

Introduction to React Native

March 29th, 2019



ralfi
SALHON

Developer & Graphic Designer

Education

Tufts University
Computer Science (BS)

Website

tufts.io/ralfi

Github

github.com/ralfisalhon

Email

rifat.salhon@tufts.edu

ABOUT ME

Skills

01 PROGRAMMING LANGUAGES

Proficient: C, C++, HTML, Javascript, Python

Familiar: C#, Java, React.js, Swift

02 SOFTWARE & LIBRARIES

Proficient: Photoshop, PyxelEdit, React Native, Unity

Familiar: After Effects, Final Cut Pro, Illustrator, Xcode

ABOUT ME

Projects



01 RED or BLUE

2D Puzzle platformer game in a dynamically changing world. Currently free to download on our website.

Qualified for Phase 2 submissions of Boston Festival of Indie Games.

Technologies Used:
Unity (C#), PyxelEdit

bit.ly/redorbluegame

github.com/mohsr/red-or-blue



02 Hearo

Mobile app that automatically summarizes conference calls using voice transcription. Currently unreleased.

Technologies Used:
React Native (JS), Xcode, Voximplant API, Google Speech-to-Text API, Twilio API

03 Jamblr

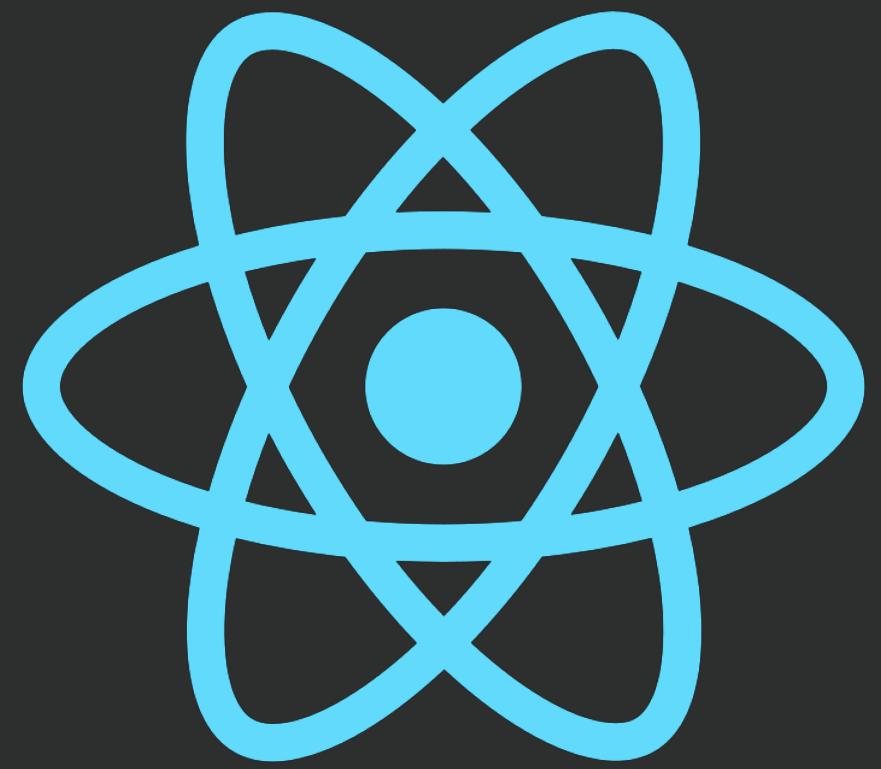
Mobile app that allows for easy music discovery and playlist creation. Currently on the App Store.

Won the HubSpot special award for best user experience and engineering values at Tufts Polyhack 2018.

