Зелинская И.Ю., 4 курс, ИВТ-1

**ЛАБОРАТОРНАЯ РАБОТА №3**

**Тема:** графика в java, пакет java.awt, javax.swing

**Цель:** в данной лабораторной работе вы получите первое представление о создании неконсольных приложений на языке java, а также научитесь использовать классы и методы пакетов java.awt и javax.swing для отображения графических фигур.

**Программно-аппаратное средство:** IntelliJ IDEA

**Выполнение задания:**

1. Модифицировать код программы bouncingcircle таким образом, чтобы вместо круга движение осуществлял экземпляр реализованного ранее (в лабораторной работе №2) класса ColorableRect()

2. Модифицировать код предыдущей программы таким образом, чтобы движение осуществляли сразу 10 экземпляров класса Rectangle, 10 класса DrawableRect и 10 экземпляров класса ColorableRect. Все созданные объекты должны хранится в одном массиве с типом класса родителя Rectangle.

public class BouncingRectSimple {  
 private static final int *BOX\_WIDTH* = 640;  
 private static final int *BOX\_HEIGHT* = 480;  
  
 Rectangle[] rects = {  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new Rectangle(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new ColoredRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10))),  
 new DrawableRect(10 + (int) (Math.*random*() \* ((100 - 10) + 10)), 10 + (int) (Math.*random*() \* ((100 - 10) + 10)))  
 };  
  
 private static final int *UPDATE\_RATE* = 60; // Number of refresh per second  
  
 */\*\*  
 \* Constructor to create the UI components and init game objects.  
 \*/* public BouncingRectSimple() {  
 this.setPreferredSize(new Dimension(*BOX\_WIDTH*, *BOX\_HEIGHT*));  
  
 for (first.Rectangle rect : rects) {  
 Thread gameThread = new Thread() {  
 public void run() {  
 while (true) { // Execute one update step  
 // Calculate the rect's new position  
 rect.move((int) rect.speedX, (int) rect.speedY);  
 // Check if the rect moves over the bounds  
 // If so, adjust the position and speed.  
 if (rect.x1 < 0) {  
 rect.speedX = -rect.speedX; // Reflect along normal  
 rect.move((int) rect.speedX, (int) rect.speedY);  
 } else if (rect.x2 > *BOX\_WIDTH*) {  
 rect.speedX = -rect.speedX;  
 rect.move((int) rect.speedX, (int) rect.speedY);  
 }  
 // May cross both x and y bounds  
 if (rect.y1 < 0) {  
 rect.speedY = -rect.speedY;  
 rect.move((int) rect.speedX, (int) rect.speedY);  
  
 } else if (rect.y2 > *BOX\_HEIGHT*) {  
 rect.speedY = -rect.speedY;  
 rect.move((int) rect.speedX, (int) rect.speedY);  
 }  
 // Refresh the display  
 repaint(); // Callback paintComponent()  
 // Delay for timing control and give other threads a chance  
 try {  
 Thread.*sleep*(1000 / *UPDATE\_RATE*); // milliseconds  
 } catch (InterruptedException ex) {  
 }  
 }  
 }  
 };  
 gameThread.start(); // Callback run()  
 }  
 }  
  
 */\*\*  
 \* Custom rendering codes for drawing the JPanel  
 \*/* @Override  
 public void paintComponent(Graphics g  
 ) {  
 super.paintComponent(g);  
  
 g.setColor(Color.*white*);  
 g.fillRect(0, 0, *BOX\_WIDTH*, *BOX\_HEIGHT*);  
 for (first.Rectangle rect : rects) {  
 rect.draw(g);  
 }  
 }  
  
 */\*\*  
 \* main program (entry point)  
 \*/* public static void main(String[] args) {  
 // Run GUI in the Event Dispatcher Thread (EDT) instead of main thread.  
 javax.swing.SwingUtilities.*invokeLater*(new Runnable() {  
 public void run() {  
 // Set up main window (using Swing's Jframe)  
 JFrame frame = new JFrame("Rectangles");  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.setContentPane(new BouncingRectSimple());  
 frame.pack();  
 frame.setVisible(true);  
 }  
 });  
 }  
}

**Список источников:**

1. Шилдт, Герберт Java 8. Руководство для начинающих / Герберт Шилдт. - М.: Вильямс, 2015. - 720 c.