**基于Android的2048游戏的设计与实现**

**摘 要**

现如今，电子游戏已慢慢渗透进人们生活中，并在扮演着越来越重的角色。可以说，随着他的迅猛发展，现今的电子游戏已经不仅仅是一种娱乐，而是形成了一种文化现象，具有很强的民族性和丰富的文化内涵。伴随着游戏软件在市场的销售，与其相关的文化也随之传播。

本文通过对Android应用开发的初步研究，设计出了一款基于Android平台的2048小游戏。本文是在通过对系统进行详细分析的情况下，基本的确定了系统的相关的功能要求，对游戏的关卡设置、菜单界面、屏幕绘制和布局进行了设计，实现了一款界面清晰、内容丰富的2048游戏，它能够让人们在这个快节奏的生活压力中得到缓解。

**关键字：**Android；游戏；益智

# 1 开发环境的介绍和搭建

本章是对Android应用开发的开发环境进行简单的介绍以及如何搭建。

## 1.1 开发环境的介绍

### 1.1.1 JDK的介绍

Java Development Kit(JDK) 是Java语言的软件开发工具包。Java的核心就是JDK，JDK主要包括了Java基础类库、Java运行环境以及Java工具。它主要是为Java程序员开发而发布的免费开发工具包[5,6]。

### 1.1.2 Eclipse的介绍

Eclipse是一个基于Java的和开源的开发平台，刚开始只是用于Java语言的开发，现在人们使用一些外挂程式使得Eclipse可以开发一些其他的计算机语言。Eclipse对他本身而言它只是一个框架[平台](http://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=3467677" \t "http://baike.sogou.com/_blank)，但最多的外挂程式使得它变得具有灵活性[7]。

### 1.1.3 Android SDK的介绍

SDK（[software](http://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=7495170) development kit）是[软件](http://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=5357" \t "http://baike.sogou.com/_blank)开发[工具包](http://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=57704855)。Android SDK 指的是Android专属的软件开发工具包[8]。

## 1.2 开发环境的搭建

### 1.2.1 JDK的安装与环境变量配置

(1）从网上下载JDK1.6或者JDK1.7版本。

(2）双击文件安装JDK。

(3）环境变量的配置：右击“我的电脑”-->"属性"-->“高级系统设置”-->"环境变量"[9]。环境变量的设置界面分“用户变量”和“系统变量”[10]。在系统变量里点击新建在变量名中输入“JAVA\_HOME”，在变量值中填入JDK的安装路径；接着新建classpath变量，变量值设为：.;%JAVA\_HOME%\lib;%JAVA\_HOME%\lib\tools.jar，然后双击“系统变量”中存在的path在弹出的对话框中添加变量值：%JAVA\_HOME%\bin;%JAVA\_HOME%\jre\bin，最后运行->cmd，输入javac，如果出现用法说明，则表示成功[11,12]。

### 1.2.2 Android SDK 与Eclipse的安装与配置

在网上搜索adt-bundle-windows并下载。下好之后直接解压缩，此文件为已经集成了Eclipse和Android SDK 的完整版，这样就可以适合一些新手。

### 1.2.3 Android虚拟机AVD的配置

在Eclipse菜单栏中点击Window->Android Virtual Device Manager[13],在弹出的窗口中点击“new”按钮，进入到创建的AVD配置窗口，可以创建自己想要的AVD[14]，如图1.1所示：

## 

**图1.1 AVD配置图**

配置成功后，选中创建好的AVD并点击“start”按钮弹出对话框点击Launch按钮，由于AVD第一次启动时间较长，需耐心等待[15]。开启成功后如图1.2所示：



