**Лабораторная работа №5**

**Иванченко Павла**

Тема: Многопоточность в Java и модификация графического приложения.

Цель:  Изучить основы многопоточности в Java и модифицировать существующее графическое приложение, добавив возможность одновременного управления несколькими объектами с использованием потоков.

**Постановка задачи.**

Необходимо модифицировать тренажер клавиатуры, добавив многопоточность для управления несколькими падающими кругами с буквами. Каждый круг должен двигаться в отдельном потоке. Игрок должен нажимать клавиши, соответствующие буквам в кругах. При правильном нажатии счет увеличивается, а круг исчезает и заменяется новым. Если круг достигает нижней границы экрана, счет уменьшается. Игра завершается при достижении 50 очков.

**Ход работы**

Основные изменения в программе:

* Добавление многопоточности. Класс FallingCircle реализует интерфейс Runnable, что позволяет каждому кругу двигаться в отдельном потоке. Для хранения кругов используется потокобезопасная коллекция CopyOnWriteArrayList.
* Новая структура классов. KeyboardTrainer отвечает за интерфейс пользователя и управление игрой. FallingCircle управляет движением круга, его отрисовкой и взаимодействием с игроком.
* Логика игры. Одновременно на экране может находиться до 4 кругов. Каждый круг движется с случайной скоростью. При нажатии правильной клавиши круг исчезает, и создается новый.

Новая диаграмма классов в соответствии с изменениями:

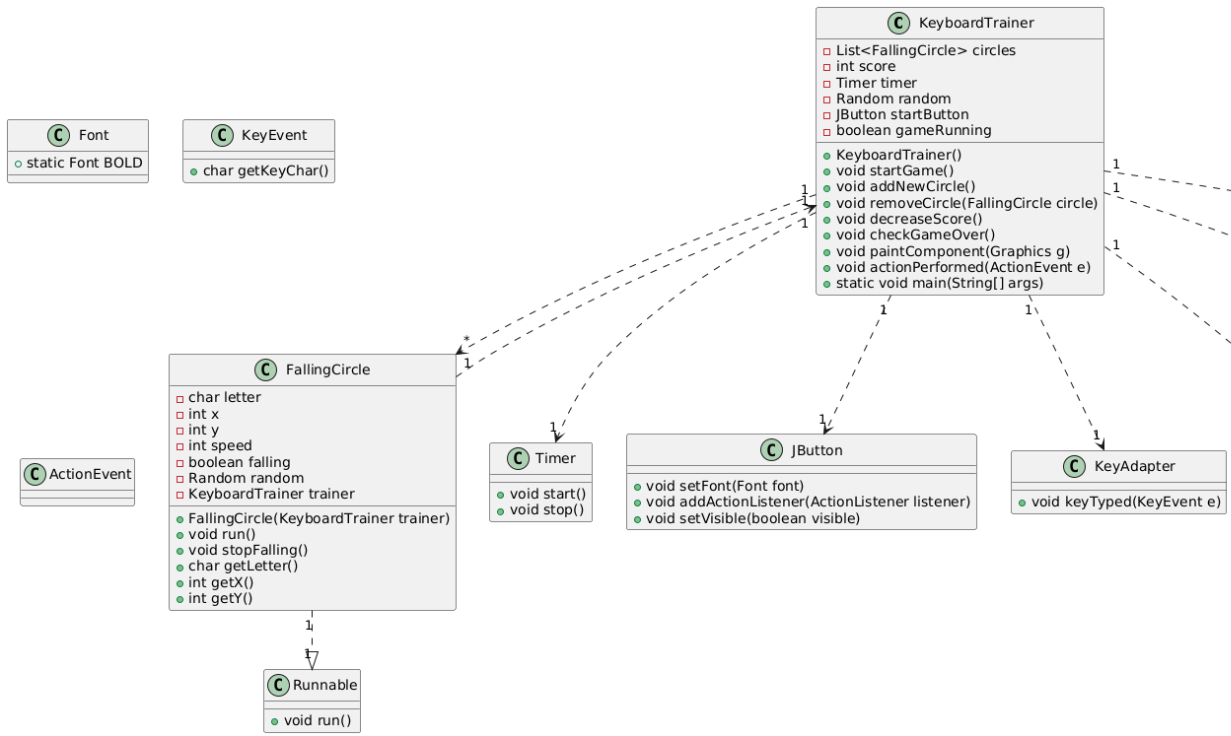


Рис.1, 2 – UML-диаграмма классов

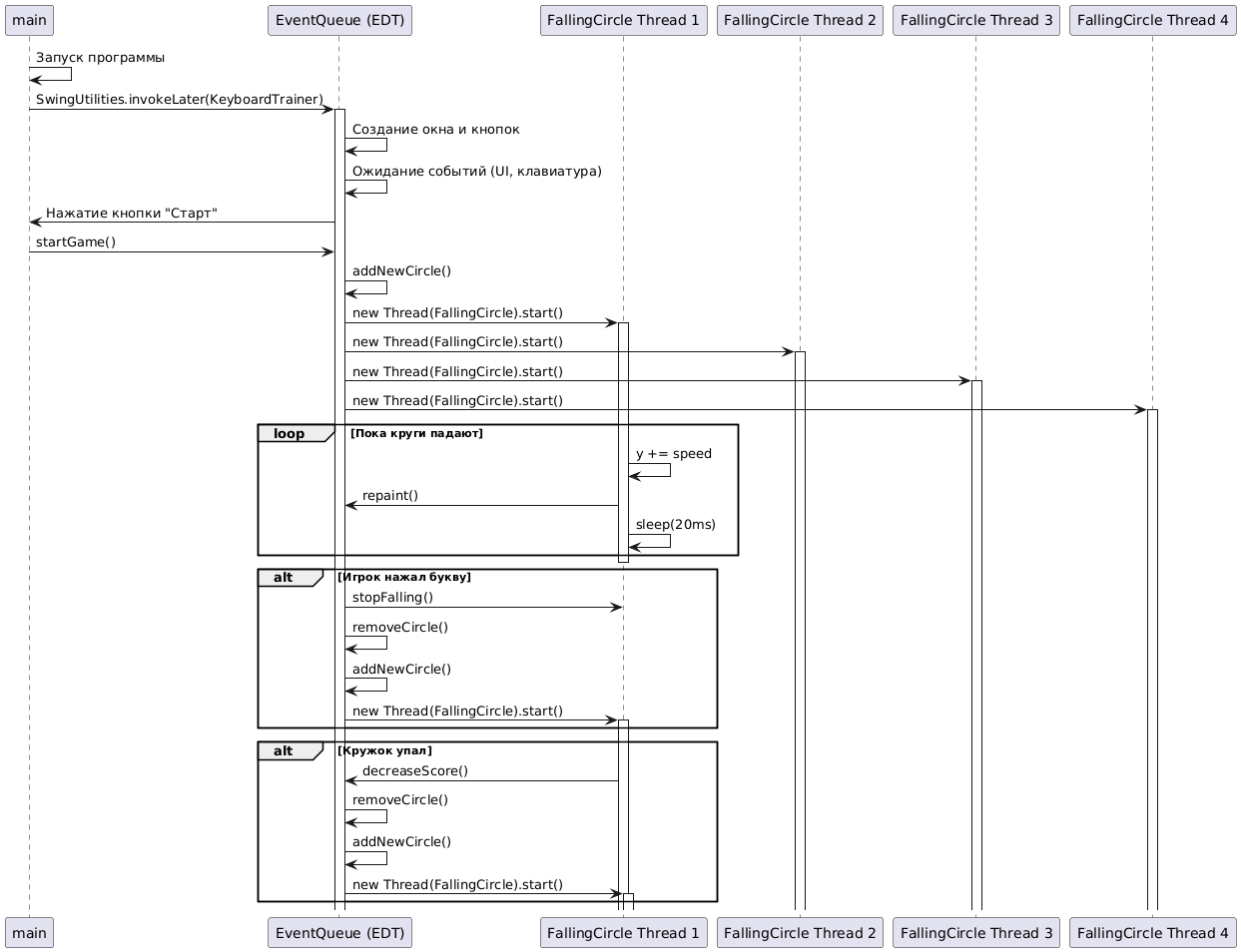


Рис. 3 – Диаграмма работы потоков в приложении

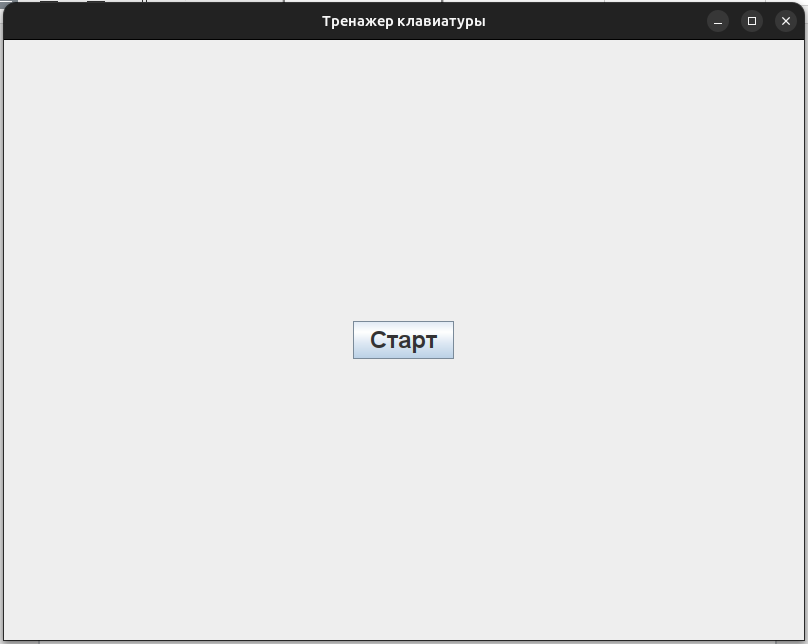


Рис. 4 – Стартовое окно программы

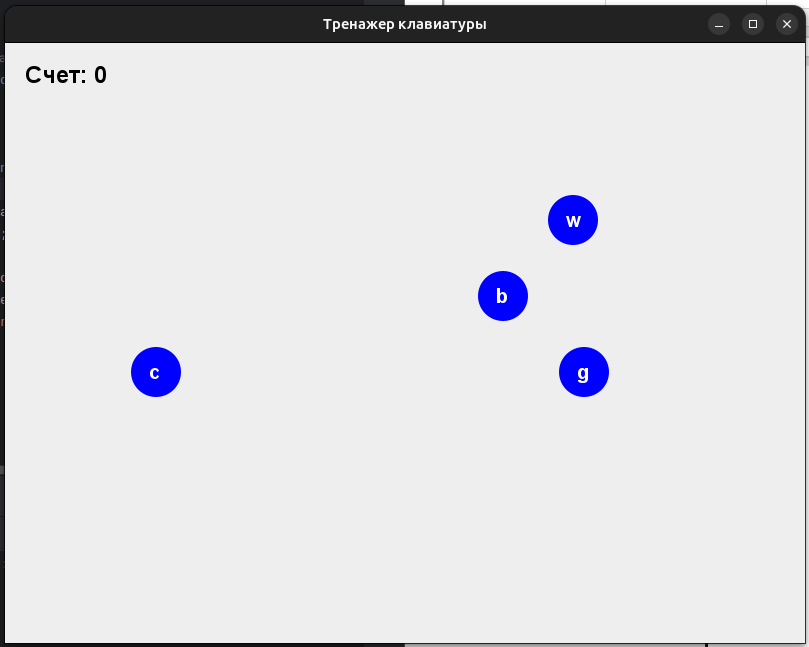


Рис. 5 – Программа в статусе gameRunning

ПРИЛОЖЕНИЕ А

Код класса KeyboarTrainer

import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.\*;  
import java.util.\*;  
import java.util.List;  
import java.util.concurrent.CopyOnWriteArrayList;  
  
public class KeyboardTrainer extends JPanel implements ActionListener {  
 private final int MAX\_CIRCLES = 4;  
 private final List<FallingCircle> circles = new CopyOnWriteArrayList<>();  
 private final Random random = new Random();  
 private final javax.swing.Timer timer = new javax.swing.Timer(5, this);  
 private final JButton startButton;  
 private int score = 0;  
 private boolean gameRunning = false;  
  
 public KeyboardTrainer() {  
 setPreferredSize(new Dimension(800, 600));  
 setLayout(new GridBagLayout());  
 setFocusable(true);  
  
 startButton = new JButton("Старт");  
 startButton.setFont(new Font("Arial", Font.*BOLD*, 24));  
 startButton.addActionListener(e -> startGame());  
  
