|  |
| --- |
| BigBasti |
| Sneaky game |
| 10/08/2014 |

|  |
| --- |
| Source: http://blog.bigbasti.com/java-snake-nachprogrammieren-demo/  Big Basti |

Game

**package** game;

**import** java.awt.\*;

**public** **class** Game {

**public** **static** Logwriter *log* = **new** Logwriter();

**public** **static** Snake *snake*;

**public** **static** WinMain *frmMain*;

**public** **static** Point *aim*;

**public** **static** **double** *playtime* = 0;

**public** **static** **int** *punkte* = 0;

**public** **static** **void** main(String[] args){

*log*.write("Spiel gestartet...");

*snake* = **new** Snake();

*frmMain* = **new** WinMain("Sneaky by Sebastian Gross");

*log*.write("Hauptfenster erstellt...");

*newAim*();

*runGame*();

}

**public** **static** **void** newAim(){

Point p;

//Einen neuen Punkt auf dem Spielfeld erzeugen

**while**(**true**){

p = **new** Point((**int**)((Math.*random*()\*20)+0.5),(**int**)((Math.*random*()\*20)+0.5));

//Darauf achten, dass das Zielfeld innerhalb des Spielfeldes liegt!

//x & y muss größer als 0 sein!

**if**(p.x >= 1 && p.y >= 1){

**break**;

}

}

*punkte* += 100; //Für das erreichen des Zielfelds 100 Punkte geben

*aim* = p;

}

**public** **static** **void** runGame(){

**int** secs = 0; //Variable um Zeit zu messen

**while**(**true**){

//Eine eunendliche Schleife starten

//die alle 100 Millisekunden (Standard) das Bild Aktualisiert und

//Die Schlange um ein Kästchen weiter bewegt

//frmMain.setTitle("S:"+snake.pos.x+snake.pos.y+" - "+aim.x+aim.y);

**if**(*snake*.moveSnake() == **false**){

//Wenn Speil zu Ende ist Programm anhalten

**break**;

}

*frmMain*.pG.paintComponent(*frmMain*.pG.getGraphics());

//Spielzeit aktualisieren (optnal)

//Muss angepasst werden wenn, die untere sleep() Einstellung geändert wird!

secs++;

**if** (secs == 10){

//Eine Sekunde vergangen

*playtime*++;

//Zähler zurücksetzen

secs = 0;

*punkte*++;

}

**try**{

Thread.*sleep*(100); //Standardeinstellung: 100

}**catch**(Exception ex){*log*.write(ex.getMessage());}

}

}

}

Logwriter

**package** game;

//Diese Klasse kümmert sich nur um die Ausgabe von Statusinformationen

//auf der Konsole zur Kontrolle und Debugging!

**public** **class** Logwriter {

**public** **void** write(String msg){

System.***out***.println(msg);

}

}

PointA

**package** game;

**import** java.awt.\*;

//Diese Klasse hat ein "A" am Ende, da der name "Point" reserviert ist!

**public** **class** PointA {

**public** Point pos;

**public** **boolean** head; //Gibt an, ob dieser Punkt der Kopf der Schlange ist

**public** **boolean** tail; //Gibt an, ob dieser Punkt das Ende der schlange ist

**private** Logwriter log = **new** Logwriter();

**public** PointA(Point p, **boolean** head, **boolean** tail){

**this**.pos = p;

**this**.head = head;

**this**.tail = tail;

log.write("Neuen Punkt erstellt...");

}

**public** Point getPos(){

**return** pos;

}

**public** **void** setPoint(Point p){

**this**.pos = p;

}

**public** **void** paintPoint(Graphics g){

**if** (head == **true**){

g.setColor(Color.***red***); //Den Kopf der Schlage Rot färben

}**else**{

g.setColor(Color.***black***);

}

g.fillRect((pos.x+1)\*5, (pos.y+1)\*5, 5, 5);

//log.write("Punkt zeichnen...");

}

}

Snake

**package** game;

**import** java.awt.\*;

**import** java.util.\*;

**public** **class** Snake {

**public** Point pos; //Aktuelle Position des Schlangen Kopfes

**public** String dir; //Die Richtung in die die schlage sich momentan bewegt (left, right, top, bottom)

**public** **boolean** newPoint = **false**;

**public** LinkedList<PointA> points = **new** LinkedList<PointA>(); //Liste mit allen Schlangenpunkten

**private** Logwriter log = **new** Logwriter();

**public** Snake(){

//Wenn eine Schlange neu erstellt wird hat sie ncoh keine Points, d.h. points ist leer

//Damit der benutzer überhaupt was sieht werden gleich 2 Punkte erzeugt

//wobei einer von den 2 Punkten als Kopf markiert wird!

