

Topic	Structuring React Native Code	
Class Description	Students learn about the Stylesheet and how to better name and use CSS styles in React Native code. They learn more about exporting and importing custom react-native components. Students learn how to write structured code by creating different components as separate files.	
Class	C56	
Class time	45 mins	
Goal	<ul style="list-style-type: none"> • Use Stylesheet to better name and use CSS styles. • Export and import custom react-native components. • Write structured code for each custom react native component differently. 	
Resources Required	<ul style="list-style-type: none"> • Teacher Resources <ul style="list-style-type: none"> ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Android/iOS Smartphone with Expo App installed ○ Expo Snack Account Login • Student Resources <ul style="list-style-type: none"> ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Android/iOS Smartphone with Expo App installed ○ Expo Snack Account Login 	
Class structure	Warm Up Teacher-led Activity Student-led Activity Wrap up	5 mins 15 min 15 min 5 min
<p style="text-align: center;"><u>CONTEXT</u></p> <ul style="list-style-type: none"> • Introduce the need for more structure in the code. 		

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	Hi! Remember the project we were working on in the last class?	ESR: Yes! Wireless Quiz Buzzer project.
	Can you recall what we did in the last class?	ESR: We decomposed the project into smaller tasks. We created a round button and added sound to it when pressed.
	What would be our next task?	ESR: Creating a screen which would allow users to pick their team and then navigate to the buzzer button.
	<p>Alright. That means making an app which has more than one screen! We have been creating an app with one screen only.</p> <p>But before we do that, you might have started noticing how our code has started to become bigger and a little unorganized. If you remember in the previous classes when we were working on Games, we had made a point to always keep our code organized and structured.</p> <p>Keeping code organized and structured is the biggest strength of a developer. We will learn a few techniques of doing that in this class.</p>	Student listens.

	<p>So we have a lot of new things to cover in this class.</p> <p>Let's get started quickly.</p>	-
Teacher Initiates Screen Share		
<p style="text-align: center;"><u>CHALLENGE</u></p> <ul style="list-style-type: none"> • Create style object and use style names for styling the components in style prop of the component. • Create screens and components as separate files. 		
Step 2: Teacher-led Activity (15 min)	<p>Teacher opens <u>Teacher Activity 1.</u></p> <p>We ended up somewhere here in the last class.</p> <p>Can you quickly recall what the code is doing?</p>	<p>ESR:</p> <p>The student quickly recalls the code from the previous class.</p>
	<p>The first thing which you might have observed is that we have a lot of styling code for button and text wherever we are rendering these components. This does not make for a very readable code.</p> <p>What would be a good idea to do this better?</p>	<p>ESR:</p> <p>Use a sheet to style it like we do in HTML and CSS.</p>

In react native, a good practice is to create a separate styles object using 'StyleSheet.create()' and use the declared styles inside the style prop of the component when rendering.

I will show you how to do it:

1. First, we import the 'StyleSheet' component defined in react native.
2. Second, we use 'StyleSheet.create()' to create the different styles. The function expects a JSON object with different styles defined on them.

Note: We use "const" to create a variable which we do not want to change while the program is running. Typically

ESR:

Student sees the code and asks questions for clarification.

we do not want the style object to change during the program.

3. We can call the styles.<name of the style> inside the style prop of a component to give a particular style to the react native component.

Teacher writes code and shows how to add styling to components using the styles object created using 'StyleSheet.create()'

1.

```

1  import * as React from 'react';
2  import { Text, View, Button, TouchableOpacity, StyleSheet } from 'react-native';
3  import { Audio } from 'expo-av';
4
5  class SoundButton extends React.Component {
6    playSound = async () => {
7      await Audio.Sound.createAsync(
8        { uri: 'http://soundbible.com/mp3/Buzzer-SoundBible.com-188422102.mp3' },
9        { shouldPlay: true }
10     );
11   }
12
13   render() {
14     return (
15       <TouchableOpacity
16         style={{
17           marginLeft: 100,
18           borderWidth: 1,
19           borderColor: 'rgba(0,0,0,0.2)',
20           alignItems: 'center',
21           justifyContent: 'center',
22           width: 200,
23           height: 200,
24           backgroundColor: 'red',
25           borderRadius: 100,
26         }}
27         onPress={this.playSound}>
28         <Text
29           style={{

