```java
import java.awt.Color;
import java.awt.Graphics;
public class MovingBall
{
    private double x;
    private double y;
    private int radius;
    private double xStep;
    private double yStep;

    public MovingBall(double x, double y, int radius, double xStep, double yStep)
    {
        this.x = x;
        this.y = y;
        this.radius = radius;
        this.xStep = xStep;
        this.yStep = yStep;
    }

    public double getX()
    {
        return x;
    }

    public double getY()
    {
        return y;
    }

    public int getRadius()
    {
        return radius;
    }

    public double getXStep()
    {
        return xStep;
    }

    public double getYStep()
    {
        return yStep;
    }

    public boolean timeHasElapsed()
    {
        return false;
    }

    public void draw(Graphics g)
    {
        g.setColor(Color.BLUE);
        g.drawOval((int) (x - radius), (int) (y - radius), 2 * radius, 2 * radius);
    }

    public void doStep(int width, int height)
    {
        if (x + radius + xStep >= width)
        {
            x = width - radius - 1;
            xStep = -xStep;
        }
        else if (x - radius + xStep < 0)
        {
            x = radius;
            xStep = -xStep;
        }
        else
            x = x + xStep;

        if (y + radius + yStep >= height)
        {
            y = height - radius - 1;
            yStep = -yStep;
        }
        else if (y - radius + yStep < 0)
        {
            y = radius;
            yStep = -yStep;
        }
        else
            y += yStep;  // même chose que: y = y + yStep;
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class Emoticon extends MovingBall
{
    public Emoticon(double x, double y, int radius, double xStep, double yStep)
    {
        super(x, y, radius, xStep, yStep);
    }

    public void draw(Graphics g)
    {
        // Accès rapide
        double x = getX();
        double y = getY();
        int r = getRadius();

        // Dessine le disque jaune et les yeux
        g.setColor(Color.YELLOW);
        g.fillOval((int) (x - r), (int) (y - r), 2 * r, 2 * r);

        // Oeil gauche / droit - partie blanche
        g.setColor(Color.WHITE);
        g.fillOval((int) (x - r * 5 / 8), (int) (y - r * 4 / 8), r / 2, r / 2);
        g.fillOval((int) (x + r * 1 / 8), (int) (y - r * 4 / 8), r / 2, r / 2);

        // Oeil gauche / droit - contour gris
        g.setColor(Color.LIGHT_GRAY);
        g.drawOval((int) (x - r * 5 / 8), (int) (y - r * 4 / 8), r / 2, r / 2);
        g.drawOval((int) (x + r * 1 / 8), (int) (y - r * 4 / 8), r / 2, r / 2);

        // Oeil gauche / droit - iris en bleu
        g.setColor(Color.BLUE);
        g.fillOval((int) (x - r * 10 / 16 + r * 2 / 16), (int) (y - r * 8 / 16 + r * 3 / 16), (int) (r / 4), (int) (r / 4));
        g.fillOval((int) (x + r * 2 / 16 + r * 2 / 16), (int) (y - r * 8 / 16 + r * 3 / 16), (int) (r / 4), (int) (r / 4));
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class EmoBigSmile extends Emoticon
{
    public EmoBigSmile(double x, double y, int radius, double xStep, double yStep)
    {
        super(x, y, radius, xStep, yStep);
    }

    public void draw(Graphics g)
    {
        // Dessine le cercle et les yeux
        super.draw(g);

        // Accès rapide
        double x = getX();
        double y = getY();
        int r = getRadius();

        // Dessiner la bouche
        g.setColor(Color.BLACK);
        g.fillArc((int) (x - r * 0.5), (int) (y - r * 0.25), r, r, 0, -180);
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class EmoSmile extends Emoticon
{
    public EmoSmile(double x, double y, int radius, double xStep, double yStep)
    {
        super(x, y, radius, xStep, yStep);
    }

    public void draw(Graphics g)
    {
        // Dessine le cercle et les yeux
        super.draw(g);

        // Accès rapide
        double x = getX();
        double y = getY();
        int r = getRadius();

        // Dessiner la bouche
        g.setColor(Color.BLACK);
        g.drawArc((int) (x - r * 0.5), (int) (y - r * 0.25), r, r, 0, -180);
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class EmoSurprised extends Emoticon
{
    public EmoSurprised(double x, double y, int radius, double xStep, double yStep)
    {
        super(x, y, radius, xStep, yStep);
    }

    public void draw(Graphics g)
    {
        // Dessine le cercle et les yeux
        super.draw(g);

        // Accès rapide
        double x = getX();
        double y = getY();
        int r = getRadius();

        // Dessiner la bouche
        g.setColor(Color.BLACK);
        g.fillOval((int) (x - r * 0.25), (int) (y + r * 0.25), (int) (r * 0.5), (int) (r * 0.5));
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class EmoNaughty extends Emoticon
{
    // Durée (en pas) aléatoire pour tirer la langue - valeur entre 10..50
    private int steps = (int) (Math.random() * 41) + 10;
    private int counter = 0;
    private boolean tongueOut = false;

    public EmoNaughty(double x, double y, int radius, double xStep, double yStep)
    {
        super(x, y, radius, xStep, yStep);
    }

    public void draw(Graphics g)
    {
        // Dessine le cercle et les yeux
        super.draw(g);

        // Accès rapide
        double x = getX();
        double y = getY();
        int r = getRadius();

        // Dessiner la langue
        if (tongueOut)
        {
            g.setColor(Color.PINK);
            g.fillArc((int) (x - r * 0.3), (int) (y - r * 0.25), (int) (r * 0.6), (int) r, 0, -180);
        }

        // Dessiner la bouche (ligne simple)
        g.setColor(Color.BLACK);
