Table of Contents

[Chapter 1: Introduction 2](#_Toc511225751)

[Chapter 2: Create a React Native application. 2](#_Toc511225752)

[Chapter 3: The basics 12](#_Toc511225753)

[Chapter 4: Redux and React-Redux 15](#_Toc511225754)

[Chapter 5: Middleware react-saga 15](#_Toc511225755)

[Chapter 6: Debugging 15](#_Toc511225756)

[Chapter 7: Build release app 15](#_Toc511225757)

Chapter 1: Introduction

React Native is a library that lets you build mobile apps using only JavaScript.

Chapter 2: Create a React Native application.

We have 2 ways to create a React Native Application: [Create React Native App](https://github.com/react-community/create-react-native-app) and The React Native CLI

Method 1: Using [Create React Native App](https://github.com/react-community/create-react-native-app)

Pros:

the easiest way to start building a new React Native application.

It allows you to start a project without installing or configuring any tools to build native code - no Xcode or Android Studio installation required

Cons:

NOT possible to include custom native modules beyond the React Native APIs and components that are available in the [Expo](https://expo.io/) client app.

Depend on Expo about support new feature or building application.

How:

Install [Node](https://nodejs.org/en/download/)

Install module create-react-native-app

Open cmd in folder that you want to create project and type:

npm install -g create-react-native-app

Create a new project named AwesomeProject.

create-react-native-app AwesomeProject

Build project

cd AwesomeProject

npm start

After building success, you will see a QR code and a link of bundle package.

Go to store, download Expo app open and scan QR code to run project.

Note: Don’t need Node or IDE go to [Snack](https://snack.expo.io/) to build an application.

Method 2: Using react-native-cli (recommend)

Installing dependencies:

Windows - Android:

Installing dependencies

You will need Node(version 6 or newer), Python2, a JDK(version 8 or newer), the React Native command line interface and Android development environment.

[Node](https://nodejs.org/en/download/), [Python2](https://www.python.org/download/releases/2.7.2/), [JDK](http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html)

React Native command line interface

npm install -g react-native-cli

Android development environment

Setting up your development environment can be somewhat tedious if you're new to Android development. If you're already familiar with Android development, there are a few things you may need to configure. In either case, please make sure to carefully follow the next few steps.

Android studio

[Download and install Android Studio](https://developer.android.com/studio/index.html). Choose a "Custom" setup when prompted to select an installation type. Make sure the boxes next to all of the following are checked:

Android SDK

Android SDK Platform

Performance (Intel ® HAXM)

Android Virtual Device

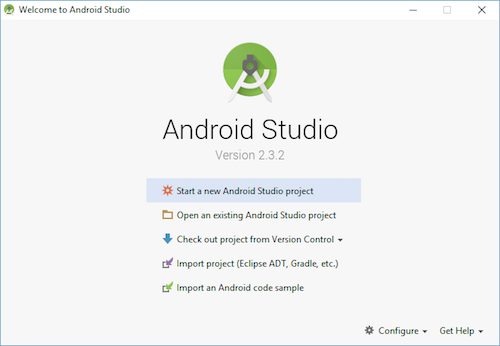
Then, click "Next" to install all of these components.

Once setup has finalized and you're presented with the Welcome screen, proceed to the next step.

3.2 Install the Android SDK

Android Studio installs the latest Android SDK by default. Building a React Native app with native code, however, requires the Android 6.0 (Marshmallow) SDK in particular. Additional Android SDKs can be installed through the SDK Manager in Android Studio.

The SDK Manager can be accessed from the "Welcome to Android Studio" screen. Click on "Configure", then select "SDK Manager".



The SDK Manager can also be found within the Android Studio "Preferences" dialog, under Appearance & Behavior → System Settings → Android SDK.