```

Prettier {} Editor ⚙️

2.

```

32 export default class App extends React.Component {
33   render() {
34     return (
35       <View>
36         <SoundButton />
37       </View>
38     );
39   }
40 }
41
42 const styles = StyleSheet.create({
43   button: {
44     marginTop: 200,
45     marginLeft: 100,
46     borderWidth: 1,
47     borderColor: 'rgba(0,0,0,0.2)',
48     alignItems: 'center',
49     justifyContent: 'center',
50     width: 200,
51     height: 200,
52     backgroundColor: 'red',
53     borderRadius: 100,
54   },
55   buttonText: {
56     fontWeight: 'bold',
57     fontSize: 20,
58   }
59 });

```

ios

Press Me

3.

```

4
5 class SoundButton extends React.Component {
6   playSound = async () => {
7     await Audio.Sound.createAsync(
8       { uri: 'http://soundbible.com/mp3/Buzzer-SoundBible.com-188422102.mp3' },
9       { shouldPlay: true }
10    );
11  }
12
13  render() {
14    return (
15      <TouchableOpacity
16        style={styles.button}
17        onPress={this.playSound}>
18        <Text
19          style={styles.buttonText}>
20          Press Me
21        </Text>
22      </TouchableOpacity>
23    );
24  }
25 }
26
27 export default class App extends React.Component {
28   render() {
29     return (
30       <View>
31         <SoundButton />
32       </View>

