# 1．选题

## （1）选题说明

2048游戏规则很简单，玩家每次可以选择上下左右其中一个方向去移动，每移动一次，所有的数字方块都会往移动的方向靠拢外，系统也会在空白的地方随机出现一个数字方块，相同数字的方块在靠拢、相撞时会相加。系统给予的数字方块不是2就是4，玩家要想办法在这小小的16格范围中凑出“2048”这个数字方块。

## 功能说明

（1）上下左右滑动进行操作

（2）重新游戏

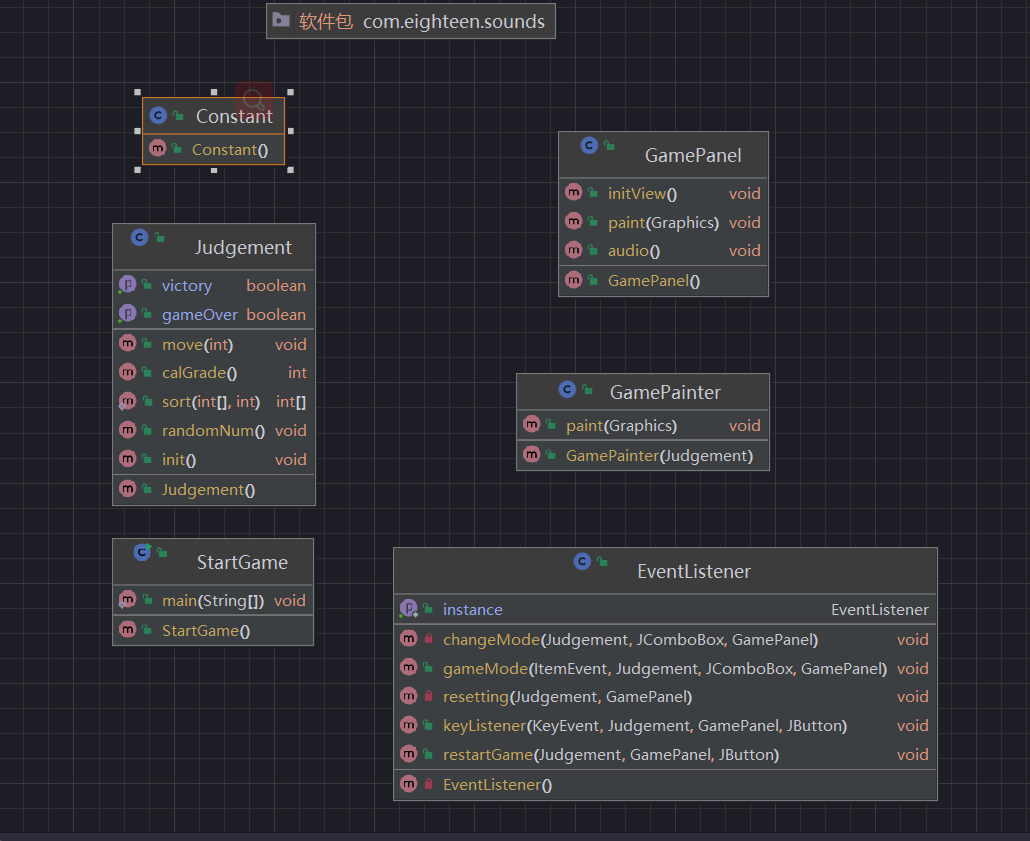
（3）播放背景音乐

（4）统计分数

（4）自动判断游戏是否结束

# 2．课程设计内容

## （1）使用的技术



## 设计内容

### StartGame主类

StartGame类Main()方法

主函数调用主要功能：GamePanel frame = new GamePanel()，实现程序运行。

### GamePanel 窗体类

GamePanel 窗体类的功能定义了JPanel可视化组件，可以在中放置按钮、文本框等非容器组件。GamePainter画图类，JComboBox下拉列表框，JButton按钮，GamePanel画图类，Judgement逻辑判断类。