**图1.2 AVD界面图**

## 1.3 本章小结

本章是对Android应用开发所需的开发环境进行简单的介绍和对开发环境如何搭建进行介绍。

# 2 系统的需求分析和概要设计

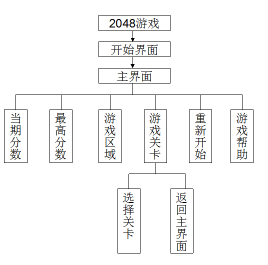
这章主要讲述了系统的需求分析，简单的描述了各个功能模块和游戏的基本规则，然后是对系统概要设计的介绍。

## 2.1系统需求分析

### 2.1.1系统功能需求分析

系统主要实现以下的几个功能：关卡选择、呈现游戏界面、重新开始游戏、当前分数和最高分数、游戏帮助等功能。

关卡选择是当玩家点击此按钮时，游戏就会从主界面跳转到关卡选择界面，当玩家选择相应的关卡时，先判断此关卡是否开启，如果没有开启此关卡就会弹出对话框提示玩家此关卡未开启，如果此关卡已经开启就会跳转到主界面同时开始此关卡。重新开始游戏是当玩家无法满足当前进度时点击此按钮就会重新开始游戏，如果玩家处于不同关卡时重新开始游戏还是停留在此关卡。游戏帮助是当新手玩此游戏时无法知道游戏玩法时给予相应的提示。呈现游戏界面是游戏开始时主界面在游戏区域会生成4×4的矩阵同时在矩阵里面随机生成两个2或4的卡片。当前分数和最高分数是显示此局玩家所获得的分数和历史最高的分数，如果当前的分数超过最高分数那么最高分显示当前的分数，如图2.1所示。



**图2.1系统功能图**

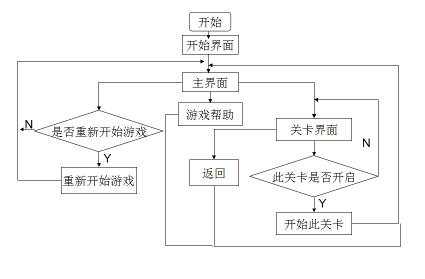
### 2.1.2游戏基本规则

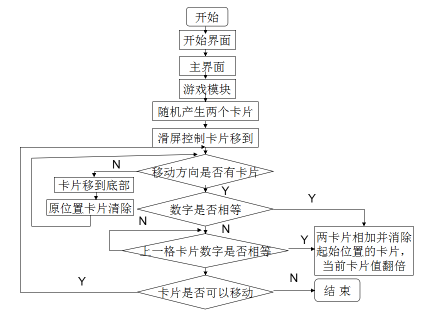
在开始游戏后玩家通过滑动屏幕来操控卡片的移动方向，当卡片滑动中如果有两张卡片相同且他们的中间也没有其他卡片时，在滑动的过程中这两张卡片会合并，显示为这两张卡片之和。在滑动之中有三张卡片相同时只会合并向滑动方向两张卡片。在滑动中如果有两张卡片一样同时又有一张卡片的值跟这两张卡片相加的值时，滑动只会使那两张相同的卡片合并而不会接着让合并后的卡片和另一张卡片合并。

## 2.2系统概要设计

### 2.2.1系统流程的设计

游戏开始进入开始页面，能够进入游戏的主界面并开始普通开局，从主界面能够重新开始游戏、查看帮助和进入关卡选择界面。当玩家点击重新开始按钮会弹出相应的对话框让玩家选择，如果玩家选择“是”时则重新开始游戏，如果选择“否”则返回游戏界面不做任何处理。当玩家点击关卡按钮会跳转到关卡选择界面，关卡界面将显示所有的关卡，当玩家点击相应的关卡时后台会判断此关卡是否开启，如果已经开启将会跳转到主界面并运行此关卡，如果没有开启将给予玩家提示“此关卡未开启”对话框。在开始界面按返回按钮时则会退出游戏。游戏的流程图如图2.2所示：