```

ios Android Web

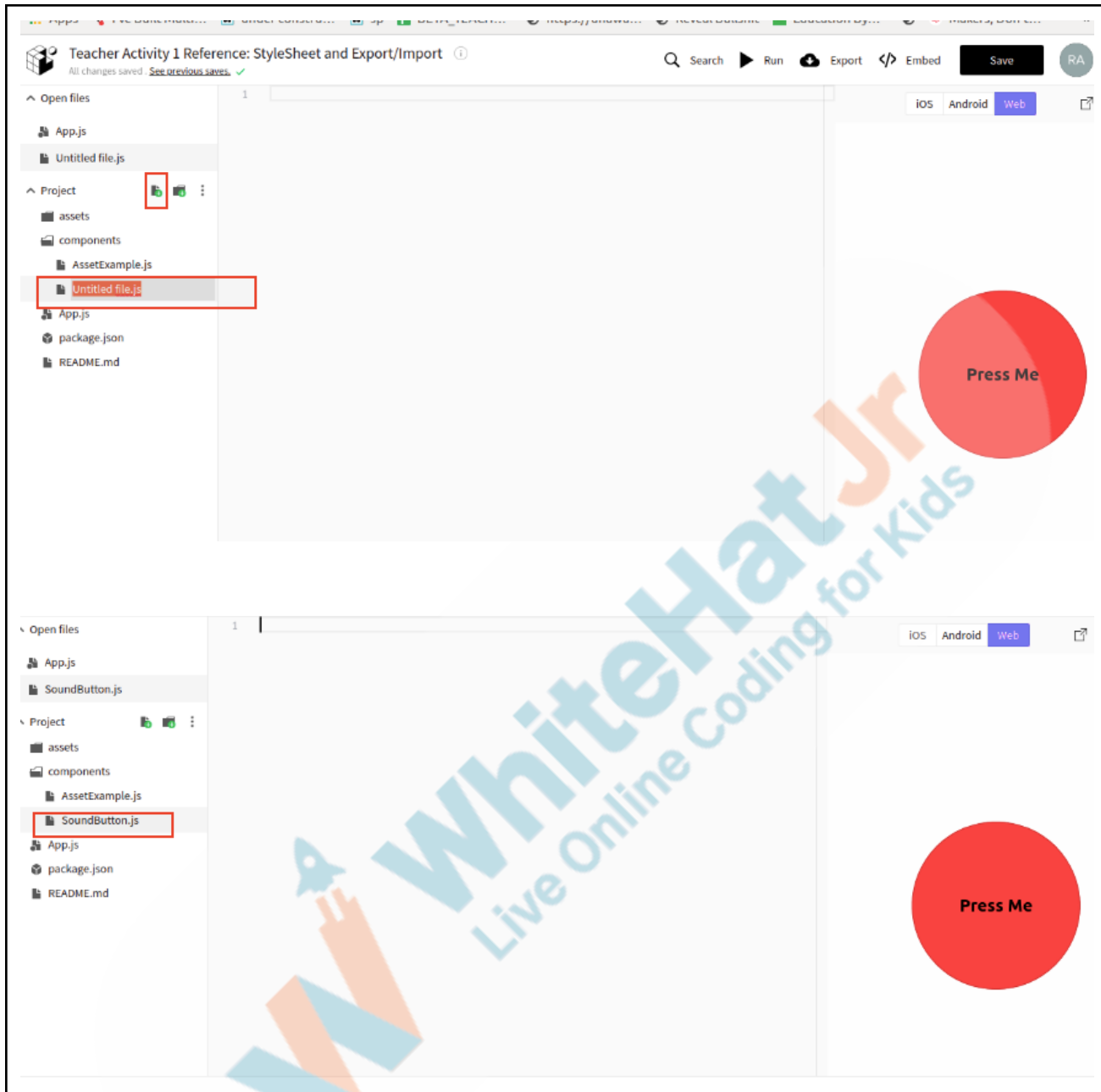
Press Me


Prettier {} Editor Expo v36.0.0 Devices 1 Preview

Does the code look simpler and more readable now?

ESR:
Yes!

	<p>Ok. We can further simplify our code.</p> <p>Right now, we are creating two component classes in a single file App.js</p> <p>There is no need to do that!!</p> <p>We can create different components in different files to keep things more organized and then import all the components in App.js to use them.</p> <p>I will show you how:</p> <ol style="list-style-type: none"> 1. Let's create a new file inside the components folder and call it 'SoundButton.js' (since it defines the SoundButton Component). 2. Now, let's copy all the code for the 'SoundButton' component into this file. 3. We will need to import components from the libraries used for this component including - react, expo-av , StyleSheet etc. 4. Lastly, we need to add an export statement - export default SoundButton. This instruction allows the SoundButton component to be imported by default when we write 'import' statement in another file. <p>We will see how we do that.</p>	<p>Student observes and asks questions.</p>
--	--	---





Teacher Activity 1 Reference: StyleSheet and Export/Import

All changes saved 2 minutes ago. [See previous saves](#) ✓

Open files

App.js
SoundButton.js

Project

assets
components
AssetExample.js
SoundButton.js
App.js
package.json
README.md

```

1  import * as React from 'react';
2  import { Text, View, Button, TouchableOpacity, StyleSheet } from 'react-native';
3  import { Audio } from 'expo-av';
4
5  class SoundButton extends React.Component {
6    playSound = async () => {
7      await Audio.Sound.createAsync(
8        { uri: 'http://soundbible.com/mp3/Buzzer-SoundBible.com-188422102.mp3' },
9        { shouldPlay: true }
10     );
11   }
12
13   render() {
14     return (
15       <TouchableOpacity
16         style={styles.button}
17         onPress={this.playSound}>
18         <Text
19           style={styles.buttonText}>
20           Press Me
21         </Text>
22       </TouchableOpacity>
23     );
24   }
25 }
26
27 export default class App extends React.Component {
28   render() {
29     return (