/\*\*

\* 窗体类

\*/

public class GamePanel extends JFrame {

//可视化组件

private JPanel panelOne;

//画图类

private GamePainter gamePainter;

//下拉列表框

private JComboBox cbGameMode;

//JButton按钮

private JButton btnRestartGame;

//画图类

public GamePanel gp;

//逻辑判断类

private Judgement jw;

public GamePanel() {

jw=new Judgement();

gamePainter=new GamePainter(jw);

gp =this;

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(Constant.width,Constant.height);

setLocation((Constant.w-Constant.width)/2,(Constant.h-Constant.height)/2);

setResizable(false);

initView();

//获取焦点

this.setFocusable(true);

/\*\*

\* 监听键盘事件

\*/

addKeyListener(new KeyListener() {

@Override

public void keyPressed(KeyEvent e) {

EventListener.getInstance().keyListener(e,jw, gp,btnRestartGame);

}

@Override

public void keyTyped(KeyEvent e) {}

@Override

public void keyReleased(KeyEvent e) {}

});

}

/\*\*

\* 绘制2048游戏

\*

\*/

@Override

public void paint(Graphics g) {

super.paint(g);

gamePainter.paint(g);

}

/\*\*

\* 播放音频文件（.wav文件）

\*

\*/

public void audio() throws MalformedURLException, InterruptedException {

//选择播放文件

File file = new File("sounds/背景音乐.wav");

//创建audioclip对象

AudioClip audioClip = null;

//将file转换为url

audioClip = Applet.newAudioClip(file.toURL());

//循环播放 播放一次可以使用audioClip.play

audioClip.loop();

Thread.sleep(0);

}

/\*\*

\* 初始化控件

\*/

public void initView(){

panelOne = new JPanel();

panelOne.setBorder(new EmptyBorder(5, 5, 5, 5));

this.setContentPane(panelOne);

panelOne.setLayout(null);

//下拉列表

cbGameMode = new JComboBox();

//监听下拉列表的Item值改变

cbGameMode.addItemListener(e -> EventListener.getInstance().gameMode(e,jw,cbGameMode, gp));

cbGameMode.addItem("4X4");

cbGameMode.addItem("5X5");

cbGameMode.setBounds(445, 360, 95, 30);

panelOne.add(cbGameMode);

btnRestartGame = new JButton("重新开始");

btnRestartGame.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

super.mouseClicked(e);

EventListener.getInstance().restartGame(jw, gp,btnRestartGame);

}

});

btnRestartGame.setBounds(445, 145, 95, 30);

panelOne.add(btnRestartGame);

}

}

### GamePainter画图类

/\*\*

\*画图类

\*/

public class GamePainter {

private Judgement jw;

/\*\*

\* 预先加载图片到数组里

\*/

public GamePainter(Judgement judgement){

this.jw = judgement;

int num=2;

for (int i = 0; i < Constant.img100.length; i++ ) {

try {

//100像素的路径和图片数组

Constant.loc100 ="images/numbers/"+num+".png";

Constant.img100[i]=ImageIO.read(new File(Constant.loc100));

Constant.loc80 ="images/numbers80/"+num+".png";

Constant.img80[i]=ImageIO.read(new File(Constant.loc80));

num=num\*2;

} catch (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* JFrame的paint方法调用的画图方法

\* @param g

\*/

public void paint(Graphics g) {

for(int x = 0; x<Constant.ACCOUNT; x++)

for(int y = 0; y<Constant.ACCOUNT; y++){

g.setColor(Color.lightGray);

g.fillRect(20 + x \* (Constant.SIZE +5), 60 + y \* (Constant.SIZE +5), Constant.SIZE, Constant.SIZE);

}

for(int x = 0; x<Constant.ACCOUNT; x++)

for(int y = 0; y<Constant.ACCOUNT; y++)

