

Министерство науки и высшего образования РФ
ФГБОУ ВО «Омский государственный технический университет»
Кафедра «Автоматизированные системы обработки информации и
управления»

ОТЧЁТ

по Лабораторной работе №4

По дисциплине: Объектно-ориентированное программирование
студента Охтеня Валерия Алексеевича группы ПИН-212

Пояснительная записка

Шифр работы От-2068998-43-ПИН-212-2 ПЗ

Специальность 09.03.04

Старший преподаватель

А.А. Кабанов

Студент

В.А. Охтеня

Омск 2022

Задание

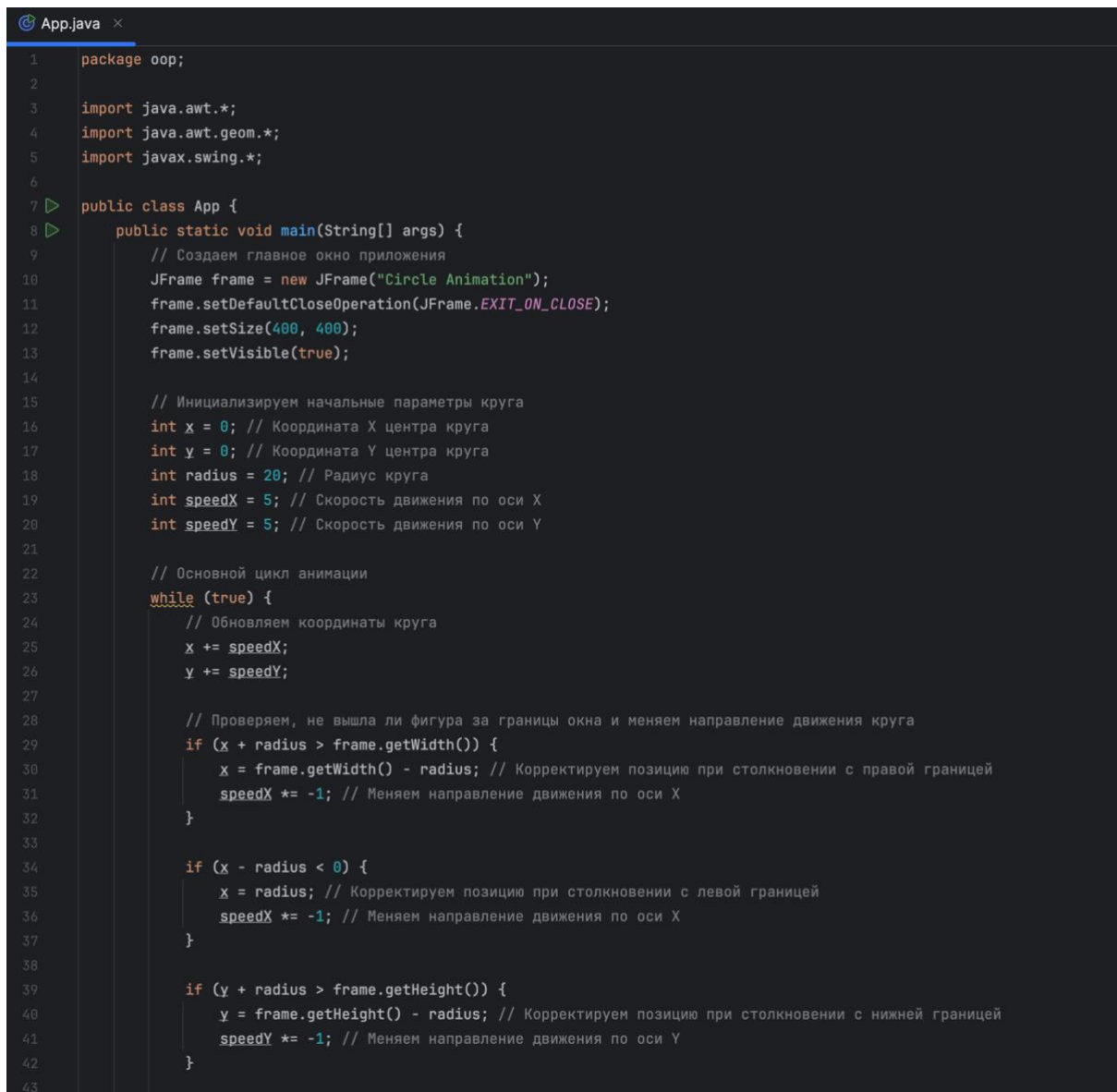
Задать движение окружности по апплету так, чтобы при касании границы окружность отражалась от нее с эффектом упругого сжатия.

Цель работы

Построить приложение с использованием графического интерфейса библиотек java.awt и javax.swing.

