12. Inheriting *protoBug* to Make Class *Bug*

As described earlier, the game being created will have a user-controlled bug that will have to dodge bug zappers and other bugs. Some of the bugs will be carnivores and will try to catch the user's Bug. All bug classes will be created from class *protoBug*.

With the exception of setting the bug to an alive state, protoBug contains just about everything that we need for the user's bug and the bugs that just fly around getting in the way. After inheriting *protoBug* via the extends keyword, all we need to add is a constructor. Here is class *Bug*. Notice the call to *super*. To call a constructor of a super class, *super* is used as a function call.

```java
import java.awt.*;

public class Bug extends protoBug {
    
    public Bug(int w, int h, int initx, int inity) {
        super(w, h, initx, inity);
        alive = true;
    }
}
```

To test our assumptions, we can alter the previous version of class *driver* so that class *Bug* objects are used instead of class *protoBug*, plus the new constructor instead of a call to *initializeBug*.

```java
import java.awt.*;
import java.applet.Applet;

public class driver extends Applet {

    public void paint(Graphics g) {
        Background b = new Background(500,400);
        Bug p = new Bug(500,400,100,275);
        Bug q = new Bug(500,400,300,50);
        Bug r = new Bug(500,400,200,350);
        Bug s = new Bug(500,400,200,50);
        p.setBugColor(0,0,255);
        q.setBugColor(255,0,0);
        r.setBugColor(0,255,255);
        s.setBugColor(255,255,0);
        b.paint(g);  p.paint(g);  q.paint(g);  r.paint(g);  s.paint(g);

        for (int j=0; j<8; j++) s.turnleft();

        for (int i=0; i<100; i++) {
            p.turnleft();  q.turnright();
        }
    }
}
```
jump start for java 2 applets

p.go(); q.go(); r.go(); s.go();
b.paint(g); p.paint(g); q.paint(g); r.paint(g); s.paint(g);
System.out.print("p=(" + p.returnx() + "," + p.returny() + ")\t");
if (r.impactBug(s)) break;
pause(50);
}

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

Since driver runs with Bug exactly as it did with protoBug, our assumption was justified.

13. Inheriting protoBug to Make Class evilBug

Like Bug, class evilBug begins by inheriting protoBug by means of the extends keyword and creating a constructor. This constructor is the same as the one created for Bug but for the inclusion of a parameter of class Bug to represent the intended victim. This is made necessary by the fact that objects of evilBug are to seek out and destroy the Bug object controlled by the user. (Notice the addition of the private instance variable of class Bug called victim.)

import java.awt.*;

public class evilBug extends protoBug {

    private Bug victim;

    public evilBug(int w, int h, int initx, int inity, Bug initVictim) {
        super(w, h, initx, inity);
        alive = true;
        victim = initVictim;
    }

    The next feature that needs to be added to complete evilBug is the ability to turn in the direction of the victim Bug object. This is the method calcDirection.

    The basic idea of calcDirection is to get the (x,y) coordinate pair of the victim, determine the slope to the victim, then set direction to point down the slope. Care must be taken to insure that if the x data of the victim and the evilBug object are the same that the slope calculation does not take place (to avoid division by zero). In that case, all that need be determined is if the victim is above or below the evilBug object. In the case
where the x data of both objects is the same, a shortcut is available because all that need be determined is whether the Bug is to the left or right of the evilBug object.

```java
public void calcDirection() {
    // *** store (x,y) location of the victim ***
    double vx = victim.returnx();
    double vy = victim.returny();

    // *** set direction to 0
    int direction = 0;

    // *** if slope 0, set direction ***
    if (y == vy) {
        if (x > vx) direction = 12;
        else direction = 4;
    }

    // *** if slope impossible, select up or down ***
    else if (x == vx) {
        if (y > vy) direction = 0;
        else direction = 8;
    }

    // *** otherwise, calculate slope & determine direction ***
    else {
        double slope = (vy - y) / (vx - x);
        if (slope > 0) {
            if (slope < .5) {
                if (x > vx) direction = 12;
                else direction = 4;
            }
            else if (slope > 2) {
                if (x > vx) direction = 0;
                else direction = 8;
            } else {
                if (x > vx) direction = 14;
                else direction = 6;
            }
        }
        else if (slope < 0) {
            if (slope > - .5) {
                if (x > vx) direction = 12;
                else direction = 4;
            }
        }
    }
}
```
else if (slope < -2) {
    if (x > vx) direction = 8;
    else direction = 0;
}
else {
    if (x > vx) direction = 10;
    else direction = 2;
}
}

