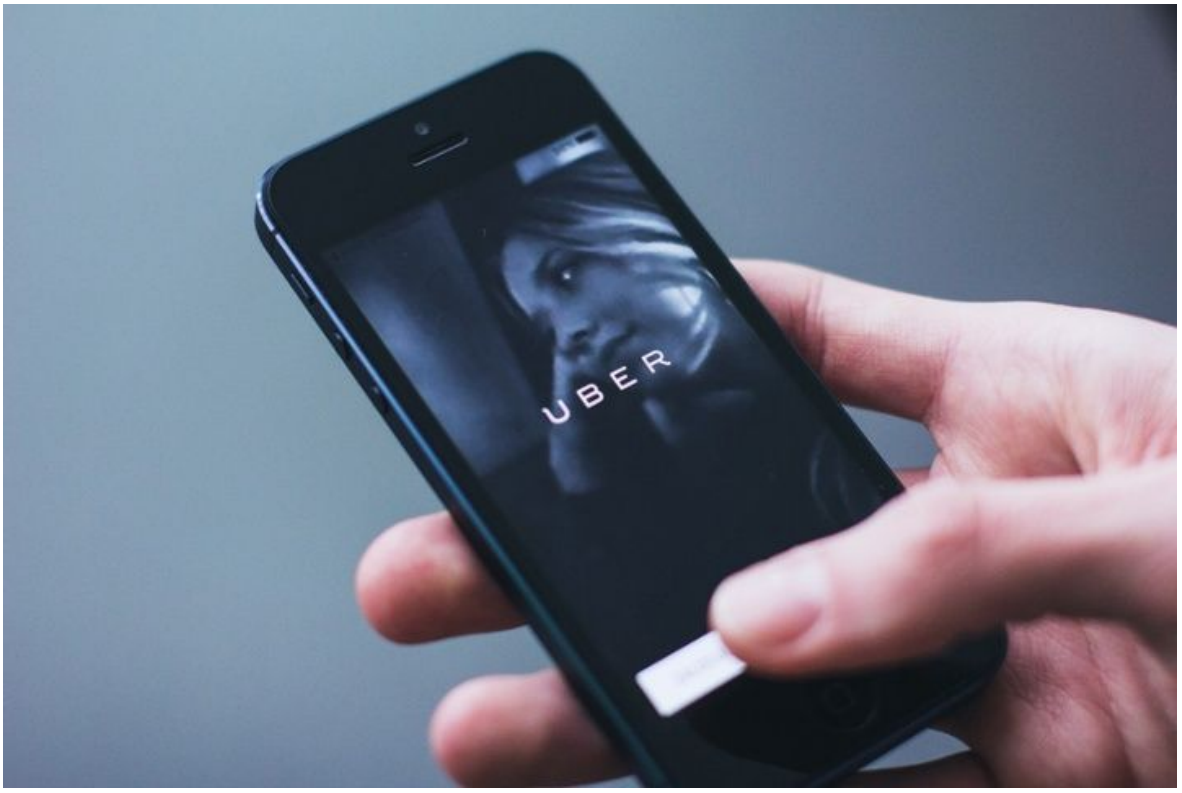
SoundCloud Pulse

SoundCloud Pulse is an application for creators that helps them manage their accounts and keep their community humming. When the company started designing the second set of native apps, they faced a few obstacles. iOS developers were impossible to find and they didn't want to have a huge gap between the iOS and Android releases. Therefore, an independent research team started to run user-testing sessions with React Native-based prototypes.

Despite a few weaknesses the team at SoundCloud spotted, their experience with the framework was generally positive. Developers found it easier to work on a React-based application than on a native application. Moreover, they were capable of building the application by themselves without frequent input from specialised mobile developers. Read more about their journey on their tech blog.

Yeti Smart Home

The dev team at Yeti Smart Home faced a similar problem as the guys at SoundCloud. They lacked iOS and Android developers, but they really wanted to start building native apps. That's why they turned to React Native, a framework that seemed to be a better option than learning from scratch, to create good native experiences in Swift and Java separately. It wasn't too easy at the beginning, as React Native was still a very immature landscape. Only with time and contribution of the growing Open Source community, the project moved forward faster and faster and then they could jump outside the mobile screen to design components that would control different devices. React Native, thanks to its modular construction, made it possible to reuse those "little bricks" to build each interface. Read more about their experience with React Native here.



