JAVA GRAPHICS ANIMATION - PART 3

In previous exercises we wrote a program to make a ball bounce on the screen.

Now we need to learn how to use objects so that we can have multiple balls on the screen.

Recall that we had the following variables for the one single ball:

```
//Global variables
int ballx = 100, bally = 100;        //location of ball
int diameter = 40;                   //diameter of ball
int xspeed = 2;                      // normally this is set to 2 or 3 pixels
int yspeed = xspeed;
```

That's five variables for one ball! What happens if we want 2 or 3 or 12 balls on the screen? It would be incredibly tiresome to track all of this.

We make a Ball object using a class (that we write) called Ball.

► class names are Uppercase. Almost everything else in Java is lowercase.

► class names are singular. A class makes ONE object. We can make a lot of balls using the Ball class, but the class only makes one at a time.

Ball Class – this must be in a separate file

```
package graphics;  //put it in the same package
import java.awt.Color;

class Ball {              //public is only needed for your main program
    int x,y;
    int vx, vy;           //vx for velocity (speed) x       vy is y-speed
    int size;
    Color colour;
}
```

- Any ball that we make will have these 6 variables.
- All the variables are set to zero by default. "colour" is set to **null**

```
Ball ball = new Ball();
System.out.println("ball.x is " + ball.x);     //this will print “ball.x is 0”
```

- You can access the variables like this: ball.size , ball.vx , etc.

- If we make two balls, the balls will both be at (0,0) with a size of zero, and zero speed.

We can set default values:

```
class Ball {
    //ball variables set to default values
    int x  = 100, y = 100;
    int vx = 3,   vy = 3;
    int size = 20;
    Color colour = Color.RED;
}
```

and in our main program somewhere …

```
    Ball b1 = new Ball();
    Ball b2 = new Ball();
```

<u>But</u> these two balls will be on top of each other, both at (100,100).

How do we make the balls at different locations? There are 3 ways ...

Method 1: ← only good for very simple programs. Methods 2 or 3 are better.

We can set the variables this way in our main program, right after creating the ball:

```
Ball b2 = new Ball();
b2.x = 200;
b2.y = 300;
```

Now the balls are in different locations.

Method 2: ← this is a good way to do it

We can pass the x,y variables to the constructor when we make the ball:

```
Ball b2 = new Ball(200,300);
```

But we now have to change the Ball class:

```
Ball b2 = new Ball();

class Ball {
    int x,y;
    int vx = 3,   vy = 3;
```

```
    int size;
    Color colour = Color.RED;

    //This is a constructor.
    //It runs every time we make a Ball object (with 2 parameters)
    Ball(int x, int y) {
        this.x = x;
        this.y = y;
        size = 20;
    }
}
```

Note:

- *Default values can be set when you make the variables (vx,vy) or in the constructor (size).*
- *The values of (x,y) that we sent to the constructor are copied to the Ball's variables. Look at the colour coding to see which is which.*
- *If you just make Ball() with no x and y values, then the location will still be at (0,0).*
- *We could make a constructor with more parameters if we also wanted to set individual vx, vy, and colour – for every ball that we make.*

Method 3: ← make random location for the balls

Our main program will simply do this: Ball b2 = new Ball();

Here is the ball class:

```
class Ball {
      int x,y;
      int vx = 3,   vy = 3;
      int size;
      Color colour = Color.RED;

      //Constructor. Set x,y to random locations
      Ball() {
            x = …  how do we do this? See below ...
            y = ...
            size = 20;  //Oops – later on I refer to this as width, not size
      }
}
```

Assume that our graphics window is 800x600:

```
GraphicsConsole gc = new GraphicsConsole(800,600);
```

and that we want a 50 pixel margin around the edges where balls won't appear.

The **purple** box shows a window of width 800 pixels. x goes from 0 to 800.
The **orange** box shows a 50 px margin, so the range of x is 50 to 750.

***However**, notice that the ball on the right side will be **outside** the orange window because the x-value refers to the **left** side of the ball.*
*To bring it back into the window, we need to subtract the ball width: see **red** line.*

So we need a random number from 50 to 750-ball.width.
→ starting number is 50
→ range is 750-ball.width-50 = 700-ball.width

```
x = (int)( Math.random() * range ) + start;       //this will do it! We'll learn how this works later

//finished constructor: each ball appears at a random location.
    Ball() {
        size = 20;
        x = (int)( Math.random() * 700-size ) + 50;
        y = (int)( Math.random() * 500-size ) + 50;
    }
```

The only final improvement would be to use the actual variable for the screen size from your main program. Thus, if your main program is called Bounce.java, and you have static final int WINW = 800; *in it, then*

```
    x = (int)( Math.random() * Bounce.WINW-100-size ) + 50;
```

TO DO:

Call this program “Draw3Balls.java”

- Use what you've learned in the PDF to make a program that has 3 balls on the screen.
- Make a separate file for the ball class: Ball.java
- Each ball should be at a different location, have a different size, and be a different colour.
- *Extra*: make one ball appear in a random location, so that each time you run the program it’s in a new place.

Tip: It might be best to put this into a new package since there will be a number of programs that use "Ball.java" and these balls will be different, and have more complex functionality as we learn more.