

Topic	Sound	
Class Description	Students decompose the Wireless Quiz Buzzer project into smaller problems. They learn how to play sound in the React Native environment. Students design a rounded buzzer button which when clicked plays the buzzer sound.	
Class	C55	
Class time	45 mins	
Goal	<ul style="list-style-type: none"> Decompose the wireless quiz buzzer project into smaller tasks. Play sound when a button is clicked in the react native environment. Design a rounded buzzer button which when clicked plays the buzzer sound. 	
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 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 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 scope of the project. 		

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	Ahh! Here we are again! Remember what we will be working on in this class?	ESR: Wireless Quiz Buzzer App
	Awesome! But before we get started, let's quickly recall what we covered in the last class? Help the student recall points when he/she is stuck.	ESR: - We learned about props of a component. - We learned to design a prop for a custom component. - We learned to perform an action (through a function) when a button is pressed. We displayed an alert box when a button was pressed. - We worked on a problem statement about a quiz buzzer app and we designed the wireframe for the app.
	Amazing. You seem to remember a great deal. Now let's quickly get started on working towards our quiz buzzer app.	
Teacher Initiates Screen Share		
<p style="text-align: center;"><u>CHALLENGE</u></p> <ul style="list-style-type: none"> • Breakdown the project into smaller tasks/problems. • Play sound 'on click' of a button. 		
Step 2: Teacher-led Activity (15 min)	Before we start working on any project, what is the first thing that we do?	ESR: varied

	<p>The first thing that we do when working on any complex project is to break it down into small/simpler tasks. Each task should be very specific and should deal with only one small part of the project.</p> <p>By combining each of these simpler tasks together, you should get the complex project done. This is called decomposition.</p> <p>Breaking down a complex programming project into smaller problems is a very important skill for any developer/coder. It makes thinking and working on any project of any complexity easier for a programmer.</p> <p>Can you try to tell what is decomposition of a project in your own words?</p> <p>(Help the student develop clarity about what decomposition is through his/her response)</p>	<p>ESR: Decomposition is breaking down the complex task into smaller/simpler problems. This makes it easier to think and work on each problem.</p>
	<p>Perfect!</p> <p>And the skill of decomposition - like any other skill - comes with practicing breaking down bigger/complex problems.</p> <p>Why don't you try breaking down the wireless quiz buzzer project into</p>	<p>ESR: (allow the student time to think.) List of tasks the student can come up with:</p> <ul style="list-style-type: none"> • Task1: Create a button which when pressed plays the sound of a buzzer.

	<p>smaller simpler problems which we can solve?</p>	<ul style="list-style-type: none"> • Task2: Create a home screen which allows the user to pick their team. • Task 3: Navigate the user from the home screen to the buzzer screen. • Task 4: Listen to which team is pressing the button first and store it in a database. • Task 5: Display the teams in an order in which they pressed the buttons.
	<p>Perfect! This is a good start. So, let's attack Task 1.</p> <p>I am going to show you how to play sound in a react native environment.</p> <p>You can then go ahead and create a rounded quiz buzzer button which plays a buzzer sound when pressed.</p> <p>Cool?</p>	<p>ESR: Yes!</p>
	<p>Teacher opens <u>Teacher Activity 1.</u></p> <p>Can you look at the code here and tell me what this is doing?</p>	<p>ESR:</p> <p>We are creating a new component called 'SoundButton'. It has a red button inside it.</p> <p>We are rendering this SoundButton inside 'View' component inside the app.</p>

<pre>1 import * as React from 'react'; 2 import { Text, View, Button } from 'react-native'; 3 4 5 class SoundButton extends React.Component { 6 render() { 7 return (8 <Button title="Sound Button" color="red"/> 9); 10 } 11 } 12 13 export default class App extends React.Component { 14 render() { 15 return (16 <View style={{marginTop:200}}> 17 <SoundButton /> 18 </View> 19); 20 } 21 } 22 23</pre>	
<p>Great recollection!</p> <p>Now, what will I do if I want the button to do something when it is pressed?</p> <p>Help the student recall and write an empty playSound function and call it inside the onPress attribute for the button.</p>	<p>ESR:</p> <p>We will write a function which can be called using the 'onPress' prop of the button.</p>

```

1  import * as React from 'react';
2  import { Text, View, Button } from 'react-native';
3
4
5  class SoundButton extends React.Component {
6    playSound = ()=> {
7
8    }
9
10   render() {
11     return (
12       <Button title="Sound Button" color="red" onPress={this.playSound} />
13     );
14   }
15 }
16
17 export default class App extends React.Component {
18   render() {
19     return (
20       <View style={{marginTop:200}}>
21         <SoundButton />
22       </View>
23     );
24   }
25 }
26
27