Uber Eats

Uber has recently shared their insights about leveraging React Native in engineering of their food delivery app. Unlike the standard Uber app, Uber Eats marketplace involves three parties: restaurants,

delivery-partners and eaters. Such model required an extra dashboard for restaurants. The original Restaurant Dashboard was built for the web and it provided limited access to the native device functionalities, such as sound notifications which was a crucial problem for user experience. As the team had a great deal of experience using React but limited iOS/Android exposure they decided to rebuild the dashboard with RN. Although the framework constitutes only a small part of technology stack used in Uber Eats, developers are very positive about its possibilities and capacity that can help them meet needs as the marketplace grows.

Naturally, there are many more applications written in React Native and we can see that there is a growing space for the framework in the future of development. With a dynamic community growth and higher popularity, we can see many more React Native applications popping up in the near future.

Migrating Your App From Ionic, Cordova, or PhoneGap to React Native

A few years back, hybrid apps built with Ionic, Cordova or PhoneGap were a perfect and cost-effective alternative to native development. However, with the technology moving forward, they might not meet users' expectations anymore. React Native is a good alternative that can save your app from going under.

Differences Between Ionic, Cordova, PhoneGap, and React Native

Ionic, Cordova and PhoneGap are hybrid apps. Cordova is a framework which runs a JavaScript app in a WebView that has additional native extensions, which is the definition of a hybrid app. Ionic is based on Cordova and comes with Angular. It has set of standard controls that mimic native controls. PhoneGap is a distribution of Cordova with a few custom packages and tweaks.
They are all essentially websites embedded in a mobile app through what we call a WebView, whereas apps built with React Native are not mobile web apps, HTML5 apps, or hybrid apps. They are written in JavaScript but are rendered using native components, which means that the user experience will generally be closer to other native apps, because they will conform to the standards imposed by the operating system. React Native comes with better performance and smoother animations. Using React, we can build truly mobile apps indistinguishable from apps built using Objective-C (native iOS apps) or Java (native Android apps).

Risks related to staying with Ionic, Cordova or PhoneGap

The biggest problem with hybrid apps is their performance. The web was originally built for web pages, not the complex apps we produce today. Hybrid apps are going to run just fine on high-end phones but still not as smoothly as you would want. Low-end phones are where the biggest problem lies.

Most smartphone owners spend the majority of their time using only a few apps, and they expect any new app they try to be as polished as Facebook, YouTube, or Uber. With high user expectations, hybrid apps can't live up to such benchmarks. They offer poorer user experience with slower animations, lack of platform-specific gesture recognition, and keyboard misbehaviour.
Another problem is that building a hybrid app, you automatically inherit all the issues that the web has. This means bugs that only appear in one or more browsers or styles working differently across different browsers.
Another thing that the popularity of hybrid apps is on the decline. Both PhoneGap's and Ionic's showcases demonstrate a noticeable shortcoming in premier apps. In contrast, the list of apps that have migrated from hybrid/native to React Native is long. It would be quite challenging to find a high-end app that has moved from native to hybrid.

Main benefits of RN over Ionic/Cordova/PhoneGap

The biggest benefit that React Native offers is its much better performance than in the case of hybrid apps. In React Native, all views are Native, so apart from higher efficiency, you get the Native feel with super smooth animations. On top of that, the hardware functionalities are processed by the specific platform and not by Cordova. React Native renders native views without using Webview, so you don't have to worry about any problems with browser compatibility. Finally, React Native is based on React, a framework with much better support – both from Facebook and the community. Thanks to the ongoing support, React Native stays up-to-date, offers higher reliability, and gives you a superior knowledge base, where you will be able to find out how to develop apps and solve problems.

Building hybrid apps was good in the past, when we had limited resources and the technology wasn't mature enough yet. In 2017, when we have React Native, we wouldn't take the risk of developing an app in Ionic/Cordova/PhoneGap and would move any app to React Native instead.

React Native is not an ultimate solution

Despite all the benefits that React Native offers, there are of course many drawbacks you should consider before kicking off the project with React Native. Some of the things mentioned below can be solved with other solutions and development tricks. On top of that, we can expect that many issues will be solved in the future as Facebook and the vibrant community actively work on making the framework better. However, for the time being, these problems with React Native need to be taken into consideration.