```

✓ No errors

Prettier {} Editor

Teacher Activity 1 Reference: StyleSheet and Export/Import ⓘ

All changes saved. [See previous saves.](#) ✓

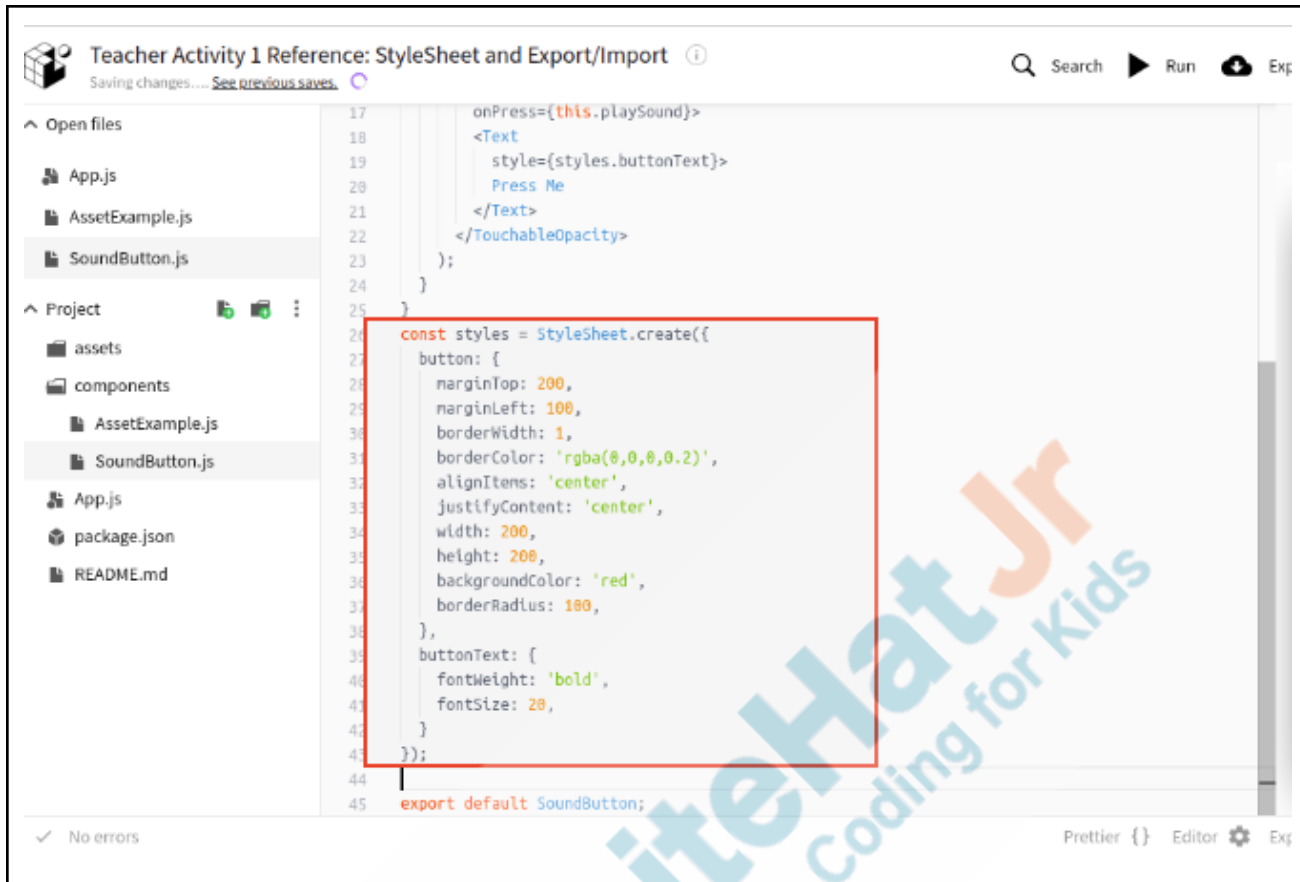
Open files

- App.js
- AssetExample.js
- SoundButton.js

Project

- assets
- components
 - AssetExample.js
 - SoundButton.js
- App.js
- package.json
- README.md

```
1 import * as React from 'react';
2 import { Text, View, TouchableOpacity, StyleSheet } from 'react-native';
3 import { Audio } from 'expo-av';
4
5 class SoundButton extends React.Component {
6   playSound = async () => {
7     await Audio.Sound.createAsync(
8       { uri: 'http://soundbible.com/mp3/Buzzer-SoundBible.com-188422102.mp3' },
9       { shouldPlay: true }
10    );
11  }
12
13  render() {
14    return (
15      <TouchableOpacity
16        style={styles.button}
17        onPress={this.playSound}>
18        <Text
19          style={styles.buttonText}>
20          Press Me
21        </Text>
22      </TouchableOpacity>
23    );
24  }
25 }
26
27 export default SoundButton;
```



Great so far.

Right now, the component 'SoundButton' is declared as a separate component in a different file. We can use it in our App.js file (or any other file).

I will show you how.



Teacher shows how to import SoundButton from the components folder.

Note: "." represents the current directory.

Student observes and asks questions.

		
	<p>Great thing about creating the components separately is that you can use the component anywhere inside another component.</p> <p>This way a complex component can be built up using several smaller and independently created components.</p>	<p>Student listens.</p>

	<p>Ok. Now you know how to create an independent component and import it in a different file.</p> <p>You also know how to write more structured styling using StyleSheet.</p> <p>Here is a challenge for you.</p> <p>Can you create a component called AppHeader which gives a header to your app (with app name)?</p>	ESR: Yes.
	<p>In the next class, when we learn about creating an app with more than one screen, we will use the same AppHeader component for both the screens.</p> <p>This way we can write a component only once and use it multiple times (Remember DRY - do not repeat yourself!)</p> <p>Let's get started.</p>	
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	
<ul style="list-style-type: none"> • Ask Student to press ESC key to come back to panel • Guide Student to start Screen Share • Teacher gets into Fullscreen 		
<p style="text-align: center;"><u>ACTIVITY</u></p> <ul style="list-style-type: none"> • Create a custom react native component. • Create default export for the component file. • Import and use the component in App.js file 		

