A Ball Releaser

It's quite easy to create a ball and add it to the game. In a game with a ball, there is logic in the game that decides when to release the ball. It's not just added to the game.

We will make a control component that does this.

- Release one ball every second until 10 balls have been released and are bouncing.
- Each ball will have a random color, size, and speed, so that each time we run the game, it will behave differently.
- Each ball will have a 50-50 chance of being either a solid circle or an outline circle.

We will be making a new game component. This component won't render anything, but it will control other components.

Create a file called ball_releaser.dart in the components directory where ball.dart is.

Game Timing

How do we release a ball once a second?

Some languages have a wait() method

Program stops for a given amount of time

We don't want the game to stop. It needs to keep running, bouncing balls

Instead, we need to make a component that checks if a certain amount of time has elapsed before it does anything.

Ball Releaser

Add the following to the ball_releaser.dart file to make another game component class.

```
import 'dart:ui';
import 'package:flutter/material.dart';
import 'package:flame/components/component.dart';
import 'package:flutter_ball/flutterball_game.dart';
import 'package:flutter_ball/components/ball.dart';

class BallReleaser extends Component {

  // instance variables
  FlutterballGame game;
  int lives=10;

  // create the component, passing in the game object
  BallReleaser(this.game) : super() {
  }

  void render(Canvas c) => null;

  void update(double t) {
  }

  // tell the game engine if this component should be destroyed
  // whenever it asks
  bool destroy() => lives <= 0;

}
```

There is a game instance variable set by the constructor

When we create this component, we need to pass it the game itself.

This gives the component access to an important method defined on the game:

currentTime()

The currentTime method gives us the number of seconds since January 1, 1970.

We don't care how long it's been since 1970, but we can use it to calculate how much time has elapsed.

```
double t = game.currentTime();  // the time right now
t = game.currentTime() + 1;  // the time one second from now
```

We use this to determine when to release the next ball.

Add this instance variable to our BallReleaser component

```
double timeOfNextBall;  // when to add another ball
```

Add this instance variable to our BallReleaser component

```
double timeOfNextBall;  // when to add another ball
```

Add this to the constructor

```
timeOfNextBall = game.currentTime() + 1;
```

This sets the time that our first ball will release – one second after the component is created.

This component doesn't render anything – so we can use an empty render() method

```
void render(Canvas c) => null;
```

Our update method does several things

- checks if it is time to release another ball
- releases a ball
- Sets the time for the next ball to be released
- Decreases the lives when the ball is released

Add this code to the update method

```
if (game.currentTime() > timeOfNextBall) {
  // make a new ball game component
  var ball = Ball(color: Colors.blue, size: 20.0, speed: 0.5, style:
PaintingStyle.fill);

  // tell the game about this component
  game.add(ball);

  lives--;  // one less ball to add
  timeOfNextBall++;  // next ball one second from now
}
```

The only thing left to do is to update our game class to add this component instead of the ball.

```
import 'package:flutter_ball/components/ball_releaser.dart';// make a new ball
releaser game component


var ballReleaser = BallReleaser(this);

// tell the game about this component
add(ballReleaser);
```

Run the Game

When you run the game, it should produce 10 identical balls, once a second.

If you have any problems, the code is here:

https://github.com/shawnlg/flutter_ball/tree/04_ball_releaser

An important part of many games is their randomness. If we run our game, we will get the exact same thing every time.

We will add some randomness to our ball releaser in the next section.