```



	<p>We want the button to play some sound, so we have created an empty playSound function.</p> <p>We have created an arrow function, so that "this" inside the function refers to the SoundButton component.</p>	<p>Student observes and asks questions.</p>
	<p>To play sound in React Native environment, we will use the Audio Class library defined in expo-av library.</p> <p>Teacher shows how to import the library.</p> <p>Press (Ctrl + Enter) to add the package to the project.</p> <p>Open 'package.json' file to show the expo-av library added in the project.</p>	<p>Student observes and asks questions.</p>

Teacher Activity: Sound

All changes saved less than 10 seconds ago. [See previous saves.](#)

Search

Run

Export

Embed

Save

Open files

App.js

package.json

Project

assets

components

App.js

package.json

README.md

1

import * as React from 'react';

2

import { Text, View, Button } from 'react-native';

3

import { Audio } from 'expo-av';

4

class SoundButton extends React.Component {

5

playSound = async () => {

6

7

8

9

10

render() {

11

return (

12

<Button title='Sound Button' color='red' onPress={this.playSound} />

13

);

14

15

16

export default class App extends React.Component {

17

render() {

18

return (

19

<View style={{marginTop:200}}>

20

<SoundButton />

21

</View>

22

);

23

24

25

IOS

Android

Web

SOUND BUTTON

Add expo-av to package.json?

ADD CTRL+P

CANCEL

All changes saved 1 minute ago. [See previous saves.](#)

Open files

App.js

package.json

Project

assets

components

App.js

package.json

README.md

1

{

2

"dependencies": {

3

"react-native-paper": "3.1.1",

4

"expo-av": "7.0.1"

5

}

6

Teacher navigates back to App.js file.

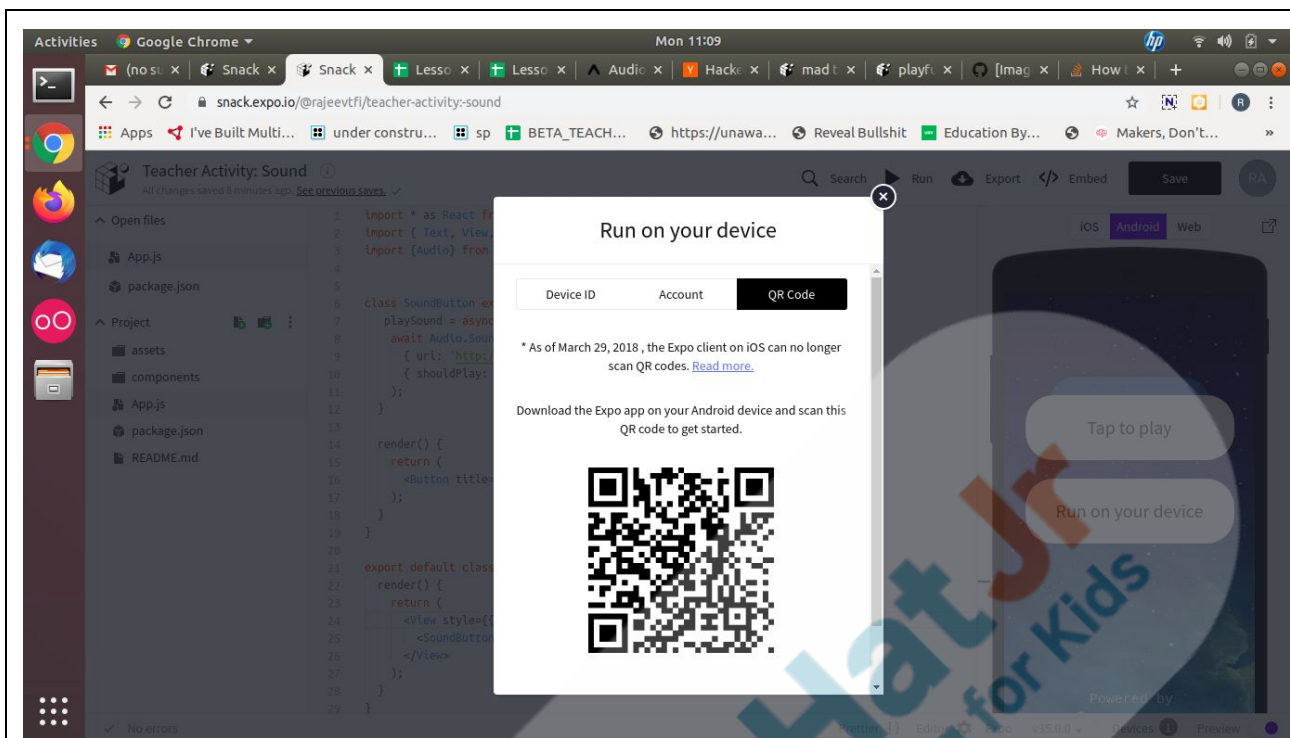
We will use
'Audio.Sound.createAsync()' to play
the sound.