**图2.2系统流程图**

### 2.2.2系统模块设计

从总体出发，将该系统划分为三大模块：“菜单设计”，“界面设计”和“算法设计”。

2.2.2.1 菜单设计

菜单的实现是在游戏界面实现，可进一步划分为三个模块，分别是：“游戏帮助”，“重新开始”，“关卡选择”，如图2.3所示：

重新开始

关卡选择

游戏帮助

**图2.3菜单设计**

2.2.2.2界面设计

可进一步划分为：“主界面设计”，“游戏界面设计”和“关卡界面设计”，如图2.4所示：

开始游戏

关卡选择

分数

游戏区域

重新

帮助

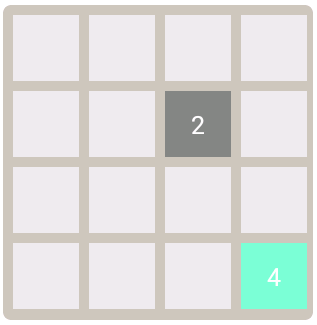
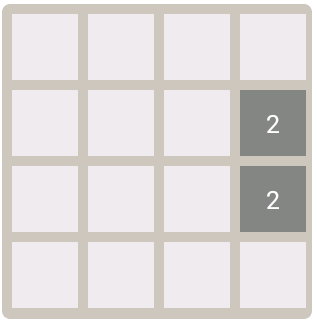
关卡

1. 开始界面 （b）游戏界面 （c）关卡选择界面

**图2.4 界面设计**

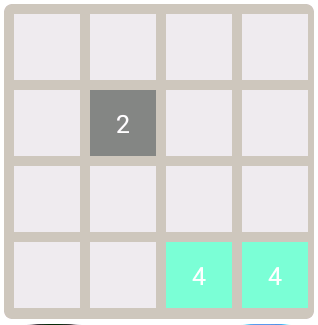
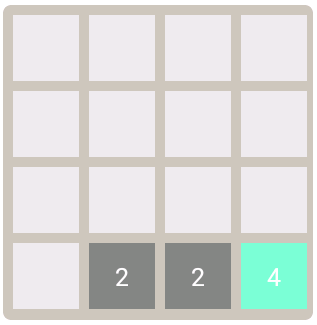
2.2.2.3算法设计

(1)当有两张卡片相同时，向它们可以碰撞的方向滑屏，卡片会移动到最底边并生成其两倍数字的卡片，并且生成一个“2”或“4”的卡片，如图2.5所示：



**图2.5 简单卡片合成**

(2)当有两张卡片相同时，且在他相同的方向有张跟它们之和的卡片，向它们可以碰撞的方向滑屏，相同的卡片会移动到无法移动的位置并生成期两倍数字的卡片，但合成的卡片不会跟那张两倍数字的卡片合并，并且生成一个“2”或“4”的卡片，如图2.6所示：



**图2.6 复杂卡片合成**

1. 当界面上没有空位并且两两相邻的卡片不相同时游戏结束。如图2.7所示：



**图2.7游戏结束**

## 2.3 本章小结

本章主要对游戏所实现的功能进行需求分析，对图形和系统性能进行了分析，分析了图形的特点和实现的可行性。对系统的性能进行了详细的分析同时也提出来解决性能问题的解决方案。 对系统的流程，系统所需的图形文件，系统的总体架构和系统用例进行了设计。通过本章的分析、设计能更加具体的了解系统功能，对系统所要实现的功能和图形文件有了更深的认识。为下一章系统功能的具体实现提供了可靠的参考依据。

# 3 系统实现

本章主要是详细介绍每个功能是怎样的实现。

## 3.1 开始界面的实现

游戏的主界面是按钮图片，只是实现了界面的跳转，当玩家点击此界面时就会调用ZuomianActivity.java此函数让页面跳转到游戏界面开始游戏,如图3.1所示：

public class ZuomianActivity extends Activity{

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main0);