Technologies Used:
React Native (JS), Node, Xcode, Spotify Web API

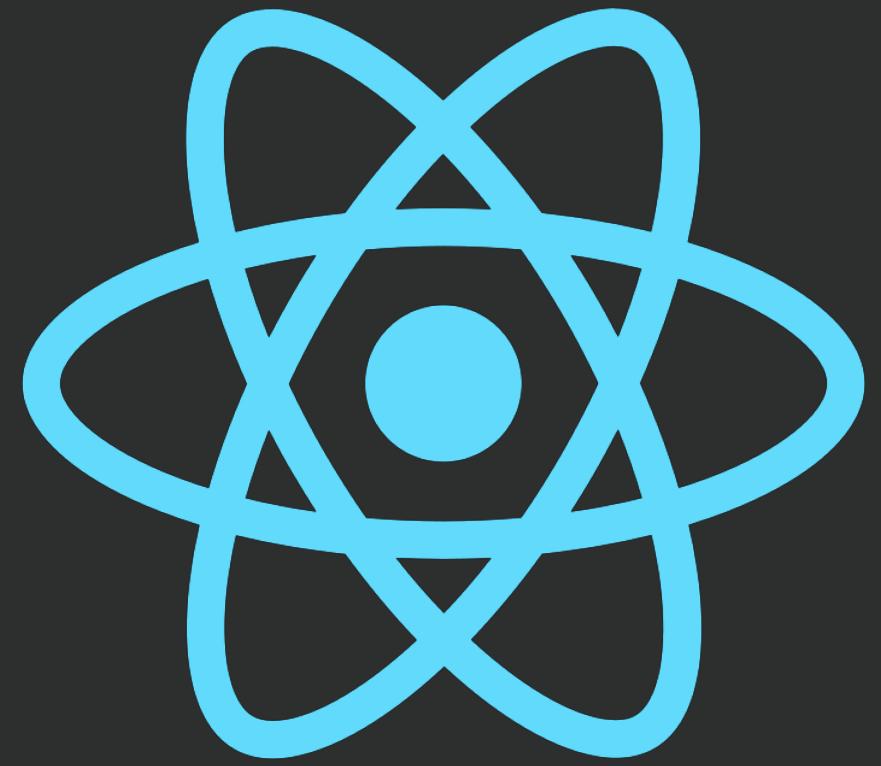
<https://apple.co/2JCioyX>





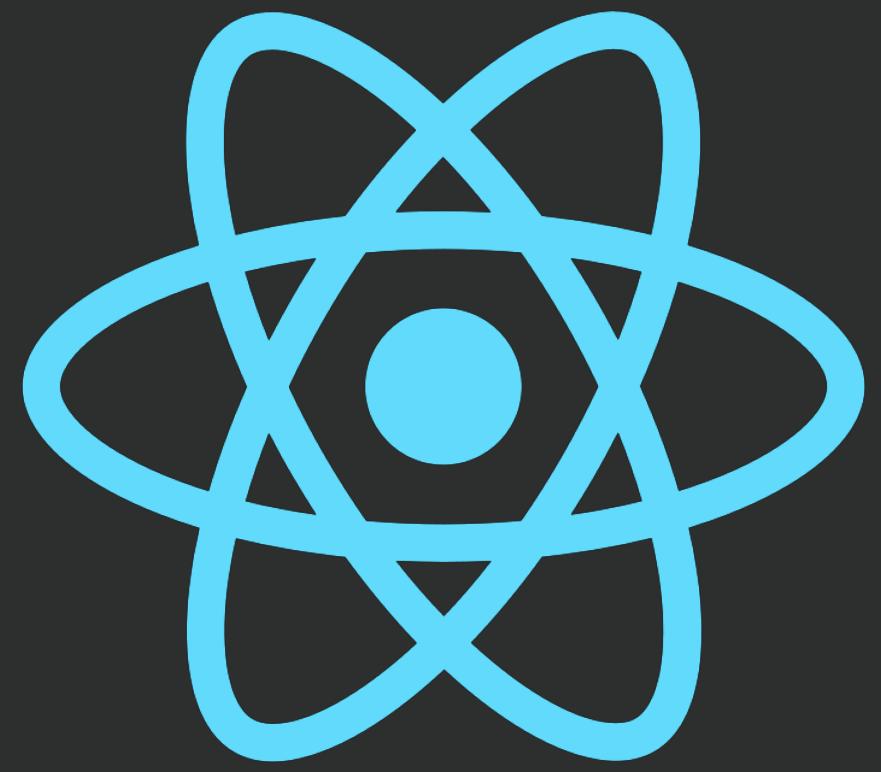
What is React Native?

- Built using React
- Build cross platform applications using Javascript
- Released on March 2015 by Facebook
- Rapidly gaining in popularity & adoption



Why React Native?

- Fast speed of development (hot reloading)
- Low cost of development (code reuse)
- Ship across multiple platforms
- Possible to ship “over the air” updates



Who's using React Native?