Less Smooth Navigation

React Native still lacks navigation components to provide users with seamless UX. There is no ideal solution in RN for navigation between displays. It will be getting better and better, but probably it will never be as good and smooth as native navigation.

Lack of Some Custom Modules

Despite its maturity, React Native still lacks some components. Others, in turn, are underdeveloped. The chances are you won't have a problem with that, as the majority of custom modules you need are available, well-documented and working properly. However, it may happen that you will have to build your solution from scratch or try to hack the existing one.

For instance, we had problems with making shadows work in our React Native application, as the custom library was only available in beta version. Therefore, we had to come up with a solution to make it look the same as it was in the native application.

Native Developers Still Needed

Implementing some native features and modules necessitates detailed knowledge of a particular platform. React Native does offer custom modules that you can refactor across operating systems, but things such as access to device sensors, camera or push-notifications require help from iOS and Android developers. Their input depends on the complexity of your project, but you need to bargain for them when kicking off with React Native.

Facebook Rules

Facebook puts a lot of resources to ensure React Native's reliability and also to guarantee a constant development of the platform. It is unlikely that Facebook kills the framework overnight. But you still operate on Facebook's license and, at the end of the day, you're always dependent on them. Finally, Facebook has a right to revoke the licence (BSD) to use React and React Native if you get into a patent-related dispute with them.

Frequenty Asked Questions

Still on the fence about choosing React Native as the framework for your next mobile app? We've got you covered. We selected and answered the most common questions that pop up when people consider using React Native in a project. We hope this short Q&A will make things much clearer.

What's the difference between React and React Native?

React Native is not a different version of React. React Native uses React. Essentially, React Native is a custom renderer for React, just like ReactDOM on Web. Apart from transforming React code to work on iOS and Android, React Native also gives access to the features that these platforms offer.

Will my React app work on mobile? Will my React Native app work on Web?

Unfortunately, no. Most of the React code for Web relies on features available in Web browsers, so it will not work on mobile and vice versa – React Native code relies on the features available on a given mobile platform. The good news is that we can still reuse some code between the mobile and web apps, and the ability to reuse the code will improve in the future.

Will React Native make my app look and run the same way on iOS and Android?

iOS and Android offer different sets of features, and it's not React Native's responsibility to make these environments equal. React Native is only a way of accessing the native components in iOS and Android. I would say that most of the time – with some effort – we can make apps on both platforms look the same, but we shouldn't. We should stick to platform guidelines when it comes to user interface. Luckily, React Native provides us with an easy way to adapt the UI to the given platform's needs.

Should I choose React Native or native (separate iOS and Android)?

React Native is great for most apps that rely heavily on the user interface, because with little effort, we can get the UI to work on both iOS and Android, and, most importantly, we can share the business logic. Apart from that, React Native uses flexbox for layout, which works the same way on iOS, Android and Web, so we can transfer our experience from Web instead of learning more different engines. On the other hand, a native app is great when we consider using all the features that a platform offers, including such modules as video/audio processing or multithreading. Since React Native focuses on the User Interface only, it can be less efficient for applications with many native features.

Is every React developer also a React Native developer?

Most of the time when developing a React Native project, we write React code, which does not really differ from the code known from React for Web, but we should be aware of the unique properties of mobile platforms – some things might not be that obvious. React developers are not React Native developers by default. That said, we can transform React developers into React Native developers in a short period of time by giving them an opportunity to gain experience in React Native.

Does React Native mean we don't need native developers?

No, native developers are still needed. They have massive experience when it comes to configuring mobile continues integration services such as Bitrise, services for automatic builds such as HockeyApp, configuring certificates in XCode and so on. Apart from that, React Native does not easily offer every feature that is available on iOS and Android, and native developers can make those features available by creating a custom module that exposes a native feature to Javascript. In the future, having native developers work with React Native teams will not be that necessary, because

React Native will become a more mature platform. By then, most of the use cases will have been solved, and packages for pretty much everything will be available.

React Native apps still have some parts written in native code, and this is okay.

The data that we've collected clearly shows that React Native has gained momentum and it doesn't seem to slow down. It might be due to the size of the very active JS community that picked up React Native quickly or to its promise to facilitate building apps more efficiently, which has proved to be right in many projects. Developers and app founders have noticed RN benefits, and we can see more and more companies jumping on the bandwagon and developing mobile apps with Facebook's framework.