Ход работы

Написал код программы (рис 1, рис 2).



```
1 package oop;
2
3 import java.awt.*;
4 import java.awt.geom.*;
5 import javax.swing.*;
6
7 public class App {
8     public static void main(String[] args) {
9         // Создаем главное окно приложения
10        JFrame frame = new JFrame("Circle Animation");
11        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
12        frame.setSize(400, 400);
13        frame.setVisible(true);
14
15        // Инициализируем начальные параметры круга
16        int x = 0; // Координата X центра круга
17        int y = 0; // Координата Y центра круга
18        int radius = 20; // Радиус круга
19        int speedX = 5; // Скорость движения по оси X
20        int speedY = 5; // Скорость движения по оси Y
21
22        // Основной цикл анимации
23        while (true) {
24            // Обновляем координаты круга
25            x += speedX;
26            y += speedY;
27
28            // Проверяем, не вышла ли фигура за границы окна и меняем направление движения круга
29            if (x + radius > frame.getWidth()) {
30                x = frame.getWidth() - radius; // Корректируем позицию при столкновении с правой границей
31                speedX *= -1; // Меняем направление движения по оси X
32            }
33
34            if (x - radius < 0) {
35                x = radius; // Корректируем позицию при столкновении с левой границей
36                speedX *= -1; // Меняем направление движения по оси X
37            }
38
39            if (y + radius > frame.getHeight()) {
40                y = frame.getHeight() - radius; // Корректируем позицию при столкновении с нижней границей
41                speedY *= -1; // Меняем направление движения по оси Y
42            }
43        }
44    }
45 }
```

Рисунок 1 – Код программы (Часть 1)

```

44         if (y - radius < 0) {
45             y = radius; // Корректируем позицию при столкновении с верхней границей
46             speedY *= -1; // Меняем направление движения по оси Y
47         }
48
49         // Удаляем все из окна и добавляем в него измененный круг
50         frame.getContentPane().removeAll();
51         frame.getContentPane().add(new Circle(x, y, radius)); // Создаем круг с обновленными координатами/радиусом
52         frame.getContentPane().validate();
53         frame.getContentPane().repaint(); // Перерисовываем окно
54
55         try {
56             Thread.sleep(20); // Задержка для управления частотой кадров
57         } catch (InterruptedException e) {
58             e.printStackTrace();
59         }
60     }
61 }
62
63 public static class Circle extends JPanel {
64     private int x;
65     private int y;
66     private int radius;
67
68     // Конструктор класса Circle
69     public Circle(int x, int y, int radius) {
70         this.x = x; // Устанавливаем координату X
71         this.y = y; // Устанавливаем координату Y
72         this.radius = radius; // Устанавливаем радиус
73     }
74
75     @Override
76     public void paintComponent(Graphics g) {
77         super.paintComponent(g); // Вызываем метод родительского класса
78         Graphics2D g2d = (Graphics2D) g; // Преобразуем Graphics в Graphics2D
79         g2d.setColor(Color.BLUE); // Устанавливаем цвет круга
80         // Рисуем круг с заданными координатами и радиусом
81         g2d.fill(new Ellipse2D.Double(x - radius, y - radius,
82             2 * radius, 2 * radius));
83     }
84 }
85 }

```

Рисунок 2 – Код программы (Часть 2)

Далее запустил и протестировал программу (рис 3, рис 4, рис 5).