//Die Schlange startet in der linken oberen ecke an Position 2,1 => wobei 2,1 der kopf und 1,1 der Körper ist

//0,0 - 0,[spielfeldlänge] ist für den Rand reserviert!

//Die anfangs laufrichtung ist dementsprechend rechts

pos = **new** Point(3,1); //Anfangsposition des Schlangenkopfes

dir = "right"; //Anfangsrichtung

//Den anfags-schlangen-körper erstellen, bestehend aus 3 Punkten Kopf-[nichtkopf & nichtschwanz]-Schwanz

points.add(**new** PointA(**new** Point(3,1),**true**,**false**)); //Schlangenkopf erstellen

points.add(**new** PointA(**new** Point(2,1),**false**,**false**)); //Schlangenkoerper erstellen

points.add(**new** PointA(**new** Point(1,1),**false**,**true**)); //Schlangenschwanz erstellen

log.write("Schlange erstellt...");

}

**public** **void** addNewPoint(PointA p){

points.add(p);

}

**public** **void** paintSneak(Graphics g){

//log.write("Punkte neu zeichnen...");

**for**(PointA p : points){ //Alle Points der Schlange neu zeichnen

p.paintPoint(g);

}

}

**public** **void** setDir(String dir){

//Prüfen ob in die Entgegengesetzte Richtung gewechselt werden soll -> geht ja nit

//Diese IF-Kette auskommentieren um das spiel um eine modi zu erweitern: instant-wenden

**if**(**this**.dir.equals("right") && dir.equals("left")){

log.write("Richtung änder ist nicht möglich...");

**return**;

}**else** **if**(**this**.dir.equals("left") && dir.equals("right")){

log.write("Richtung änder ist nicht möglich...");

**return**;

}**else** **if**(**this**.dir.equals("top") && dir.equals("bottom")){

log.write("Richtung änder ist nicht möglich...");

**return**;

}**else** **if**(**this**.dir.equals("bottom") && dir.equals("top")){

log.write("Richtung änder ist nicht möglich...");

**return**;

}

log.write("Laufrichtung auf "+dir+" ändern...");

**this**.dir = dir; //Laufrichtung der Schlage ändern

}

**public** **boolean** moveSnake(){

//Diese Funktion gibt true zurück wenn die Schlange weiterrücken konnte

//Oder false wenn die schlange gegen die Wand oder gegen Sich selbst gefahren ist!

/\*

\* Mir sind für diesen Vorgang 2 Vorgehensweisen eingefallen:

\* 1:

\* Wenn die Schlange weiterfahren soll nimmt man einfach jeden Punkt und

\* ersetzt dessen koordinaten mit denen des Vordermanns

\* O--X--X--X--X--X--X <= Das soll die schlange sein

\* ^<|^<|^<|^<|^<|^<|^<| => Alles rückt einen Schritt weiter

\*

\* 2:

\* Oder man nimmt den Letzten point und setzt ihn an den Anfang der Schlange:

\* O--X--X--X--X--X--X <= Das soll die schlange sein

\* ^------------------<| => Nur der Letzte point wird bewegt!

\*

\* Ich habe mich an dieser Stelle mal für die zweite Variante entschieden,

\* da ich denke dass diese flüssiger ist und Recourcensparender!

\*/

//Prüfen, ob die Schlange sich in die angegebene Richtung bewegen kann

Point nPos = **null**;

**if**(**this**.dir.equals("left")){

nPos = **new** Point(**this**.pos.x-1, **this**.pos.y);

}**else** **if**(**this**.dir.equals("top")){

nPos = **new** Point(**this**.pos.x, **this**.pos.y-1);

}**else** **if**(**this**.dir.equals("bottom")){

nPos = **new** Point(**this**.pos.x, **this**.pos.y+1);

}**else** **if**(**this**.dir.equals("right")){

nPos = **new** Point(**this**.pos.x+1, **this**.pos.y);

}

**if**(nPos.x <= 0 || nPos.x >= 40 || nPos.y <= 0 || nPos.y >= 40){ //Wenn Spielfeldrand erreicht ist

**return** **false**;

}**else**{

**for**(PointA p : points){

**if**(p.pos.equals(nPos)){ //Wenn Schlange sich selbst berührt

log.write("Wand berührt, Speil zu Ende...");

**return** **false**;

}

}

//Wenn noch Platz ist:

//Den aktuellen Kopf als "nicht-kopf" markieren

**for**(PointA p : points){

**if**(p.head == **true**){

p.head = **false**;

}

}

//und den letzten Punkt vor den Kopf setzen

**for**(PointA p : points){

**if**(p.tail == **true**){

p.tail = **false**;

p.head = **true**;

p.setPoint(nPos);