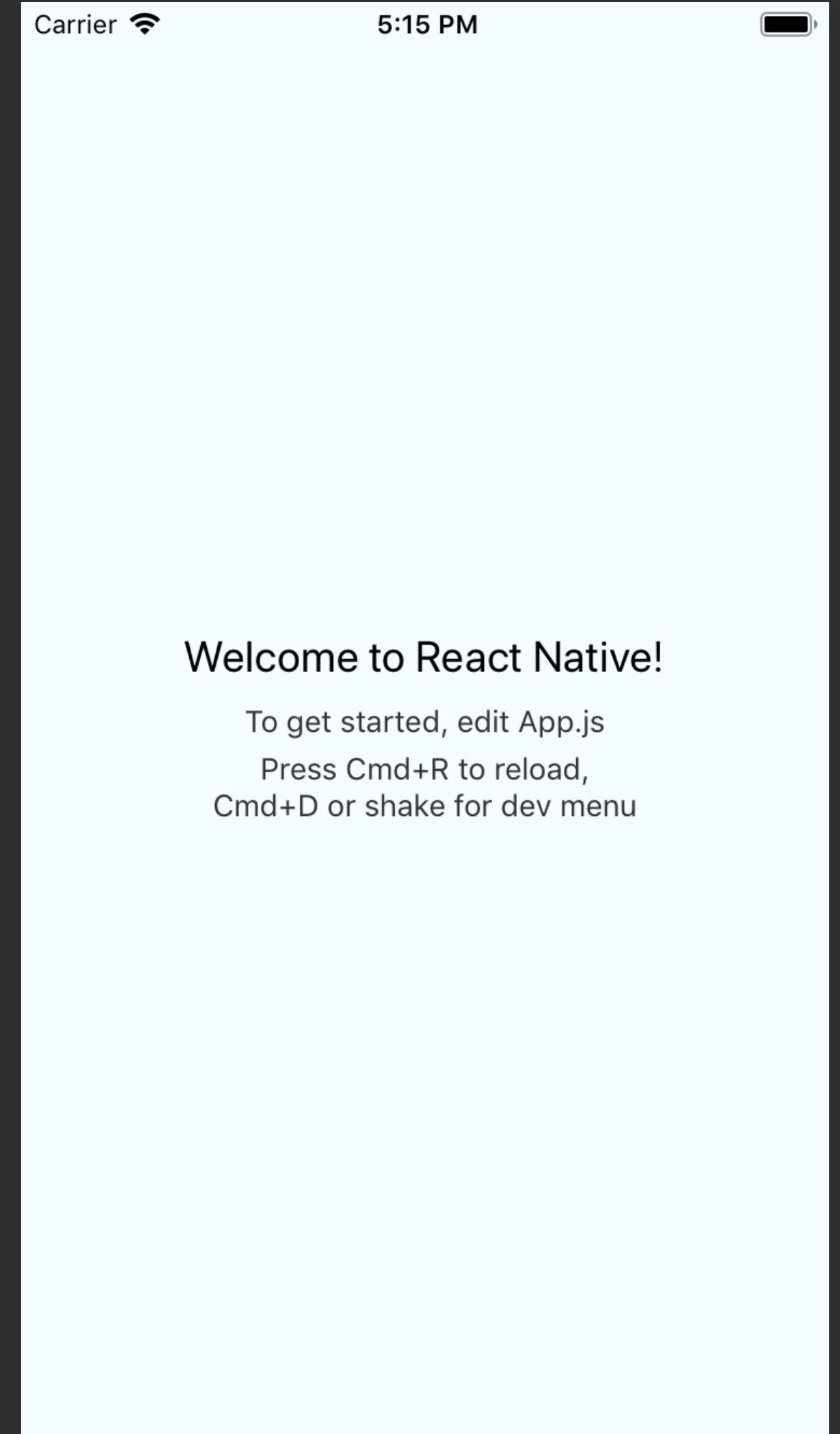
- Instagram
- AirBNB
- Skype
- Discord

Mapping to Web UI Primitives

- **View == div**
- **Text == span / p / h**
- **Image == img**
- **TextInput == input**

Creating a new Project

- *brew install node*
- *brew install watchman*
- *npm install -g react-native-cli*
- **react-native init AwesomeProject**



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

Package Imports

```
1 import React from "react";
2 ∵ import {
3   StyleSheet,
4   View,
5   Image,
6   Text,
7 } from "react-native";
8
```