}



**图3.1 开始界面**

## 3.2游戏界面的实现

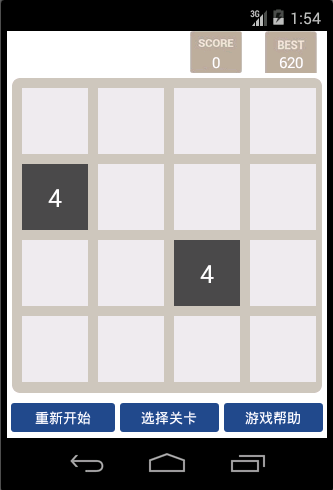
游戏界面主要是在activity\_main.xml中当前分数、最高分数、游戏区域、重新开始按钮、关卡选择按钮、帮助按钮，当跳转到游戏界面时就会调用并执行MainActivity.java函数来展示游戏界面，如图3.2所示。

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

}



**图3.2 主界面**

## 3.3 游戏滑屏卡片移动的实现

当玩家滑动屏幕时，主要是通过initGameView函数来监听玩家手指滑动的位置，先通过获取开始坐标和结束坐标，然后通过比较结束坐标跟开始坐标的差值来判断玩家是怎样滑动屏幕的。判断出玩家的滑动轨迹后，通过调用swipeLeft、swipeRight、swipeUp、swipeDown方法来实现卡片的移动，如图3.3所示。

public void initGameView() {

setColumnCount(4);

setOnTouchListener(new OnTouchListener() {

private float startX, startY, offsetX, offsetY;

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)) {

if (offsetX < -5) {

swipeLeft();

} else if (offsetX > 5) {

swipeRight();

}

} else {

if (offsetY < -5) {

swipeUp();

} else if (offsetY > 5) {

swipeDown();

}

}

break;

}

return true;

}

});

}

private void swipeLeft() {

boolean merge = false;

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

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

for (int x1 = x + 1; x1 < 4; x1++) {

if (cardsMap[x1][y].getNum() > 0) {

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

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

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

x--;

merge = true;

} else if (cardsMap[x][y].equals(cardsMap[x1][y])) {

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

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

MainActivity.getMainActivity().addScore(

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

merge = true;

}

break;

}

}

}

}

if (merge) {

addRandomNum();

checkComplete();

}

}

private void swipeRight() {

boolean merge = false;

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

for (int x = 3; x >= 0; x--) {

for (int x1 = x - 1; x1 >= 0; x1--) {

if (cardsMap[x1][y].getNum() > 0) {

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

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

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

x++;

merge = true;

} else if (cardsMap[x][y].equals(cardsMap[x1][y])) {

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

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

MainActivity.getMainActivity().addScore(

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

merge = true;

}

break;

}

}

}

}

if (merge) {

addRandomNum();

checkComplete();

}

}

private void swipeUp() {

boolean merge = false;

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

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

for (int y1 = y + 1; y1 < 4; y1++) {

if (cardsMap[x][y1].getNum() > 0) {

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

cardsMap[x][y].setNum(cardsMap[x][y1].getNum());

cardsMap[x][y1].setNum(0);

y--;

merge = true;

} else if (cardsMap[x][y].equals(cardsMap[x][y1])) {

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

cardsMap[x][y1].setNum(0);

MainActivity.getMainActivity().addScore(

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

merge = true;

}

break;

}

}

}

}

if (merge) {

addRandomNum();

checkComplete();

}

}

private void swipeDown() {

boolean merge = false;

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

for (int y = 3; y >= 0; y--) {

for (int y1 = y - 1; y1 >= 0; y1--) {

if (cardsMap[x][y1].getNum() > 0) {

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

cardsMap[x][y].setNum(cardsMap[x][y1].getNum());

cardsMap[x][y1].setNum(0);

y++;

merge = true;

} else if (cardsMap[x][y].equals(cardsMap[x][y1])) {

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

cardsMap[x][y1].setNum(0);