This function will help us load the
sound and control how we want to

Student observes and asks
questions.

	<p>play it. We will be exploring this more in our next project. For now, know that it accepts two objects - as json. One object defines the source of the file and the other object tells the status of the file.</p> <p>We will be passing the url of the sound file we want to play in the first object and in the second object we will pass the shouldPlay status as true.</p> <p>Also, remember Javascript is synchronous by default. But here we want to wait for the sound object to load and then play the sound.</p> <p>We will attach 'await' before our instruction. To let the computer know that playSound is an asynchronous function now, we will add 'async' while defining it.</p>	
--	---	--

	<p>Now, let's open the project on our device and see the output.</p> <p>Note: The sound might not play on the expo web because of the cross origin policy of the browser. Please test the app on the phone using an expo client.</p> <p>Teacher scans the QR code on her android/iOS device and shows the output to the student.</p>	<p>The student scans the QR code on his/her android/iOS device and sees the output.</p> <p>He/She presses the button to hear the buzzer sound.</p>
--	--	--



Awesome!

This seems like a good start to our Wireless Quiz Buzzer Project.

Now here is a challenge for you.
Create a round large button to represent a buzzer button and then add sound to it when pressed.

Ready to do that?

ESR:
yes!!

Teacher Stops Screen Share

Now it's your turn. Please share your screen with me.

- Ask Student to press ESC key to come back to panel
- Guide Student to start Screen Share
- Teacher gets into Fullscreen

ACTIVITY

- Design a rounded buzzer button which when clicked plays a buzzer sound.

Step 3: Student-Led Activity (15 min)	Guide the student to open <u>Student Activity 1</u>	Student opens <u>Student Activity 1.</u>
	<p>You see, buttons were built mostly for the web and do not suit the app interface.</p> <p>We use another component for the app, which we call 'TouchableOpacity'.</p> <p>It is a component which responds when touched.</p> <p>Unlike Button, TouchableOpacity can enclose anything inside them - Text, View, etc.</p> <p>Guide the student to replace Button in the code with TouchableOpacity with a text inside it.</p>	<p>The student imports the 'TouchableOpacity' Component.</p> <p>He/She replaces Button with TouchableOpacity component with a Text component inside it.</p>

```

1  import * as React from 'react';
2  import { Text, View, Button, TouchableOpacity } from 'react-native';
3
4
5  class SoundButton extends React.Component {
6    playSound = async () => {
7    }
8
9    render() {
10     return (
11       <TouchableOpacity>
12         <Text>Press Me</Text>
13       </TouchableOpacity>
14     );
15   }
16 }
17
18 export default class App extends React.Component {
19   render() {
20     return (
21       <View style={{marginTop:200}}>
22         <SoundButton />
23       </View>
24     );
25   }
26 }
27
28
29

```

	<p>What do you see?</p> <p>Yes! That is because 'TouchableOpacity' is transparent by default. But you can add color to it using the styles prop of the component.</p>	<p>ESR:</p> <p>Only the Text "Press Me".</p>
	<p>Guide the student to add color to the 'TouchableOpacity'.</p>	<p>The student adds the 'backgroundColor' property to 'TouchableOpacity' under the styles property.</p>

```

1  import * as React from 'react';
2  import { Text, View, Button, TouchableOpacity } from 'react-native';
3
4
5  class SoundButton extends React.Component {
6    playSound = async () => {
7    }
8
9    render() {
10     return (
11       <TouchableOpacity style={{
12         backgroundColor: 'red'
13       }}>
14         <Text>Press Me</Text>
15       </TouchableOpacity>
16     );
17   }
18 }
19
20 export default class App extends React.Component {
21   render() {
22     return (
23       <View style={{marginTop:200}}>
24         <SoundButton />
25       </View>
26     );
27   }
28 }
29


```

IOS Android Web

Press Me

Now, go ahead and add more styles to make this a rounded button.

