

React Native

Notes for Professionals

Chapter 3: Props

Props, or properties, are data that is passed to child components in a React application. React UI elements based on their props and their internal state. The props that a component takes in how it can be controlled from the outside.

Section 3.1: PropTypes

The prop-types package allows you to add runtime type checking to your component that is passed to the component are correct. For instance, if you don't pass a name or surname component below it will throw an error in development mode. In production mode the prop done. Defining propTypes can make your components more readable and maintainable.

```
import React, { Component } from 'react';
import PropTypes from 'prop-types';
import { AppRegistry, Text, View } from 'react-native';

import styles from './styles.js';

class Recipe extends Component {
  static propTypes = {
    name: PropTypes.string.isRequired,
    surname: PropTypes.bool.isRequired
  };
  render() {
    return (
      <View style={styles.container}>
        <Text style={styles.name}>{this.props.name}</Text>
        <Text style={styles.surname}>{this.props.surname}</Text>
      </View>
    );
  }
}

AppRegistry.registerComponent('Recipe', () => Recipe);
```

Multiple PropTypes
You can also have multiple propTypes for one prop. For example, the name I can write it as:

```
static propTypes = {
  name: PropTypes.oneOfType([
    PropTypes.string,
    PropTypes.object
  ])
}
```

Children Props
There is also a special prop called children, which is not passed in like

Chapter 8: Styling

Styles are defined within a JSON object with similar styling attribute names like in CSS. Such an object can either be put inline in the style prop of a component or it can be passed to the function StyleSheet.create(StyleObject) and be stored in a variable for shorter inline access by using a selector name for it similar to a class in CSS.

Section 8.1: Conditional Styling

```
<View style={[(this.props.isTrue) ? styles.backgroundColorBlack : styles.backgroundColorWhite]}>
```

If the value of isTrue is true then it will have black background color otherwise white.

Section 8.2: Styling using inline styles

Each React Native component can take a style prop. You can pass it a JavaScript object with CSS-style properties.

```
<Text style={{color:red}}>Red Text</Text>
```

This can be inefficient as it has to recreate the object each time the component is rendered. Using a stylesheet is preferred.

Section 8.3: Styling using a stylesheet

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

const styles = StyleSheet.create({
  red: {
    color: 'red'
  },
  big: {
    fontSize: 30
  }
});

class Example extends Component {
  render() {
    return (
      <View>
        <Text style={styles.red}>Red</Text>
        <Text style={styles.big}>Big</Text>
      </View>
    );
  }
}
```

StyleSheet.create() returns an object where the values are numbers. React Native knows to convert the numeric IDs into the correct style object.

Section 8.4: Adding multiple styles

You can pass an array to the style prop to apply multiple styles. When there is a conflict, the last one in takes precedence.

React Native Notes for Professionals

Chapter 15: HTTP Requests

Section 15.1: Using Promises with the fetch API and Redux

Redux is the most common state management library used with React Native. The following example demonstrates how to use the fetch API and dispatch changes to your applications state reducer using redux-thunk.

```
export const fetchRecipes = (action) => {
  return (dispatch, getState) => {
    fetch('recipes')
      .then((response) => {
        headers: {
          'Accept': 'application/json',
          'Content-Type': 'application/json'
        },
        body: JSON.stringify({
          recipeName,
          instructions,
          ingredients
        })
      })
      .then((res) => {
        // If response was successful parse the json and dispatch an update
        if (res.ok) {
          res.json().then((recipe) => {
            dispatch({
              type: 'UPDATE_RECIPE',
              recipe
            });
          });
        }
        // response wasn't successful so dispatch an error
      })
      .catch((err) => {
        // But if there is a general JavaScript error,
        dispatch(error({
          message: err.message,
          status: err.status
        }));
      });
    };
  };
}
```

Section 15.2: HTTP with the fetch API

It should be noted that Fetch does not support progress callbacks. See <https://github.com/github/fetch/issues/15>. The alternative is to use XMLHttpRequest. See <https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest>.