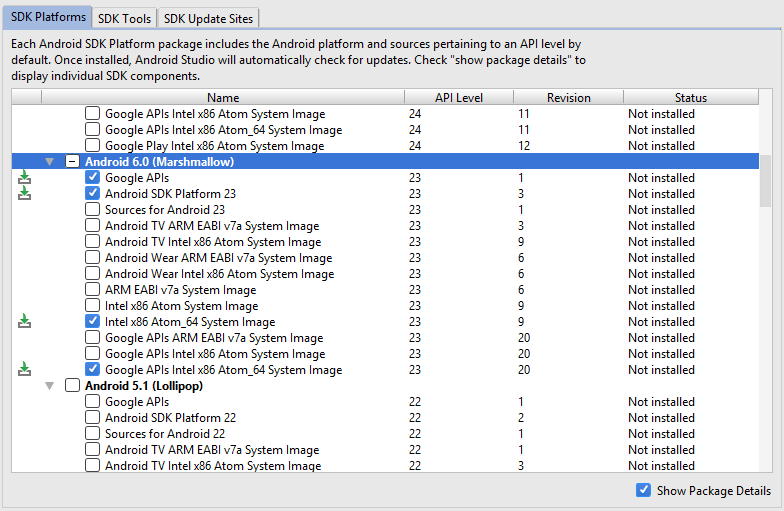
Select the "SDK Platforms" tab from within the SDK Manager, then check the box next to "Show Package Details" in the bottom right corner. Look for and expand the Android 6.0 (Marshmallow)entry, then make sure the following items are all checked:

Google APIs

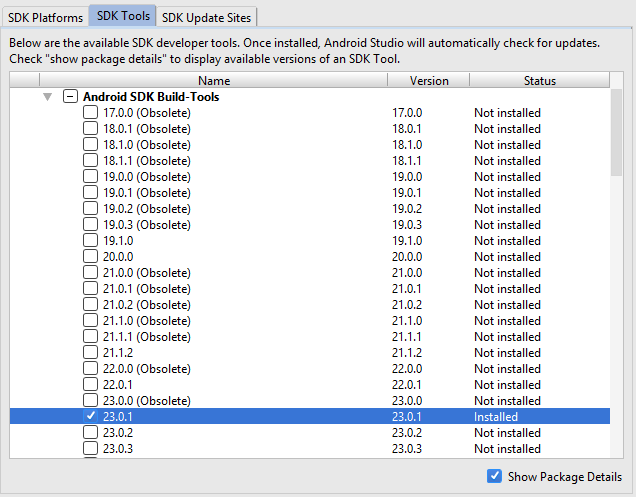
Android SDK Platform 23

Intel x86 Atom\_64 System Image

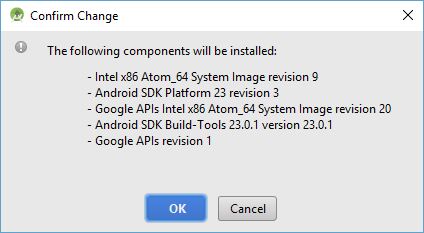
Google APIs Intel x86 Atom\_64 System Image



Next, select the "SDK Tools" tab and check the box next to "Show Package Details" here as well. Look for and expand the "Android SDK Build-Tools" entry, then make sure that 23.0.1 is selected.



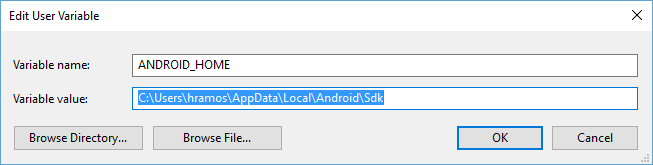
Finally, click "Apply" to download and install the Android SDK and related build tools.



3. Configure the ANDROID\_HOME environment variable

The React Native tools require some environment variables to be set up in order to build apps with native code.

Open the System pane under System and Security in the Control Panel, then click on Change settings.... Open the Advanced tab and click on Environment Variables.... Click on New... to create a new ANDROID\_HOME user variable that points to the path to your Android SDK:



The SDK is installed, by default, at the following location:

C:\Users\YOUR\_USERNAME\AppData\Local\Android\Sdk

You can find the actual location of the SDK in the Android Studio "Preferences" dialog, under Appearance & Behavior → System Settings → Android SDK.

Open a new Command Prompt window to ensure the new environment variable is loaded before proceeding to the next step.

Creating a new application

Use the React Native command line interface to generate a new React Native project called "AwesomeProject":

react-native init AwesomeProject

Preparing the Android device

You will need an Android device to run your React Native Android app. This can be either a physical Android device, or more commonly, you can use an Android Virtual Device which allows you to emulate an Android device on your computer.

Either way, you will need to prepare the device to run Android apps for development.

