**classpage:Snake.jave**

**Coding:**

package snake\_game;

import javax.swing.\*;

import java.awt.event.\*;

import java.awt.event.ActionListener;

import java.awt.event.ActionEvent;

import java.awt.\*;

import java.util.\*;

public class snake extends JFrame implements KeyListener,Runnable{

String Name;

JPanel p1, p2;

JButton[] lb = new JButton[200];

JButton bounsfood;

JTextArea t;

int x = 1000, y = 500, gu = 2, directionx = 1, directiony = 0, speed = 100, differences = 0, oldx, oldy, score = 0;

int[] lbx = new int[600];

int[] lby = new int[600];

Point[] lbp = new Point[600];

Point bfp = new Point();

Thread myt;

boolean food = false, run1 = false, runr = true, runu = true, rund = true, bounsflag = true;

Random r = new Random();

JMenuBar mybar;

JMenu game, help, level;

public void initializeVlaues() {

gu = 3;

lbx[0] = 200;

lby[0] = 300;

directionx = 10;

directiony = 0;

score = 0;

food = false;

run1 = false;

runr = true;

runu = true;

rund = true;

bounsflag = true;

}

snake() {

super("snake");

setSize(1000, 590);

//create menubar with function

createbar();

//initialize all variable

initializeVlaues();

p1 = new JPanel();

p2 = new JPanel();

//t will view the score

t = new JTextArea("Score==>" + score);

t.setEnabled(false);

t.setBackground(Color.BLACK);

//snake have to eat bouncefood to frowup

bounsfood = new JButton();

bounsfood.setEnabled(false);

//will make first snake

createFirstSnake();

p1.setLayout(null);

p2.setLayout(new GridLayout(0, 1));

p1.setBounds(0, 0, x, y);

p1.setBackground(Color.blue);

p2.setBounds(0, y, x, 30);

p2.setBackground(Color.RED);

p2.add(t);

getContentPane().setLayout(null);

getContentPane().add(p1);

getContentPane().add(p2);

show();

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

addKeyListener(this);

//start thread

myt = new Thread(this);

myt.start(); //go to run () method

}

public void createFirstSnake() {

//initially the snake has small length 3

for (int i = 0; i < 3; i++) {

lb[i] = new JButton("1b" + i);

lb[i].setEnabled(false);

p1.add(lb[i]);

lb[i].setBounds(lbx[i], lby[i], 10, 10);

lbx[i + 1] = lbx[i] - 10;

lby[i + 1] = lby[i];

}

}

public void createbar() {

mybar = new JMenuBar();

game = new JMenu("Game");

JMenuItem newgame = new JMenuItem("new game");

JMenuItem exit = new JMenuItem("Exit");

newgame.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

reset();

}

});

exit.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

System.exit(0);

}

});

game.add(newgame);

game.addSeparator();

game.add(exit);

mybar.add(game);

level = new JMenu("Level");

help = new JMenu("Help");

JMenuItem creator = new JMenuItem("creator");

JMenuItem instruction = new JMenuItem("instruction");

creator.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

JOptionPane.showMessageDialog(p2, "Name" + ":Group Work");

}

});

help.add(creator);

help.add(instruction);

mybar.add(help);

setJMenuBar(mybar);

}

void reset() {

initializeVlaues();

p1.removeAll();

myt.stop();

createFirstSnake();

t.setText("Score==>"+score);

myt = new Thread(this);

myt.start();

}

void growup() {

lb[gu] = new JButton();

lb[gu].setEnabled(false);

p1.add(lb[gu]);

int a = 10 + (10 \* r.nextInt(48));

int b = 10 + (10 \* r.nextInt(23));

lbx[gu] = a;

lby[gu] = b;

lb[gu].setBounds(a, b, 10, 10);

gu++;

}

void moveForward() {

for (int i = 0; i < gu; i++) {

lbp[i] = lb[i].getLocation();

}

lbx[0] += directionx;

lby[0] += directiony;

lb[0].setBounds(lbx[0],lby[0],10,10);

for (int i = 1; i < gu; i++) {

lb[i].setLocation(lbp[i - 1]);

}

if (lbx[0] == x){

lbx[0] = 10;

}

else if (lbx[0] == 0){

lbx[0] = x - 10;}

else if (lby[0] == y){

lby[0] = 10;

}

else if (lby[0] == 0){

lby[0] = y - 10;

}

if (lbx[0] == lbx[gu - 1] && lby[0] == lby[gu - 1])

{

food = false;

score += 5;

t.setText("Score==>"+score);

if (score % 50 == 0 && bounsflag == true) {

p1.add(bounsfood);

bounsfood.setBounds((10 \* r.nextInt(50)), (10 \* r.nextInt(25)), 15, 15);

bfp = bounsfood.getLocation();

bounsflag = false;

}

}

if(bounsflag==false){

if(bfp.x <= lbx[0] && bfp.y <=lby[0] && bfp.x+10>=lbx[0]&&bfp.y+10>=lby[0]){

p1.remove(bounsfood);

score += 100;

t.setText("Score ==> " + score);

bounsflag = true;

}

}

if (food==false){

growup();

food=true;

}

else{

lb [gu-1].setBounds(lbx[gu-1],lby[gu-1],10,10);

}

for (int i = 1; i <gu; i++) {

if (lbp[0]==lbp[i]){

t.setText("Game Over -"+score);

try{

myt.join();

}

catch(InterruptedException ie){}

break;

}

}

p1.repaint();

show();

}

public void KeyPressed(KeyEvent e){

//left wnen player pressed left arrow

if (run1 ==true && e.getKeyCode()==37){

directionx =- 10;

directiony = 0;

runr=false;

runu=true;

rund=true;

}

//up when up arrow

if(runu==true && e.getKeyCode()==38){

directionx=0;

directiony =- 10;

rund=false;

runr=true;

run1=true;

}

//right when right arrow

if(runr == true && e.getKeyCode()==39){

directionx =+ 10;

directiony = 0;

run1=false;

runu=true;

rund=true;

}

//down when down arrow

if(rund==true && e.getKeyCode() == 40){

directionx = 0;

directiony =+ 10;

runu=false;

runr=true;

run1=true;

}

}

public void KeyReleased(KeyEvent e){

}

public void KeyTyped(KeyEvent e){

}

public void run(){

for(;;){

moveForward();

try{

Thread.sleep(speed);

}

catch(InterruptedException ie){

}

}

}

@Override

public void keyTyped(KeyEvent e) {

throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody

}

@Override

public void keyPressed(KeyEvent e) {

throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody

}

@Override

public void keyReleased(KeyEvent e) {

throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody

}

}