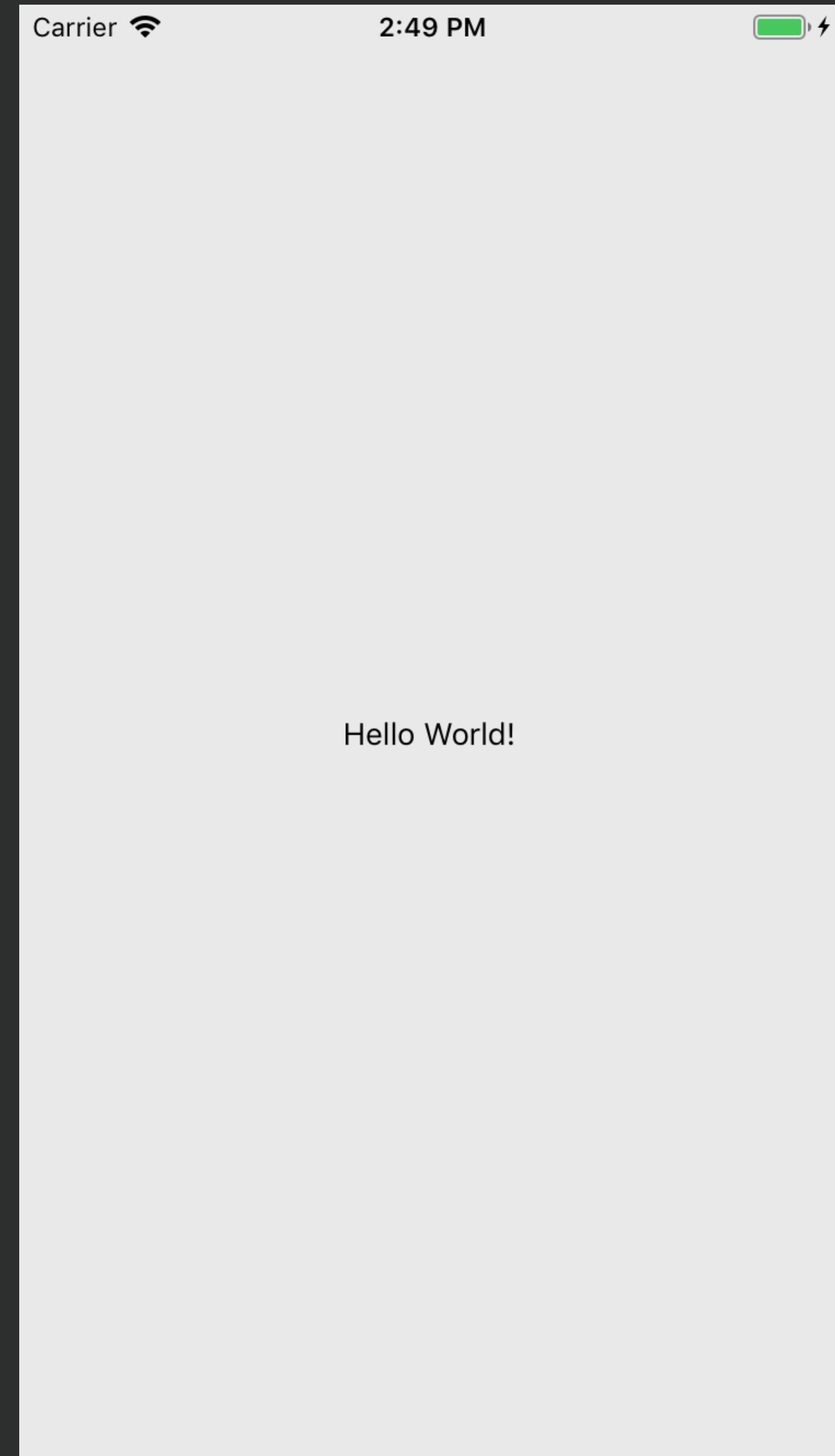
Basic Class Structure

```
9 ∵ export class HomeScreen extends React.Component {
10 ∵   render() {
11 ∵     return (
12 ∵       <View style = {styles.container}>
13 ∵         <Text> Hello World! </Text>
14 ∵       </View>
15     );
16   }
17 }
```

Hello World!

Styling (like CSS)

```
18
19 ∵ const styles = StyleSheet.create({
20 ∵   container: {
21 ∵     flex: 1,
22 ∵     justifyContent: 'center',
23 ∵     alignItems: 'center',
24 ∵   },
25 ∵});
```



The Power of Hot Loading

A screenshot of a mobile application development interface. On the left, a code editor displays the file `App.js` with the following content:

```
1 import React from "react";
2 import {
3   StyleSheet,
4   View,
5   Image,
6   Text,
7 } from "react-native";
8
9 export default class HomeScreen extends React.Component {
10   render() {
11     return (
12       <View style = {styles.container}>
13         <Text> Hello World! </Text>
14       </View>
15     );
16   }
17 }
18
19 const styles = StyleSheet.create({
20   container: {
21     flex: 1,
22     justifyContent: 'center',
23     alignItems: 'center',
24   },
25 });
26
```

The code defines a `HomeScreen` component that contains a single `Text` element with the text "Hello World!". It also includes a `StyleSheet` object with a `container` style. The code editor has a dark theme.

