Tapping the Buttons

It's time to add some functionality to our buttons

- Tapping a button makes a “button tap” sound
- Tapping REC will show the STOP button and set the state to RECORD
- Tapping PLAY will show the STOP button and set the state to PLAY
- Tapping STOP will show the PLAY and RECORD buttons and set the state to TAP

We are going to have some constant values as we enhance the drum controller

- Number of beats per pattern
- Length of a beat
- Name of sound file to play when button is tapped

We will make a section for our constants underneath our enum for the states

Add our first constant.

```
// constants
const BUTTON_PRESS_SOUND = 'drum/metro main.wav';  // what to play when button is pressed
```

Why constants?

- One location where we can change values as needed
- Code is easier to read – no hard-coded strings
- All caps makes them stand out as constant values

Feel free to go through the other components looking for hard-coded values

Turn them into constants

The drum controller component will be playing a sound when the buttons are pressed.

See if you can find the needed import so that the play() method can be called.

The drum controller component will be playing a sound when the buttons are pressed.

See if you can find the needed import so that the play() method can be called.

```
import 'package:flame/flame.dart';
```

We check for the PLAY and RECORD button taps when we are in the TAP state. Remember, this is the state where you can just tap on the drums to play them.

We add another case for the TAP state and test for the record button tap

```
case State.TAP:
  // see if user tapped on record button
  if (game.wasTapped && button1Rect.contains(Offset(game.tapX,game.tapY))) {
    game.wasTapped = false;  // reset tap
    Flame.audio.play(BUTTON_PRESS_SOUND);
    state = State.RECORD;
  }

  // see if user tapped on play button

  break;
```

It's easy to test if a button is tapped

- First, the user has to have tapped the screen
- Next, the tap must be inside the rectangle of the image
- Remember to set tapped to false if you have been tapped so that the next update doesn't think it was tapped again
- If tapped, play a sound and change the state to what is appropriate.

Now see if you can add the test for the PLAY button being tapped.

It's easy to test if a button is tapped

- First, the user has to have tapped the screen
- Next, the tap must be inside the rectangle of the image
- Remember to set tapped to false if you have been tapped so that the next update doesn't think it was tapped again
- If tapped, play a sound and change the state to what is appropriate.

Now see if you can add the test for the PLAY button being tapped.

```
// see if user tapped on play button
if (game.wasTapped && button2Rect.contains(Offset(game.tapX,game.tapY))) {
  game.wasTapped = false;  // reset tap
  Flame.audio.play(BUTTON_PRESS_SOUND);
  state = State.PLAY;
}
```

Extra Credit

We have two more states to add. See if you can add them. Here are some hints.

- We need to handle the RECORD and PLAY states in our switch statement
- Both states do the same thing
- The STOP button is in the second button rectangle

Good luck!

The code is very much like the code you already added.

```
case State.RECORD:
  // see if user tapped on stop button
  if (game.wasTapped && button2Rect.contains(Offset(game.tapX,game.tapY))) {
    game.wasTapped = false;  // reset tap
    Flame.audio.play(BUTTON_PRESS_SOUND);
    state = State.TAP;
  }
  break;
case State.PLAY:
  // see if user tapped on stop button
  if (game.wasTapped && button2Rect.contains(Offset(game.tapX,game.tapY))) {
    game.wasTapped = false;  // reset tap
    Flame.audio.play(BUTTON_PRESS_SOUND);
    state = State.TAP;
  }
```

Try It Out

Run the game. The buttons should tap and display the right ones

Drums should still play

PLAY and RECORD will show STOP

STOP will take you back to PLAY and RECORD.

The code for this section can be found here:

https://github.com/shawnlg/flutter_ball/tree/10_working_buttons

Next time, we add a metronome to keep the beat when recording a drum pattern.