The student adds more styles to give the 'TouchableOpacity' the appearance of a rounded button.

	<p>You can style the text inside 'TouchableOpacity' too!</p> <p>Guide the student to add style to Text.</p>	<p>The student adds styling to the Text inside 'TouchableOpacity'.</p>
--	---	--

```

1  import * as React from 'react';
2  import { Text, View, Button, TouchableOpacity } from 'react-native';
3
4
5  class SoundButton extends React.Component {
6    playSound = async () => {
7    }
8
9    render() {
10     return (
11       <TouchableOpacity style={{
12         marginLeft: 100,
13         borderWidth: 1,
14         borderColor: 'rgba(0,0,0,0.2)',
15         alignItems: 'center',
16         justifyContent: 'center',
17         width: 200,
18         height: 200,
19         backgroundColor: 'red',
20         borderRadius: 100,
21       }}>
22         <Text style={{
23           fontWeight: 'bold',
24           fontSize: 20
25         }}>
26           >Press Me</Text>
27       </TouchableOpacity>
28     )
29   }

```

	Now, do you see the advantage of 'TouchableOpacity' over Buttons?	ESR: Yes!
	<p>Now add the sound to your 'TouchableOpacity' using its 'onPress' property.</p> <p>This is similar to Button.</p> <p>Guide the student wherever they get stuck.</p>	<p>The student imports Audio from the expo-av library.</p> <p>He/She writes the async playSound function.</p> <p>He/She calls the function with 'onPress' prop of the 'TouchableOpacity'.</p>


```

1 import * as React from 'react';
2 import { Text, View, Button, TouchableOpacity } 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 Exj

Now run your code on your device and check the output

The student uses the expo client to scan the QR code and run the project on his/her device.

Teacher Guides Student to Stop Screen Share

FEEDBACK




- Encourage the student to play around with another component called **TouchableOpacity** which works exactly like a button.
- 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)

Awesome!

We have solved the first task out of the different tasks we listed at the start of the day. And it was not that difficult - was it?

ESR:
No!

	<p>Great. Can you quickly recall what we learned in Today's class?</p>	<p>ESR:</p> <ul style="list-style-type: none"> - We learned how to import external libraries. - We learned how to add sounds in the React Native environment. - We also learned how to use a component called 'TouchableOpacity' and style it.
	<p>Awesome!</p> <p>In the next class we will attack the other problems.</p> <p>We will learn how to create a two screen app rather than a single screen app.</p> <p>Users will be able to choose their team and then navigate to the Buzzer button.</p> <p>Are you excited about learning new stuff in the next class?</p>	<p>ESR:</p> <p>Yes!</p>
	<p>You get a "hats off".</p> <p>Till next class then. See you. Bye!</p>	<p>Make sure you have given at least 2 Hats Off during the class for:</p> <div>Creatively Solved Activities  +10</div> <div>Great Question  +10</div> <div>Strong Concentration  +10</div>

<p>Project Pointers and Cues (5 min)</p>	<p>DJ AUDIO MIXER APP - STAGE 2</p> <p>Goal of the Project:</p> <p>Today you learned how to play sound using the expo-av library and designed a buzzer button.</p> <p>In this project, you will have to practice and apply what you have learned in the class and add some sounds to our DJ audio Mixer App.</p> <p><i>*This is a continuation of Project 54. So make sure to complete that project before you attempt this one.*</i></p> <p>Story:</p> <p>Your cousin Nikhil loves music and wants to become a Disc Jockey (DJ). You have started creating a DJ mixer app for him. So far you have designed the buttons for the app. Now you need to add sounds to all the buttons, so that a sound comes whenever any button is pressed.</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	Encourage the student to look at the documentation for 'TouchableOpacity' Component and Audio library from expo-av package by googling online.	The student googles online to look at the documentation for audio library from expo-av and 'TouchableOpacity' component.
	<p>Encourage the student to write reflection notes in their reflection journal using markdown.</p> <p>Use these as guiding questions:</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? 	The student uses the markdown editor to write her/his reflection in a reflection journal.

Activity	Activity Name	Links
Teacher Activity 1	Teacher Activity : Sound	https://snack.expo.io/@rajeevtfi/teacher-activity:-sound
Teacher Activity 2	Reference 1	https://snack.expo.io/@rajeevtfi/90df1e
Teacher Activity 3	Reference 2	https://snack.expo.io/@rajeevtfi/teacher-reference-2
Student Activity 1	Student Activity: Sound	https://snack.expo.io/@rajeevtfi/30c803