MainActivity.getMainActivity().addScore(

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

merge = true;

}

break;

}

}

}

}

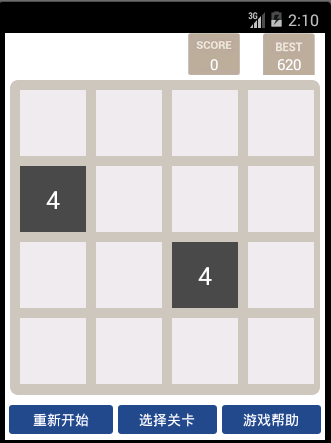
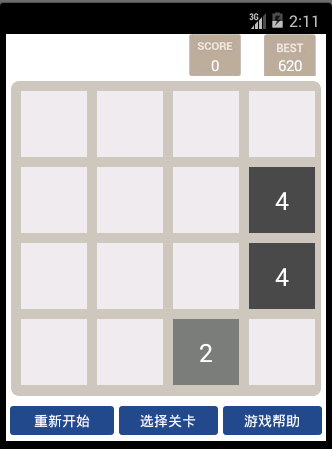
if (merge) {

addRandomNum();

checkComplete();

}

}

**图3.3 卡片移到**

## 3.4 重新开始游戏功能的实现

当玩家点击游戏界面的重新开始游戏时，会弹出给玩家选择的对话框，让玩家选择是否重新开始游戏。当玩家选择“是”时游戏会重新开始，重新开始游戏会根据玩家当前所选的关卡来重置游戏，如果玩家选择“否”时，游戏将继续下去，如图3.4所示。

btnNewGame = (Button) findViewById(R.id.btnNewGame);

btnNewGame.setOnClickListener(new View.OnClickListener() {

public void onClick(View v) {

new AlertDialog.Builder(MainActivity.this)

.setTitle(" 是否重新再来！")

.setNegativeButton("取消",new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog,int which) {

}

})

.setPositiveButton("确定",new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog,int which) {

if (getGameid() == 0)

gameView.startGame();

else if (getGameid() == 1)

gameView.startGame64();

else if (getGameid() == 2)

gameView.startGame128();

else if (getGameid() == 3)

gameView.startGame256();

else if (getGameid() == 4)

gameView.startGame512();

else if (getGameid() == 5)

gameView.startGame1024();

else if (getGameid() == 6)

gameView.startGame2048();

else if (getGameid() == 7)

gameView.startGame4096();

else if (getGameid() == 8)

gameView.startGame8192();

else if (getGameid() == 9)

gameView.startGame16384();

else if (getGameid() == 10)

gameView.startGame32768();

else if (getGameid() == 11)

gameView.startGame65536();

}

}).show();

}

});

}



**图3.4 重新开始游戏**

## 3.5 关卡的实现

当玩家的分数达到一定的高度的时候就会解开相应的关卡，玩家可以在游戏界面点击游戏关卡的按钮，后台会通过调用MainActivity中关卡相关的代码跳转到关卡界面执行GqActivity将写好的activity\_guanqia.xml显示出来，当玩家点击相应的关卡时会传递相应的值给MainActivity，MainActivity根据传过来的值调用相应的开局方式，如果点击的关卡已开启则跳转到游戏界面开启此关卡，如果没有开启则弹出对话框提示玩家此关卡没开启，如图3.6所示。

public class GqActivity extends Activity {

private Button fanhui;

private Button butpt;

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_guanqia);

fanhui = (Button) findViewById(R.id.fanhui);

fanhui.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View arg0) {

Intent intent = new Intent(GqActivity.this, MainActivity.class);

startActivity(intent);

}

});

butpt = (Button) findViewById(R.id.butpt);

butpt.setOnClickListener(new View.OnClickListener() {

MainActivity mat = MainActivity.getMainActivity();

public void onClick(View arg0) {

Intent intent = new Intent(GqActivity.this, MainActivity.class);

setResult(0, intent);

