实验三 2048游戏

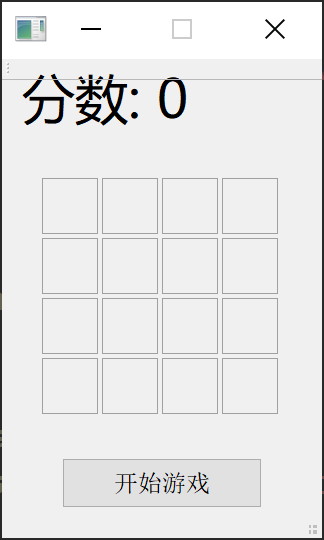
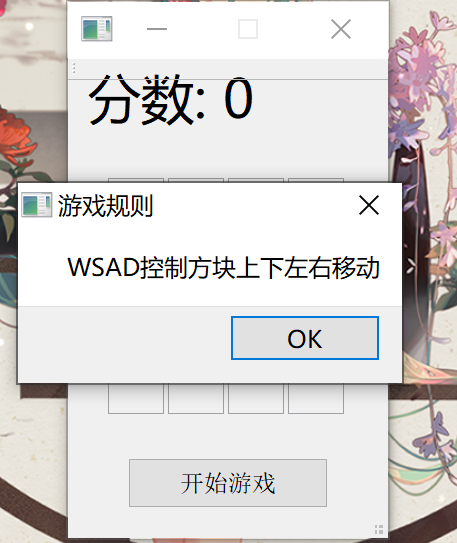
# 1.程序介绍

该程序为名称为2048游戏。2048是一款非常有趣的益智游戏，该游戏使用方向键让方块整体上下左右移动。如果两个带有相同数字的方块在移动中碰撞，则它们会合并为一个方块，且所带数字变为两者之和。每次移动时，会有一个值为2或者4的新方块出现，所出现的数字都是2的幂。当值为2048的方块出现时，游戏即胜利，该游戏因此得名。本项目利用Qt框架结合c++来实现器游戏规则。

# 2.操作说明

每次控制所有方块向同一个方向运动，两个相同数字的方块撞在一起之后合并成为他们的和，每次操作之后会在空白的方格处随机生成一个2或者4，最终得到一个“2048”的方块就算胜利了。如果16个格子全部填满并且相邻的格子都不相同也就是无法移动的话，那么恭喜你，gameover。

点击开始游戏，WSAD为上下左右控制，记录分数，效果如图1所示。

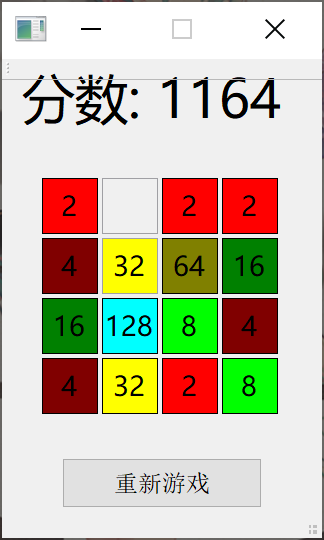
 

图 1操作展示图

# 3.设计理念

## 3.1设计目标

该项目是想设计一个益智小游戏2048，通过该游戏使用方向键让方块整体上下左右移动。如果两个带有相同数字的方块在移动中碰撞，则它们会合并为一个方块，且所带数字变为两者之和。每次移动时，会有一个值为2或者4的新方块出现，所出现的数字都是2的幂。不同数字对应不同颜色的出现直到出现2048游戏胜利。

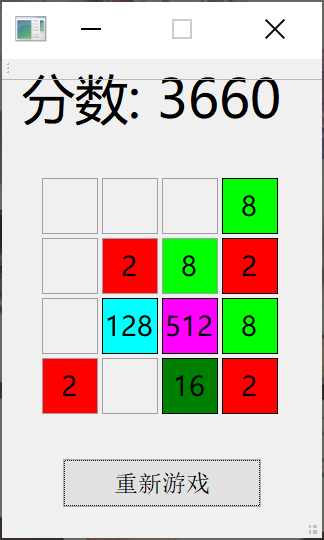


图 2 2048整体展示

## 3.2 设计分析和算法分析

### 3.2.1设计整体思路

1. 游戏界面初始化，共有4行4列，总计16个位置，游戏开始时，在任意的两个位置上，随机产生数字2；这里用二维数组来表示。
2. 玩家可通过W\A\S\D或者键盘方向键来控制所有数字的移动，游戏过程中，要符合2048游戏的基本规则；
3. 当游戏中无空余位置，且相邻数字之间无法合并，则 game over；
4. 数字移动和合并的算法实现在2048游戏中，数字移动和合并为游戏的核心，在游戏过程中，无论数字向那个方向移动，其实现所用的算法都是相同的。
5. 对于不同数字用不同颜色进行显示，初始化为0（灰色）。每次操作后在一个随机位置生成一个2，并判断游戏是否继续。

