Random

Right now, we only have a single speed scale for our ball. This means that our x and y movements always are at the same speed. For a completely random speed and direction, we need each to have their own speed scale. Make these changes to Ball.dart

```
double speedScaleX;  // how many times screen width the x speed will be
double speedScaleY;  // how many times screen width the y speed will be

Ball({color:Colors.white, size:10.0, speedX=1.0, speedY=1.0, style = PaintingStyle.stroke,
this.lives:10}) : super() {
  paint.color = color;
  paint.strokeWidth = 1;
  paint.style = style;
  ballSize = size;
  speedScaleX = speedX;
  speedScaleY = speedY;
}

speedX = width*speedScaleX;
speedY = height*speedScaleY;
```

We will now add some randomness to the balls released by the ball releaser.

We need a random number generator

Add this import, which will give us access to the Random class.

```
import 'dart:math';
```

Add this instance variable

```
Random rnd = Random();  // random number generator
```

We now have a class that will give us random numbers whenever we need them

rnd.nextDouble() gives us a random number between 0 and 1

rnd.nextBool() gives us a random true or false (like flipping a coin)

rnd.nextInt(10) gives us a random integer between 0 and 9

Example: Roll a die – get a number between 1 and 6

```
int die = rnd.nextInt(6) + 1;
```

Why the + 1? It is because nextInt(6) will return 6 possible integers starting with 0 (in other words, 0,1,2,3,4,5). We add 1 to this so it will return random numbers from 1 to 6.

What if we want a random speed between some low and high number?

```
double speed = rnd.nextDouble()*(high-low) + low;
```

Let's try this with real numbers to make it clearer. Give a random speed between 8 and 10 – rnd.nextDouble()*(10-8) + 8

The high-low gives us the range of numbers we need. We need numbers between 8 and 10, so we need a range of 2 All the numbers will be no more than 2 apart.

We multiply the random number by this range to get a random number between 0 and 2

We add low to that number to bump up the lowest part of the range to the low number.

Now we have numbers between 8 and 10.

Random Colors

Let's make balls with random colors. Here is some code that picks a random color out of 6 possible colors

```
Color color;
switch (rnd.nextInt(6)) {
  case 0:
    color = Colors.white;
    break;
  case 1:
    color = Colors.blue;
    break;
  case 2:
    color = Colors.green;
    break;
  case 3:
    color = Colors.deepOrange;
    break;
  case 4:
    color = Colors.pink;
    break;
  case 5:
    color = Colors.purple;
    break;
}
```

Other Random Numbers

With what we know about random numbers, we can get the rest of our random values. Add this code to the update() method.

```
// make a new ball game component with some random values
double speedX = rnd.nextDouble()*1.8 + 0.2;  // x speed between 0.2 and 2
double speedY = rnd.nextDouble()*1.8 + 0.2;  // y speed between 0.2 and 2
double size = rnd.nextDouble()*15 + 5;  // size between 5 and 20
PaintingStyle style = rnd.nextBool() ? PaintingStyle.stroke : PaintingStyle.fill;
```

We change how we create the ball.

```
var ball = Ball(color: color, size: size, speedX: speedX, speedY: speedY, style: style);
```

Finally, we change how often a new ball is created.

```
timeOfNextBall += rnd.nextDouble()*1.5 + 0.5;
```

Run the game. You can see that it is a little more interesting with randomness.

The code can be found here:

https://github.com/shawnlg/flutter_ball/tree/05_random_balls

On Your Own

- Add a named parameter on the BallReleaser constructor so that lives can be used when creating it. Right now we hard code to creating 10 balls.
- Add another named parameter, ballLives, and use it to control how many lives each ball will have. Right now we use the default value.
- See how many bouncing balls your phone or emulator can handle before the game slows or crashes.

Next up – human interaction in our game