PointA nP = p;

points.remove(p);

points.add(0, nP);

}

}

**this**.pos = nPos;

points.get(points.size()-1).tail = **true**;

}

//Prüfen, ob das "Futter" gefunden wurde und die schlange erweitern

**if**(nPos.equals(Game.*aim*)){

log.write("Futter gefunden, wachsen...");

//Gucken in welche Richtung die Schlange grade läuft

**if**(**this**.dir.equals("left")){

**for**(PointA p : points){

//Den Kopf der Schlange um eine Stelle in Laufrichtung verschieben

**if**(p.head == **true**){

p.setPoint(**new** Point(**this**.pos.x-1,**this**.pos.y));

}

}

//Und den neuen Punkt hinter den Kopf hängen

**this**.pos = **new** Point(**this**.pos.x-1,**this**.pos.y);

points.add(1,**new** PointA(**new** Point(0,0),**false**,**false**));

}**else** **if**(**this**.dir.equals("top")){

**for**(PointA p : points){

**if**(p.head == **true**){

p.setPoint(**new** Point(**this**.pos.x,**this**.pos.y-1));

}

}

**this**.pos = **new** Point(**this**.pos.x,**this**.pos.y-1);

points.add(1,**new** PointA(**new** Point(0,0),**false**,**false**));

}**else** **if**(**this**.dir.equals("bottom")){

**for**(PointA p : points){

**if**(p.head == **true**){

p.setPoint(**new** Point(**this**.pos.x,**this**.pos.y+1));

}

}

**this**.pos = **new** Point(**this**.pos.x,**this**.pos.y+1);

points.add(1,**new** PointA(**new** Point(0,0),**false**,**false**));

}**else** **if**(**this**.dir.equals("right")){

**for**(PointA p : points){

**if**(p.head == **true**){

p.setPoint(**new** Point(**this**.pos.x+1,**this**.pos.y));

}

}

**this**.pos = **new** Point(**this**.pos.x+1,**this**.pos.y);

points.add(1,**new** PointA(**new** Point(0,0),**false**,**false**));

}

//Nenen Zielpunkt setzen

Game.*newAim*();

}

**return** **true**;

}

}

WinMain

**package** game;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**public** **class** WinMain **extends** JFrame{

**public** gPanel pG; //Panel für den Spielablauf

**public** JPanel pE; //Panel für das Ende des Spiels

**public** WinMain(String title){

//Die Vrbereitungen für das Hauptfenster treffen

Game.*log*.write("Hauptfenster erstellen...");

**this**.setTitle(title);

**this**.setSize(230,270); //Eigentlich müsste es 200,200 sein aber leider passen diese Angaben bei Windows nciht pixel genau und es wird ein Teil abgeschnitten!

**this**.setLayout(**null**);

**this**.setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

//Panels initialisieren

Game.*log*.write("Spielpanel erstellen...");

**this**.pG = **new** gPanel();

**this**.pG.setBounds(0, 0, 250, 250);

**this**.pG.setBackground(Color.***black***);

//Tasten-Events abfangen

**this**.addKeyListener(**new** KeyListener(){

//@Override

**public** **void** keyPressed(KeyEvent e) {}

//@Override

**public** **void** keyReleased(KeyEvent e) {

//Auf Tastendruck reagieren

**if**(e.getKeyCode() == KeyEvent.***VK\_RIGHT***){

Game.*snake*.setDir("right");

}**else** **if**(e.getKeyCode() == KeyEvent.***VK\_LEFT***){

Game.*snake*.setDir("left");

}**else** **if**(e.getKeyCode() == KeyEvent.***VK\_UP***){

Game.*snake*.setDir("top");

}**else** **if**(e.getKeyCode() == KeyEvent.***VK\_DOWN***){

Game.*snake*.setDir("bottom");

}

}

//@Override

**public** **void** keyTyped(KeyEvent e) {}

});

**this**.add(pG);

**this**.setVisible(**true**);

}

**public** **class** gPanel **extends** JPanel{

@Override **public** **void** paintComponent(Graphics gr){

//Game.log.write("Spielpanel zeichnen...");

Graphics2D g2d = (Graphics2D) gr;

g2d.setColor(Color.***white***);

g2d.fillRect(0, 0, 250, 250);

g2d.setColor(Color.***black***);

g2d.drawRect(10, 10, 195, 195);

g2d.setColor(Color.***green***);

g2d.fillRect((Game.*aim*.x+1)\*5, (Game.*aim*.y+1)\*5, 5, 5);

g2d.setColor(Color.***black***);

g2d.drawString("Punkte: "+Game.*punkte*+" - Zeit: "+Game.*playtime*, 10, 220);

Game.*snake*.paintSneak(g2d);

}

}

}