Using a physical device

If you have a physical Android device, you can use it for development in place of an AVD by plugging it in to your computer using a USB cable and following the instructions [here](https://facebook.github.io/react-native/docs/running-on-device.html).

Using a virtual device

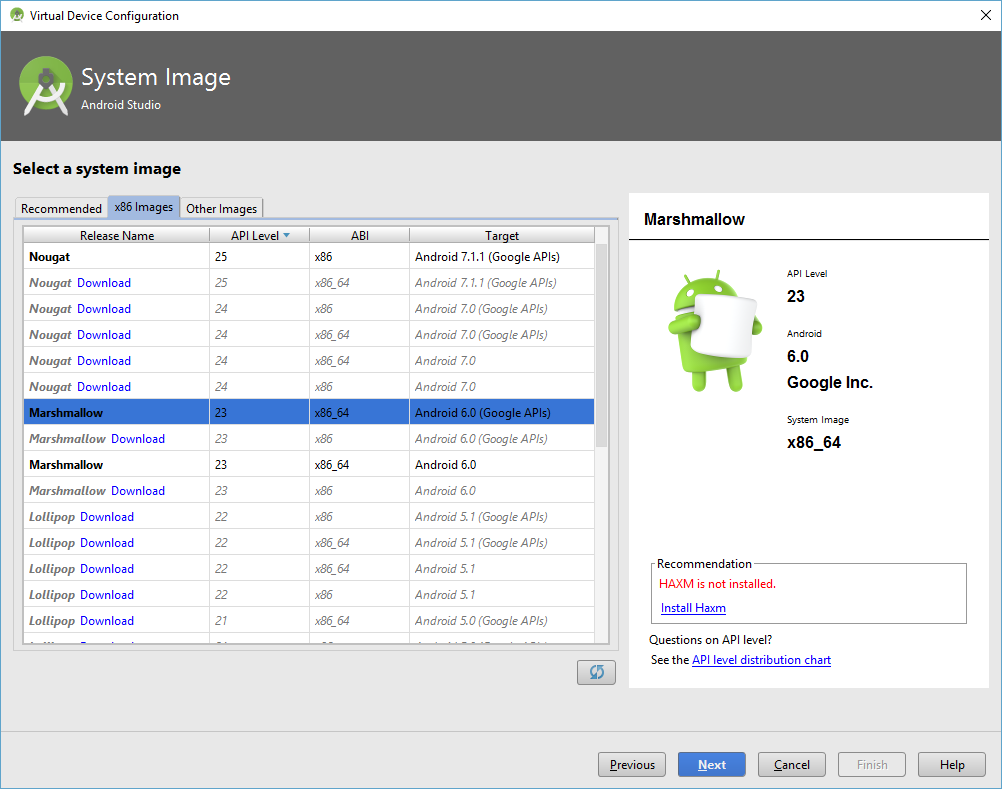
You can see the list of available Android Virtual Devices (AVDs) by opening the "AVD Manager" from within Android Studio. Look for an icon that looks like this:

Android Studio AVD Manager

If you have just installed Android Studio, you will likely need to [create a new AVD](https://developer.android.com/studio/run/managing-avds.html). Select "Create Virtual Device...", then pick any Phone from the list and click "Next".



Select the "x86 Images" tab, then look for the Marshmallow API Level 23, x86\_64 ABI image with a Android 6.0 (Google APIs) target.



AVD List

Click "Next" then "Finish" to create your AVD. At this point you should be able to click on the green triangle button next to your AVD to launch it, then proceed to the next step.

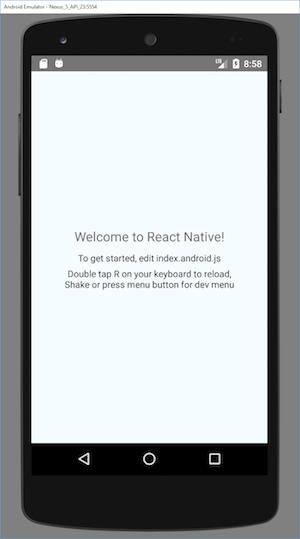
Running your React Native application

Run react-native run-android inside your React Native project folder:

cd AwesomeProject

react-native run-android

If everything is set up correctly, you should see your new app running in your Android emulator shortly.



react-native run-android is just one way to run your app - you can also run it directly from within Android Studio or [Nuclide](https://nuclide.io/).

