INSTITUT SUPERIEUR POLYTECHNIQUE DE MADAGASCAR

TP JAVA n°8 – Polymorphisme et Animation swing

1) Implémentez les codes sources suivantes :

AnimationObject.java

```
1   package animation;
2   import java.awt.Graphics2D;
3
4   public abstract class AnimationObject {
5       protected float x, y;           // Position
6       protected float width, height;  // Dimensions
7       public AnimationObject(float x, float y, float width, float height) {
8           this.x = x;
9           this.y = y;
10          this.width = width;
11          this.height = height;
12      }
13      public abstract void update(float deltaTime, int containerWidth, int containerHeight);
14      public abstract void render(Graphics2D g2d);
15   }
16
```

Ball.java

```
1   package animation;
2   import java.awt.Color;
3   import java.awt.Graphics2D;
4
5   public class Ball extends AnimationObject {
6       private Color color;
7       private float speedX, speedY;
8
9       public Ball(float x, float y, float width, float height, Color color) {
10          super(x, y, width, height);
11          this.color = color;
12          this.speedX = (float) (Math.random() * 100 - 50); // Random initial speed
13          this.speedY = (float) (Math.random() * 100 - 50);
14      }
15      @Override
16      public void update(float deltaTime, int containerWidth, int containerHeight) {
17          // Update position based on speed
18          x += speedX * deltaTime;
19          y += speedY * deltaTime;
20          // Simple boundary collision detection using provided container boundaries
21          if (x < 0 || x + width > containerWidth) {
22              speedX *= -1;
23              x = Math.max(0, Math.min(x, containerWidth - width));
24          }
25          if (y < 0 || y + height > containerHeight) {
26              speedY *= -1;
27              y = Math.max(0, Math.min(y, containerHeight - height));
28          }
29      }
30      @Override
31      public void render(Graphics2D g2d) {
32          g2d.setColor(color);
33          g2d.fillOval(Math.round(x), Math.round(y), Math.round(width), Math.round(height));
34      }
35  }
```

AnimationPanel.java

```
1  package animation;
2  import java.awt.*;
3  import java.awt.event.*;
4  import javax.swing.*;
5
6  public class AnimationPanel extends JPanel {
7      private AnimationObject[] objects;
8      private Timer timer;
9      private long lastUpdateTime;
10     private int fps;
11     public AnimationPanel(AnimationObject[] objects, int fps) {
12         this.objects = objects;
13         this.fps = fps;
14         setBackground(Color.WHITE);
15         lastUpdateTime = System.currentTimeMillis();
16         int delay = 1000 / fps;
17         timer = new Timer(delay, new ActionListener() {
18             @Override
19             public void actionPerformed(ActionEvent e) {
20                 long currentTime = System.currentTimeMillis(); // Get the system time
21                 float deltaTime = (currentTime - lastUpdateTime) / 1000f; // Convert to seconds
22                 lastUpdateTime = currentTime;
23                 for (AnimationObject obj : objects) {
24                     obj.update(deltaTime, getWidth(), getHeight());
25                 }
26                 repaint(); // repaint the component
27             }
28         });
29     }
30     public void startAnimation() {
31         timer.start();
32     }
33     @Override
34     protected void paintComponent(Graphics g) {
35         super.paintComponent(g);
36         Graphics2D g2d = (Graphics2D) g;
37         for (AnimationObject obj : objects) { // Render all objects
38             obj.render(g2d);
39         }
40     }
41 }
```

Main.java

```
1  package animation;
2  import java.awt.Color;
3  import javax.swing.JFrame;
4
5  public class Main {
6      public static void main(String[] args) {
7          JFrame frame = new JFrame("Animation et Polymorphisme Sucré");
8          frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
9          frame.setSize(800, 600);
10         AnimationObject[] animationObjects = new AnimationObject[]{
11             new Ball(100, 100, 50, 50, Color.RED),
12             new Ball(200, 200, 30, 30, Color.BLUE),
13         };
14         AnimationPanel panel = new AnimationPanel(animationObjects, 15); //15 FPS
15         frame.add(panel);
16         frame.setVisible(true);
17         panel.startAnimation();
18     }
19 }
```

2) Ecrire une classe `Rectangle` (extends `AnimationObject`), un rectangle qui tombe sous l'effet de son poids. Ajouter dans Main.