On the right, a mobile device simulator is shown. The status bar indicates "Carrier" with a signal icon, the time "3:05 PM", and a battery level. The main screen of the simulator displays the text "Hello World!" centered on a white background.

CONS of React Native

```
Loading from localhost:8081...

Failed to load bundle(http://localhost:8081/index.bundle?
platform=ios&dev=true&minify=false) with
error:(SyntaxError: /Users/ralfisalhon/
Desktop/rnpresent/node_modules/react-
native/App.js: Adjacent JSX elements must
be wrapped in an enclosing tag. Did you want
a JSX fragment <>...</>? (15:12)

[0m [90m 13 | [39m[0m
[0m [90m 14 | [39m      [33m<[39m[33m/
[39m[33mView[39m[33m>[39m[0m
[0m[31m[1m>[22m[39m[90m 15 | [39m
[33m<[39m[33mText[39m[33m>[39m
[33mHello[39m [33mWorld[39m[33m![39m
[33m<[39m[33m/
[39m[33mText[39m[33m>[39m[0m
[0m [90m   | [39m
[31m[1m^>[22m[39m[0m
[0m [90m 16 | [39m    )[33m;[39m[0m
[0m [90m 17 | [39m  )[0m
[0m [90m 18 | [39m){[0m (null)

  38-[RCTCxxBridge loadSource:onProgress
:]_block_invoke.228
RCTCxxBridge.mm:414
__ZL36attemptAsynchronousLoad0fBundleAt
URLP5NSURLU13block_pointerFvP18RCTLoadin

Dismiss (ESC) Reload (⌘R) Copy (⌥⌘C) Extra Info (⌘E)
```

Google

jsx elements must be wrapped in an enclosing tag

All News Videos Images More Settings Tools

About 36,100 results (0.47 seconds)

Parse Error: Adjacent JSX elements must be wrapped in an enclosing ...

<https://stackoverflow.com/.../parse-error-adjacent-jsx-elements-must-be-wrapped-in-a...> ▾

10 answers

Dec 28, 2016 - this gets converted into this(Just for illustration purpose, actually you will get error : Adjacent JSX elements must be wrapped in an enclosing tag)

Parse Error: Adjacent JSX ...

Parse Error: Adjacent JSX elements must be wrapped in ...

[More results from stackoverflow.com »](#)

Adjacent JSX elements must be wrapped in an enclosing tag · Issue ...

<https://github.com/zeit/next.js/issues/3251> ▾

Nov 7, 2017 - Next.js 4 supports **React 16**,and now **React 16** supports return an array of **elements** from a component's render method, but why it throw an error of **Adjacent JSX elements must be wrapped in an enclosing tag** when I attempt to use it in Next.js 4 ? ... Module build failed: SyntaxError ...

React Native is not beginner friendly → Google is your best friend.
Debugging can take hours → WRITE CLEAN CODE!

Learning flex

```
export default class HomeScreen extends React.Component {
  render() {
    return (
      <View style = {{flex: 1}}>
        <View style = {{flex: 1, backgroundColor: 'red'}>
        </View>
        <View style = {{flex: 1, backgroundColor: 'green'}>
        </View>
        <View style = {{flex: 1, backgroundColor: 'blue'}>
        </View>
      </View>
    );
  }
}
```

by default,
flexDirection: 'column'