Modifying your app

Now that you have successfully run the app, let's modify it.

Open App.js in your text editor of choice and edit some lines.

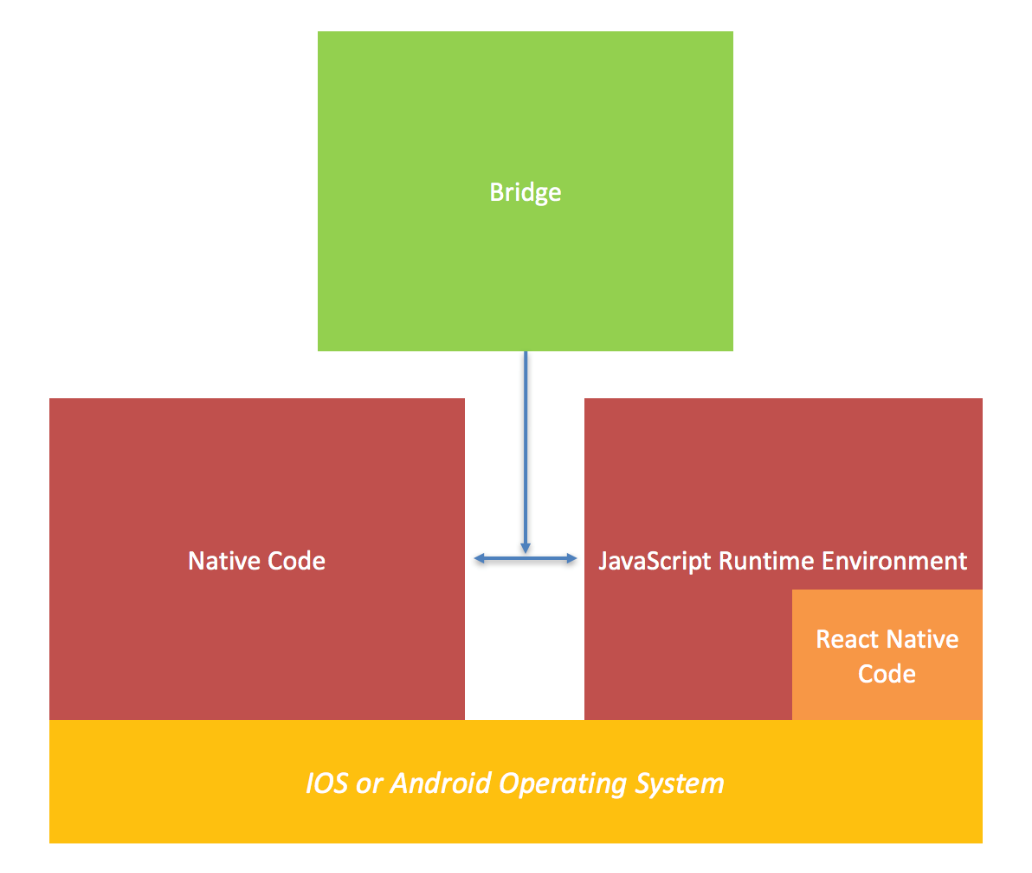
Press the R key twice or select Reload from the Developer Menu (Ctrl + M) to see your changes!

That's it!

Congratulations! You've successfully run and modified your first React Native app.

Read the [document](https://facebook.github.io/react-native/docs/getting-started.html) for more detail.

Application architecture



When you fire up the test project from the React Native website you can be left scratching your head. You have to think about the phone, the emulator, the JavaScript engine, the browser debugger and hell… where does the native even code run?

To begin building apps with React Native we need read the following [document](https://www.logicroom.co/react-native-architecture-explained/) to understand