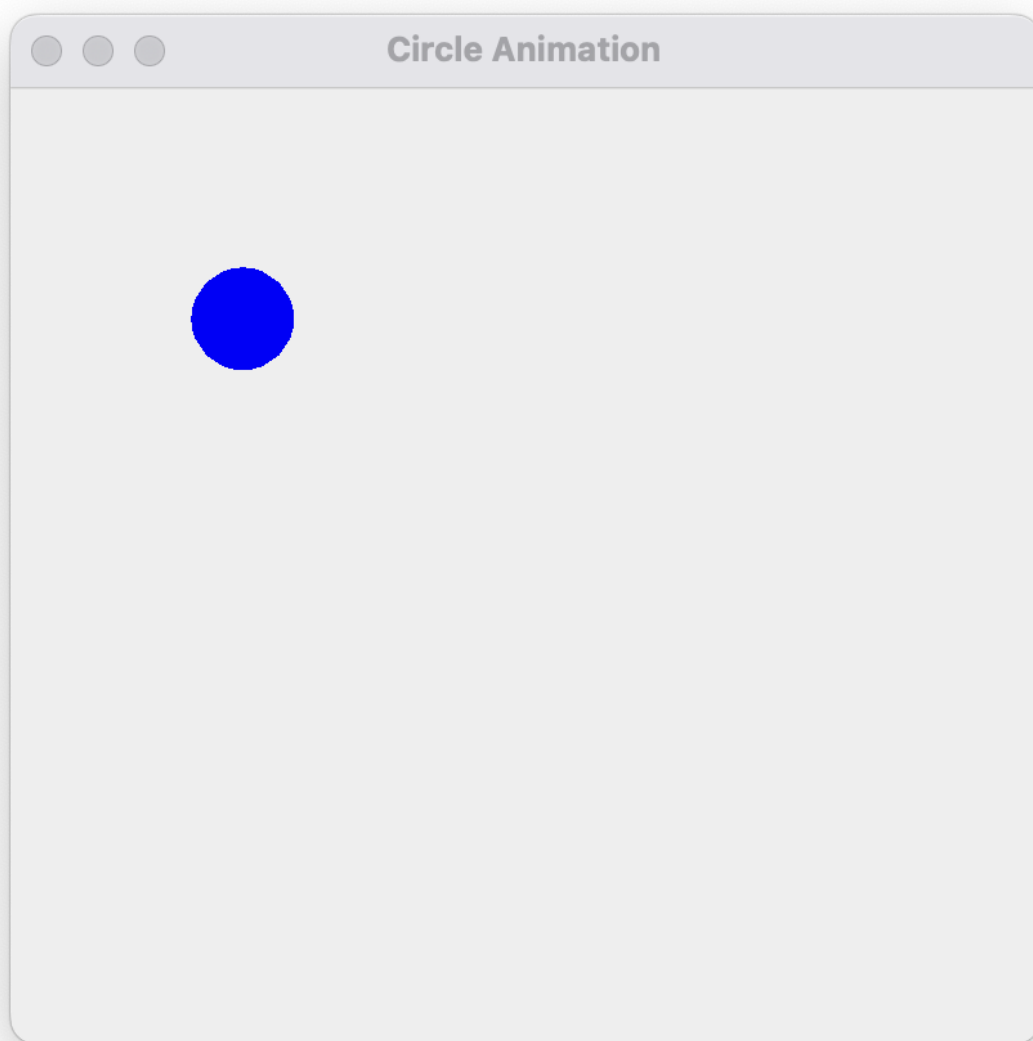


Рисунок 4 – Тестирование программы (Часть 1)