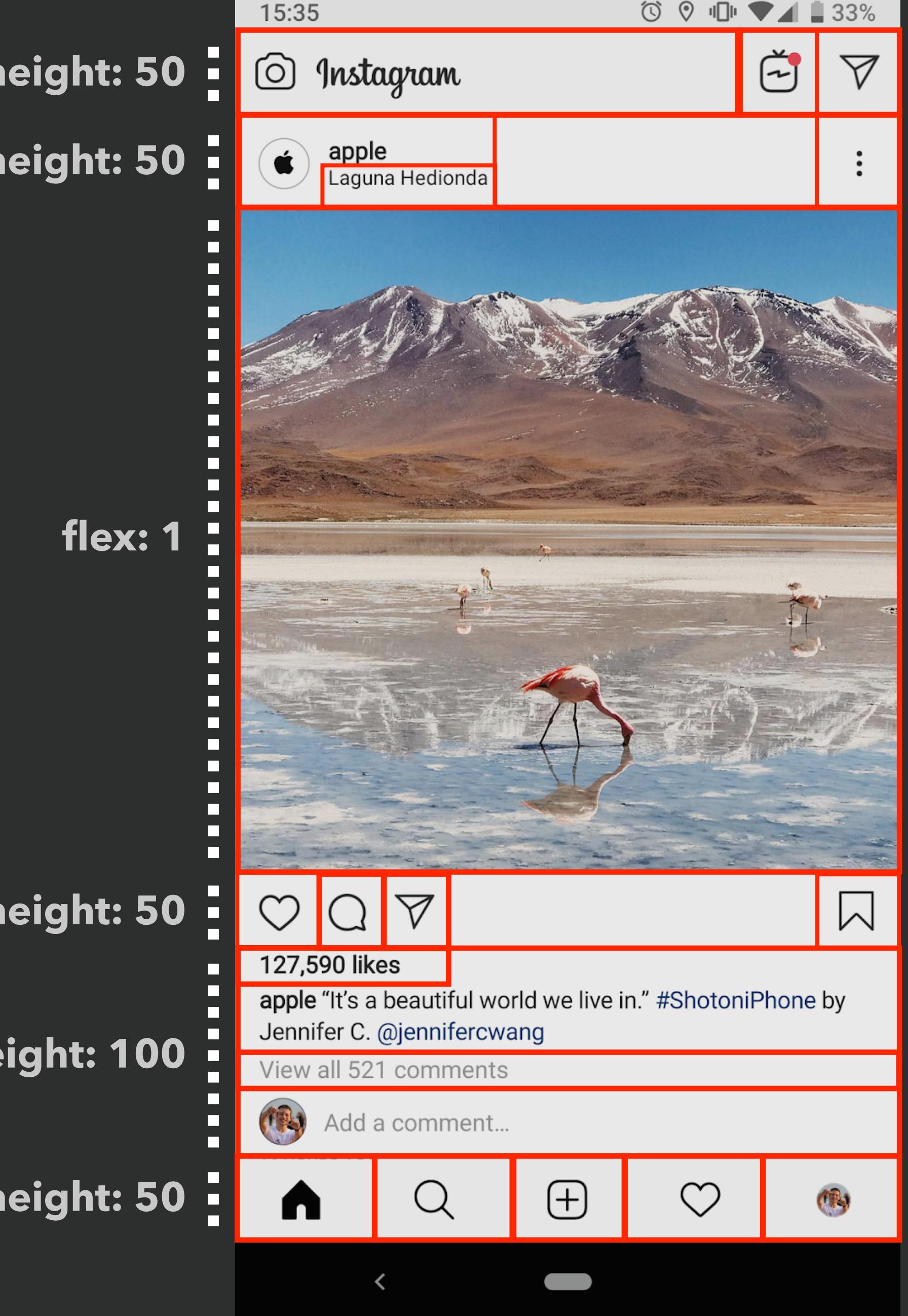
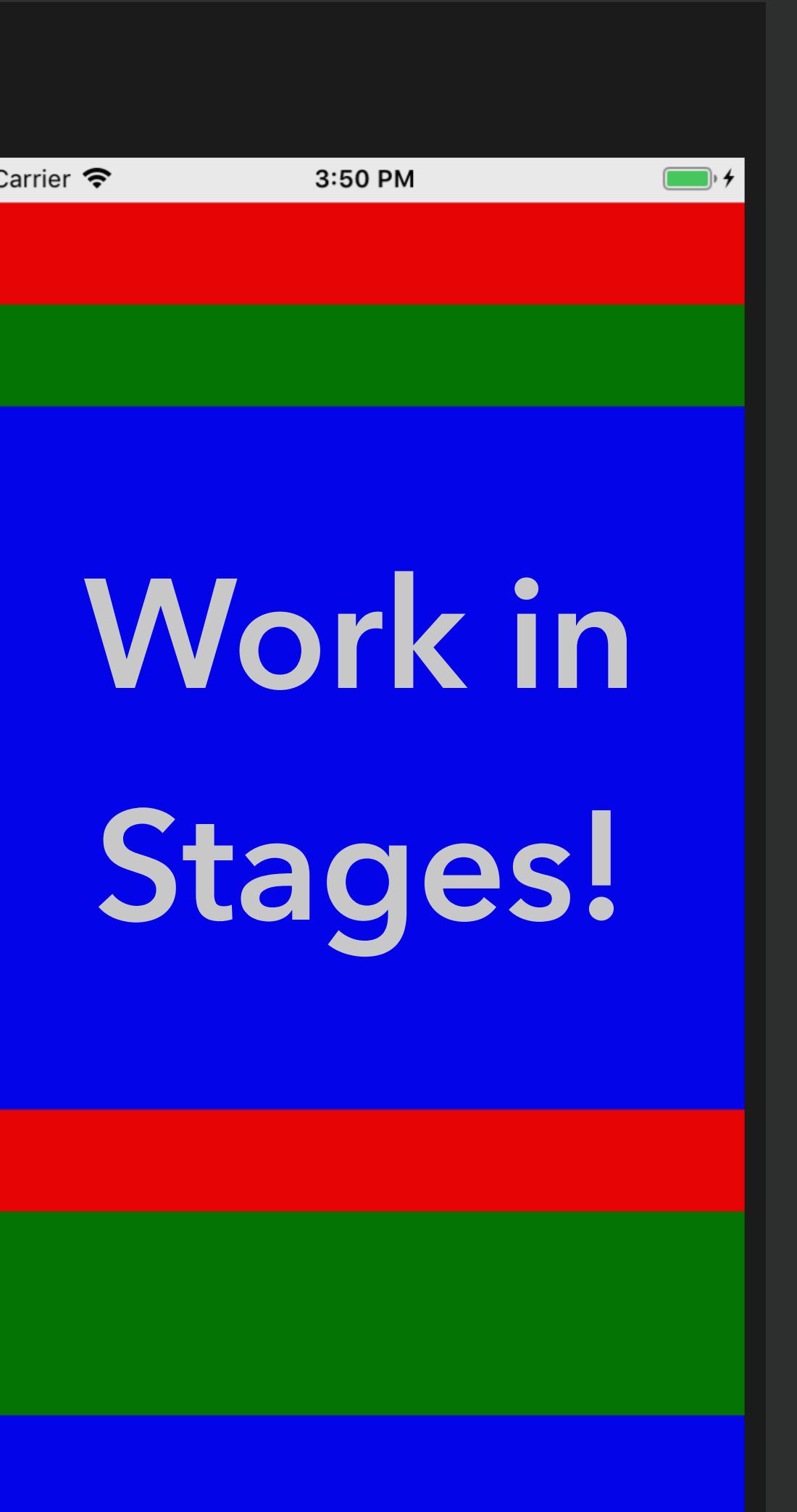


```
export default class HomeScreen extends React.Component {
  render() {
    return (
      <View style = {{flex: 1, flexDirection: 'row'}}>
        <View style = {{flex: 1, backgroundColor: 'red'}>
        </View>
        <View style = {{flex: 1, backgroundColor: 'green'}>
        </View>
        <View style = {{flex: 1, backgroundColor: 'blue'}>
        </View>
      </View>
    );
  }
}
```