// *** call method setdirection to transfer local ***
// *** direction to global direction       ***
setdirection(direction);

} // *** end of calcDirection ***

Full Listing of Class evilBug

import java.awt.*;

public class evilBug extends protoBug {

    private Bug victim;

    public evilBug(int w, int h, int initx, int inity, Bug initVictim) {
        super(w, h, initx, inity);
        alive = true;
        victim = initVictim;
    }

    public void calcDirection() {
        double vx = victim.returnx();
        double vy = victim.returny();
        int direction = 0;
        if (y == vy) {
            if (x > vx) direction = 12;
            else direction = 4;
        } else if (x == vx) {
            if (y > vy) direction = 8;
            else direction = 0;
        } else {
            double slope = (vy - y) / (vx - x);
            if (slope > 0) {
                if (slope < .5) {
if (x > vx) direction = 12;
else direction = 4;
}
else if (slope > 2) {
    if (x > vx) direction = 0;
    else direction = 8;
}
else {
    if (x > vx) direction = 14;
    else direction = 6;
}
}
else if (slope < 0) {
    if (slope > -.5) {
        if (x > vx) direction = 12;
        else direction = 4;
    }
    else if (slope < -2) {
        if (x > vx) direction = 8;
        else direction = 0;
    }
    else {
        if (x > vx) direction = 10;
        else direction = 2;
    }
}
}
setdirection(direction);

**Testing Class evilBug**

Testing class evilBug involves adding evilBug objects to the driver program and selecting suitable targets for them. Notice that objects of Bug class and evilBug class can be tested for crashes via the method impactBug that both inherited from protoBug.

To give the evilBug objects a bit of a workout, two Bug objects are set to move randomly. This is accomplished with the method random or class Math is called to make a value for the new instance variable temp. Since random returns a value from 0 to 1, multiplying the return of the call to random by 8 gives a value from 0 to 8. Casting it allows the result to be assigned to the integer variable temp, which can then be tested to determine the direction that two Bug objects should turn.
import java.awt.*;
import java.applet.Applet;

public class driver extends Applet {

    public void paint(Graphics g) {
        int temp;

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

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

        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);

        b.paint(g);  p.paint(g);  q.paint(g);  r.paint(g);
        s.paint(g);  t.paint(g);  u.paint(g);  v.paint(g);  w.paint(g);

        for (int j=0; j<8; j++) r.turnleft( );

        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();
b.paint(g); p.paint(g); q.paint(g); r.paint(g);
s.paint(g); t.paint(g); u.paint(g); v.paint(g); w.paint(g);

if (s.impactBug(r)) System.out.print("crash\t");

    pause(50);
}

private void pause (int time) {
    try { Thread.sleep(time); } 
    catch (Exception e) {} 
}
}
Here is an example of the run of *driver.class*:

```
C:\jdk1.3\projects\bugs>go
C:\jdk1.3\projects\bugs>del driver.class
C:\jdk1.3\projects\bugs>del evilBug.class
C:\jdk1.3\projects\bugs>javac evilBug.java
C:\jdk1.3\projects\bugs>pause
Press any key to continue . . .

C:\jdk1.3\projects\bugs>javac driver.java
C:\jdk1.3\projects\bugs>pause
Press any key to continue . . .

C:\jdk1.3\projects\bugs>appletviewer driver.html
```

```