**基于IOS的2048游戏说明文档**

1511班－nb15033－王国军

**1、选题依据及游戏介绍**

最近以来，移动手游越来越成为当下游戏产业中重要的一环，市场也在加大对这一产业的投入，涌现出了愤怒的小鸟，水果忍者，flappy bird，2048等诸多优秀的手游，然而任何一款手游能够风靡，主要还是依赖其简单的操作性，因此本次IOS课程大作业我选择其中经典的2048游戏作为本次作业的课题。

在前不久，新出的一款叫做2048的手游十分的火爆，几乎游戏下载排行榜的前10名都能看到“它的身影”。然而并不是每个人都知道一件事情，那是一个关于“2048”诞生的故事，故事的发展是这样的，“2048”的祖先是一款名为“Threes”（中文名为“小3传奇”）的手游。“小3传奇”的两位开发者花了近一年半的时间才开发出这个游戏的核心玩法，却在其上架App Store后仅21天就被“1024”抄了过去。最让人意想不到的是，“2048”则更为迅速，19岁的Gabriele Cirulli只用了一个星期改编前两者游戏，却获得最大的成功，“Threes”却不被人知。而到了现在，这类游戏从开发到现在，不管是在App Store还是Android市场，下载量已经远远超越了千万，网上各个版本的都有，比如朝代版、后宫版、甲乙丙版、生肖版等。

由于本人非常喜欢“2048”这款游戏，在了解本次IOS大作业要基于IOS开发一款手机应用之后，就选定“2048”作为本次课程大作业的课题，在整个大作业的过程中，我运用IOS课堂学到的知识以及OC语言基础知识，通过网上现有资料的查阅，完成了这款经典的手机游戏。

**2、基本流程**

系统程序流程图，如下图所示：

开始

随机产生两个方块

键盘控制移动方块

移动方向最底部的方格是否有方块

方块移到底部

数字是否相等

原来位置方块清除

上一格方格数字是否相等

两方块数字相加把上面的行数据下移并置

方块是否能够移动

结束游戏

**3、功能说明**

（1）实现可以显示2或4两张数字的方块；

（2）实现各个方块上下左右移动；

（3）实现方块之间数字判断与相加。

**4、主要算法的设计**

**4.1 方块值的随机数产生**

-(void)bornNewLabel

{

if(self.currentExistArray.count<16){

//设置空余方格的数列

[self.emptyPlaceArray removeAllObjects];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:11]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:21]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:31]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:41]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:12]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:22]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:32]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:42]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:13]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:23]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:33]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:43]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:14]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:24]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:34]];

[self.emptyPlaceArray addObject:[NSNumber numberWithInt:44]];

for (CLLuckyLabel \*label in self.currentExistArray) {

[self.emptyPlaceArray removeObject:[NSNumber numberWithInt:label.placeTag]];

}

//随机数必须小于空余方格的个数

int random;

if (self.emptyPlaceArray.count>1) {

random=arc4random()%(self.emptyPlaceArray.count-1);

}else{

random=0;

}

//每次移动新增一个随机数字，设置每次移动的数字随机位置

NSNumber \*place=[self.emptyPlaceArray objectAtIndex:random];

CLLuckyLabel \*label=[[CLLuckyLabel alloc]init];

label.placeTag=[place intValue];

NSDictionary \*dic=[self caculatePosition:place];

//设置每次移动的数字大小

int random2=arc4random()%2;

NSNumber \*textNumber=[self.labelArray objectAtIndex:random2];

label.numberTag=[textNumber intValue];

label.text=[NSString stringWithFormat:@"%@",textNumber];

#warning flaffffff

label.backgroundColor = [self getLabelColor:label.text.intValue];

CGRect frame=CGRectMake([[dic objectForKey:kPlaceX] intValue], [[dic objectForKey:kPlaceY] intValue] , kOneLabelwidth, kOneLabelHeight);

label.frame=frame;

[self.currentExistArray addObject:label];

[self.view addSubview:label];

}

else{

[self isGameOver];

}

}

**4.2 鼠标移动的算法实现**

-(void)moveLabel:(int) directionFlag

{

NSMutableArray \*array1 = [[NSMutableArray alloc]initWithCapacity:4];

NSMutableArray \*array2 = [[NSMutableArray alloc]initWithCapacity:4];

NSMutableArray \*array3 = [[NSMutableArray alloc]initWithCapacity:4];

