# 16. Class BugZapper

The next class to be tackled is the class *BugZapper*. Class *BugZapper* will be housed inside of a file called *BugZapper.java*. It will begin with the usual include of all AWT and Swing classes and will include three private constants. *x* and *y* will give the initial location of the rock while *scale* will give its size.

```
import java.awt.*;
import javax.swing.*;

public class BugZapper {
   private final int x;
   private final int y;
   private final int scale;
        :
        :
}
```

As you would expect, the constructor will receive as parameters the initial values for *x*, *y* and *scale*.

```
   public BugZapper(int initX,int initY,int initScale) {
        x = initX;
        y = initY;
        scale = initScale;
   }
```

Lastly, a *paintComponent* method will be created to draw the actual bug zappers. Since the necessary drawing methods that will be used is part of class *Graphics*, to work, *paintComponent* will receive a variable of class *Graphics* as its single parameter.

Three methods will be used. Two are of class *Graphics* and one of class *Color*. First, the constructor of class *Color* will be used to create two new colors stored in two instance variables of class *Color*. These calls are:

```
Color runnyGray = new Color(128,96,64);
     :
Color runnyBrown = new Color(128,84,64);
     :
```

The *Graphics* method *setColor* is called twice to set the color to be used to draw with. Once the color is set to *runnyGray*, the next time it is set to *runnyBrown*.

```
g.setColor(runnyGray);
     :
g.setColor(runnyBrown);
     :
```

To finish drawing our bug zapper, the *Graphics* method *fillArc* is called liberally after each *setColor* call. *fillArc* requires 6 parameters. These are, from left to right:

- *x, y* – a pair that gives the upper left point of the rectangle that will enclose the arc
- *width, height* – a pair that will give the width (right) from x and the height (down) from y of the rectangle that will enclose the arc
- *start angle* – gives the starting angle of the arc in degrees; 0 degrees is located on the horizontal at 3 o'clock
- *sweep angle* – angular distance from the start angle in degrees; negative draws clockwise, positive draws counter-clockwise

```
public void paintComponent(Graphics g) {
     Color runnyGray = new Color(128,96,64);
     g.setColor(runnyGray);
     g.fillArc(x,y,scale/2,scale/2,0,45);
     g.fillArc(x,y,scale/2,scale/2,90,45);
     g.fillArc(x,y,scale/2,scale/2,180,45);
     g.fillArc(x,y,scale/2,scale/2,270,45);
     Color runnyBrown = new Color(128,84,64);
     g.setColor(runnyBrown);
     g.fillArc(x,y,scale/2,scale/2,45,45);
     g.fillArc(x,y,scale/2,scale/2,135,45);
     g.fillArc(x,y,scale/2,scale/2,225,45);
     g.fillArc(x,y,scale/2,scale/2,315,45);
}
```

This would be all there is to class *BugZapper* were it not for the need in our example game to have a bug zapper that could collide with any type of bug. Method *impactBug* is created here to detect such collisions.

```
public boolean impactBug(Bug p) {
     int halfscale = scale/2;
     if (p.returnx() > x &&
         p.returnx() < x+halfscale &&
         p.returny() > y && p.returny() < y+halfscale)
          return true;
     else return false;
}
```

Here is the complete listing of class *BugZapper*:

```
import java.awt.*;
import javax.swing.*;


public class BugZapper {

   private final int x;
   private final int y;
   private final int scale;


    public BugZapper(int initX,int initY,int initScale) {
        x = initX;
        y = initY;
        scale = initScale;
   }


   public void paintComponent(Graphics g)  {
        Color runnyGray = new Color(128,96,64);
        g.setColor(runnyGray);
        g.fillArc(x,y,scale/2,scale/2,0,45);
        g.fillArc(x,y,scale/2,scale/2,90,45);
        g.fillArc(x,y,scale/2,scale/2,180,45);
        g.fillArc(x,y,scale/2,scale/2,270,45);
        Color runnyBrown = new Color(128,84,64);
        g.setColor(runnyBrown);
        g.fillArc(x,y,scale/2,scale/2,45,45);
        g.fillArc(x,y,scale/2,scale/2,135,45);
        g.fillArc(x,y,scale/2,scale/2,225,45);
        g.fillArc(x,y,scale/2,scale/2,315,45);
   }


   public boolean impactBug(Bug p) {
        int halfscale = scale/2;
        if (p.returnx() > x &&
            p.returnx() < x+halfscale &&
            p.returny() > y &&
            p.returny() < y+halfscale)
             return true;
        else return false;
   }
}
```

To test class *BugZapper*, an object of class *BugZapper* needs to be added to *MainPanel* as a global variable:

```
BugZapper abug_zapper;
```

Next, the bug zapper needs to be created in the constructor of *MainPanel*:

```
abug_zapper = new BugZapper(180,150,50);
```