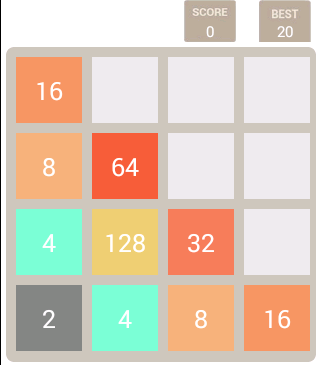
finish();

}

});

}





**图3.6 关卡功能**

## 3.6 游戏帮助的实现

当新玩家进入到游戏且不知道此游戏玩法，玩家可以点击游戏帮助按钮来了解游戏玩法，点击按钮时游戏会弹出对话框显示游戏玩法，如图3.7所示。

btnHelpInfo.setOnClickListener(new View.OnClickListener() {

public void onClick(View v) {

showGameHelpInfo();

}

});

private void showGameHelpInfo() {

TextView view = new TextView(MainActivity.this);

view.setText(R.string.gametactics);

view.setMaxLines(10);

new AlertDialog.Builder(MainActivity.this).setTitle("2048游戏攻略")

.setView(view)

.setNeutralButton("OK", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int which) {

dialog.dismiss();

}

}).show();

}



**图3.7 游戏帮助**

## 3.7 本章小结

本章主要阐述本游戏相关功能的实现，详细的讲述了主界面的实现和个按钮功能的实现。

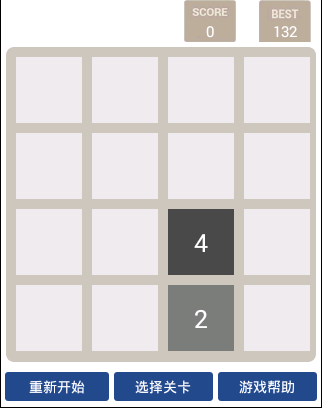
# 4 测 试

本章主要对系统的功能进行测试，此次的测试只是进行简单的调试，来确定游戏的各项功能是否能够正常运行。

## 4.1 游戏系统测试

### 4.1.1游戏流程测试

该测试主要验证游戏能否实现场景的切换，当界面在开始界面时只显示按钮画面如图4.1所示，当玩家点击此界面时跳转到游戏界面如图4.2所示，最后是关卡选择时界面的跳转，当玩家点击关卡选择按钮时界面会跳转到关卡选择界面如图4.3所示，当玩家选择相应的关卡时就会跳转到游戏界面如图4.4所示为选择相应的关卡后跳转的界面。

**图4.1开始界面**  **图4.2游戏界面**

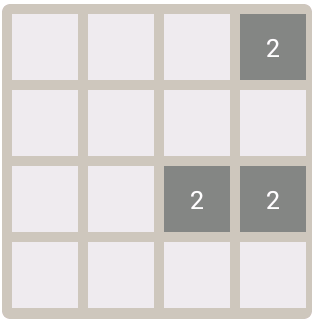
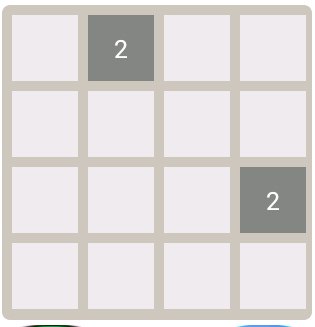


**图4.3关卡界面**

### 4.1.2 游戏模式

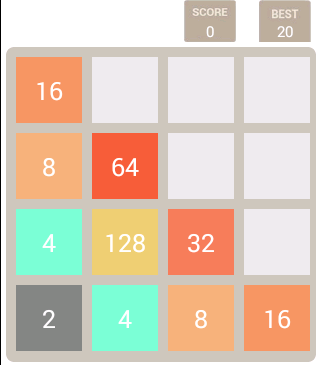
该测试主要是测试游戏能否正常的运行，当玩家滑动屏幕时卡片能否正常的移动和当卡片相同时是否能够相加，还有就是测试游戏是否正常结束，玩家选择相应的关卡是否能够执行相应的关卡。