{

if(jw.loc[x][y]!=-1) {

//如果选中4X4就加载100像素打图片，否在加载80像素的图片

if (Constant.isImg100) {

g.drawImage(Constant.img100[jw.loc[x][y]], 20 + y \* (Constant.SIZE + 5), 60 + x \* (Constant.SIZE + 5), null);

}else {

g.drawImage(Constant.img80[jw.loc[x][y]], 20 + y \* (Constant.SIZE + 5), 60 + x \* (Constant.SIZE + 5), null);

}

}

}

//显示分数

g.setFont(new Font("Tahoma", Font.BOLD, 21));

g.setColor(Color.BLACK);

g.drawString("分数:",50,40);

g.drawString(Integer.toString(Constant.grade),110,40);

g.drawString("游戏步数:",255,40);

g.drawString(Integer.toString(Constant.count),360,40);

}

}

### EventListener 事件监听类

/\*\*

\* 窗体类

\*/

public class GamePanel extends JFrame {

//可视化组件

private JPanel panelOne;

//画图类

private GamePainter gamePainter;

//下拉列表框

private JComboBox cbGameMode;

//JButton按钮

private JButton btnRestartGame;

//画图类

public GamePanel gp;

//逻辑判断类

private Judgement jw;

public GamePanel() {

jw=new Judgement();

gamePainter=new GamePainter(jw);

gp =this;

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setSize(Constant.width,Constant.height);

setLocation((Constant.w-Constant.width)/2,(Constant.h-Constant.height)/2);

setResizable(false);

initView();

//获取焦点

this.setFocusable(true);

/\*\*

\* 监听键盘事件

\*/

addKeyListener(new KeyListener() {

@Override

public void keyPressed(KeyEvent e) {

EventListener.getInstance().keyListener(e,jw, gp,btnRestartGame);

}

@Override

public void keyTyped(KeyEvent e) {}

@Override

public void keyReleased(KeyEvent e) {}

});

}

/\*\*

\* 绘制2048游戏

\*

\*/

@Override

public void paint(Graphics g) {

super.paint(g);

gamePainter.paint(g);

}

/\*\*

\* 播放音频文件（.wav文件）

\*

\*/

public void audio() throws MalformedURLException, InterruptedException {

//选择播放文件

File file = new File("sounds/背景音乐.wav");

//创建audioclip对象

AudioClip audioClip = null;

//将file转换为url

audioClip = Applet.newAudioClip(file.toURL());

//循环播放 播放一次可以使用audioClip.play

audioClip.loop();

Thread.sleep(0);

}

/\*\*

\* 初始化控件

\*/

public void initView(){

panelOne = new JPanel();

panelOne.setBorder(new EmptyBorder(5, 5, 5, 5));

this.setContentPane(panelOne);

panelOne.setLayout(null);

//下拉列表

cbGameMode = new JComboBox();

//监听下拉列表的Item值改变

cbGameMode.addItemListener(e -> EventListener.getInstance().gameMode(e,jw,cbGameMode, gp));

cbGameMode.addItem("4X4");

cbGameMode.addItem("5X5");

cbGameMode.setBounds(445, 360, 95, 30);

panelOne.add(cbGameMode);

btnRestartGame = new JButton("重新开始");

btnRestartGame.addMouseListener(new MouseAdapter() {

@Override

public void mouseClicked(MouseEvent e) {

super.mouseClicked(e);

EventListener.getInstance().restartGame(jw, gp,btnRestartGame);

}

});

btnRestartGame.setBounds(445, 145, 95, 30);

panelOne.add(btnRestartGame);

}

}

### 5.Judgement 逻辑判断类

/\*\*

\* 逻辑判断类

\*/

public class Judgement {

public int loc[][]=new int[Constant.SIZE][Constant.SIZE];

public Judgement(){

init();

