```
package inputs;

import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;

import main.Game;
import main.GamePanel;

public class KeyboardInputs implements KeyListener {

	private GamePanel gamePanel;

	public KeyboardInputs(GamePanel gamePanel) {
		this.gamePanel = gamePanel;
	}

	@Override
	public void keyTyped(KeyEvent e) {
		// TODO Auto-generated method stub

	}

	@Override
	public void keyReleased(KeyEvent e) {
		// TODO Auto-generated method stub

	}

	@Override
	public void keyPressed(KeyEvent e) {
```

```
            switch (e.getKeyCode()) {
            case KeyEvent.VK_W:
                  gamePanel.changeYDelta(-5);
                  break;
            case KeyEvent.VK_A:
                  gamePanel.changeXDelta(-5);
                  break;
            case KeyEvent.VK_S:
                  gamePanel.changeYDelta(5);
                  break;
            case KeyEvent.VK_D:
                  gamePanel.changeXDelta(5);
                  break;
            }

      }

}
```

```
package inputs;

import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.MouseMotionListener;

import main.GamePanel;

public class MouseInputs implements MouseListener, MouseMotionListener {
```

```
	private GamePanel gamePanel;

	public MouseInputs(GamePanel gamePanel) {
		this.gamePanel = gamePanel;
	}

	@Override
	public void mouseDragged(MouseEvent e) {
		// TODO Auto-generated method stub

	}

	@Override
	public void mouseMoved(MouseEvent e) {
//		gamePanel.setRectPos(e.getX(), e.getY());

	}

	@Override
	public void mouseClicked(MouseEvent e) {
//		System.out.println("Mouse clicked!");
		gamePanel.spawnRect(e.getX(),e.getY());

	}

	@Override
	public void mousePressed(MouseEvent e) {
		// TODO Auto-generated method stub

	}
```

```
	@Override
	public void mouseReleased(MouseEvent e) {
		// TODO Auto-generated method stub

	}

	@Override
	public void mouseEntered(MouseEvent e) {
		// TODO Auto-generated method stub

	}

	@Override
	public void mouseExited(MouseEvent e) {
		// TODO Auto-generated method stub

	}

}
```

```
package main;

public class Game implements Runnable {

	private GameWindow gameWindow;
	private GamePanel gamePanel;
	private Thread gameThread;
	private final int FPS_SET = 120;
```

```
public Game() {

	gamePanel = new GamePanel();
	gameWindow = new GameWindow(gamePanel);
	gamePanel.requestFocus();
	startGameLoop();

}

private void startGameLoop() {
	gameThread = new Thread(this);
	gameThread.start();
}

@Override
public void run() {

	double timePerFrame = 1000000000.0 / FPS_SET;
	long lastFrame = System.nanoTime();
	long now = System.nanoTime();

	int frames = 0;
	long lastCheck = System.currentTimeMillis();

	while (true) {

		now = System.nanoTime();
		if (now - lastFrame >= timePerFrame) {
			gamePanel.repaint();
```

```
                        lastFrame = now;
                        frames++;
                    }

                    if (System.currentTimeMillis() - lastCheck >= 1000) {
                        lastCheck = System.currentTimeMillis();
                        System.out.println("FPS: " + frames);
                        frames = 0;
                    }
                }

            }

}

package main;

import java.awt.Color;
import java.awt.Graphics;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.util.ArrayList;
import java.util.Random;

import javax.swing.JPanel;

import inputs.KeyboardInputs;
import inputs.MouseInputs;

public class GamePanel extends JPanel {
```

```
private MouseInputs mouseInputs;
private float xDelta = 100, yDelta = 100;
private float xDir = 1f, yDir = 1f;
private Color color = new Color(150, 20, 90);
private Random random;

// Temp
private ArrayList<MyRect> rects = new ArrayList<>();

public GamePanel() {
	random = new Random();
	mouseInputs = new MouseInputs(this);
	addKeyListener(new KeyboardInputs(this));
	addMouseListener(mouseInputs);
	addMouseMotionListener(mouseInputs);

}

public void changeXDelta(int value) {
	this.xDelta += value;

}

public void changeYDelta(int value) {
	this.yDelta += value;

}

public void setRectPos(int x, int y) {
```

```
		this.xDelta = x;
		this.yDelta = y;
	}

	public void spawnRect(int x, int y) {
		rects.add(new MyRect(x, y));
	}

	public void paintComponent(Graphics g) {
		super.paintComponent(g);

		// Temp Rects
		for (MyRect rect : rects) {
			rect.updateRect();
			rect.draw(g);
		}

		updateRectangle();

		g.setColor(color);
		g.fillRect((int) xDelta, (int) yDelta, 200, 50);

	}

	private void updateRectangle() {
		xDelta += xDir;
		if (xDelta > 400 || xDelta < 0) {
			xDir *= -1;
			color = getRndColor();
		}
```

```
	yDelta += yDir;
	if (yDelta > 400 || yDelta < 0) {
		yDir *= -1;
		color = getRndColor();
	}
}

private Color getRndColor() {
	int r = random.nextInt(255);
	int g = random.nextInt(255);
	int b = random.nextInt(255);

	return new Color(r, g, b);
}

// Temp
public class MyRect {
	int x, y, w, h;
	int xDir = 1, yDir = 1;
	Color color;

	public MyRect(int x, int y) {
		this.x = x;
		this.y = y;
		w = random.nextInt(50);
		h = w;
		color = newColor();
	}
```

```
	public void updateRect() {
		this.x += xDir;
		this.y += yDir;

		if ((x + w) > 400 || x < 0) {
			xDir *= -1;
			color = newColor();
		}
		if ((y + h) > 400 || y < 0) {
			yDir *= -1;
			color = newColor();
		}

	}

	private Color newColor() {
		return new Color(random.nextInt(255), random.nextInt(255),
random.nextInt(255));
	}

	public void draw(Graphics g) {
		g.setColor(color);
		g.fillRect(x, y, w, h);
	}

}

}

package main;
```

```
import javax.swing.JFrame;

public class GameWindow {
	private JFrame jframe;

	public GameWindow(GamePanel gamePanel) {

		jframe = new JFrame();

		jframe.setSize(400, 400);
		jframe.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		jframe.add(gamePanel);
		jframe.setLocationRelativeTo(null);
		jframe.setVisible(true);

	}

}
```

```
package main;

public class MainClass {

	public static void main(String[] args) {
		new Game();

	}

}
```