The runtime architecture

The build architecture

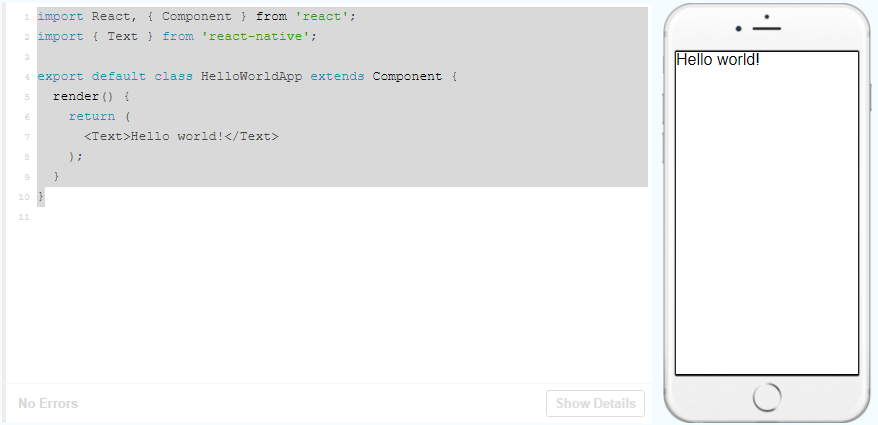
The debugging architecture

Chapter 3: The basics

To understand the basic structure of a React Native app, you need to understand some of the basic React concepts, like JSX, components, state, and props.

Hello World

In accordance with the ancient traditions of our people, we must first build an app that does nothing except say "Hello world". Here it is:



Components

When you're building a React Native app, you'll be making new components a lot. Anything you see on the screen is some sort of component. A component can be pretty simple - the only thing that's required is a render function which returns some JSX to render.

JSX

Example: <Text>This is a Text </Text> is a JSX

Props and state

There are two types of data that control a component: props and state.

props are set by the parent and they are fixed throughout the lifetime of a component. For data that is going to change, we have to use state.

Props

Props contain data that transferred from parent component.

<https://facebook.github.io/react-native/docs/props.html>

State

There are two types of data that control a component: props and state. props are set by the parent and they are fixed throughout the lifetime of a component. For data that is going to change, we have to use state.

In general, you should initialize state in the constructor, and then call setState when you want to change it.