NSMutableArray \*array4 = [[NSMutableArray alloc]initWithCapacity:4];

switch (directionFlag)

{

case 1: //right

for (CLLuckyLabel \*label in self.currentExistArray) {

switch (label.placeTag/10) {

case 4:

[array1 addObject:label];

break;

case 3:

[array2 addObject:label];

break;

case 2:

[array3 addObject:label];

break;

case 1:

[array4 addObject:label];

break;

default:

break;

}

}

for (CLLuckyLabel \*childLabel in array2) {

[self checkFrontLabel:childLabel andDirection:1];

}

for (CLLuckyLabel \*childLabel in array3) {

[self checkFrontLabel:childLabel andDirection:1];

}

for (CLLuckyLabel \*childLabel in array4) {

[self checkFrontLabel:childLabel andDirection:1];

}

break;

case 2: //left

for (CLLuckyLabel \*label in self.currentExistArray) {

switch (label.placeTag/10) {

case 1:

[array1 addObject:label];

break;

case 2:

[array2 addObject:label];

break;

case 3:

[array3 addObject:label];

break;

case 4:

[array4 addObject:label];

break;

default:

break;

}

}

for (CLLuckyLabel \*childLabel in array2) {

[self checkFrontLabel:childLabel andDirection:2];

}

for (CLLuckyLabel \*childLabel in array3) {

[self checkFrontLabel:childLabel andDirection:2];

}

for (CLLuckyLabel \*childLabel in array4) {

[self checkFrontLabel:childLabel andDirection:2];

}

break;

case 3: // up

for (CLLuckyLabel \*label in self.currentExistArray) {

switch (label.placeTag%10) {

case 1:

[array1 addObject:label];

break;

case 2:

[array2 addObject:label];

break;

case 3:

[array3 addObject:label];

break;

case 4:

[array4 addObject:label];

break;

default:

break;

}

}

for (CLLuckyLabel \*childLabel in array2) {

[self checkFrontLabel:childLabel andDirection:3];

}

for (CLLuckyLabel \*childLabel in array3) {

[self checkFrontLabel:childLabel andDirection:3];

}

for (CLLuckyLabel \*childLabel in array4) {

[self checkFrontLabel:childLabel andDirection:3];

}

break;

case 4: // down

for (CLLuckyLabel \*label in self.currentExistArray) {

switch (label.placeTag%10) {

case 4:

[array1 addObject:label];

break;

case 3:

[array2 addObject:label];

break;

case 2:

[array3 addObject:label];

break;

case 1:

[array4 addObject:label];

break;

default:

break;

}

}

for (CLLuckyLabel \*childLabel in array2) {

[self checkFrontLabel:childLabel andDirection:4];

}

for (CLLuckyLabel \*childLabel in array3) {

[self checkFrontLabel:childLabel andDirection:4];

}

for (CLLuckyLabel \*childLabel in array4) {

[self checkFrontLabel:childLabel andDirection:4];

}

break;

default:

break;

}

}

**4.3 判断游戏是否结束的算法实现**

-(void)isGameOver

{

self.isOver = YES;

// [self moveLabel:1];

// [self moveLabel:2];

// [self moveLabel:3];

// [self moveLabel:4];

bestScore=[[NSUserDefaults standardUserDefaults] integerForKey:@"bestScore"];

if (self.isOver == YES) {

if (totalScore>bestScore) {

bestScore=totalScore;

[[NSUserDefaults standardUserDefaults] setInteger:bestScore forKey:@"bestScore"];

[self updateGameBestScore];

UIAlertView \*alertView = [[UIAlertView alloc]initWithTitle:@"Game Over" message:[NSString stringWithFormat:@"恭喜你，获得了新的记录：%d分，好棒哟！再来一局？",totalScore] delegate:self cancelButtonTitle:@"再来一局" otherButtonTitles:@"截图",@"分享", nil];

alertView.tag=100;

[alertView show];

}

else{

UIAlertView \*alertView = [[UIAlertView alloc]initWithTitle:@"Game Over" message:[NSString stringWithFormat:@"你获得了%d分，继续努力！再来一局？",totalScore] delegate:self cancelButtonTitle:@"再来一局" otherButtonTitles:@"截图",@"截图并分享", nil];

alertView.tag=101;

[alertView show];

