package com.EightNumber.view;

import java.util.\*;

public class EightNumPath {

final static int dx[] = {-1, 1, 0, 0};

final static int dy[] = { 0, 0,-1, 1};

final static String dir = "UDLR";

static int maxstate = 400000;

static int [][]st = new int[maxstate][9];

static int []goal = {1,2,3,4,5,6,7,8,0};

static int []dist = new int[maxstate];

static int []fa = new int[maxstate];

static int []move = new int[maxstate];

static boolean []vis = new boolean[maxstate];

static int []fact = new int[9];

static StringBuffer path;

public static boolean isok(int []a) {

int sum=0;

for(int i=0; i < 9; i++)

for(int j=i+1; j < 9; j++)

if(a[j] != 0 && a[i] != 0 && a[i] > a[j])

sum++;

if(sum % 2 == 0) {

return true;

}

return false;

}

private static void init\_lookup\_table() {

fact[0] = 1;

for(int i = 1; i < 9; i++) {

fact[i] = fact[i-1] \* i;

}

Arrays.fill(vis, false);

}

private static boolean try\_to\_insert(int s) {

int code = 0;

for(int i = 0; i < 9; i++) {

int cnt = 0;

for(int j = i+1; j < 9; j++) {

if(st[s][j] < st[s][i]) {

cnt++;

}

}

code += fact[8-i] \* cnt;

}

if(vis[code]) {

return false;

}

return vis[code] = true;

}

private static void print\_path(int cur) {

while(cur != 1) {

path.insert(0,dir.charAt(move[cur]));

cur = fa[cur];

}

}

private static int bfs() {

init\_lookup\_table();

int front = 1 , rear = 2;

try\_to\_insert(front);

while(front < rear) {

if(Arrays.equals(st[front], goal)) {

return front;

}

int z;

for(z = 0; z < 9; z++) {

if(st[front][z] == 0) {

break;

}

}

int x = z/3, y = z%3;

for(int d = 0; d < 4; d++) {

int newx = x + dx[d];

int newy = y + dy[d];

int newz = newx \* 3 + newy;

if(newx >= 0 && newx < 3 && newy >= 0 && newy < 3) {

st[rear] = Arrays.copyOf(st[front], st[front].length);

st[rear][newz] = st[front][z];

st[rear][z] = st[front][newz];

dist[rear] = dist[front] + 1;

if(try\_to\_insert(rear)) {

fa[rear] = front;

move[rear] = d;

rear++;

}

}

}

front++;

}

return 0;

}

public static String solve(String state) {

path = new StringBuffer();

for(int i = 0; i < state.length(); i++) {

st[1][i] = Integer.valueOf(state.charAt(i)) - '0';

}

int ans = bfs();

print\_path(ans);

return path.toString();

}

}

package com.EightNumber.view;

import java.awt.\*;

import javax.swing.\*;

import java.awt.event.\*;

import java.util.\*;

public class EightNumFrame extends Frame implements ActionListener,KeyListener

{

MenuBar menubar=new MenuBar();

Menu menu\_file = new Menu("文件(F)");

MenuItem restart = new MenuItem("重新开始");

MenuItem nextPath = new MenuItem("提示");

MenuItem printPath = new MenuItem("还原");

MenuItem exit = new MenuItem("退出");

Button[] button;

Panel panel;

int row,col;

private static int position,cellNum;

final int dr[] = { 0,-1, 0, 1};

final int dc[] = {-1, 0, 1, 0};

public EightNumFrame(int row,int col) {

super.setMenuBar(menubar);

//Windows风格

try {

UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");

} catch (Exception e) {

e.printStackTrace();

}

this.row = row;

this.col = col;

cellNum = row\*col;

restart.addActionListener(this);

exit.addActionListener(this);

nextPath.addActionListener(this);

printPath.addActionListener(this);

menu\_file.add(restart);

menu\_file.add(nextPath);

menu\_file.add(printPath);

menu\_file.add(exit);

menubar.add(menu\_file);

panel = new Panel(new GridLayout(row,col)) ;

button = new Button[cellNum];

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

if(i == cellNum - 1) {

button[i] = new Button(" ");

}else {

button[i] = new Button(String.valueOf(i + 1));

}

button[i].setFont(new Font("Courier", 1, 20));

button[i].addActionListener(this);

button[i].addKeyListener(this);

panel.add(button[i]);