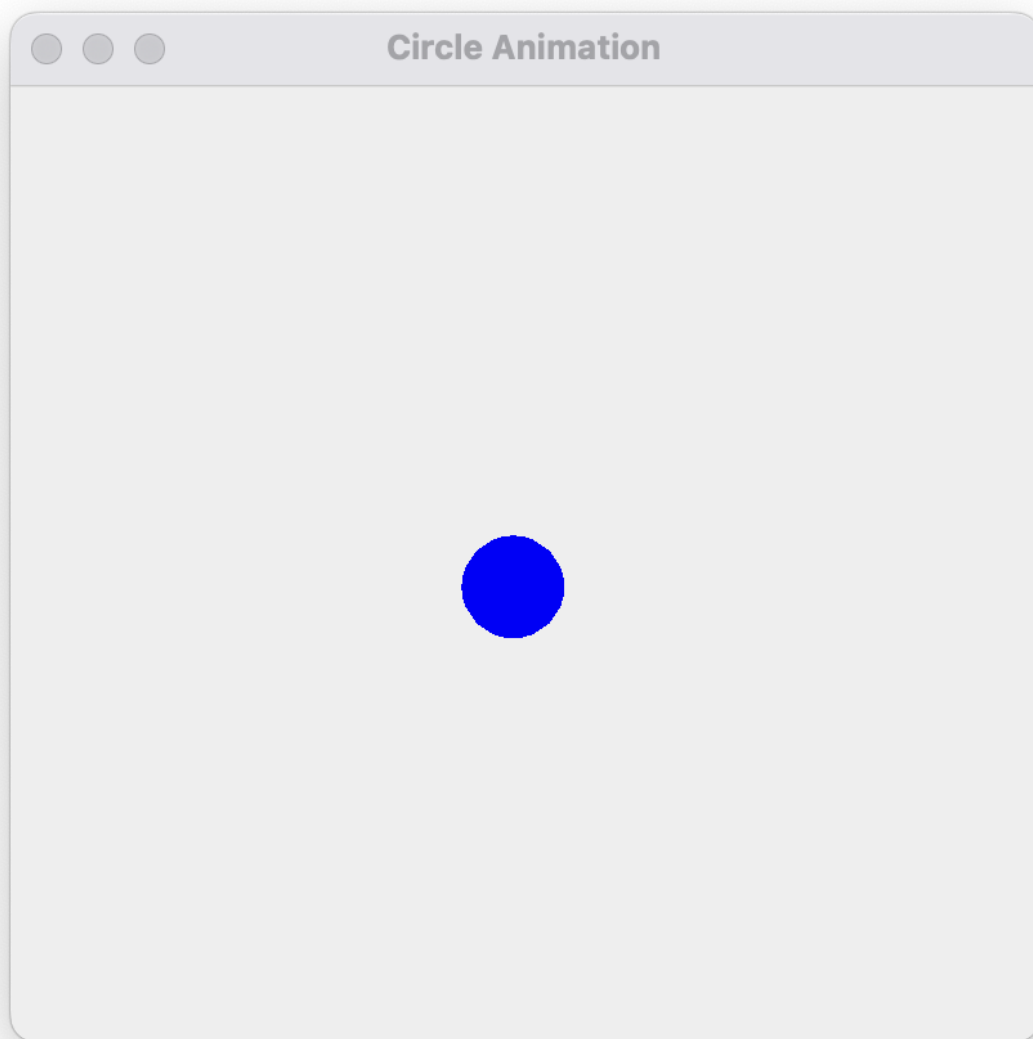


Рисунок 5 – Тестирование программы (Часть 2)

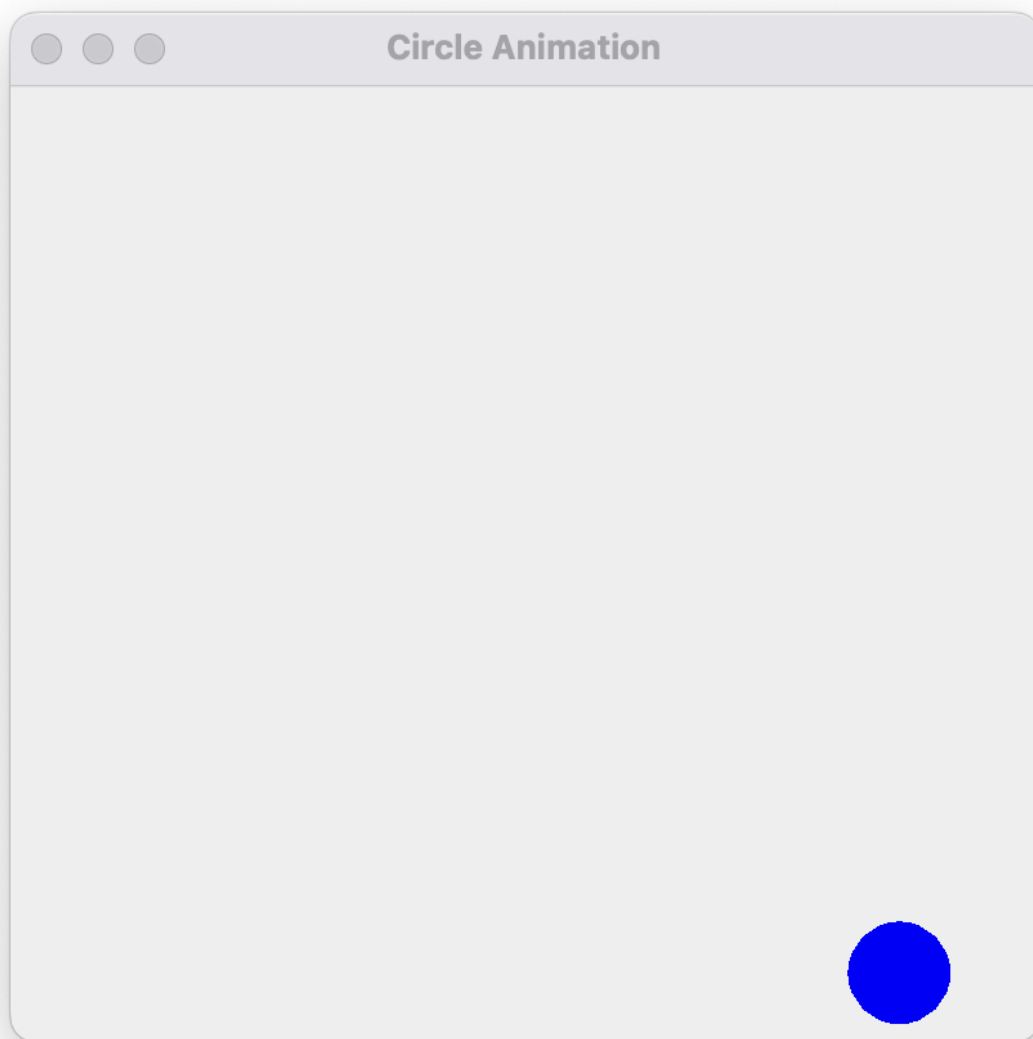


Рисунок 6 – Тестирование программы (Часть 3)

Заключение

В ходе лабораторной работы получила навыки по построению приложений с использованием графического интерфейса библиотек `java.awt` и `javax.swing`.