Working with Audio Files:XyloPhone App

Prashanth B S[1]

[1]Department of Information Science & Engineering, Nitte Meenakshi Institute of Technology,
Yelahanka - 560064, Bengaluru

1 XyloPhone App

The Xylophone is a simple app that demonstrates how to play the audio files that is present in the flutter asset folder. To play audio files, we need to make use of third party library. A list of proven good third party packages can be found in the link https://pub.dev/. In order to play the audio files we use a third party library called as "audioplayers" which is available in https://pub.dev/packages/audioplayers. To use this package we need to add the dependencies in pubspec.yaml file as shown below, and click on pub get to sync the files into user's flutter project.

```
dependencies:
  audioplayers: ^0.20.1
```

A sample code snippet of its usage is shown below. The code snippet demonstrates how we can play the audio using audioplayer package. The steps are as follows,

1. Create an object of type AudioPlayer using the constructor AudioPlayer()
2. Define the String asset which holds the file to be played under the asset folder. In this case, under the asset folder we have "audio_file.wav" file
3. Convert the audio asset into bytes using rootBundle.load and typecast it to bytes
4. Convert the bytes into UnsignedInt8, which is a fixed-length list of 8-bit unsigned integers. This is done by reading the bytes generated into Uint8List datatype
5. Use the player.playBytes to the converted data [1].

Note: You add the async* keyword to make a function that returns a bunch of future values one at a time[2].The async and await for the null safety, i.e, adding async to the function makes the function to generate future values. and await is to interrupt the process flow until the async method completes.

```
void PlayAudio() async{
    AudioPlayer player = AudioPlayer(); // Step 1
    String audioAsset = 'assets/audio_file.wav' ; // Step 2
    ByteData bytes = await rootBundle.load(audioAsset); // Step 3
    Uint8List audiobytes = bytes.buffer.asUint8List(bytes.offsetInBytes,
        bytes.lengthInBytes); // Step 4
    int result = await player.playBytes(audiobytes); // Step 5
    if(result == 1){ //play success
      print("audio is playing.");
    }else{
```

[1]https://pub.dev/packages/audioplayers
[2]https://www.codegrepper.com/code-examples/dart/what+is+async+and+await+in+flutter

```
    print("Error while playing audio.");
   }
 }
```

2 Exercise

Build a Flutter App called as Xylophone which imitates the behaviour of a physical Xylophone, where in it has bunch of buttons, with different colors stretched to the screen. Upon click of each button, different audio notes need to be played.
The following are the steps to be followed,

1. Open the browser and download the stub project from this URL https://github.com/londonappbrewery/xylophone-flutter.

2. Copy the asset folder which consists of 7 different audio files into the user's project folder. The project folder upon copying the asset folder into our project directory is as shown in figure 1,

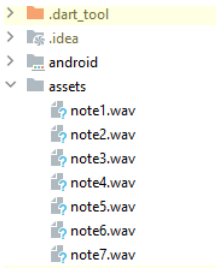

Figure 1: Project Structure

3. Sync the assets folder and add the audioplayers dependencies into the pubspec.yaml file as shown below, and click on pub get and return to main.dart and click on get-dependencies to sync the project.

```
dependencies:
  flutter:
    sdk: flutter

  cupertino_icons: ^1.0.2
  audioplayers: ^0.20.1 # adding the 3rd party audioplayers package
dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_lints: ^1.0.0

flutter:
  uses-material-design: true
  assets:   # syncing asset folder
    - assets/
```

4. Build the Widget Tree as shown in figure 2.

5. The starter code would be a stateful widget and it is as shown below,

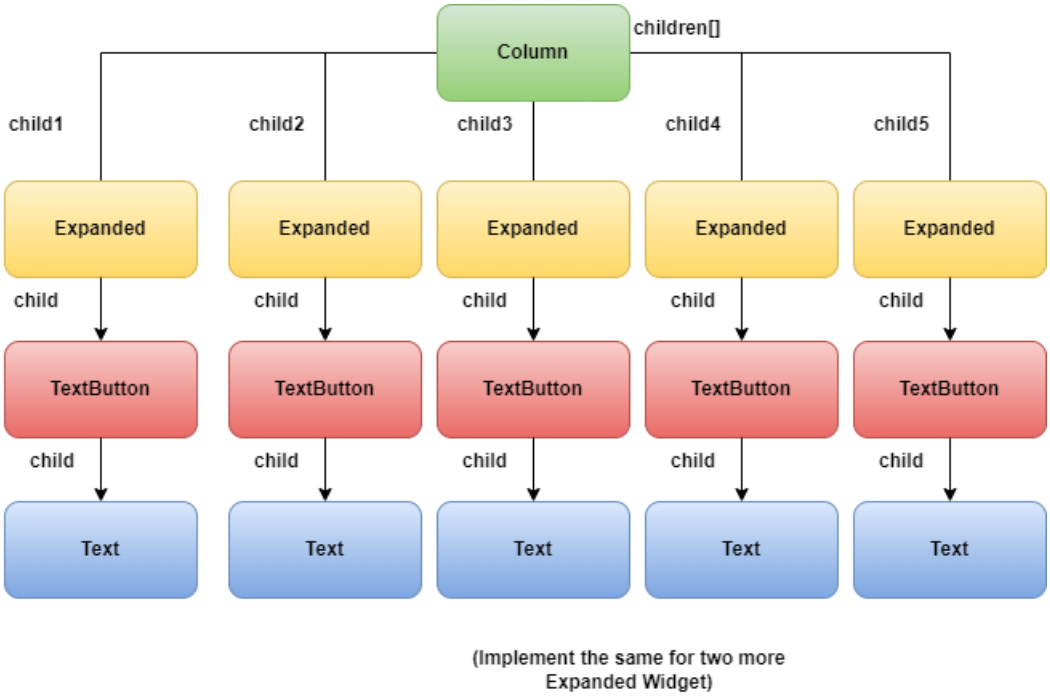

Figure 2: Xylophone: Widget Tree

```
void main() {
  runApp(MaterialApp(
      home:Scaffold(
        appBar: AppBar(title: Text('XYLOPHONE'),centerTitle: true,),
        body:XyloPage(),
      )
  ));
}
class XyloPage extends StatefulWidget {
  const XyloPage({Key? key}) : super(key: key);

  @override
  _XyloPageState createState() => _XyloPageState();
}

class _XyloPageState extends State<XyloPage> {
 @override
  Widget build(BuildContext context) {
    return Container(); // Build the Widget Tree here
    }
}
```

6. Define the function PlayAudio(int i) where it takes a number and plays the respective audio file from the asset folder as shown below,

```
class _XyloPageState extends State<XyloPage> {
  void PlayAudio(int i) async{ // i is the number we pass to String note$i.wav
    AudioPlayer player = AudioPlayer();
    String audioAsset = 'assets/note$i.wav' ; // plays the passed ith audio file
    ByteData bytes = await rootBundle.load(audioAsset);
    Uint8List audiobytes = bytes.buffer.asUint8List(bytes.offsetInBytes,
```

```
      bytes.lengthInBytes);
  int result = await player.playBytes(audiobytes);
  if(result == 1){ //play success
    print("audio is playing.");
  }else{
    print("Error while playing audio.");
  }
}
```

7. Building the App using Widget Tree, the code snippet would look as shown below,

```
@override
 Widget build(BuildContext context) {
   return Column(
     crossAxisAlignment: CrossAxisAlignment.stretch, // Setting the column to
         occupy full screen width horizontally
     children: [
       Expanded(child: TextButton(
         style: TextButton.styleFrom(
             backgroundColor: Colors.amber
         ),
         onPressed: (){
           PlayAudio(1);
         }, child: Text('First'),
       ),),
       Expanded(child: TextButton(
         style: TextButton.styleFrom(
             backgroundColor: Colors.cyanAccent
         ),
       onPressed: (){
           PlayAudio(2);
         }, child: Text('Second'),
       ),),
  // similarly implement the other expanded widgets 5 times and call
      PlayAudio(number)
```

3 Results

The output of the App is shown in figure 3

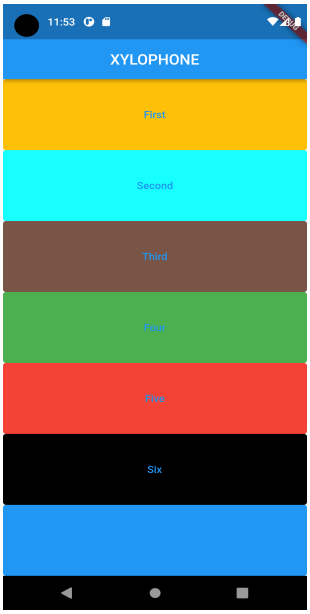

Figure 3: Result