```
fetch('https://www.bbc.com/health')
  .then((res) => {
    if (!res.ok) {
      console.log('Error');
    }
  });
```

React Native Notes for Professionals

80+ pages
of professional hints and tricks

Contents

About	1
Chapter 1: Getting started with React Native	2
Section 1.1: Setup for Mac	2
Section 1.2: Setup for Linux (Ubuntu)	8
Section 1.3: Setup for Windows	10
Chapter 2: Hello World	12
Section 2.1: Editing <code>index.ios.js</code> or <code>index.android.js</code>	12
Section 2.2: Hello world!	12
Chapter 3: Props	13
Section 3.1: PropTypes	13
Section 3.2: What are props?	14
Section 3.3: Use of props	14
Section 3.4: Default Props	15
Chapter 4: Multiple props rendering	16
Section 4.1: render multiple variables	16
Chapter 5: Modal	17
Section 5.1: Modal Basic Example	17
Section 5.2: Transparent Modal Example	18
Chapter 6: State	20
Section 6.1: <code>setState</code>	20
Section 6.2: Initialize State	22
Chapter 7: Routing	23
Section 7.1: Navigator component	23
Chapter 8: Styling	24
Section 8.1: Conditional Styling	24
Section 8.2: Styling using inline styles	24
Section 8.3: Styling using a stylesheet	24
Section 8.4: Adding multiple styles	24
Chapter 9: Layout	26
Section 9.1: Flexbox	26
Chapter 10: Components	35
Section 10.1: Basic Component	35
Section 10.2: Stateful Component	35
Section 10.3: Stateless Component	35
Chapter 11: ListView	37
Section 11.1: Simple Example	37
Chapter 12: RefreshControl with ListView	38
Section 12.1: Refresh Control with ListView Full Example	38
Section 12.2: Refresh Control	39
Section 12.3: <code>onRefresh</code> function Example	39
Chapter 13: WebView	41
Section 13.1: Simple component using <code>webview</code>	41
Chapter 14: Command Line Instructions	42
Section 14.1: Check version installed	42
Section 14.2: Initialize and getting started with React Native project	42

Section 14.3: Upgrade existing project to latest RN version	42
Section 14.4: Add android project for your app	42
Section 14.5: Logging	43
Section 14.6: Start React Native Packager	43
Chapter 15: HTTP Requests	44
Section 15.1: Using Promises with the fetch API and Redux	44
Section 15.2: HTTP with the fetch API	44
Section 15.3: Networking with XMLHttpRequest	45
Section 15.4: WebSockets	45
Section 15.5: Http with axios	45
Section 15.6: Web Socket with Socket.io	47
Chapter 16: Platform Module	49
Section 16.1: Find the OS Type/Version	49
Chapter 17: Images	50
Section 17.1: Image Module	50
Section 17.2: Image Example	50
Section 17.3: Conditional Image Source	50
Section 17.4: Using variable for image path	50
Section 17.5: To fit an Image	51
Chapter 18: Custom Fonts	52
Section 18.1: Custom fonts for both Android and IOS	52
Section 18.2: Steps to use custom fonts in React Native (Android)	53
Section 18.3: Steps to use custom fonts in React Native (iOS)	53
Chapter 19: Animation API	56
Section 19.1: Animate an Image	56
Chapter 20: Android - Hardware Back Button	57
Section 20.1: Detect Hardware back button presses in Android	57
Section 20.2: Example of BackAndroid along with Navigator	57
Section 20.3: Hardware back button handling using BackHandler and Navigation Properties (without using deprecated BackAndroid & deprecated Navigator)	58
Section 20.4: Example of Hardware back button detection using BackHandler	59
Chapter 21: Run an app on device (Android Version)	60
Section 21.1: Running an app on Android Device	60
Chapter 22: Native Modules	61
Section 22.1: Create your Native Module (IOS)	61
Chapter 23: Linking Native API	63
Section 23.1: Outgoing Links	63
Section 23.2: Incomming Links	63
Chapter 24: ESLint in React Native	65
Section 24.1: How to start	65
Chapter 25: Integration with Firebase for Authentication	66
Section 25.1: Authentication In React Native Using Firebase	66
Section 25.2: React Native - ListView with Firebase	66
Chapter 26: Navigator Best Practices	69
Section 26.1: Navigator	69
Section 26.2: Use react-navigation for navigation in react native apps	71
Section 26.3: react-native Navigation with react-native-router-flux	72
Chapter 27: Navigator with buttons injected from pages	74

Section 27.1: Introduction	74
Section 27.2: Full commented example	74
Chapter 28: Create a shareable APK for android	77
Section 28.1: Create a key to sign the APK	77
Section 28.2: Once the key is generated, use it to generate the installable build:	77
Section 28.3: Generate the build using gradle	77
Section 28.4: Upload or share the generated APK	77
Chapter 29: PushNotification	78
Section 29.1: Push Notification Simple Setup	78
Section 29.2: Navigating to scene from Notification	80
Chapter 30: Render Best Practises	82
Section 30.1: Functions in JSX	82
Chapter 31: Debugging	83
Section 31.1: Start Remote JS Debugging in Android	83
Section 31.2: Using console.log()	83
Chapter 32: Unit Testing	84
Section 32.1: Unit Test In React Native Using Jest	84
Credits	85
You may also like	87

About

Please feel free to share this PDF with anyone for free,
latest version of this book can be downloaded from:

<https://goalkicker.com/ReactNativeBook>

This *React Native Notes for Professionals* book is compiled from [Stack Overflow Documentation](#), the content is written by the beautiful people at Stack Overflow. Text content is released under Creative Commons BY-SA, see credits at the end of this book whom contributed to the various chapters. Images may be copyright of their respective owners unless otherwise specified

This is an unofficial free book created for educational purposes and is not affiliated with official React Native group(s) or company(s) nor Stack Overflow. All trademarks and registered trademarks are the property of their respective company owners

The information presented in this book is not guaranteed to be correct nor accurate, use at your own risk

Please send feedback and corrections to web@petercv.com

[Click here to download full PDF material](#)