Contents

Book Revision

Revision 6pa

We'd love to hear from you!

Did you like the book? Did you find it helpful? We'd love to add your face to our list of testimonials on the website! Email us at: rn@fullstack.io[1].

Bug Reports

If you'd like to report any bugs, typos, or suggestions just email us at: rn@fullstack.io[2].

Be notified of updates via Twitter

If you'd like to be notified of updates to the book on Twitter, follow @fullstackreact[3]

[1] mailto:rn@fullstack.io?Subject=React%20Native%20testimonial
[2] mailto:rn@fullstack.io?Subject=Fullstack%20React%20Native%20book%20feedback
[3] https://twitter.com/fullstackreact

Introduction

One of the major problems that teams face when writing native mobile applications is *becoming familiar with all the different technologies*. iOS and Android - the two dominant mobile platforms - support different languages. For iOS, Apple supports the languages Swift[4] and Objective-C[5]. For Android, Google supports the languages Java[6] and Kotlin[7].

And the differences don't end there. These platforms have different toolchains. And they have different interfaces for the device's core functionality. Developers have to learn each platform's procedure for things like accessing the camera or checking network connectivity.

One trend is to write mobile apps that are powered by **WebViews**. These types of apps have minimal native code. Instead, the interface is a web browser running an app written in HTML, CSS, and JS. This web app can use the native wrapper to access features on the device, like the camera roll.

Tools like Cordova[8] enable developers to write these hybrid apps. The advantage is that developers can write apps that run on multiple platforms. Instead of learning iOS and Android specifics, they can use HTML, CSS, and JS to write a "universal" app.

The disadvantage, though, is that it's hard to make these apps look and feel like *real* native applications. And users can tell.

While universal WebView-powered apps were built with the idea of *build once, run anywhere*, React Native was built with the goal of *learn once, write anywhere*.

React is a JavaScript framework for building rich, interactive web applications. With React Native, we can build native mobile applications for multiple platforms using

[4] https://developer.apple.com/swift/

[5] https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html

[6] https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html

[7] https://developer.android.com/kotlin/index.html

[8] https://cordova.apache.org/

JavaScript and React. Importantly, the interfaces we build are *translated into native views*. React Native apps are not composed of WebViews.

We'll be able to share a lot of the code we write between iOS and Android. And React Native makes it easy to write code specific to each platform when the need arises. We get to use one language (JavaScript), one framework (React), one styling engine, and one toolchain to write apps for both platforms. *Learn once, write anywhere.*

At its core, React Native is composed of React components. We'll dig deep into components throughout this book, but here's an example of what a React component looks like:

```
import React from 'react';
import { StyleSheet, Text } from 'react-native';

export default class StyledText extends React.Component {
  render() {
    return (
      <Text style={styles.text}>{this.props.content}</Text>
    );
  }
}

const styles = StyleSheet.create({
  text: {
    color: 'red',
    fontWeight: 'bold',
  },
});
```

React Native **works**. It is currently being used in production at Facebook, Instagram, Microsoft, Amazon, and thousands of other companies.

About This Book

This book aims to be an extensive React Native resource. By the time you're done reading this book, you (and your team) will have everything you need to build

reliable React Native applications.

React Native is rich and feature-filled, but that also means it can be tricky to understand all of its parts. In this book, we'll walk through everything, such as installing its tools, writing components, navigating between screens, and integrating native modules.

But before we dig in, there are a few guidelines we want to give you in order to get the most out of this book. Specifically:

- how to approach the code examples
- how to get help if something goes wrong

Running Code Examples

This book comes with a library of runnable code examples. If you purchased a digital copy of this book, the code is available to download from the same place where you downloaded the book. If you purchased a physical copy, you can find download instructions right after the table of contents and before this Introduction chapter.

We use yarn[9] to run every example in this book. This means you can type the following commands to run any example:

- `yarn start` will start the React Native packager and print a QR code. If you're on an Android mobile device, scanning this code with the Expo[10] app will load the application. For iOS devices, see the instructions for loading apps onto your phone at the beginning of the first chapter.
- `yarn run ios` will start the React Native packager and open your app in the iOS Simulator if you are using a Mac.
- `yarn run android` will start the React Native packager and open your app on a connected Android device or emulator.

In the next chapter we'll explain each of these commands in detail.

[9] https://yarnpkg.com/en/
[10] https://expo.io/

Code Blocks and Context

Nearly every code block in this book is pulled from a runnable code example, which you can find in the sample code. For example, here is a code block pulled from the first chapter:

weather/1/App.js

```
import React from 'react';
import { StyleSheet, Text, View } from 'react-native';

export default class App extends React.Component {
  render() {
    return (
      <View style={styles.container}>
        <Text>Open up App.js to start working on your app!</Text>
      </View>
    );
  }
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
    alignItems: 'center',
    justifyContent: 'center',
  },
});
```

Notice that the header of this code block states the path to the file which contains this code: `code/weather/1/App.js`.

This book is written with the expectation that you'll also be looking at the example code alongside the chapter. If you ever feel like you're missing the context for a code example, open up the full code file using your favorite text editor.