Exploring Instagram's <View>'s

```
App.js
1 import React from "react";
2 import {
3   StyleSheet,
4   View,
5   Image,
6   Text,
7 } from "react-native";
8
9 export default class HomeScreen extends React.Component {
10   render() {
11     return (
12       <View style = {styles.container}>
13         <View style = {{height: 50, backgroundColor: 'red'}>
14         </View>
15         <View style = {{height: 50, backgroundColor: 'green'}>
16         </View>
17         <View style = {{flex: 1, backgroundColor: 'blue'}>
18         </View>
19         <View style = {{height: 50, backgroundColor: 'red'}>
20         </View>
21         <View style = {{height: 100, backgroundColor: 'green'}>
22         </View>
23         <View style = {{height: 50, backgroundColor: 'blue'}>
24         </View>
25       </View>
26     );
27   }
28 }
29
30 const styles = StyleSheet.create({
31   container: {
32     flex: 1,
33     marginTop: 22,
34   },
35 });
36 
```



Let's make a To-Do Application

The screenshot shows a mobile application interface for a To-Do application. At the top, the status bar displays "Carrier" and "12:12 AM". The main screen contains a list of three items:

- Take out garbage (checkmark)
- Make bed (checkmark)
- Submit essay (checkmark)

Below the list is a button labeled "Add Todo". At the bottom of the screen, a message says "You completed 0/3 tasks!".