 GridBagConstraints gbc = new GridBagConstraints();  
 add(startButton, gbc);  
  
 addKeyListener(new KeyAdapter() {  
 @Override  
 public void keyTyped(KeyEvent e) {  
 if (gameRunning) {  
 char pressedKey = e.getKeyChar();  
 for (FallingCircle circle : circles) {  
 if (circle.getLetter() == pressedKey) {  
 score++;  
 circle.stopFalling();  
 circles.remove(circle); // Теперь это безопасно  
 addNewCircle(); // Добавляем новый круг  
 }  
 }  
 checkGameOver();  
 }  
 }  
 });  
 }  
  
 private void startGame() {  
 gameRunning = true;  
 startButton.setVisible(false);  
 score = 0;  
 circles.clear();  
 timer.start();  
 requestFocusInWindow();  
  
 // Запускаем начальные 4 круга  
 for (int i = 0; i < MAX\_CIRCLES; i++) {  
 addNewCircle();  
 }  
 }  
  
 private void addNewCircle() {  
 FallingCircle circle = new FallingCircle(this);  
 synchronized (circles) {  
 circles.add(circle);  
 }  
 new Thread(circle).start(); // Запускаем поток с шариком  
 }  
  
 public void removeCircle(FallingCircle circle) {  
 circles.remove(circle);  
 addNewCircle();  
 }  
  
 public void decreaseScore() {  
 if (score > 0) { // Чтобы не уходило в минус  
 score--;  
 }  
 }  
  
 private void checkGameOver() {  
 if (score >= 50) {  
 gameRunning = false;  
 timer.stop();  
 synchronized (circles) {  
 for (FallingCircle circle : circles) {  
 circle.stopFalling();  
 }  
 }  
 circles.clear();  
 JOptionPane.*showMessageDialog*(this, "Поздравляем! Вы набрали 50 очков!");  
 startButton.setVisible(true);  
 }  
 }  
  
 @Override  
 protected void paintComponent(Graphics g) {  
 super.paintComponent(g);  
 if (gameRunning) {  
 Graphics2D g2d = (Graphics2D) g;  
 g2d.setRenderingHint(RenderingHints.*KEY\_ANTIALIASING*, RenderingHints.*VALUE\_ANTIALIAS\_ON*);  
  
 synchronized (circles) {  
 for (FallingCircle circle : circles) {  
 g2d.setColor(Color.*BLUE*);  
 g2d.fillOval(circle.getX(), circle.getY(), 50, 50);  
  
 g2d.setColor(Color.*WHITE*);  
 g2d.setFont(new Font("Arial", Font.*BOLD*, 20));  
 g2d.drawString(String.*valueOf*(circle.getLetter()), circle.getX() + 18, circle.getY() + 32);  
 }  
 }  
  
 g2d.setColor(Color.*BLACK*);  
 g2d.setFont(new Font("Arial", Font.*BOLD*, 24));  
 g2d.drawString("Счет: " + score, 20, 40);  
 }  
 }  
  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 repaint();  
 }  
  
 public static void main(String[] args) {  
 JFrame frame = new JFrame("Тренажер клавиатуры");  
 KeyboardTrainer trainer = new KeyboardTrainer();  
 frame.add(trainer);  
 frame.pack();  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.setLocationRelativeTo(null);  
 frame.setVisible(true);  
 }  
}

ПРИЛОЖЕНИЕ Б

Код класса FallingCircle

import java.util.Random;  
  
public class FallingCircle implements Runnable {  
 private final char letter;  
 private int x, y;  
 private final int speed;  
 private final Random random = new Random();  
 private volatile boolean falling = true;  
 private final KeyboardTrainer trainer;  
  
 public FallingCircle(KeyboardTrainer trainer) {  
 this.trainer = trainer;  
 letter = (char) ('a' + random.nextInt(26));  
 x = random.nextInt(750);  
 y = 0;  
 speed = 2 + random.nextInt(3);  
 }  
  
 @Override  
 public void run() {  
 while (falling && y < 600) {  
 y += speed;  
 try {  
 Thread.sleep(20);  
 } catch (InterruptedException e) {  
 Thread.currentThread().interrupt();  
 }  
 }  
  
 if (falling) {  
 trainer.decreaseScore(); // Уменьшаем счет  
 trainer.removeCircle(this); // Удаляем текущий круг и создаем новый  
 }  
 }  
  
  
 public void stopFalling() {  
 falling = false;  
 }  
  
 public char getLetter() { return letter; }  
 public int getX() { return x; }  
 public int getY() { return y; }  
}