/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package javafxapplication2;

import java.util.ArrayList;

import javafx.application.Application;

import javafx.event.EventHandler;

import javafx.scene.Group;

import javafx.scene.Scene;

import javafx.scene.canvas.Canvas;

import javafx.scene.canvas.GraphicsContext;

import javafx.scene.image.Image;

import javafx.scene.input.KeyEvent;

import javafx.scene.input.MouseEvent;

import javafx.scene.paint.Color;

import javafx.stage.Stage;

/\*\*

\*

\* @author SUGIARTO COKROWIBOWO

\*/

public class JavaFXApplication2 extends Application implements Runnable {

//Loop Parameters

private final static int MAX\_FPS = 60;

private final static int MAX\_FRAME\_SKIPS = 5;

private final static int FRAME\_PERIOD = 1000 / MAX\_FPS;

//Thread

private Thread thread;

private volatile boolean running = true;

//Canvas

Canvas canvas = new Canvas(1024, 700);

//KEYBOARD HANDLER

ArrayList<String> inputKeyboard = new ArrayList<String>();

//ATRIBUT KOTAK

float sisi = 100f;

float sudutRotasi = 0f;

float cx = 100;

float cy = 0;

//ATRIBUT GJB

float g = 0.1f;

float t = 0f;

float v = 0f;

float vUP = 10f;

private int yo;

public JavaFXApplication2() {

resume();

}

@Override

public void start(Stage primaryStage) {

Group root = new Group();

Scene scene = new Scene(root);

root.getChildren().add(canvas);

//HANDLING KEYBOARD EVENT

scene.setOnKeyPressed(

new EventHandler<KeyEvent>() {

public void handle(KeyEvent e) {

String code = e.getCode().toString();

if (!inputKeyboard.contains(code)) {

inputKeyboard.add(code);

System.out.println(code);

}

}

}

);

scene.setOnKeyReleased(

new EventHandler<KeyEvent>() {

public void handle(KeyEvent e) {

String code = e.getCode().toString();

inputKeyboard.remove(code);

}

}

);

//HANDLING MOUSE EVENT

scene.setOnMouseClicked(

new EventHandler<MouseEvent>() {

public void handle(MouseEvent e) {

}

}

);

//primaryStage.getIcons().add(new Image(getClass().getResourceAsStream("logo.jpg")));

primaryStage.setTitle("Visual Loop");

primaryStage.setScene(scene);

primaryStage.show();

}

public static void main(String[] args) {

launch(args);

}

//THREAD

private void resume() {

reset();

thread = new Thread(this);

running = true;

thread.start();

}

//THREAD

private void pause() {

running = false;

try {

thread.join();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

//LOOP

private void reset() {

}

//LOOP

private void update() {

if(inputKeyboard.contains("RIGHT")) {

cx+=2;//menggerakkan kotak ke kanan saat key cx+2=2;

}else if (inputKeyboard.contains("LEFT")) {

cx-=2;//menggerakkan kotak ke kiri saat cx-=2;

}

if (inputKeyboard.contains("UP")) {

cy-= 2;//menggerakkan kotak ke atas saat key UP cy-=2;

}else if (inputKeyboard.contains("DOWN")) {

cy+=2;//menggerakkan kotak ke bawah saat cy+=2;

}

if (inputKeyboard.contains("R")) {

sudutRotasi+= 2;//merotasi kotak se arah gerakan jarum jam sudutRotasi+=2;

}

//JATUH BEBAS

if (cy<canvas.getHeight() -0.5\*sisi){

t++;

v = g\*t;

cy += v;

}

if (inputKeyboard.contains("SPACE")){

t = 0;

cy-=vUP;

}

}

//LOOP

private void draw() {

try {

if (canvas != null) {

GraphicsContext gc = canvas.getGraphicsContext2D();

gc.clearRect(0, 0, canvas.getWidth(), canvas.getHeight());

//CONTOH MENGGAMBAR KOTAK YANG DAPAT DITRANSLASI DAN DI ROTASI

gc.save();

gc.translate(cx, cy);

gc.rotate(sudutRotasi);

gc.setFill(Color.PURPLE);

gc.fillRect(-sisi / 2.0f, -sisi / 2.0f, sisi, sisi);

gc.restore();

}

} catch (Exception e) {

e.printStackTrace();

}

}

@Override

public void run() {

long beginTime;

long timeDiff;

int sleepTime = 0;

int framesSkipped;

//LOOP WHILE running = true;

while (running) {

try {

synchronized (this) {

beginTime = System.currentTimeMillis();

framesSkipped = 0;

update();

draw();

}

timeDiff = System.currentTimeMillis() - beginTime;

sleepTime = (int) (FRAME\_PERIOD - timeDiff);

if (sleepTime > 0) {

try {

Thread.sleep(sleepTime);

} catch (InterruptedException e) {

}

}

while (sleepTime < 0 && framesSkipped < MAX\_FRAME\_SKIPS) {

update();

sleepTime += FRAME\_PERIOD;

framesSkipped++;

}

} finally {

}

}

}

}