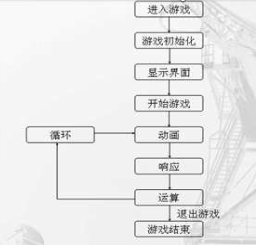
}

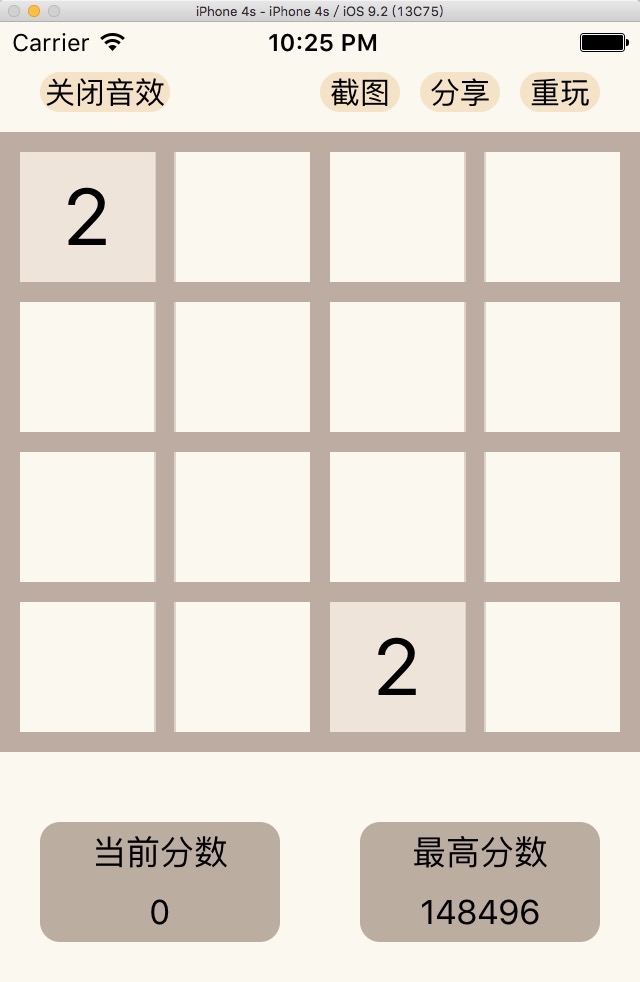
}

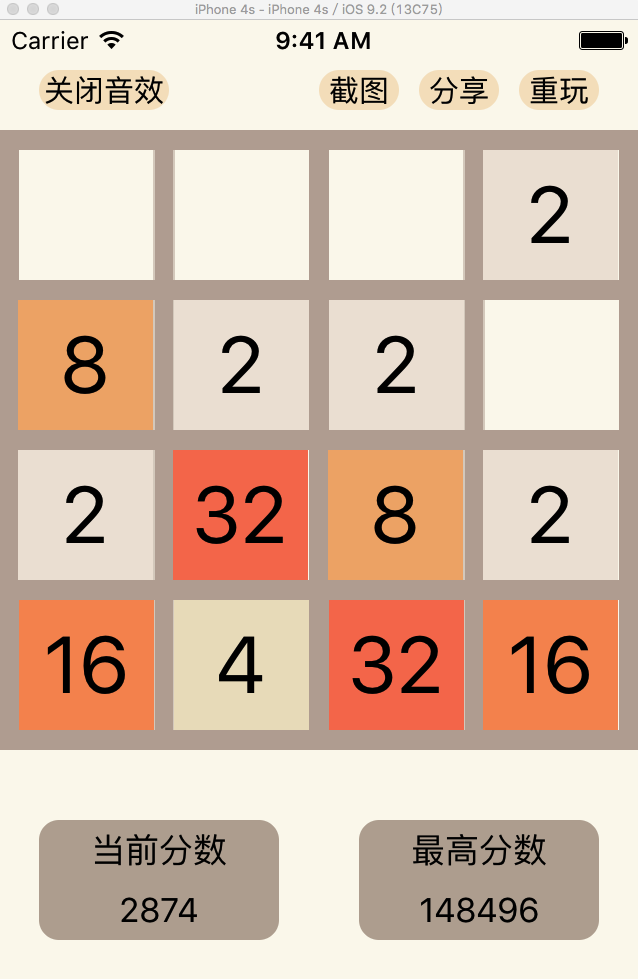
}

**5、游戏运行说明**

**1、游戏运行流程图**



**2、游戏开始 **

**3、游戏进行**

**4、游戏结束**

**5、截图保存**

**6、微博分享**