南京邮电大学本科生程序设计课程报告

2048小游戏开发

|  |  |
| --- | --- |
| 课程 | 嵌入式智能终端开发 |
| 学生姓名 | 茹兴奥 |
| 专业 | 信息安全 |
| 班级学号 | B1030824 |
| 学院 | 计算机学院、软件学院、网络空间安全学院 |
| 授课教师 | 陈向东 |

日期： 2021 年 6 月 15 日

**目 录**

[**一、 课程设计题目与要求 3**](#_Toc44065337)

[1. 设计题目 3](#_Toc44065338)

[2. 设计要求 3](#_Toc44065339)

[**二、 总体设计 3**](#_Toc44065340)

[1. 总体功能框架 3](#_Toc44065341)

[2. 数据结构概要设计 3](#_Toc44065342)

[**三、 详细设计 3**](#_Toc44065343)

[1. 数据结构详细设计 3](#_Toc44065344)

2. 系统功能详细设计 6

3. 所有代码…………………………………………………………………...16

[**四、 运行结果 25**](#_Toc44065346)

[**五、 课程设计总结 26**](#_Toc44065347)

[1. 编程中的问题及解决方法 27](#_Toc44065348)

[2. 小结 27](#_Toc44065349)

[3. 心得体会 27](#_Toc44065350)

[4. 程序设计方法 28](#_Toc44065351)

[**六、 参考文献 28**](#_Toc44065352)

# 课程设计题目与要求

## 设计题目

2048小游戏的开发

## 设计要求

功能完备，界面简洁。

# 总体设计

## 总体功能框架

设计一个简单的手机程序，可以玩经典的2048小游戏。

## 数据结构概要设计

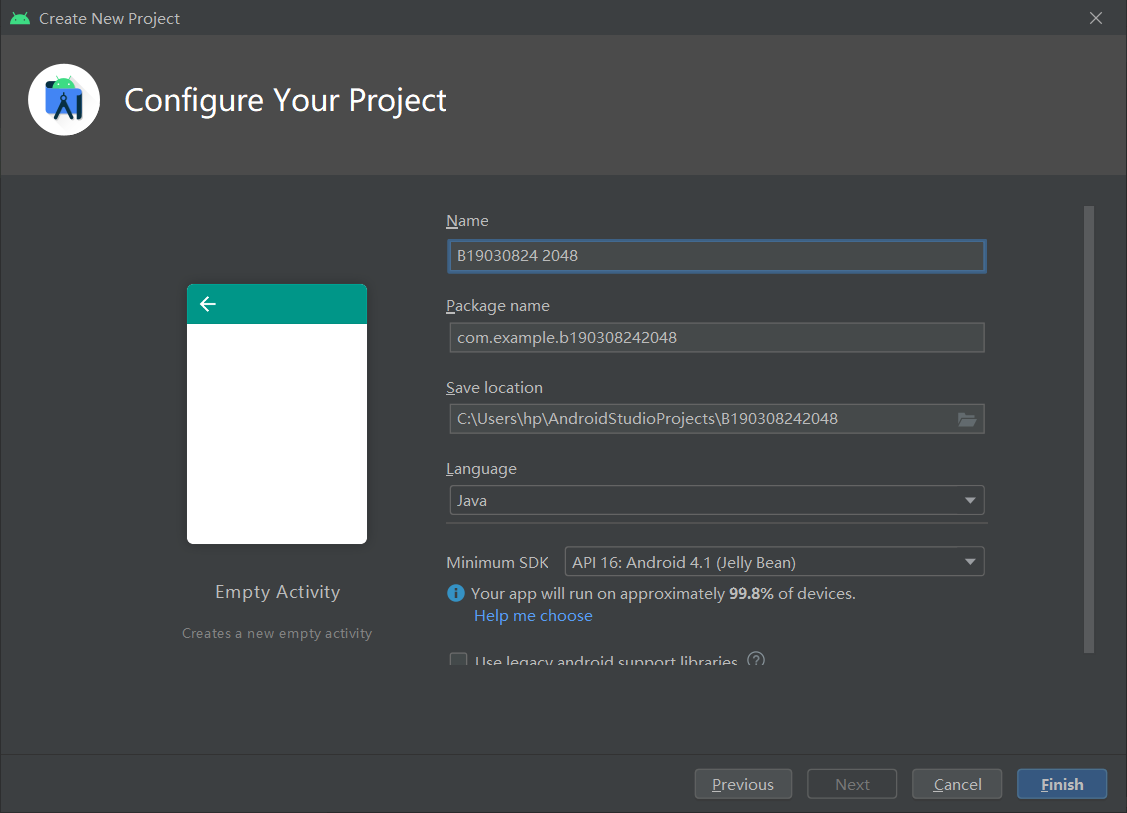
2048是一个曾经非常火爆的一个小游戏，本项目用到的主要知识点是：如何创建游戏项目，如何管理游戏布局，如何编写游戏主类GameView，如何在android平台触控交互设计，编写2048游戏的卡片类Card，将数字卡片类添加到场景，在游戏中添加随机数，实现游戏逻辑（最主要的部分），检查游戏计分，以及如何判断游戏是否结束。用到随机生成方块，还根据不同生成，改变不同方块数字的出现概率。

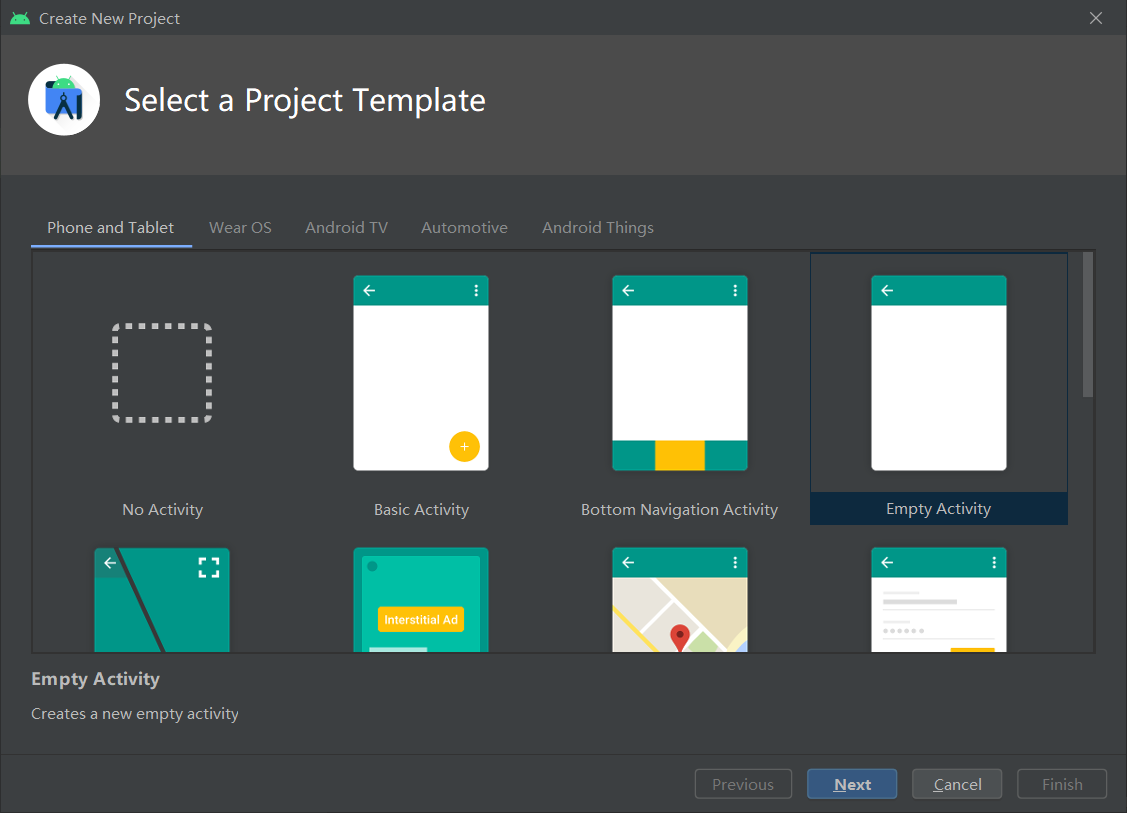
# 详细设计

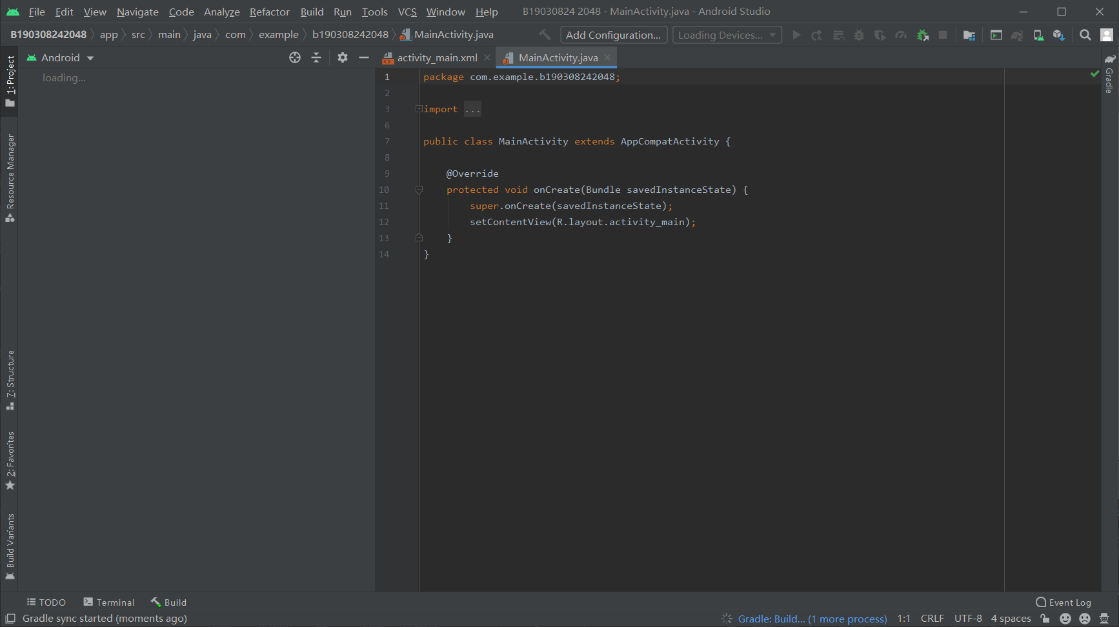
## 数据结构详细设计

首先我们新建一个安卓项目。   
填入创建的名称，公司域(项目的修饰)，项目路径，include c++ 和 include kotlin support 不选择   
。   
进入下一步。

新建一个空白项目

定一个面如下

页面如下



首先创建一个项目，接下来设计2048的游戏布局（activity\_main），上方是一个LinearLayout用于计分，下方是游戏布局GameView，继承自GridLayout，布局效果如下所示：

    接下来实现游戏主类GameView（在xml文件中访问它），GameView继承自GridLayout，用于放置2048游戏的16个卡片，关于网格布局GridLayout的使用，推荐一篇博客：http://blog.csdn.net/pku\_android/article/details/7343258

    下面是关于2048游戏的触控交互设计，无非是检测手指的down和up动作，利用坐标的变化值来判断手指的上下左右滑动，这里有一个需要注意的地方：要解决好当手指斜向滑动的时候如何判断的问题。触控交互的具体代码如下，相信也比较好理解：

setOnTouchListener(new View.OnTouchListener() { // 设置交互方式

// 监听上下左右滑动的这几个动作，再由这几个动作去执行特定的代码，去实现游戏的逻辑

private float startX, startY, offsetX, offsetY;

@Override

public boolean onTouch(View v, MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

break;

case MotionEvent.ACTION\_UP:

offsetX = event.getX() - startX;

offsetY = event.getY() - startY;

if (Math.abs(offsetX) > Math.abs(offsetY)) { // 加此判断是为了解决当用户向斜方向滑动时程序应如何判断的问题

if (offsetX < -5) {

swipeLeft(); //向左滑动的处理

} else if (offsetX > 5) {

swipeRight();

}

} else { // 判断向上向下

if (offsetY < -5) {

swipeUp();

} else if (offsetY > 5) {

swipeDown();

}

}

break;

}

return true; // 此处必须返回true

}

});

## 系统功能详细设计

程序结构图接下来再去实现2048游戏的卡片类，将16个卡片对象添加到GameView（继承自GridLayout）中去，新建Card类，继承自Framelayout，每个卡片都要有显示的数字，所有这个类有成员变量：

private int num = 0;

private TextView label; //卡片需要呈现文字

具体代码如下：

public class Card extends FrameLayout {

private int num = 0;

private TextView label; //卡片需要呈现文字

public Card(Context context) {

super(context);

label = new TextView(getContext()); //初始化

label.setTextSize(32); //设置文本大小

label.setBackgroundColor(0x33ffffff); //设置文字背景或颜色

label.setGravity(Gravity.CENTER); //居中文字显示 否则数字都在卡片的左上角

LayoutParams lp = new LayoutParams(-1, -1); //设置布局参数 填充满整个父级容器

lp.setMargins(10, 10, 0, 0); //设置文字间的间隔 用以区分各个card

addView(label, lp);

setNum(0); //默认情况下卡片数字为0 !!!!!!!!!顺序不能错 否则会出bug

}

public int getNum() {

return num;

}

public void setNum(int num) {

this.num = num;

if(num <= 0){

label.setText(""); //如果卡片中的数字是0，则不显示

}else {

label.setText(num+""); //如果卡片中的数字不是0，则显示，此时num(int)会变成一个字符串

}

}

public boolean equals(Card card) { //判断两个卡片上的数字是否相同

return this.getNum() == card.getNum();

}

}

接下来当然是往布局里面添加卡片类，将16个卡片添加到4\*4方阵中，其中为了解决适配不同手机的分辨率的问题，所以不能给出具体的宽高值，需要动态的计算卡片的宽高，通过重写onSizeChanged(int w, int h, int oldw, int oldh)来实现，计算布局宽高并往GameView中添加16个卡片的具体代码如下：

// 只有第一次创建的时候才会执行一次 只可能会执行一次

// 手机横放的时候不会执行，因为布局宽高不会发生改变，在AndroidManifest文件中配置了横放手机布局不变的参数

@Override

protected void onSizeChanged(int w, int h, int oldw, int oldh) { // 动态计算卡片宽高

super.onSizeChanged(w, h, oldw, oldh);

int cardWidth = (Math.min(w, h) - 10) / 4; //留出图片和边缘的大小

int cardHeight = cardWidth;

addCards(cardWidth, cardHeight);

startGame(); // 这里开启游戏

// 因为第一次创建游戏时，onSizeChanged()会被调用，且仅被调用一次，所以在这里开启游戏很合适

}

private void addCards(int cardWidth, int cardHeight) {

Card card;

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

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

card = new Card(getContext());

card.setNum(0); // 刚开始全部添加0 此时并不会显示数字

addView(card, cardWidth, cardHeight); // card是一个继承自FrameLayout的View

// 在initGameView()中指明这个GridLayout是四列的方阵

cardsMap[x][y] = card; //用一个二维数组来记录下这16个卡片

}

}

}

这里需要注意一下，因为手机横放时手机的布局（宽高）可能会发生变化，这时候onSizeChanged(int w, int h, int oldw, int oldh)方法会被调用，但并不希望这时候它被调用，所以需要需要配置一下，在AndroidManifest文件中设置屏幕一直是直立状态，代码如下所示

android:screenOrientation="portrait

这样的话onSizeChanged(int w, int h, int oldw, int oldh)只会被执行一次，只有布局第一次被创建的时候才会被调用执行，手机横放的时候不会执行。

接下里分别设置GridLayout和Card的背景颜色，此时需要注意要给Card的背景颜色留出一定的margin，设置card的颜色的代码如下所示：

LayoutParams lp = new LayoutParams(-1, -1); //设置布局参数 填充满整个父级容器

lp.setMargins(10, 10, 0, 0); //设置文字间的间隔 用以区分各个card

addView(label, lp); //lable是一个TextView

此时各个card就可以明显区分了。

下面来添加随机数，首先把所有的位置（空点的位置）全放在一个数组里面，方便随机地去取，新建一个List用于存储空点位置，把所有的位置（空点的位置）全放在一个数组里面，方便随机地去取：

private List<Point> emptyPoints = new ArrayList<Point>();

添加随机数的代码如下：

private void addRandomNum() { // 添加随机数 首先需要遍历所有卡片

emptyPoints.clear(); // 添加随机数之前先清空emptyPoints,然后把每一个空点都添加进来

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

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

if (cardsMap[x][y].getNum() <= 0) {

emptyPoints.add(new Point(x, y)); // 把空点的位置添加进去

// 因为只有空点才能够去添加数字

// 已经有数字的话肯定不会去添加了

}

}

}

Point p = emptyPoints

.remove((int) (Math.random() \* emptyPoints.size())); // 随机地移除一个点

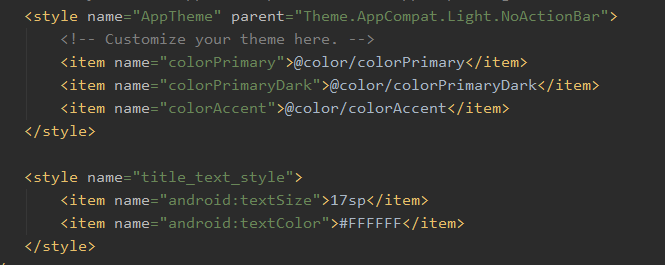
// 注意这里仅仅是移除emptyPoints中记录的点了，并没有移除card

cardsMap[p.x][p.y].setNum(Math.random() > 0.1 ? 2 : 4); // 给这个空点添加一个数，2或4，概率为9：1

}

接下来是最重要的游戏逻辑部分！！！！

以手指向左滑动为例（其他方向类似），此处要实现游戏逻辑，包括游戏的玩法逻辑，游戏计分逻辑和判断游戏是否结束的逻辑。代码如下所示：



// 实现游戏逻辑 只要有位置的改变就添加新的 最重要的部分：游戏逻辑

private void swipeLeft() {

boolean merge = false; // 判断是否有合并，如果有的话就进行一些处理

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

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

for (int x1 = x + 1; x1 < 4; x1++) { // 从当前的位置往右边去遍历

if (cardsMap[x1][y].getNum() > 0) { // 如果往右去遍历得到的card的值（获取到的值）不是空的，则做如下逻辑判断

if (cardsMap[x][y].getNum() <= 0) { // 如果当前位置上的值是空的，则将获取到的值移动到当前位置上

cardsMap[x][y].setNum(cardsMap[x1][y].getNum());

cardsMap[x1][y].setNum(0);

x--; // 这里非常重要！！！！ 可以测试理解

merge = true;

} else if (cardsMap[x][y].equals(cardsMap[x1][y])) { // 如果当前位置上的值不是空的，而且获取到的值与当前位置上的值相等，则做相加处理，并将结果放在当前位置上

cardsMap[x][y].setNum(cardsMap[x][y].getNum() \* 2);

cardsMap[x1][y].setNum(0);

// 合并时加分 有合并就有添加（分数）

MainActivity.getMainActivity().addScore(

cardsMap[x][y].getNum());

merge = true;

}

break; // 这个break的位置非常重要！！！！！ 只能写在这里！！ eg:方格最下面一行是2 32

// 64 2，然后往左滑动的情况！

}

}

}

}

if (merge) { // 在添加数字时判断游戏是否结束

addRandomNum();

checkComplete(); // 添加新项后都要检查游戏是否结束：没空位置，而且已经不能再合并

}

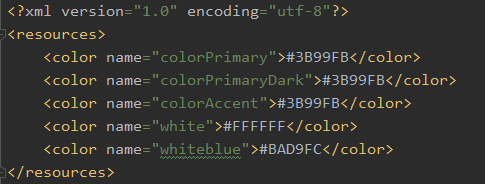
}

简要说一下此处实现的游戏玩法算法：首先用for循环一行一行地去遍历每一个card，然后从当前的位置往右去遍历，判断如果获取到了某一个值不是空的，此时有两种情况，一是当前位置上的值是空的，此时把获取到的值放到当前位置上，同时把获取到的位置上的数字清掉；二是当前位置上的值不是空的，并且获取到的值和当前位置上的值相同，则把合并这两个卡片，把当前位置上的值乘以二，同时把获取到的位置上的数字清掉。

还有一种情况是，如果我们当前位置上是空的，然后把右边的值放到当前的位置上去了，此时继续往后边（右边）去遍历，后边的位置还是空的，然后右边又有一个数字和之前放过去的数字是一样的情况的话，也是把它放到这个空位置上去了，这时会发生一个状况：这两张数字实际是一样的，但是它们并不合并。为了避免这种情况的发生，我们再让它去遍历一次，即让x-- ，这样这个问题就解决了。

这里还有一个非常关键的点：注意break的位置，写在别的地方是不对的，应该是只要检测到获取到的值不是空的,不管是否合并了，都应该break掉，这里可能比较难理解，最好亲手测试一下。

此处还设置了一个boolean类型的标志merge，用来记录卡片是否发生了合并，如果卡片发生了合并，就要添加一个随机数并且判断游戏是否结束，判断游戏是否结束的逻辑代码如下：



// 判断游戏是否结束的逻辑

private void checkComplete() {

boolean complete = true;

All: for (int y = 0; y < 4; y++) {

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

// 游戏没有结束的判定情况 5种情况

if (cardsMap[x][y].getNum() == 0

|| (x > 0 && cardsMap[x][y].equals(cardsMap[x - 1][y]))

|| (x < 3 && cardsMap[x][y].equals(cardsMap[x + 1][y]))

|| (y > 0 && cardsMap[x][y].equals(cardsMap[x][y - 1]))

|| (y < 3 && cardsMap[x][y].equals(cardsMap[x][y + 1]))) {

complete = false;

break All; // 写一个标签，跳出所有循环

}

}

}

if (complete) {

AlertDialog.Builder dialog = new AlertDialog.Builder(getContext());

dialog.setTitle("你好")

.setMessage("游戏结束")

.setPositiveButton("重来",

new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog,

int which) {

startGame();

}

});

dialog.setNegativeButton("关闭程序", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

MainActivity.getMainActivity().finish();

}

});

dialog.show();

}

**3.全部代码**

setOnTouchListener(new View.OnTouchListener() { // 设置交互方式

// 监听上下左右滑动的这几个动作，再由这几个动作去执行特定的代码，去实现游戏的逻辑

private float startX, startY, offsetX, offsetY;

@Override

public boolean onTouch(View v, MotionEvent event) {

switch (event.getAction()) {

case MotionEvent.ACTION\_DOWN:

startX = event.getX();

startY = event.getY();

break;

case MotionEvent.ACTION\_UP:

offsetX = event.getX() - startX;

offsetY = event.getY() - startY;

if (Math.abs(offsetX) > Math.abs(offsetY)) { // 加此判断是为了解决当用户向斜方向滑动时程序应如何判断的问题

if (offsetX < -5) {

swipeLeft(); //向左滑动的处理

} else if (offsetX > 5) {

swipeRight();

}

} else { // 判断向上向下

if (offsetY < -5) {

swipeUp();

} else if (offsetY > 5) {

swipeDown();

}

}

break;

}

return true; // 此处必须返回true

}

});

public class Card extends FrameLayout {

private int num = 0;

private TextView label; //卡片需要呈现文字

public Card(Context context) {

super(context);

label = new TextView(getContext()); //初始化

label.setTextSize(32); //设置文本大小

label.setBackgroundColor(0x33ffffff); //设置文字背景或颜色

label.setGravity(Gravity.CENTER); //居中文字显示 否则数字都在卡片的左上角

LayoutParams lp = new LayoutParams(-1, -1); //设置布局参数 填充满整个父级容器

lp.setMargins(10, 10, 0, 0); //设置文字间的间隔 用以区分各个card

addView(label, lp);

setNum(0); //默认情况下卡片数字为0 !!!!!!!!!顺序不能错 否则会出bug

}

public int getNum() {

return num;

}

public void setNum(int num) {

this.num = num;

if(num <= 0){

label.setText(""); //如果卡片中的数字是0，则不显示

}else {

label.setText(num+""); //如果卡片中的数字不是0，则显示，此时num(int)会变成一个字符串

}

}

public boolean equals(Card card) { //判断两个卡片上的数字是否相同

return this.getNum() == card.getNum();

}

}

// 只有第一次创建的时候才会执行一次 只可能会执行一次

// 手机横放的时候不会执行，因为布局宽高不会发生改变，在AndroidManifest文件中配置了横放手机布局不变的参数

@Override

protected void onSizeChanged(int w, int h, int oldw, int oldh) { // 动态计算卡片宽高

super.onSizeChanged(w, h, oldw, oldh);

int cardWidth = (Math.min(w, h) - 10) / 4; //留出图片和边缘的大小

int cardHeight = cardWidth;

addCards(cardWidth, cardHeight);

startGame(); // 这里开启游戏

// 因为第一次创建游戏时，onSizeChanged()会被调用，且仅被调用一次，所以在这里开启游戏很合适

}

private void addCards(int cardWidth, int cardHeight) {

Card card;

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

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

card = new Card(getContext());

card.setNum(0); // 刚开始全部添加0 此时并不会显示数字

addView(card, cardWidth, cardHeight); // card是一个继承自FrameLayout的View

// 在initGameView()中指明这个GridLayout是四列的方阵

cardsMap[x][y] = card; //用一个二维数组来记录下这16个卡片

}

}

}

LayoutParams lp = new LayoutParams(-1, -1); //设置布局参数 填充满整个父级容器

lp.setMargins(10, 10, 0, 0); //设置文字间的间隔 用以区分各个card

addView(label, lp); //lable是一个TextView

此时各个card就可以明显区分了。

添加随机数的代码如下：

private void addRandomNum() { // 添加随机数 首先需要遍历所有卡片

emptyPoints.clear(); // 添加随机数之前先清空emptyPoints,然后把每一个空点都添加进来

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

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

if (cardsMap[x][y].getNum() <= 0) {

emptyPoints.add(new Point(x, y)); // 把空点的位置添加进去

// 因为只有空点才能够去添加数字

// 已经有数字的话肯定不会去添加了

}

}

}

Point p = emptyPoints

.remove((int) (Math.random() \* emptyPoints.size())); // 随机地移除一个点

// 注意这里仅仅是移除emptyPoints中记录的点了，并没有移除card

cardsMap[p.x][p.y].setNum(Math.random() > 0.1 ? 2 : 4); // 给这个空点添加一个数，2或4，概率为9：1

}

接下来是最重要的游戏逻辑部分！！！！

以手指向左滑动为例（其他方向类似），此处要实现游戏逻辑，包括游戏的玩法逻辑，游戏计分逻辑和判断游戏是否结束的逻辑。代码如下所示：

// 实现游戏逻辑 只要有位置的改变就添加新的 最重要的部分：游戏逻辑

private void swipeLeft() {

boolean merge = false; // 判断是否有合并，如果有的话就进行一些处理

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

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

for (int x1 = x + 1; x1 < 4; x1++) { // 从当前的位置往右边去遍历

if (cardsMap[x1][y].getNum() > 0) { // 如果往右去遍历得到的card的值（获取到的值）不是空的，则做如下逻辑判断

if (cardsMap[x][y].getNum() <= 0) { // 如果当前位置上的值是空的，则将获取到的值移动到当前位置上

cardsMap[x][y].setNum(cardsMap[x1][y].getNum());

cardsMap[x1][y].setNum(0);

x--; // 这里非常重要！！！！ 可以测试理解

merge = true;

} else if (cardsMap[x][y].equals(cardsMap[x1][y])) { // 如果当前位置上的值不是空的，而且获取到的值与当前位置上的值相等，则做相加处理，并将结果放在当前位置上

cardsMap[x][y].setNum(cardsMap[x][y].getNum() \* 2);

cardsMap[x1][y].setNum(0);

// 合并时加分 有合并就有添加（分数）

MainActivity.getMainActivity().addScore(

cardsMap[x][y].getNum());

merge = true;

}

break; // 这个break的位置非常重要！！！！！ 只能写在这里！！ eg:方格最下面一行是2 32

// 64 2，然后往左滑动的情况！

}

}

}

}

if (merge) { // 在添加数字时判断游戏是否结束

addRandomNum();

checkComplete(); // 添加新项后都要检查游戏是否结束：没空位置，而且已经不能再合并

}

// 判断游戏是否结束的逻辑

private void checkComplete() {

boolean complete = true;

All: for (int y = 0; y < 4; y++) {

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

// 游戏没有结束的判定情况 5种情况

if (cardsMap[x][y].getNum() == 0

|| (x > 0 && cardsMap[x][y].equals(cardsMap[x - 1][y]))

|| (x < 3 && cardsMap[x][y].equals(cardsMap[x + 1][y]))

|| (y > 0 && cardsMap[x][y].equals(cardsMap[x][y - 1]))

|| (y < 3 && cardsMap[x][y].equals(cardsMap[x][y + 1]))) {

complete = false;

break All; // 写一个标签，跳出所有循环

}

}

}

if (complete) {

AlertDialog.Builder dialog = new AlertDialog.Builder(getContext());

dialog.setTitle("你好")

.setMessage("游戏结束")

.setPositiveButton("重来",

new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog,

int which) {

startGame();

}

});

dialog.setNegativeButton("关闭程序", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

MainActivity.getMainActivity().finish();

}

});

dialog.show();

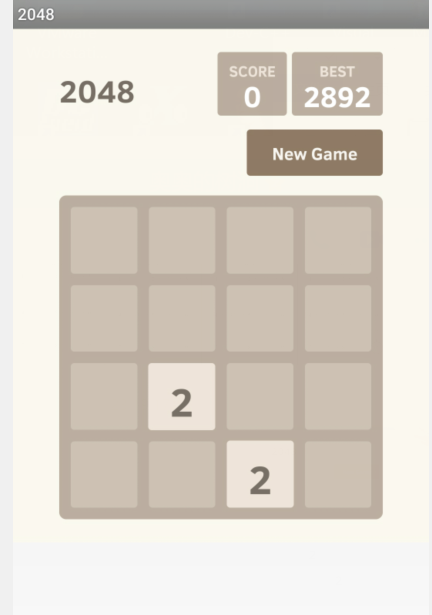
}

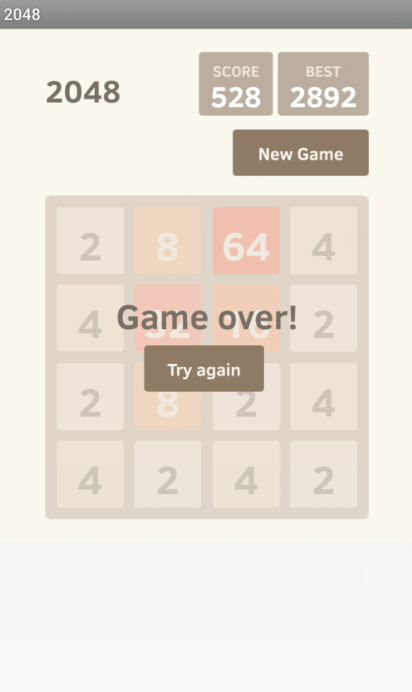
}

# 运行结果

在系统桌面上，点击“B19030824 2048”运行程序。

进入后，界面如下图所示：





# 课程设计总结

## 编程中的问题及解决方法

（1）布局问题

由于是在本课程时初学安卓开发，因此在一开始的时候，我并不是十分熟悉布局的方法。作为一个对界面设计十分注意的开发者，我一定要尽量让我的程序变得美观易用。我的想法是能够像“铁路12306”官方App一样，实现非自带的标题栏，同时加入渐变的蓝底图片，在图片上显示列车状态文字，以增加界面的美观效果。而ImageView无法直接增加可变的动态文字，标题栏也没有现成控件。最终实现效果如下图：

解决方法：网上查阅了各种资料，发现基本上都是使用谷歌于2014年的ToolBar控件，以工具栏的形式实现标题栏。但是查阅相关文档，发现操作起来不是十分简洁，且尝试后出现了较多的错误，对于初学者不易解决。因此，我联想自己之前在Windows平台的开发经验，决定通过设置LinearLayout背景图片+标签文本的方式来实现，并在styles.xml中设置为不带标题栏的样式。这样，问题就被“简单粗暴”地解决了，虽然并不是正道，但确实收到了良好的效果。

（2）文件读写问题

在我的程序中，车次信息的读入需要设计纯文本格式的文件内容读取，并且还是在安卓平台而非Windows，我一开始感到束手无策。查阅网络资料，发现能找到的资料实践之后，基本上都是读取assets文件夹下的文本，而我需要读取的，是位于SD卡下的特定文件夹下面的文件。

解决方法：联想到Android作为操作系统，其路径原理应当和Windows类似。因此，我查阅了获取SD卡路径的有关资料，结合自己手机上的一些文件路径格式，以及查到的关于assets文件夹下文本的读取，自己编写了有关的代码，成功实现。

（3）代码表达问题

由于安卓开发和Java语言都是初学，很多时候自己写错了代码都无法及时发现，或者给了提示也不知道该如何修改。

例如，下面一句获取字符串的代码，从语义上看去似乎也很符合逻辑，但却出现了错误。

解决方法：其实，这确实是字符串的一种获取方式，但并不能在这里使用。查阅之前的代码，找到该类型表达式的一处正确的用法如下：

mStateShow.setText(R.string.TrainState\_LastStationApproach);

查阅资料可知，这里的R.string.TrainState\_LastStationApproach的内容实际上是一个用于标注资源参考位置的ID编号，并不是实际的资源内容。这就类似于编程中常提到的“普通变量值”与“指针”之间的关系。既然这里只是一个资源编号，那么自然就不能使用toString方法，直接转换为字符串类型，从逻辑上也就理解了。

因此，正确的用法应该是：

cStateText == getResources().getString(R.string.TrainState\_StartStationWait)

这说明，在学习Java语言及安卓开发时，还是应当注重对于不同方法、不同参数、不同常用表达式类型的充分理解，只有这样才能避免犯类似的错误。

## 小结

作为一名安卓开发和Java开发的初学者，本次的大作业开发过程可以说是第一个具体化作品的尝试。在本次程序设计大作业的完成过程中，学习了Java语言和安卓环境开发的一些基础，并在项目实践中得到了操练。

在布局方面，尝试了Android Studio的相对布局之后，对其有了更加深刻的理解。结合之前自学过的HTML布局，对于此类相对布局又有了新的认识，这对于之后进一步学习CSharp下的WPF等界面布局方式也将有所启示和帮助。

在核心代码方面，通过查阅资料并实现了文件读取、音频播放等功能后，我对于Java语言基础以外的内容，以及安卓平台下的文件操作也有了一定的认识，大致体会到了其与Windows平台下文件操作的异同。同时，由于这两个操作都需要用到来自外部的库，实际上这也让我在实际操作中，对Java语言如何超越基础语法、调用外部包/API等的操作，有了更全面的理解。

## 心得体会

从一开始对于Java语言的茫然，只能结合较为相似的C++去理解，到后面逐步学习相关概念，并在安卓环境下开发出一款具有可用性的App，心里还是挺有成就感的。整个App的开发，也见证了我对于Java和Android开发的入门过程。

在纯Android设计层面，一个重要的问题就是对于各概念的理解（如Activity、包、系统结构、监听器等）。对于Activity这一概念来说，由于之前曾经进行过不少的VB开发，结合着VB的不同窗体，以及各窗体、控件的各类响应事件，理解起来也并不困难。而对于布局，在之前已经对HTML的盒子模型及其Table、Flex、Grid等布局有所了解后，现在直接面对Android Studio这样类似的相对布局，理解起来也并不困难，很快便能上手。不过，一些细节处理上还是有一些区别，在较容易地获得了感性的理解之后，还需要仔细斟酌，才能游刃有余地运用这一布局。

## 程序设计方法

本程序主要采用的是Android Studio开发环境+Java语言进行开发。在界面搭建的过程中，为了美观，App图标及部分图片资源使用了Photoshop等软件进行加工。

程序的需求设计来自于平时对于铁路的爱好。根据实际观察，以及与列车乘务员的交流，CRH2A/B/C/E、CRH3C等系列的早期动车组列车没有现代化的PA-PIS操作界面，无法实现新车型的简单化播放。目前，此类早期车型的报站均由乘务员将报站音频导入SD卡后，用收音机进行操作，并使用AUX音频线连接列车后进行播放，操作十分繁琐。既然有标准的3.5mm音频接口，加之近期列车电子客票普及，每趟列车都会有至少一部手机用于列车管理。于是我想，完全可以开发一款手机App，智能化地操作报站播报，然后通过音频线将报站声音传送到列车上实现全车播放。本程序的开发灵感由此而生。

# 参考文献

1. AndroidStudio-3.2.1（十二）4种文件读写方式

https://blog.csdn.net/SHENCHEN1987/article/details/88822047

1. Android实现播放音乐列表

https://blog.csdn.net/LosingCarryJie/article/details/79119694

1. Android studio 操作1 ：去掉默认的标题框

https://blog.csdn.net/guqigongzi/article/details/80844562

1. Android Studio的使用（十）--读取assets、Raw文件夹下文件，以及menu、drawable文件夹

https://www.cnblogs.com/begin1949/p/4967363.html

1. Android五大布局：FrameLayout、LinearLayout、AbsoluteLayout、RelativeLayout和TableLayout

http://c.biancheng.net/view/2976.html

1. android studio 的下拉菜单Spinner使用详解

https://blog.csdn.net/qq\_41893274/article/details/85528297

1. 廖雪峰的Java教程

https://www.liaoxuefeng.com/wiki/1252599548343744

1. 《Android编程权威指南》（[美]Bill Phillips等 著，人民邮电出版社）（任课老师推荐教材）