Step 3: Student-Led Activity (15 min)	Guide the student to open <u>Student Activity 1.</u>	The student opens <u>Student Activity 1</u> and renames the project.
	<p>Write a component 'AppHeader.js'.</p> <p>Import necessary components at the top.</p> <p>Render the component inside the render() function.</p> <p>Create appropriate styling for it using StyleSheet</p> <p>Export the class as default export.</p>	The student creates the 'AppHeader' component in 'AppHeader.js' and exports it as default.
		
	Now try using the 'AppHeader' component you created inside 'App.js' file.	The student imports the 'AppHeader' component and uses it while rendering the App.

Great work!




Now you know how to properly structure react native components and use them inside your App.

This would keep your app organized and help you scale your application easily.

Teacher Guides Student to Stop Screen Share

FEEDBACK

- Encourage the student to create more components for the screen.
- Encourage the student to make reflection notes in the markdown format.
- Complement the student for her/his effort in the class.

Step 4: Wrap-Up (5 min)	<p>Before we close the class, can you quickly recall what we learned in this class?</p>	<p>ESR:</p> <ul style="list-style-type: none"> - We learned to create style for the components using StyleSheet. - We learned to create independent components and how to export/import them in our files.
	<p>This style of creating independent components which can live on their own and can be re-used anywhere in our code is central to React Philosophy.</p> <p>Can you guess why that would be useful to a programmer?</p> <p>It is like constructing anything in real-life. We create complex structures using simpler parts and then use these complex structures to build more complex stuff and so on.</p>	<p>ESR:</p> <p>Yes! It helps us make new complex components using the existing/custom-defined simple components.</p> <p>ESR:</p> <p>Yes!</p>
	<p>You get a “hats off”.</p> <p>In the next class, we will learn how to convert our single screen app to a two-screen app and how to navigate and pass information from one screen to another.</p> <p>Sounds interesting?</p> <p>See you in the next class then.</p>	<p>Make sure you have given at least 2 Hats Off during the class for:</p> <div data-bbox="1019 1346 1312 1444">  </div> <div data-bbox="1019 1497 1312 1591">  </div> <div data-bbox="1019 1644 1312 1738">  </div>

Project Pointers and Cues (5 min)	<p>*This Project will take only 30 mins to complete. Motivate students to try and finish it immediately after the class.</p> <p>DJ AUDIO MIXER APP - STAGE 2</p> <p>Goal of the Project:</p> <p>In Class 56, you learned about Stylesheet and how to write structured code by creating different components as separate files.</p> <p>In this project, you will apply what you have learned in the class to achieve the following goals.</p> <p>Story:</p> <p>Your cousin has made a DJ mixer App. But the buttons are looking quite boring. Please help your cousin make the App exciting by adding colours and styles.</p> <p>I am very excited to see your project solution and I know you both will do really well.</p> <p>Bye Bye!</p>	
<div> <div>Teacher Clicks</div> <div>✕ End Class</div> </div>		
Additional Activities	<p>Encourage the student to write reflection notes in their reflection journal using markdown.</p> <p>Use these as guiding questions:</p>	<p>The student uses the markdown editor to write her/his reflection in a reflection journal.</p>

	<ul style="list-style-type: none"> • What happened today? <ul style="list-style-type: none"> - Describe what happened - Code I wrote • How did I feel after the class? • What have I learned about programming and developing games? • What aspects of the class helped me? What did I find difficult? 	
--	---	--

Activity	Activity Name	Links
Teacher Activity 1	Teacher Activity 1: Stylesheet	https://snack.expo.io/@rajeevtfi/7cda70
Teacher Reference 1	Teacher Activity 1 Reference	https://snack.expo.io/@rajeevtfi/teacher-activity-1-reference:-stylesheet-and-export-import
Student Activity 1	Student Activity : App Header	https://snack.expo.io/@rajeevtfi/student-activity-1-app-header
Project Solution	DJ Audio Mixer App- Stage-2	https://snack.expo.io/@snerrus/dj-app-solution:-pro-c56