①测试游戏能否正常运行，如图4.4所示：



**图4.4游戏是否正常运行测试**

②测试关卡选择，如图4.5所示：



**图4.5关卡测试**

## 4.2 本章小结

本章是对游戏系统进行简单的测试，通过测试可以看出此游戏可以正常的工作，同时一些功能也能够实现。

# 参考文献

[1] [黄彬华](http://www.dangdang.com/author/%BB%C6%B1%F2%BB%AA_1" \t "http://product.dangdang.com/_blank).Android手机/平板电脑程序开发教练[M].北京:[水利水电出版社](http://www.dangdang.com/publish/%CB%AE%C0%FB%CB%AE%B5%E7%B3%F6%B0%E6%C9%E7_1" \t "http://product.dangdang.com/_blank),2012:58-65.

[2] 李嘉诚.Android应用程序开发研究与应用[J].软件导刊,2014，13(11):65-67.

[3] 崔浩然.基于Android平台的手机游戏的设计与实现[D].西安科技大学,2011.

[4] 王亮.基于Android平台冒险类手机游戏的设计与实现[D].北京工业大学，2013.

[5] [孙宏明.](http://search.china-pub.com/s/?key1=%cb%ef%ba%ea%c3%f7" \t "http://product.china-pub.com/_blank)Android手机程序设计入门、应用到精通[M].北京:[中国水利水电出版社](http://www.china-pub.com/love/chubanshe/chubanshe.asp?newid=98" \t "http://product.china-pub.com/_blank),2012:78-89.

[6] [郭少豪](http://search.china-pub.com/s/?key1=%b9%f9%c9%d9%ba%c0" \t "http://product.china-pub.com/_blank).创意之钥:Android手机交互应用开发[M].北京:[中国铁道出版社](http://www.china-pub.com/love/chubanshe/chubanshe.asp?newid=122" \t "http://product.china-pub.com/_blank),2011:230-256.

[7] [杨丰盛](http://search.china-pub.com/s/?key1=%d1%ee%b7%e1%ca%a2" \t "http://product.china-pub.com/_blank).Android技术内幕[M].北京:[机械工业出版社](http://www.china-pub.com/love/chubanshe/chubanshe.asp?newid=549" \t "http://product.china-pub.com/_blank)，2011:134-165.

[8] 孟晓龙.Win7系统下Android开发平台的搭建[J].科协论坛,2011,1(08):72-73.

[9] 吴想想.基于Android平台软件开发方法的研究与应用[D].北京邮电大学，2011.

[10] 王琳.基于Android平台音频播放系统的设计与实现[D].吉林大学,2013.

[11] [李宁.](http://search.china-pub.com/s/?key1=%c0%ee%c4%fe" \t "http://product.china-pub.com/_blank)Android开发完全讲义[M].北京:[中国水利水电出版社](http://www.china-pub.com/love/chubanshe/chubanshe.asp?newid=98" \t "http://product.china-pub.com/_blank)，2012:89-102.

[12] 王明超.基于Android游戏模式研究与设计[J].电脑知识与技术,2012,8(36):8776-8779.

[13] 韩超.Android经典应用程序开发[M].北京:[电子工业出版社](http://www.china-pub.com/love/chubanshe/chubanshe.asp?newid=499" \t "http://product.china-pub.com/_blank),2012:79-97.

[14] 杨云君.Android的设计与实现[M].北京:[机械工业出版社](http://www.china-pub.com/love/chubanshe/chubanshe.asp?newid=549" \t "http://product.china-pub.com/_blank),2013:323-341.

[15] 陈淮.基于Android游戏开发中常用类库的设计与实现[D].西安电子科技大学,2010.