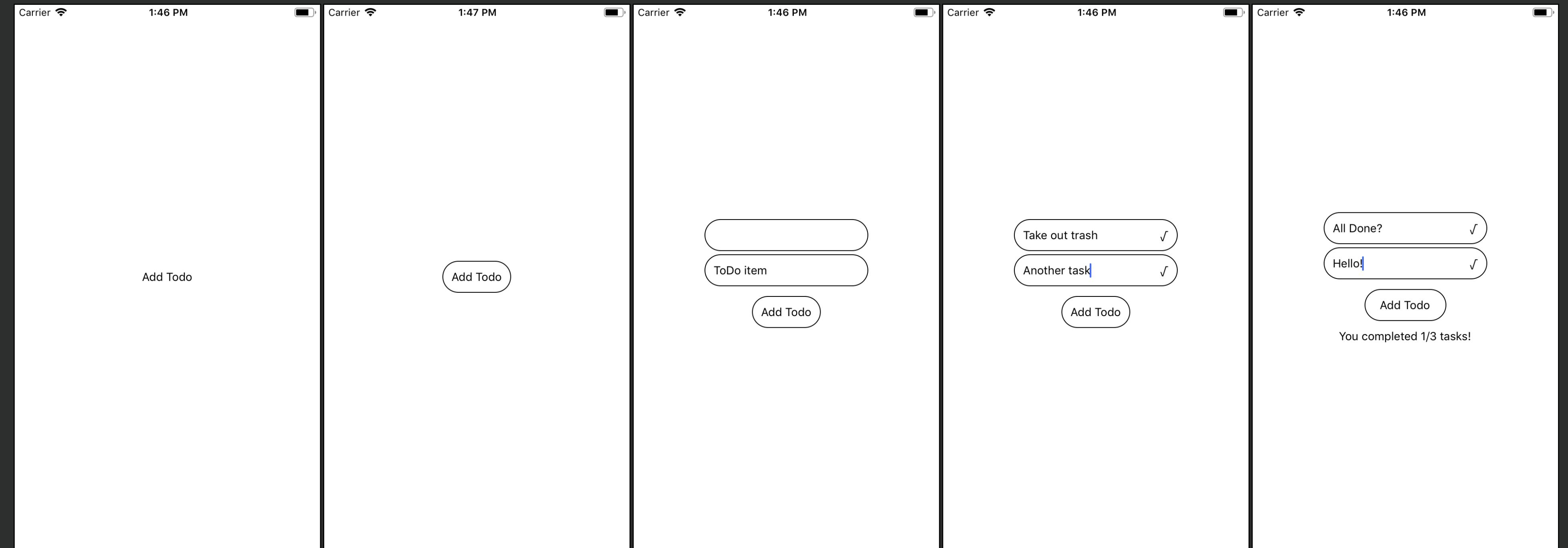
On the right side of the image, the corresponding `HomeScreen.js` code is shown:

```
1 import React from "react";
2 import {
3   StyleSheet,
4   View,
5   Text,
6   TouchableOpacity,
7   TextInput,
8 } from "react-native";
9
10 export default class HomeScreen extends React.Component {
11   constructor() {
12     super();
13     this.state = { };
14   }
15
16   isCompleted(index) { }
17
18   completeItem(index) { }
19
20   addTodo() { }
21
22   renderTodos = () => { }
23
24   render() {
25     return (
26       <View style = {styles.container}>
27         {this.renderTodos()}
28         <TouchableOpacity
29           style = {styles.button}
30           onPress={() => this.addTodo()}>
31           <Text>Add Todo</Text>
32         </TouchableOpacity>
33         <Text>=</Text>
34       </View>
35     );
36   }
37 }
38
39 const styles = StyleSheet.create({});
```

Annotations with arrows point from the UI elements to specific lines of code:

- A red arrow points from the first task ("Take out garbage") to line 16 ("isCompleted(index) { }").
- A green arrow points from the second task ("Make bed") to line 18 ("super();").
- A pink arrow points from the third task ("Submit essay") to line 20 ("completeItem(index) { }").
- A blue arrow points from the "Add Todo" button to line 24 ("renderTodos = () => { }").
- A purple arrow points from the "You completed 0/3 tasks!" message to line 33 ("<Text>=</Text>").

Stages of Development



01 Start new React Native App. For your main view, set Justify Content and Align Items to 'center'.
Test by rendering a Text element in the middle. Call it "Add ToDo"

02 Wrap the text in a TouchableOpacity. Give it some styling like a borderWidth.
Test by clicking on the item. You should see the Opacity change.

03 Create a state to keep track of how many ToDo items you have. Make a render method to create that many ToDo boxes. Use TextInput inside of the boxes to be able to write text.

04 Add new button next to each button to complete it. This should call a function to remove that item. You could keep track of the deleted items in another state variable, like an array of completed item indexes.

05 Create a Text component on the bottom that displays the amount of tasks left, and the amount of tasks completed.

Live Coding

Introduction to React Native

March 29th, 2019



ralfi
SALHON

Developer & Graphic Designer

Education

Tufts University
Computer Science (BS)

Website

tufts.io/ralfi

Github

github.com/ralfisalhon

Email

rifat.salhon@tufts.edu