### 3.2.2整体的成员和成员函数展示

|  |
| --- |
| #ifndef MAINWINDOW\_H  #define MAINWINDOW\_H  #include <QMainWindow>  #include <QKeyEvent>  #include <QPushButton>  #include <QPainter>  #include <QTime>  #include <QDebug>  #include <QMessageBox>  *namespace* **Ui** {  *class* **MainWindow**;  }  *class* **MainWindow** : *public* QMainWindow  {  Q\_OBJECT  *public*:  *explicit* **MainWindow**(QWidget \*parent = *nullptr*);  ~***MainWindow***();  void ***paintEvent***(QPaintEvent \*);  void ***keyPressEvent***(QKeyEvent \*event);  void **PressUp**();  void **PressDown**();  void **PressLeft**();  void **PressRight**();  void **myRand**();  *//开始游戏的按钮，指针*  QPushButton \*button;  *//4\*4的数据集*  int s[4][4];  *//得分*  int score = 0;  *//状态*  bool state;  *//随机的一个点*  *struct* **Ns**  {  int i;  int j;  };  *//槽函数*  *public* slots:  void **slotStart**();  *private*:  Ui::MainWindow \*ui;  };  #endif *//* *MAINWINDOW\_H* |

代码块 1成员函数展示

### 3.2.3开始按钮

在点击开始游戏按钮后，游戏开始，随机生成一个2，并刷新屏幕显示。

|  |
| --- |
| *//开始游戏按钮*  void MainWindow::**slotStart**()  {  QMessageBox::about(*this*,"游戏规则","WSAD控制方块上下左右移动");  score = 0;  *for* (int i = 0;i < 4;i++)  {  *for* (int j = 0;j < 4;j++)  {  s[i][j] = 0;  }  }  button->setText("重新游戏");  *//随机生成第一个2*  int randi = qrand() % 4;  int randj = qrand() % 4;  s[randi][randj] = 2;  *//结束状态*  state = *true*;  *//刷新*  update();  } |

代码块 2开始游戏代码

### 3.2.4刷新绘制算法

这里应用Qt自带的paintEvent(QPaintEvent \*)函数进行重载。用于主要游戏界面绘制，内置刷新函数进行更新。先绘制顶层分数，进行显示。后遍历4×4数组。对不同数字进行不同颜色绘制。代码如下：

|  |
| --- |
| *//绘制每一个方块*  void MainWindow::***paintEvent***(QPaintEvent \*)  {  *//基本绘图，开始构建方块*  QPainter p(*this*);  p.setBrush(Qt::*blue*);  p.setFont(QFont("微软雅黑",20,700,*false*));  *//分数显示*  QString strscore;  p.drawText(QPoint(20,60),"分数: "+QString::number(score));  *for* (int i = 0;i < 4;i++)  {  *for* (int j = 0;j < 4;j++)  {  p.setBrush(Qt::*transparent*);  *//值为0时为灰色*  *if*(s[i][j] == 0)  {  p.setPen(Qt::*gray*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  }  *else* *if* (s[i][j] == 2)  {  p.setBrush(Qt::*red*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(2),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==4)  {  p.setBrush(Qt::*darkRed*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(4),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==8)  {  p.setBrush(Qt::*green*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(8),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==16)  {  p.setBrush(Qt::*darkGreen*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(16),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==32)  {  p.setBrush(Qt::*yellow*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(32),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==64)  {  p.setBrush(Qt::*darkYellow*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(64),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==128)  {  p.setBrush(Qt::*cyan*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(128),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==256)  {  p.setBrush(Qt::*darkCyan*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(256),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==512)  {  p.setBrush(Qt::*magenta*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(512),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==1024)  {  p.setBrush(Qt::*darkMagenta*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(1024),QTextOption(Qt::*AlignCenter*));  }  *else* *if* (s[i][j]==2048)  {  p.setBrush(Qt::*blue*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(2048),QTextOption(Qt::*AlignCenter*));  QMessageBox::about(*this*,"游戏胜利","分数为：" + QString::number(score)+" ");  *return*;  }  *else*  {  p.setBrush(Qt::*darkBlue*);  p.drawRect(i \* 60 + 40, j \* 60 + 120, 55, 55);  p.setPen(Qt::*black*);  p.setFont(QFont("微软雅黑",10,700,*false*));  p.drawText(QRectF(i \* 60 + 40, j \* 60 + 120, 55, 55),QString::number(s[i][j]),QTextOption(Qt::*AlignCenter*));  }  }  }  } |