}

/\*\*

\* 随机位置，随机2或4

\*/

public void randomNum(){

int i1=(int)(Math.random()\*Constant.ACCOUNT);

int j1=(int)(Math.random()\*Constant.ACCOUNT);

int i2=(int)(Math.random()\*Constant.ACCOUNT);

int j2=(int)(Math.random()\*Constant.ACCOUNT);

//预防两个位置重合，再随机一次

if (i1==i2 && j1==j2){

i1=(int)(Math.random()\*Constant.ACCOUNT);

}

int num1=(int)(1+Math.random()\*3);

if(num1==1){

num1=0;

}else{

num1=1;

}

int num2=(int)(1+Math.random()\*3);

if(num2==1){

num2=0;

}else{

num2=1;

}

//-1为空，0-11分别代表2-4096

for(int x = 0; x<Constant.ACCOUNT; x++)

for(int y = 0; y<Constant.ACCOUNT; y++)

{

loc[x][y]=-1;

}

loc[i1][j1]=num1;

loc[i2][j2]=num2;

}

public void init(){

randomNum();

calGrade();

}

public static int[] sort(int[] list,int flag) {

if(flag==0){

for(int i = 0; i<Constant.ACCOUNT; i++){

if(list[i]==-1)continue;

for(int j = i+1; j<Constant.ACCOUNT; j++)

if(list[j]!=-1){

if(list[i]==list[j]){

list[i]=list[i]+1;

list[j]=-1;

Constant.change=true;

}

break;

}

}

int k=0;

for(int i = 0; i<Constant.ACCOUNT; i++){

if(list[i]!=-1){

list[k]=list[i];

if(k<i){

list[i]=-1;

Constant.change=true;

}

k++;

}

}

}

else{

for(int i = Constant.ACCOUNT -1; i>=0; i--){

if(list[i]==-1)continue;

for(int j=i-1;j>=0;j--)

if(list[j]!=-1){

if(list[i]==list[j]){

list[i]=list[i]+1;

list[j]=-1;

Constant.change=true;

}

break;

}

}

int k=Constant.ACCOUNT-1;

for(int i = Constant.ACCOUNT -1; i>=0; i--){

if(list[i]!=-1){

list[k]=list[i];

if(k>i){

list[i]=-1;

Constant.change=true;

}

k--;

}

}

}

return list;

}

/\*\*

\* 判断游戏是否结束

\* @return

\*/

public boolean isGameOver(){

for(int h = 0; h<Constant.ACCOUNT; h++)

for(int l = Constant.ACCOUNT -1; l>=0; l--){

if(loc[h][l]==-1)

return false;

if(h>0 && loc[h][l]== loc[h-1][l])

return false;

if(l<Constant.ACCOUNT-1 && loc[h][l]== loc[h][l+1])

return false;

}

return true;

}

public boolean isVictory(){

for(int h = 0; h<Constant.ACCOUNT; h++)

for(int l = Constant.ACCOUNT-1; l>=0; l--){

if(!Constant.isContinue){

if(loc[h][l]==10)

return true;

}

if (Constant.isContinue){

if(loc[h][l]==11)

return true;

}

}

return false;

}

/\*\*

\* 计算分值

\* @return

\*/

public int calGrade()

{

int res=0;

for(int h = 0; h<Constant.ACCOUNT; h++)

for(int l = 0; l<Constant.ACCOUNT; l++)

if(loc[h][l]!=-1)

res+=Math.pow(2, loc[h][l])\* loc[h][l];

return res;

}

/\*\*

\*把相邻的又数字相同的合并，并且计算分数

\* @param key

\*/

public void move(int key){

int h,l;

int tmp[]=new int[Constant.ACCOUNT];

//扫描每一行

if(key== KeyEvent.VK\_LEFT){

for(h=0; h<Constant.ACCOUNT; h++)

loc[h]=sort(loc[h],0);

Constant.grade = calGrade();

Constant.count++;