<https://facebook.github.io/react-native/docs/state.html>

Style

<https://facebook.github.io/react-native/docs/style.html>

Height and Width

<https://facebook.github.io/react-native/docs/height-and-width.html>

Flex Dimensions

<https://facebook.github.io/react-native/docs/flexbox.html>

Layout with Flexbox

<https://facebook.github.io/react-native/docs/flexbox.html>

Handling Text Input

<https://facebook.github.io/react-native/docs/handling-text-input.html>

Handling Touches

<https://facebook.github.io/react-native/docs/handling-touches.html>

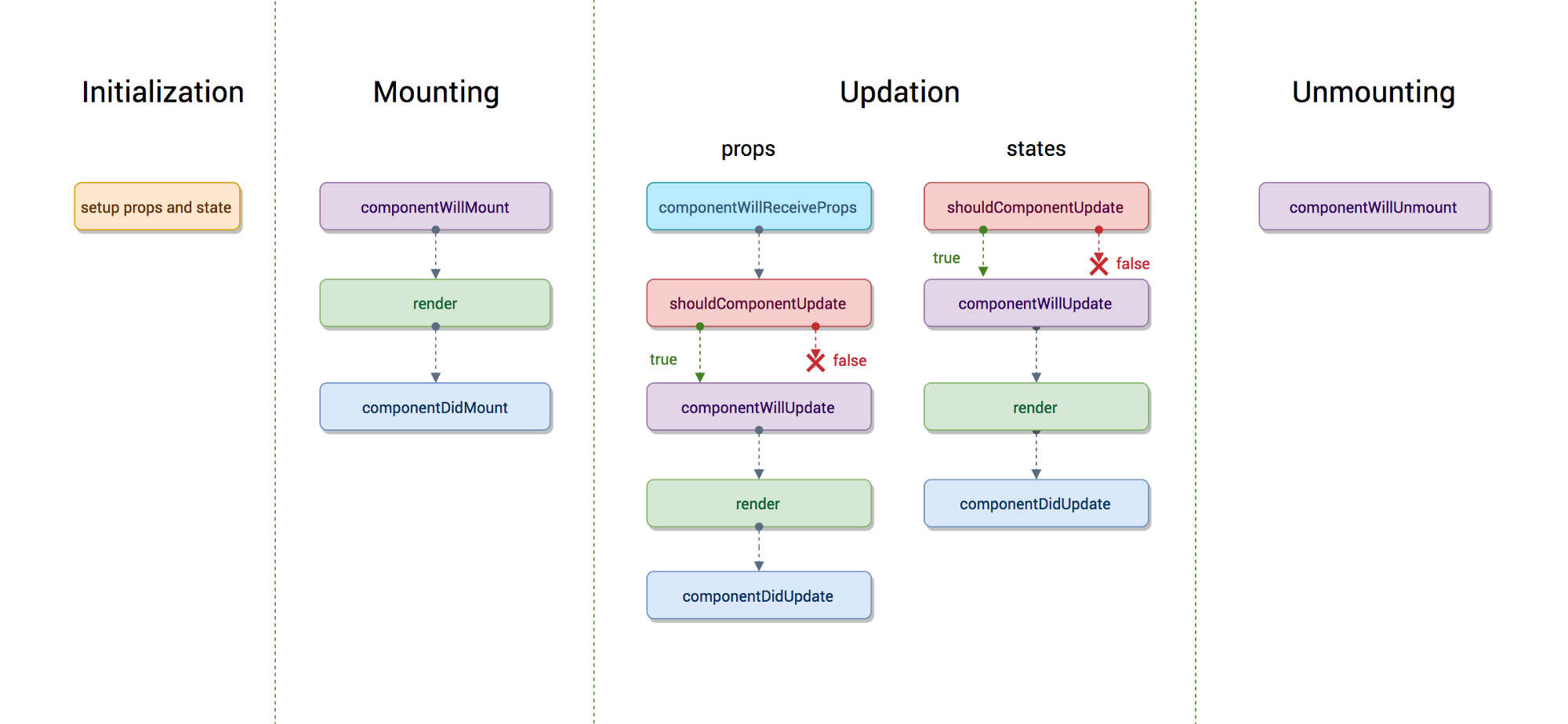
Using a ScrollView

<https://facebook.github.io/react-native/docs/using-a-scrollview.html>

Using list view

<https://facebook.github.io/react-native/docs/using-a-listview.html>

Lifecycle



<http://busypeoples.github.io/post/react-component-lifecycle/>

<https://medium.com/@baphemot/understanding-reactjs-component-life-cycle-823a640b3e8d>

Code an example to demonstrade (convert to react-native)

<https://www.tutorialspoint.com/reactjs/reactjs_component_life_cycle.htm>

Networking (todolist to server)

<https://facebook.github.io/react-native/docs/network.html>

AsyncStorage (todolist to storage)

<https://facebook.github.io/react-native/docs/asyncstorage.html>

Navigation

Document: <https://reactnavigation.org/docs/getting-started.html>

Have 4 types: stack, tab, drawer, switch

[StackNavigator](https://reactnavigation.org/docs/stack-navigator)

[TabNavigator](https://reactnavigation.org/docs/tab-navigator.html)

[DrawNavigator](https://reactnavigation.org/docs/drawer-navigator.html)

[SwitchNavigator](https://reactnavigation.org/docs/switch-navigator.html)

Chapter 4: Redux and React-Redux

<https://redux.js.org/>

The basic concepts: action, reducer, store.

React redux

<https://redux.js.org/basics/usage-with-react>

Container and component

<https://medium.com/@dan_abramov/smart-and-dumb-components-7ca2f9a7c7d0>

Chapter 5: Middleware react-saga

<https://redux-saga.js.org/>

Chapter 6: Build release app

Android: <https://facebook.github.io/react-native/docs/signed-apk-android.html>

iOS: <https://facebook.github.io/react-native/docs/running-on-device.html#building-your-app-for-production>

Chapter 7: Debugging

<https://facebook.github.io/react-native/docs/debugging.html>

Training Schedule Estimate

|  |  |  |  |
| --- | --- | --- | --- |
| No | Content | Subcontent | Estimate time (days) |
| 1 | Chapter 1: Introduction | Introduction | 0 |
| 2 | Chapter 2: Create a React Native application. | Create project by 2 ways | 1 |
| 3 | Chapter 3: The basics | Component, props, state | 1 |
| 4 | Style | 2 |
| 5 | Handle Input, Touches, ScrollView, ListView | 2 |
| 6 | Lifecycle, Networking, Async | 2 |
| 7 | Navigation | 2 |
| 8 | Chapter 4: Redux and React-Redux | Redux and React-redux | 2 |
| 9 | Chapter 5: Middleware react-saga | React-saga | 1,5 |
|  | Chapter 6: Build release app | iOS | 1 |
| 10 | Android | 0,5 |
|  |  | Total | 15 |