package visualloop;

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 VisualLoop 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>();

//attribut kotak

float xo = 100;

float yo = 10;

float velocity = 2f;

float sudutRotasi = 0f;

float sisi = 100f;

float size = 100f;

//Simulasi Gerak jatuh bebas

float g = 0.1f; //Percepatan grafitasi

float t = 0f; //Waktu

float v = 0f; //Kecepatan jatuh

float vUP = 10f;

public VisualLoop(){

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.print(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() {

//GERAKAN HORIZONTAL

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

xo+=velocity;//saat key RIGHT di Keyboard ditekan makan Kotak akan bergerak ke kanan dengan kecepatan sebesar velocity

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

xo-=velocity;//kotak bergerak ke kiri secepat velocity saat key LEFT di tekan

}

//GERAKAN VERTICAL

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

yo-=velocity;//Kotak bergerak keatas saat key UP ditekan

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

yo+=velocity;//kotak Bergerak kebawah saat key DOWN ditekan

}

//UPDATE VELOCITY

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

velocity++;//Key Z untuk menambah kecepatan

}else if(inputKeyboard.contains("X")&&velocity>0){

velocity--;

}

//ROTASI

if(inputKeyboard.contains("R")){//Merotasi Kotak se arah gerakan jarum jam saat Key R ditekan

sudutRotasi+=2;

}

// JATUH

// v = gt

if (yo<canvas.getHeight()-0.5f\*size){

t++;

v = g\*t;

yo+=v;

}

if(inputKeyboard.contains("SPACE")&&yo>0){

yo-=vUP; // Melompat dengan kecepatan grafitasi vUP

t = 0;

}

}

//LOOP

private void draw() {

try {

if (canvas != null) {

GraphicsContext gc = canvas.getGraphicsContext2D();

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

//CONTOH MENGGAMBAR KOTAK

gc.save();

gc.translate(xo, yo);

gc.rotate(sudutRotasi);

gc.setFill(Color.CRIMSON);

gc.fillRect(-sisi\*0.5f, -sisi\*0.5f, 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 {

}

}

}

}