代码块 3页面函数代码

### 3.2.5游戏操作设计

调用keyPressEvent函数检测键盘操作，并调用先对应函数。

|  |
| --- |
| *//按键控制wsad，*  void MainWindow::***keyPressEvent***(QKeyEvent \*event)  {  *if*(!state)  {  *return* ;  }  *switch* (event->key())  {  *case* Qt::*Key\_W*:  PressUp();  *break*;  *case* Qt::*Key\_S*:  PressDown();  *break*;  *case* Qt::*Key\_A*:  PressLeft();  *break*;  *case* Qt::*Key\_D*:  PressRight();  *break*;  *default*:  *//忽略其他按钮*  *return*;  }  *//随机生成一个2*  myRand();  *//强制刷新*  update();  } |

代码块 4keyPressEvent函数代码

W/S/A/D操作函数算法原理一致，先进行移动操作，后进行判断，如果这个方向上前一个数字与这个相同着进行合并操作。应用俩次数组遍历来实现。

|  |
| --- |
| void MainWindow::**PressUp**()  {  *//移动*  *for* (int i = 0;i < 4;i++)  {  *for* (int j = 1;j < 4;j++)  {  *if*(s[i][j] == 0)  {  *continue*;  }  *for* (int p = 0;p < j;p++)  {  *//查看前面是否有空格子可移动*  *if*(s[i][p] == 0)  {  s[i][p] = s[i][j];  s[i][j] = 0;  *break*;  }  }  }  }  *//相加*  *for* (int i = 0;i < 4;i++)  {  *for* (int j = 0;j < 3;j++)  {  *if*(s[i][j] == s[i][j+1])  {  s[i][j] = 2 \* s[i][j];  s[i][j+1] = 0;  score += s[i][j];  *for* (int p = j + 2;p < 4;p++)  {  s[i][p-1] = s[i][p];  }  }  }  }  }  void MainWindow::**PressDown**()  {  *//移动*  *for* (int i = 0;i < 4;i++)  {  *for* (int j = 2;j >= 0;j--)  {  *if*(s[i][j] == 0)  {  *continue*;  }  *for* (int p = 3;p > j;p--)  {  *//查看前面是否有空格子可移动*  *if*(s[i][p] == 0)  {  s[i][p] = s[i][j];  s[i][j] = 0;  *break*;  }  }  }  }  *//相加*  *for* (int i = 0;i < 4;i++)  {  *for* (int j = 3;j > 0; j--)  {  *if*(s[i][j] == s[i][j-1])  {  s[i][j] = 2\*s[i][j];  s[i][j-1] = 0;  score += s[i][j];  *for* (int p = j - 2;p >= 0;p--)  {  s[i][p+1] = s[i][p];  }  }  }  }  }  void MainWindow::**PressLeft**()  {  *//移动*  *for* (int j = 0;j < 4;j++)  {  *for* (int i = 1;i < 4;i++)  {  *if*(s[i][j] == 0)  {  *continue*;  }  *for* (int p = 0;p < i; p++)  {  *//查看前面是否有空格可移入*  *if*(s[p][j] == 0)  {  s[p][j] = s[i][j];  s[i][j] = 0;  *break*;  }  }  }  }  *//相加*  *for* (int j = 0;j < 4;j++)  {  *for* (int i = 0;i < 3;i++)  {  *if*(s[i][j] == s[i+1][j])  {  s[i][j] = s[i][j] \* 2;  score += s[i][j];  s[i+1][j] = 0;  *for*(int p = i + 2;p < 4; p++)  {  s[p-1][j] = s[p][j];  }  }  }  }  }  void MainWindow::**PressRight**()  {  *//移动*  *for* (int j = 0;j < 4; j++)  {  *for* (int i = 2;i >= 0; i--)  {  *if*(s[i][j] == 0)  {  *continue*;  }  *for* (int p = 3;p > i;p--)  {  *//查看前面是否有空格可移入*  *if*(s[p][j] == 0)  {  s[p][j] = s[i][j];  s[i][j] = 0;  *break*;  }  }  }  }  *//相加*  *for* (int j = 0;j < 4;j++)  {  *for* (int i = 3;i >= 0;i--)  {  *if*(s[i][j] == s[i-1][j])  {  s[i][j] = s[i][j] \* 2;  s[i-1][j] = 0;  score += s[i][j];  *for*(int p = i - 2; p >= 0; p--)  {  s[p+1][j] = s[p][j];  }  }  }  }  } |