}

//扫描每一行

if(key==KeyEvent.VK\_RIGHT){

for(h=0; h<Constant.ACCOUNT; h++)

loc[h]=sort(loc[h],1);

Constant.grade = calGrade();

Constant.count++;

}

//扫描每一列

if(key==KeyEvent.VK\_UP){

for(l=0; l<Constant.ACCOUNT; l++){

for(h=0;h<Constant.ACCOUNT;h++)

tmp[h]= loc[h][l];

tmp=sort(tmp,0);

for(h=0; h<Constant.ACCOUNT; h++)

loc[h][l]=tmp[h];

}

Constant.grade = calGrade();

Constant.count++;

}

//扫描每一列

if(key==KeyEvent.VK\_DOWN){

for(l=0; l<Constant.ACCOUNT; l++){

for(h=0;h<Constant.ACCOUNT;h++)

tmp[h]= loc[h][l];

tmp=sort(tmp,1);

for(h=0; h<Constant.ACCOUNT; h++)

loc[h][l]=tmp[h];

}

Constant.grade = calGrade();

Constant.count++;

}

}

}

### 6.Constant 常量类

/\*\*

\* 常量类

\*/

public class Constant {

//游戏模式4X4或5X5

public static int ACCOUNT =4;

public static int SIZE =100;

//软件宽高

public static final int width=600;

public static final int height=600;

//获取屏幕宽和高

public static int w= Toolkit.getDefaultToolkit().getScreenSize().width;

public static int h=Toolkit.getDefaultToolkit().getScreenSize().height;

//100像素的路径和图片数组

public static String loc100;

public static BufferedImage img100[]=new BufferedImage[12];

//80像素的路径和图片数组

public static String loc80;

public static BufferedImage img80[]=new BufferedImage[12];

public static int loc[][]=new int[SIZE][SIZE];

//当前是否选中100像素的图片

public static boolean isImg100=true;

//是否继续玩到4096

public static boolean isContinue=false;

public static boolean change;

//计算游戏步数

public static int count=0;

//计算游戏分数

public static int grade=0;

public static boolean isOver =false;

}

# 调试与难点

## 1.2048游戏算法

（1）游戏开始时随机的产生2个数值为2/4的方格，其中1个方格置于4个角中的一个位置，另一个方格随机的置于其它位置，其余方格值为0。

（2）每次移动鼠标或按方向键后，逐行计算移动后的方格值。每行移动的算法是：先将所有值为0的数移至行首。能后从行尾开始逐一和前一个数比较，如果相等则合并这2个子。 （3）每合并一次格子，将其值累计到游戏总分中。

（4）一次移动结束后，在所有值为0的方格中随机的分配一个2/4的值。

（5）所有方格值不为0且上下、左右相邻的方格都不相等，则游戏结束。

2.模块化写代码

项目创建应该从三方面入手：

1.界面 2.游戏逻辑 3.交互

  写完一块要测试，且测试不要放在负载的环境中，否则给改bugs带来不便。

## 3.jdk版本不同导致异常

遇到jdk版本不同导致的异常，我检查了jdk版本。软件版本一般是向下兼容的，java虚拟机也不例外，即低版本虚拟机生成的class文件可以在高版本虚拟机中运行，反之则未必可以(向上兼容)。修改版本后，在不同机型都可以运行。

# 4．总结

通过本次课程设计，对java语言有了更多的了解，不过我也发现我的好多不足之处，首先是自己编程粗心大意，经常出错，通过学习也有所改进；再有对java类库的一些标准方法不太熟悉，还以及对java语言中经常出现的错误也不了解，通过实践，使我在这几个方面的认识有所提高。

通过实践的学习，我认识到学好计算机要重视实践操作，不仅仅是学习java语言，还是其它的语言，以及其它的计算机方面的知识都要重在实践，所以后在学习过程中，我会更加注视实践操作，遇到问题一定要查阅相关资料，使自己更好地学好计算机。