The bug zapper needs to be drawn in *paintComponent*:

```
abug_zapper.paintComponent(hiddenG);
```

And lastly, code needs to inserted to detect collisions of a bug with the bug zapper in *run*:

```
if (abug_zapper.impactBug(p))
     System.out.print("BugZapper Crash\t");
if (abug_zapper.impactBug(q))
     System.out.print("BugZapper Crash\t");
if (abug_zapper.impactBug(r))
     System.out.print("BugZapper Crash\t");
if (abug_zapper.impactBug(s))
     System.out.print("BugZapper Crash\t");
if (abug_zapper.impactBug(t))
     System.out.print("BugZapper Crash\t");
if (abug_zapper.impactBug(u))
     System.out.print("BugZapper Crash\t");
if (abug_zapper.impactBug(v))
     System.out.print("BugZapper Crash\t");
if (abug_zapper.impactBug(w))
     System.out.print("BugZapper Crash\t");
```

The following is the full listing of the new version of *MainPanel*:

```
import java.awt.*;
import javax.swing.*;

public class MainPanel extends JPanel implements Runnable {

   private final int WIDTH;
   private final int HEIGHT;

   private int temp;

   private Image myBuffer;
   private Thread animate;
```

```
Bug p, q, r;
EvilBug s, t, u, v, w;
BugZapper abug_zapper;

public MainPanel(int wInit, int hInit) {
      WIDTH  = wInit;
      HEIGHT = hInit;

      animate = new Thread(this);

      p = new Bug(500,400,100,275);
      q = new Bug(500,400,300,50);
      r = new Bug(500,400,200,50);

      s = new EvilBug(500,400,200,350,r);
      t = new EvilBug(500,400,0,0,r);
      u = new EvilBug(500,400,500,400,r);
      v = new EvilBug(500,400,0,400,p);
      w = new EvilBug(500,400,500,0,q);

      abug_zapper = new BugZapper(180,150,50);

      p.setBugColor(0,0,255);
      q.setBugColor(255,0,0);
      r.setBugColor(0,255,255);
      s.setBugColor(255,255,0);
      t.setBugColor(127,127,127);
      u.setBugColor(127,127,127);
      v.setBugColor(127,127,127);
      w.setBugColor(127,127,127);
}


public void paintComponent(Graphics g) {
      super.paintComponent(g);

      myBuffer = createImage(WIDTH,HEIGHT);
      Graphics hiddenG = myBuffer.getGraphics();
      hiddenG.setColor(Color.black);
      hiddenG.fillRect(0, 0, WIDTH, HEIGHT);

      r.turnleft();

      p.paintComponent(hiddenG);
      q.paintComponent(hiddenG);
      r.paintComponent(hiddenG);
      s.paintComponent(hiddenG);
      t.paintComponent(hiddenG);
      u.paintComponent(hiddenG);
      v.paintComponent(hiddenG);
      w.paintComponent(hiddenG);
      abug_zapper.paintComponent(hiddenG);

      g.drawImage(myBuffer,0,0,this);
}
```

```
    public void start() {
          animate.start();
    }

    public void run() {
          for (int i=0; i<200; i++) {
                temp = (int)(Math.random( )*8);

                if (temp > 6) p.turnleft( );
                else if (temp < 2) p.turnright( );

                if (temp > 6) q.turnleft();
                else if (temp < 2) q.turnright( );

                p.go( );  q.go( );  r.go( );

                s.calcDirection();  s.go( );
                t.calcDirection();      t.go( );
                u.calcDirection();      u.go( );
                v.calcDirection();      v.go( );
                w.calcDirection();      w.go( );

                repaint();

                if (s.impactBug(r)) System.out.print("Bug Crash!\t");
                      if (abug_zapper.impactBug(p))
                            System.out.print("BugZapper Crash\t");
                if (abug_zapper.impactBug(q))
                            System.out.print("BugZapper Crash\t");
                if (abug_zapper.impactBug(r))
                            System.out.print("BugZapper Crash\t");
                if (abug_zapper.impactBug(s))
                            System.out.print("BugZapper Crash\t");
                if (abug_zapper.impactBug(t))
                            System.out.print("BugZapper Crash\t");
                if (abug_zapper.impactBug(u))
                            System.out.print("BugZapper Crash\t");
                if (abug_zapper.impactBug(v))
                            System.out.print("BugZapper Crash\t");
                if (abug_zapper.impactBug(w))
                            System.out.print("BugZapper Crash\t");

                pause(50);
          }
    }

    private void pause (int time) {
          try { animate.sleep(time); }
          catch (Exception e) {}
    }

}
```

When *BugZapper.java* and *MainPanel.java* are compiled,

```
javac BugZapper.java
javac MainPanel.java
```

then the program is run

```
java JavaByBugs
```

the following animation is output: