

**《移动应用开发技术》**

**课程设计**

**题目：Android平台数独游戏的设计与实现**

**院、 系： 计算机信息与技术系**

**学科专业： 软件工程**

**学生姓名： 范 卓**

**学 号： B14060405**

**指导教师： 张 琳**

×××2017年6月21日

# 摘 要

移动互联网时代的到来极大地改变了我们的生活，而Android是一种以Linux为基础的开放源码操作系统，迅速占领了智能机操作系统，所以在Android环境下开发显得尤为必要。Android是开放性体系架构，不仅具有非常好的开发、调试环境，而且还支持各种可扩展的用户体验，包括丰富的图形组件、多媒体支持功能以及强大的浏览器。因此，对于软件从业人员来说，Android平台具有与无限的吸引力。

本文就在分析讨论Android手机软件开发技术原理的基础上，为用户开发出基于Android平台的数独游戏软件。通过对Android系统架构和应用的开发进行初步探究，设计一款数独游戏。在对系统进行详细需求分析基础上，确定了系统的功能和性能要求，对游戏运行界面、菜单界面、屏幕绘制、布局算法、按钮监听等进行了设计。

本位详细介绍在Android平台上以Java语言为基础编写游戏的过程，最后实现了一个界面友好、操作简单的手机益智游戏。

**关键词：**Android游戏； Java；数独游戏

目 录

[摘 要 1](#_Toc485890954)

[第一章 绪论 3](#_Toc485890955)

[1.1 研究背景 3](#_Toc485890956)

[1.2 需求分析 3](#_Toc485890957)

[第二章 概要设计 4](#_Toc485890958)

[2.1 设计原理 4](#_Toc485890959)

[2.2 设计步骤 4](#_Toc485890960)

[2.3 设计结构 4](#_Toc485890961)

[2.4 流程图 5](#_Toc485890962)

[第三章 详细设计 5](#_Toc485890963)

[3.1 总体设计 5](#_Toc485890964)

[3.2 布局设计 7](#_Toc485890965)

[3.3 按钮监听 14](#_Toc485890966)

[3.4 算法设计 15](#_Toc485890967)

[3.5 存储设计 19](#_Toc485890968)

[第四章 测试分析 21](#_Toc485890969)

[第五章 总 结 24](#_Toc485890970)

[第六章 参考文献 24](#_Toc485890971)

# 第一章 绪论

## 1.1 研究背景

手机游戏，简称手游，是指用户依靠Android平台运行在移动终端的应用程序。不仅具有网络连接功能的手游，而且还具备单机游戏，将数据保存在本地手机，进行运行，同时手游还可以满足用户随时随地进行游戏运行的需求，它是需要一定的硬件环境和一定的系统环境作为其运行的基础。手机游戏吸引用户最大的特点就是其便利性，由于气是移动终端，所以，用户可以随时随地进行游戏。

数独游戏的前身为“九宫格”，最早起源于中国。但当时的算法比现在复杂，要求纵、横、斜向三个数字之和等于15。现在的数独游戏很简单，只需要在空白中填写1~9的数字，使横向、纵向、以及小九宫格数字不重复。因此，该功能通过相应的算法可以实现。

## 1.2 需求分析

1、 点击软件运行，进入菜单界面；

2、 选择第一关，进入游戏；

3、 进入第一关数独游戏界面，开始游戏；

4、 点击界面剩余空白方格，进行数字填充；

5、 数字填充满后，结束游戏；

6、 进行第二关、第三关测试；

7、 每关不少于5种初始局，且能随机选择其中一局运行；

8、 用户每次通关播放动画（或音频、图片），并显示用户当前的分数；

# 第二章 概要设计

## 2.1 设计原理

通过java中的canvas进行绘图设计，同时通过main.xml进行菜单界面布局设计。最后加上相应的游戏设计算法和自定义弹窗进行游戏的运行。将1~9数字分别填充到各个方框，最终使得横行、竖行、小九宫格里没有重复的数字。空格填满，游戏结束。

## 2.2 设计步骤

1、主界面菜单的设计，包括文本说明、关卡按钮等；

2、游戏主界面的设计，格子、线条、界面等；

3、游戏自定义对话弹窗的设计；

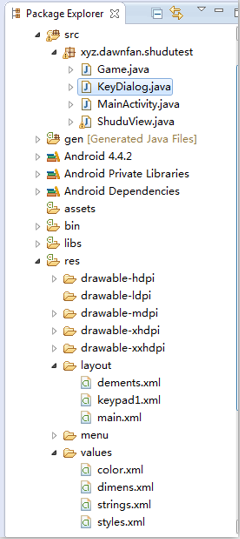
4、算法的设计；

5、通过弹窗和算法的功能，计算并抓取相应合适数据，填在文本格子；

6、退出游戏。

## 2.3 设计结构

设计总体结构如图2.1所示

 **图2.1**

## 2.4 流程图

菜单选项

第一关

第二关

第三关

游戏运行界面

退 出

# 第三章 详细设计

## 3.1 总体设计

1、通过main.xml文件来实现对菜单关卡界面的布局（如图3.1所示）

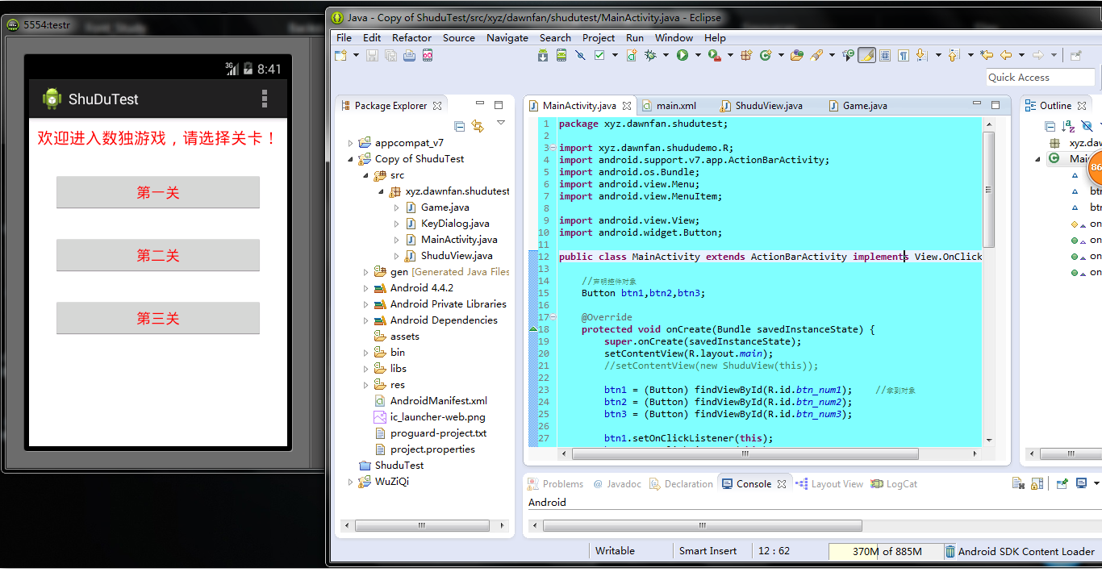


图3.1

2、通过ShuduView.java类中的Canvas来实现对游戏主界面格子的绘制以及背景颜色的添加实现（如图3.2所示）

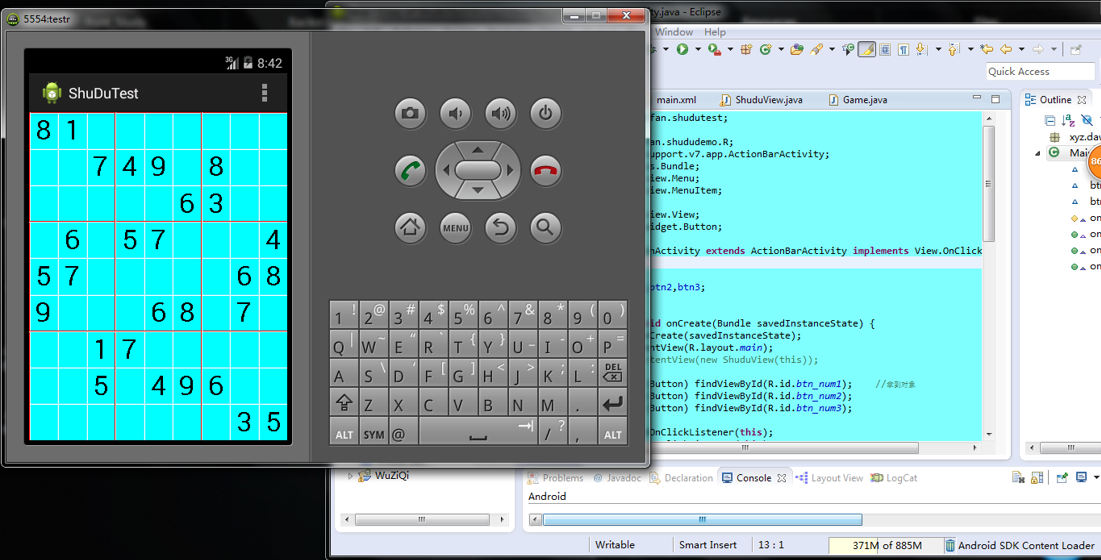


图3.2

3、通过KeyDialog.java类来实现弹窗，以及数字的显示功能（如图3.3、图3.4所示）

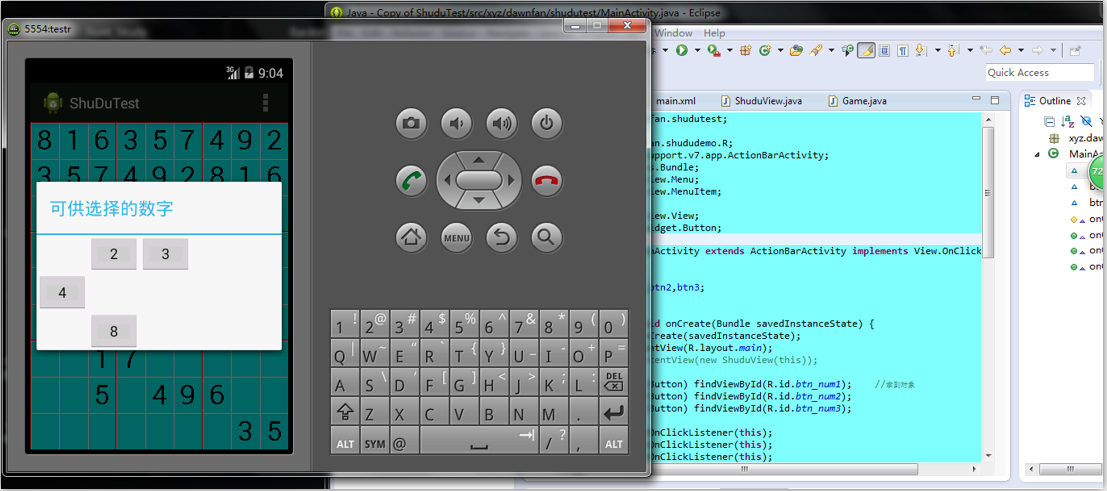


图3.3

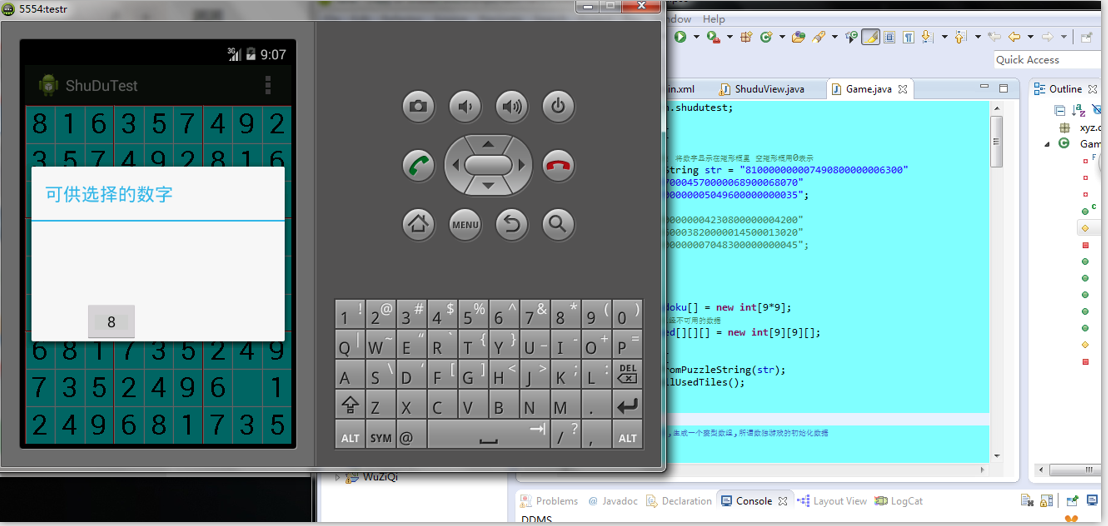
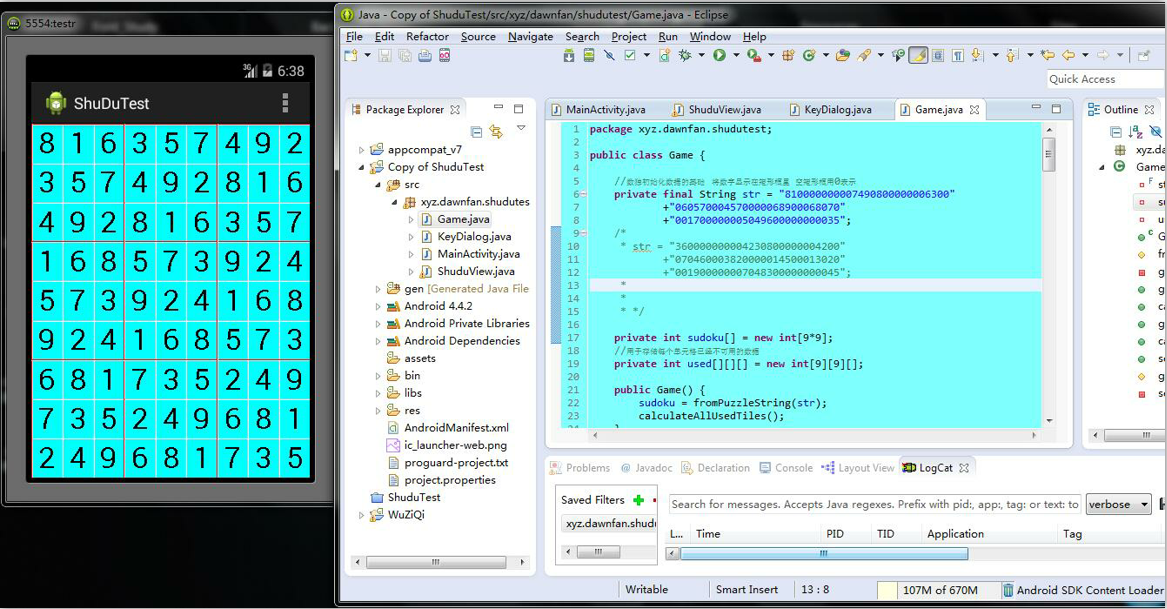


图3.4

4、通过Game.java类来实现具体算法，以及弹窗中数字的获取，同时保证横、纵、小九宫格中没有重复数字，直到所有小格都填满。数独游戏结束。（如图3.5所示）



## 3.2 布局设计

1、主菜单布局文件main.xml

<!-- LinearLayout布局 -->

<TextView

android:id=*"@+id/sd\_view"*

android:text=*"@string/sd\_text"*

android:layout\_marginTop=*"10dp"*

android:layout\_marginLeft=*"10dp"*

android:textColor=*"@color/colorRed"*

android:textSize=*"@dimen/sdtext"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

/>

<Button

android:id=*"@+id/btn\_num1"*

android:layout\_width=*"260sp"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginLeft=*"30dp"*

android:layout\_marginTop=*"30dp"*

android:text=*"@string/num1\_label"*

android:textColor=*"@color/colorRed"*

/>

<Button

android:id=*"@+id/btn\_num2"*

android:layout\_width=*"260sp"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginLeft=*"30dp"*

android:layout\_marginTop=*"30dp"*

android:text=*"@string/num2\_label"*

android:textColor=*"@color/colorRed"*

/>

<Button

android:id=*"@+id/btn\_num3"*

android:layout\_width=*"260sp"*

android:layout\_height=*"wrap\_content"*

android:layout\_marginLeft=*"30dp"*

android:layout\_marginTop=*"30dp"*

android:text=*"@string/num3\_label"*

android:textColor=*"@color/colorRed"*

/>

2、主界面绘制类ShuduView.java

**import** xyz.dawnfan.shududemo.R;

**import** android.content.Context;

**import** android.graphics.Canvas;

**import** android.graphics.Color;

**import** android.graphics.Paint;

**import** android.graphics.Paint.FontMetrics;

**import** android.view.MotionEvent;

**import** android.view.View;

**public** **class** ShuduView **extends** View{

//单元格的宽度和高度

**private** **float** width;

**private** **float** height;

**private** Game game = **new** Game();

**private** **int** selectedX;

**private** **int** selectedY;

**public** ShuduView(Context context) {

**super**(context);

}

/\*\*

\* w:当前view的宽度

\* h:当前view的高度

\*

\*/

@Override

**protected** **void** onSizeChanged(**int** w, **int** h, **int** oldw, **int** oldh) {

//计算当前单元格的宽度和高度

**this**.width = w / 9f;

**this**.height = h / 9f;

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

}

@Override

**protected** **void** onDraw(Canvas canvas) {

//生成用于绘制背景色的画笔

Paint backgroundPaint = **new** Paint();

//设置画笔的颜色

backgroundPaint.setColor(getResources().getColor(R.color.*shudu\_background*);

//绘制背景色

canvas.drawRect(0, 0,getWidth(),getHeight(),backgroundPaint);

Paint darkPaint = **new** Paint();

darkPaint.setColor(getResources().getColor(R.color.*shudu\_dark*));

Paint hilitePaint = **new** Paint();

hilitePaint.setColor(getResources().getColor(R.color.*shudu\_hilite*));

Paint lightPaint = **new** Paint();

lightPaint.setColor(getResources().getColor(R.color.*shudu\_light*));

/\*\*

\* 绘制用于分割小九宫格的线(即将屏幕分成81个格子)

\*/

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

/\*\*

\* canvas.drawLine(0, i\*height, getWidth(),i\*height, lightPaint)

\* 第1、2个参数: 起点的坐标

\* 第3、4个参数: 终点的坐标

\* 第5个参数: 所使用的画笔

\*/

canvas.drawLine(0, i\*height, getWidth(),i\*height, lightPaint);//划横线

canvas.drawLine(0, i\*height + 1, getWidth(), i\*height + 1, hilitePaint);//也是划横线,为了对比突出那种"刻出来"的效果

canvas.drawLine(i\*width, 0, i\*width, getHeight(), lightPaint);

canvas.drawLine(i\*width + 1, 0, i\*width + 1, getHeight() , hilitePaint);

}

/\*\*

\* 绘制用于将屏幕分成9个小九宫格的线

\* 3、6条线加深颜色

\*/

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

**if**(i % 3 != 0){

**continue**;

}

canvas.drawLine(0, i\*height, getWidth(),i\*height, darkPaint);//划横线

canvas.drawLine(0, i\*height + 1, getWidth(), i\*height + 1, hilitePaint);//也是划横线,为了对比突出那种"刻出来"的效果

canvas.drawLine(i\*width, 0, i\*width, getHeight(), darkPaint);

canvas.drawLine(i\*width + 1, 0, i\*width + 1, getHeight() , hilitePaint);

}

//绘制数字

Paint numberPaint = **new** Paint();

numberPaint.setColor(Color.*BLACK*);

numberPaint.setStyle(Paint.Style.*FILL*); //设置绘图为实心，空心为Style.STROKE

numberPaint.setTextSize(height\*0.75f);

numberPaint.setTextAlign(Paint.Align.*CENTER*);//设置数字对齐方式 ，居中对齐

//文字在矩形方框中居中的方法

FontMetrics fm = numberPaint.getFontMetrics();

**float** x = width / 2;

**float** y = height/2 - (fm.ascent + fm.descent)/2;

//生成数独的初始化数据

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

**for**(**int** j = 0 ; j < 9 ; j++){

canvas.drawText(game.getTileString(i, j), i\*width + x, j\*height + y, numberPaint);

}

}

**super**.onDraw(canvas);

}

@Override

**public** **boolean** onTouchEvent(MotionEvent event) {

**if**(event.getAction() != MotionEvent.*ACTION\_DOWN*){

**return** **super**.onTouchEvent(event);

}

//判断用户点击的是哪一个单元格

selectedX = (**int**)(event.getX() / width);

selectedY = (**int**)(event.getY() / height);

**int** used[] = game.getUsedTilesByCoor(selectedX, selectedY);

StringBuffer sb = **new** StringBuffer();

**for**(**int** i = 0 ; i < used.length ; ++i){//用来验证一下看对不对

sb.append(used[i]);

}

KeyDialog keyDialog = **new** KeyDialog(getContext(),used,**this**);

keyDialog.show();

**return** **true**;

}

**public** **void** setSelectedTile(**int** tile) {

**if**(game.setTileIfValid(selectedX,selectedY,tile)){

invalidate();

}

}

}

3、自定义弹窗类KeyDialog.java

**import** xyz.dawnfan.shududemo.R;

**import** android.app.Dialog;

**import** android.content.Context;

**import** android.os.Bundle;

**import** android.view.View;

/\*\*

\* 该类用于实现Dialog,实现自定义的对话框功能...

\* **@author** Administrator

\*

\*/

**public** **class** KeyDialog **extends** Dialog{

//用来存放代表对话框当中按钮的对象

**private** **final** View keys[] = **new** View[9];

**private** **final** **int** used[];

**private** ShuduView shuduView;

/\*\*

\*

\* 构造函数的第二个参数中保存着当前单元格已经使用过的数字

\* **@param** context

\* **@param** used

\*/

**public** KeyDialog(Context context , **int**[] used , ShuduView shuduView) {

**super**(context);

**this**.used = used;

**this**.shuduView = shuduView;

}

/\*\*

\* 当一个Dialog第一次显示的时候,会调用其onCreate方法

\*/

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

//设置对话框的标题

setTitle("可供选择的数字");

//用于为该Dialog设置布局文件

setContentView(R.layout.*keypad1*);

findViews();

//显示某一单元格中可用的数字

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

**if**(used[i] != 0){

keys[used[i] - 1].setVisibility(View.*INVISIBLE*);

}

}

//为对话框当中所有按钮设置监听器

setListeners();

}

//获取1~9数字

**private** **void** findViews() {

keys[0] = findViewById(R.id.*keypad\_1*);

keys[1] = findViewById(R.id.*keypad\_2*);

keys[2] = findViewById(R.id.*keypad\_3*);

keys[3] = findViewById(R.id.*keypad\_4*);

keys[4] = findViewById(R.id.*keypad\_5*);

keys[5] = findViewById(R.id.*keypad\_6*);

keys[6] = findViewById(R.id.*keypad\_7*);

keys[7] = findViewById(R.id.*keypad\_8*);

keys[8] = findViewById(R.id.*keypad\_9*);

}

/\*\*

\* 通知ShuduView对象,刷新挣个九宫格显示的数据

\* **@param** tile

\*/

**private** **void** returnResult(**int** tile){

System.*out*.println("shuduView: " + shuduView);

shuduView.setSelectedTile(tile);

dismiss();//取消对话框的显示

}

**private** **void** setListeners(){

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

**final** **int** t = i + 1;

keys[i].setOnClickListener(**new** View.OnClickListener() {

@Override

**public** **void** onClick(View v) {

returnResult(t);

}

});

}

}

}

## 3.3 按钮监听

//MainActivity.java类；实现按钮监听和主界面类的调用

**import** xyz.dawnfan.shududemo.R;

**import** android.support.v7.app.ActionBarActivity;

**import** android.os.Bundle;

**import** android.view.Menu;

**import** android.view.MenuItem;

**import** android.view.View;

**import** android.widget.Button;

**public** **class** MainActivity **extends** ActionBarActivity **implements** View.OnClickListener {

//声明控件对象

Button btn1,btn2,btn3;

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*main*);

//setContentView(new ShuduView(this));

btn1 = (Button) findViewById(R.id.*btn\_num1*); //拿到对象

btn2 = (Button) findViewById(R.id.*btn\_num2*);

btn3 = (Button) findViewById(R.id.*btn\_num3*);

btn1.setOnClickListener(**this**);

btn2.setOnClickListener(**this**);

btn3.setOnClickListener(**this**);

}

@Override

**public** **void** onClick(View v) {

ShuduView ch=**new** ShuduView(**this**);

**switch** (v.getId()){

**case** R.id.*btn\_num1*:

setContentView(ch);

**break**;

**case** R.id.*btn\_num2*:

setContentView(ch);

**break**;

**case** R.id.*btn\_num3*:

setContentView(ch);

**break**;

}

}

## 3.4 算法设计

//算法类Game.java

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

//数独初始化数据的基础 将数字显示在矩形框里 空矩形框用0表示

**private** **final** String str = "810000000007490800000006300"

+"060570004570000068900068070"

+"001700000005049600000000035";

**private** **int** sudoku[] = **new** **int**[9\*9];

//用于存储每个单元格已经不可用的数据

**private** **int** used[][][] = **new** **int**[9][9][];

**public** Game() {

sudoku = fromPuzzleString(str);

calculateAllUsedTiles();

}

/\*\*

\* 根据一个字符串数据,生成一个整型数组,所谓数独游戏的初始化数据

\* **@param** src

\* **@return**

\*/

**protected** **int**[] fromPuzzleString(String src){

**int**[] sudo = **new** **int**[src.length()];

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

sudo[i] = src.charAt(i) - '0';

}

**return** sudo;

}

/\*\*

\* 根据九宫格当中的坐标,返回该坐标所应该填写的数字

\* **@param** x

\* **@param** y

\* **@return**

\*/

**private** **int** getTile(**int** x, **int** y){

**return** sudoku[y\*9 + x];

}

/\*\*

\* 根据x轴坐标和y轴坐标得到这一单元格不可用的数据

\* **@param** x

\* **@param** y

\* **@return**

\*/

**public** String getTileString(**int** x, **int** y){

**int** v = getTile(x,y);

**if**(v == 0){

**return** "";

}**else**{

**return** String.*valueOf*(v);

}

}

/\*\*

\* 计算所有单元格对应的不可用的数据

\*/

**public** **void** calculateAllUsedTiles(){

**for**(**int** x = 0 ; x < 9 ; ++x){

**for**(**int** y = 0 ; y < 9 ; ++y){

used[x][y] = calculateUsedTiles(x, y);

}

}

}

/\*\*

\* 取出某一单元格中已经不可用的数据

\* **@param** x

\* **@param** y

\* **@return**

\*/

**public** **int**[] getUsedTilesByCoor(**int** x, **int** y){

**return** used[x][y];

}

/\*\*

\* 计算某一单元格之中已经不可用的数据

\* **@param** x

\* **@param** y

\*/

**public** **int**[] calculateUsedTiles(**int** x,**int** y) {

**int** c[] = **new** **int**[9];

//计算在y轴(列)方向上那些数字不可用

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

**if**(i == y){//如果这是用户点击的格子

**continue**;

}

**int** t = getTile(x,i);

**if**(t != 0){

c[t - 1] = t;

}

}

//计算在x轴(行)方向上那些数字不可用

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

**if**(i == x){

**continue**;

}

**int** t = getTile(i,y);

**if**(t != 0){

c[t - 1] = t;

}

}

//计算在小的九宫格中有那些数字已经用过了

**int** startX = (x/3)\*3;

**int** startY = (y/3)\*3;

**for**(**int** i = startX ; i < startX + 3 ; ++i){

**for**(**int** j = startY ; j < startY + 3 ; ++j){

**if**(i == x && j == y){

**continue**;

}

**int** t = getTile(i, j);

**if**(t != 0 ){

c[t - 1] = t;

}

}

}

//去掉数组c中的0,将非0的数组中的数字累计为nused

**int** nused = 0;

**for**(**int** t : c){

**if**(t != 0){

nused++;

}

}

//定义新数组，大小为nused

**int** c1[] = **new** **int**[nused];

nused = 0;

**for**(**int** t : c){

**if**(t != 0){

c1[nused++] = t;

}

}

**return** c1;

}

**public** **boolean** setTileIfValid(**int** x, **int** y, **int** value) {

**int** tiles[] = getUesdTiles(x,y);

**if**(value != 0){

**for**(**int** tile : tiles){

**if**(tile == value){

**return** **false**;

}

}

}

setTile(x,y,value);//把用户输入的数字添加到九宫格中

calculateAllUsedTiles();//更新该单元格可以使用的数字

**return** **true**;

}

**protected** **int**[] getUesdTiles(**int** x, **int** y) {

**return** used[x][y];

}

**private** **void** setTile(**int** x, **int** y, **int** value){

sudoku[y\*9 + x] = value;

}

}

## 3.5 存储设计

本程序所有的存储全用数组实现

**private** **final** String str = "810000000007490800000006300"

+"060570004570000068900068070"

+"001700000005049600000000035";

/\*

\* str = "360000000004230800000004200"

+"070460003820000014500013020"

+"001900000007048300000000045";

\*

\*

\* \*/

**private** **int** sudoku[] = **new** **int**[9\*9];

//用于存储每个单元格已经不可用的数据

**private** **int** used[][][] = **new** **int**[9][9][];

//获取1~9数字

**private** **void** findViews() {

keys[0] = findViewById(R.id.*keypad\_1*);

keys[1] = findViewById(R.id.*keypad\_2*);

keys[2] = findViewById(R.id.*keypad\_3*);

keys[3] = findViewById(R.id.*keypad\_4*);

keys[4] = findViewById(R.id.*keypad\_5*);

keys[5] = findViewById(R.id.*keypad\_6*);

keys[6] = findViewById(R.id.*keypad\_7*);

keys[7] = findViewById(R.id.*keypad\_8*);

keys[8] = findViewById(R.id.*keypad\_9*);

}

Keypad1.xml自定义弹窗代码实现

<TableLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:id=*"@+id/keypad"*

android:orientation=*"vertical"*

>

<TableRow >

<Button android:id=*"@+id/keypad\_1"*

android:text=*"1"*

/>

<Button android:id=*"@+id/keypad\_2"*

android:text=*"2"*

/>

<Button android:id=*"@+id/keypad\_3"*

android:text=*"3"*

/>

</TableRow>

<TableRow >

<Button android:id=*"@+id/keypad\_4"*

android:text=*"4"*

/>

<Button android:id=*"@+id/keypad\_5"*

android:text=*"5"*

/>

<Button android:id=*"@+id/keypad\_6"*

android:text=*"6"*

/>

</TableRow>

<TableRow >

<Button android:id=*"@+id/keypad\_7"*

android:text=*"7"*

/>

<Button android:id=*"@+id/keypad\_8"*

android:text=*"8"*

/>

<Button android:id=*"@+id/keypad\_9"*

android:text=*"9"*

/>

</TableRow>

</TableLayout>

# 第四章 测试分析

本次程序设计由Adroid studio和eclipse编辑器均可导入操作，最后通过Genymotion手机模拟器、Eclipse自带模拟器、Android自带模拟器、华为真机等设备均测试无误。但存在有的功能基本没能完善，用户需求较不完善等。（测试结果如图4.1、图4.2、图4.3、图4.4所示）



图4.1



图4.2

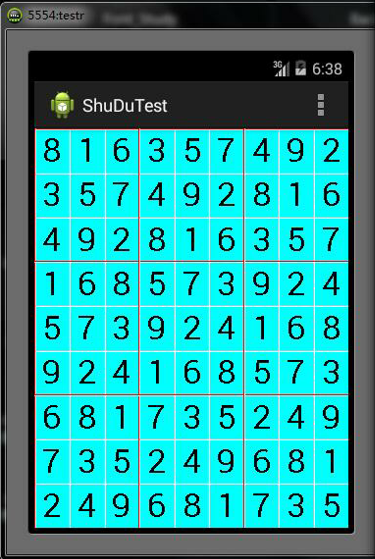
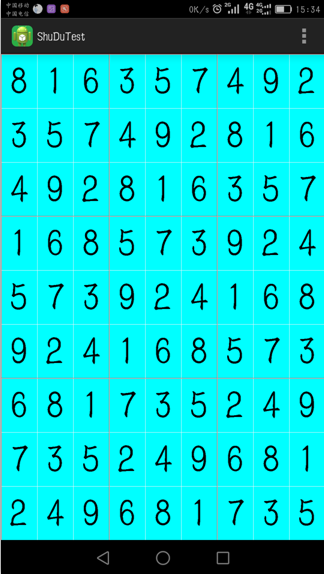
 

图4.3 图4.4

# 第五章 总 结

本次课程设计，让我获益匪浅，以前在学习Android时，虽然上课也好好听老师讲课，但是在课后自己亲自动手实践的不太多，现在有许多知识都忘记了，所以导致编写程序的时候不能有效完善。同时在程序的设计中思想存在许多不足，导致最后代码的不完善，遗留问题居多。当然，在开发数独游戏的时候，我会不停的翻阅java与Android资料，同时也会借助网络上的数独的相关资料来实现。通过这次安卓软件开发的课程设计，我更加深入的了解了java和Android这两门学科，更使自己有了继续探索的兴趣。

于个人而言，在程序设计的过程中，我深感“认真严谨”这个词的重要性，一点点小的马虎，便会导致整个程序不能正常运行。在今后的学习生活中，我定将“认真严谨”时刻最为自己的谨言。与此同时，通过这次课程设计，让我也学习到了开发一个软件、一个程序应该具备的技能。墙高基下，虽得必失。切记避免盲目的敲代码，一定要有思路的去开发，在以后的程序道路上一定要避免眼高手低，从最基础的做起，多练、多看，才会在开发程序的时候不至于太过于迷茫。总之，此次课程设计在我的学生生涯中起到了至关重要的作用。

# 第六章 参考文献

[1] 马越. Android 的架构与应用[D]. 北京：中国地质大学，2008:330-357.

[2] 李刚. 疯狂 Android 讲义[M]. 北京：电子工业出版社，2013:25-42.

[3] 杨丰盛. Android 技术内幕[M]. 北京： 机械工业出版社，2011:77-89.

[4] 柯元旦. Android 内核剖析[M]. 北京：电子工业出版社，2011:59-70.

[5] 丰生强. Android 软件安全与逆向分析[M]. 北京：人民邮电出版社，2013:78-90.

[6] 杨云君. Android 的设计与实现[M]. 北京：机械工业出版社，2013:45-49.