        g.drawLine((int) (x - r * 0.5), (int) (y + r * 0.25), (int) (x + r * 0.5), (int) (y + r * 0.25));
    }

    public void doStep(int width, int height)
    {
        super.doStep(width, height);

        // Vérifier s'il faut sortir ou rentrer la langue
        counter++;
        if (counter >= steps)
        {
            tongueOut = !tongueOut;
            counter = 0;
        }
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class EmoSad extends Emoticon
{
    // Durée (en pas) aléatoire pour disparaître entre 500..1000 --> entre 5s et 10s
    private int steps = (int) (Math.random() * (1000-500+1)) + 500;

    public EmoSad(double x, double y, int radius, double xStep, double yStep)
    {
        super(x, y, radius, xStep, yStep);
    }

    public boolean timeHasElapsed()
    {
        return (steps == 0);
    }

    public void doStep(int width, int height)
    {
        super.doStep(width, height);

        // diminuer le temps de "vie"
        steps--;
    }

    public void draw(Graphics g)
    {
        // Dessine le cercle et les yeux
        super.draw(g);

        // Accès rapide
        double x = getX();
        double y = getY();
        int r = getRadius();

        // Dessiner la bouche
        g.setColor(Color.BLACK);
        g.drawArc((int) (x - r * 0.5), (int) (y + r * 0.25), r, r, 0, 180);
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
import java.util.ArrayList;
public class MovingBalls
{
    private ArrayList<MovingBall> alMovingBalls = new ArrayList<>();

    public void addBall(MovingBall movingBall)
    {
        alMovingBalls.add(movingBall);
    }

    public void clear()
    {
        alMovingBalls.clear();
    }

    public void draw(Graphics g)
    {
        for (int i = 0; i < alMovingBalls.size(); i++)
            alMovingBalls.get(i).draw(g);
    }

    public void doStep(int width, int height)
    {
        for (int i = 0; i < alMovingBalls.size(); i++)
            alMovingBalls.get(i).doStep(width, height);

        // vérifier s'il faut enlever un EmoSad
        // WARNING: toujours commencer pas la fin de la liste
        for (int i = alMovingBalls.size() -1; i>=0; i--)
        {
            if (alMovingBalls.get(i).timeHasElapsed())
                alMovingBalls.remove(i);
        }
    }
}
```

```java
import java.awt.Color;
import java.awt.Graphics;
public class DrawPanel extends javax.swing.JPanel
{
    private MovingBalls movingBalls = null;

    public DrawPanel()
    {
        initComponents();
    }

    public void setMovingBalls(MovingBalls movingBalls)
    {
        this.movingBalls = movingBalls;
    }

    public void paintComponent(Graphics g)
    {
        // clean the background
        g.setColor(Color.WHITE);
        g.fillRect(0, 0, getWidth(), getHeight());

        // draw the ball
        if (movingBalls != null)
            movingBalls.draw(g);
    }
// Skipped: ... initComponents { ... }
    // Variables declaration - do not modify//GEN-BEGIN:variables
    // End of variables declaration//GEN-END:variables
}
```

```java
import javax.swing.Timer;
public class MainFrame extends javax.swing.JFrame
{
    private MovingBalls movingBalls = null;

    private int delay = 10;
    private Timer timer = null;

    public MainFrame()
    {
        initComponents();
        movingBalls = new MovingBalls();
        drawPanel.setMovingBalls(movingBalls);
        timer = new Timer(delay, stepButton.getActionListeners()[0]);
        stepButton.setVisible(false);
    }
// Skipped: ... initComponents { ... }
    public double random(double min, double max)
    {
        return (Math.random() * (max - min)) + min;
    }

    private void startStopButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_startStopButtonActionPerformed
        if (timer.isRunning())
        {
            timer.stop();
            startStopButton.setText("Start");
        }
        else
        {
            movingBalls.clear();
            for (int i = 0; i < 30; i++)
            {
                int radius = 30;
                double x = random(radius, drawPanel.getWidth() - radius);
                double y = random(radius, drawPanel.getHeight() - radius);
                double xStep = random(-5, 5);
                double yStep = random(-5, 5);

                // on ne dessine plus le "MovingBall" bleu
                int kind = (int) random(0, 5); // valeurs: 0, 1, 2, 3, 4
                Emoticon emoticon;
                if (kind == 0)
                    emoticon = new EmoSmile(x, y, radius, xStep, yStep);
                else if (kind == 1)
                    emoticon = new EmoBigSmile(x, y, radius, xStep, yStep);
                else if (kind == 2)
                    emoticon = new EmoSad(x, y, radius, xStep, yStep);
                else if (kind == 3)
                    emoticon = new EmoSurprised(x, y, radius, xStep, yStep);
                else
                    emoticon = new EmoNaughty(x, y, radius, xStep, yStep);

                movingBalls.addBall(emoticon);
            }
            timer.start();
            startStopButton.setText("Stop");
        }

        repaint();
    }//GEN-LAST:event_startStopButtonActionPerformed

    private void stepButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_stepButtonActionPerformed
        movingBalls.doStep(drawPanel.getWidth(), drawPanel.getHeight());
        repaint();
    }//GEN-LAST:event_stepButtonActionPerformed
// Skipped: ... Look & Feel
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private DrawPanel drawPanel;
    private javax.swing.JButton startStopButton;
    private javax.swing.JButton stepButton;
    // End of variables declaration//GEN-END:variables
}
```