}

position = cellNum - 1;

this.add(BorderLayout.CENTER,panel);

this.setTitle("八数码");

this.setVisible(true);

this.setSize(300,300);

Toolkit kit = Toolkit.getDefaultToolkit();

Dimension screenSize = kit.getScreenSize();

int screenWidth = screenSize.width/2;

int screenHeight = screenSize.height/2;

int height = this.getHeight();

int width = this.getWidth();

this.setLocation(screenWidth-width/2, screenHeight-height/2);

this.addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

System.exit(0);

}

});

}

void start() {

int a[] = new int[9];

do {

int k = 0;

Random random=new Random();

Set<Integer> set=new HashSet<Integer>();

while(set.size() < cellNum-1) {

int n=random.nextInt(cellNum-1)+1;

if(!set.contains(n)) {

set.add(n);

a[k++] = n;

}

}

a[k] = 0;

}while(!EightNumPath.isok(a));

for(int i = 0; i < 9; i++)

button[i].setLabel(String.valueOf(a[i]));

button[cellNum-1].setLabel(" ");

position = cellNum - 1;

}

boolean win() {

for(int i = 0; i < cellNum - 1; i++) {

if(button[i].getLabel().equals(" ")) {

return false;

}else if(Integer.valueOf(button[i].getLabel()) != i+1) {

return false;

}

}

return true;

}

private boolean judge(Button a, Button b) {

for(int i = 0; i < 4; i++) {

if( (a.getX() == b.getX() + dr[i]\*a.getWidth())

&& (a.getY() == b.getY() + dc[i]\*a.getHeight())) {

return true;

}

}

return false;

}

public void actionPerformed(ActionEvent e) {

StringBuffer state = new StringBuffer();

if(e.getSource() == restart) {

start();

return;

}else if(e.getSource() == exit) {

System.exit(0);

return;

}else if(e.getSource() == nextPath) {

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

if(button[i].getLabel().equals(" ")) {

state.append('0');

}else {

state.append(button[i].getLabel());

}

}

String path = EightNumPath.solve(state.toString());

JOptionPane.showMessageDialog(this,"建议走："+path);

System.out.println(path);

return;

}else if(e.getSource() == printPath) {

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

if(button[i].getLabel().equals(" ")) {

state.append('0');

}else {

state.append(button[i].getLabel());

}

}

String path = EightNumPath.solve(state.toString());

for(int i = 0; i < path.length(); i++) {

switch(path.charAt(i)) {

case 'U':

go(KeyEvent.VK\_UP);

break;

case 'D':

go(KeyEvent.VK\_DOWN);

break;

case 'L':

go(KeyEvent.VK\_LEFT);

break;

case 'R':

go(KeyEvent.VK\_RIGHT);

break;

}

try {

Thread.sleep(1000);

} catch (InterruptedException e1) {

e1.printStackTrace();

}

}

}

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

if(e.getSource() == button[i]) {

if(!button[i].getLabel().equals(" ") && judge(button[i],button[position])) {

button[position].setLabel(button[i].getLabel());

button[i].setLabel(" ");

position = i;

}

}

}

if(win()) {

JOptionPane.showMessageDialog(this,"Congratulations");

}

}

void go(int dir) {

int x = position / col;

int y = position % col;

switch(dir) {

case KeyEvent.VK\_UP:

if(x != 0) {

button[position].setLabel(button[position-col].getLabel());

button[position-col].setLabel(" ");

position -= col;

}

break;

case KeyEvent.VK\_DOWN:

if(x != row-1) {

button[position].setLabel(button[position+col].getLabel());

button[position+col].setLabel(" ");

position += col;

}

break;

case KeyEvent.VK\_LEFT:

if(y != 0) {

button[position].setLabel(button[position-1].getLabel());

button[position-1].setLabel(" ");

position -= 1;

}

break;

case KeyEvent.VK\_RIGHT:

if(y != col-1) {

button[position].setLabel(button[position+1].getLabel());

button[position+1].setLabel(" ");

position += 1;

}

break;

}

}

public void keyPressed(KeyEvent e) {

go(e.getKeyCode());

if(win()) {

JOptionPane.showMessageDialog(this,"Congratulations");

}

}

public void keyReleased(KeyEvent e) {}

public void keyTyped(KeyEvent e) {}

public static void main(String[] args) {

new EightNumFrame(3, 3);

}

}