代码块 5操作控制代码

### 3.2.6随机生成原理

这里采用一个struct结构来说实现，每次寻找空闲的格子，随机生成一个数对应这个格子。代码如下：

|  |
| --- |
| *//随机生成*  void MainWindow::**myRand**()  {  int i = 0;  int j = 0;  *//找出格子*  *struct* Ns n[15];  int ni = 0;  *for* (i = 0;i < 4;i++)  {  *for* (j = 0; j < 4; j++)  {  *if*(s[i][j] == 0)  {  n[ni].i = i;  n[ni].j = j;  ni++;  }  }  }  *//判断游戏是否结束*  *if* (ni == 0)  {  *for* (i = 0;i < 4;i++)  {  *for* (j = 0;j < 3;j++)  {  *if*(s[i][j] == s[i][j+1])  {  *return*;  }  }  }  *for* (j = 0;j < 4;j++)  {  *for* (i = 0;i < 3;i++)  {  *if*(s[i][j] == s[i+1][j])  {  *return*;  }  }  }  QMessageBox::about(*this*,"游戏失败","分数为：" + QString::number(score)+" ");  *return*;  }  int rand = qrand() % ni;  s[n[rand].i][n[rand].j] = 2;  } |

代码块 6随机函数代码

## 3.4类图关系

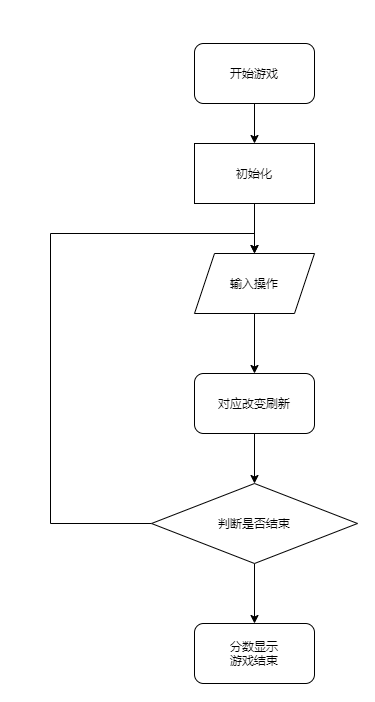
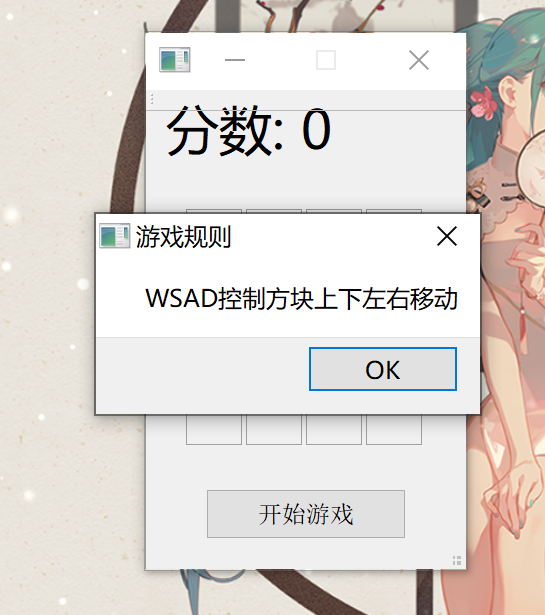
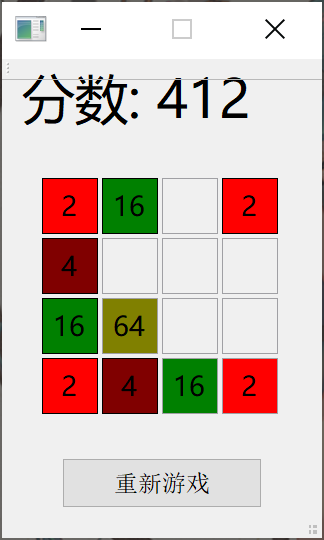


图 3类关系流程图

# 4.程序展示

一些操作实际展示，展示效果如下图：

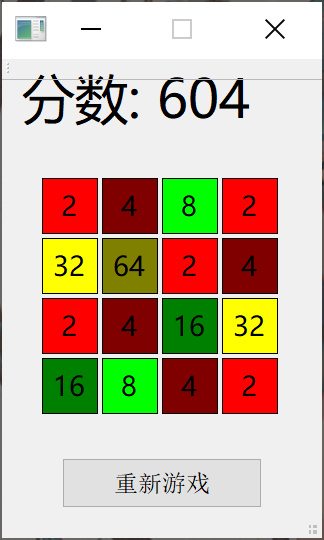
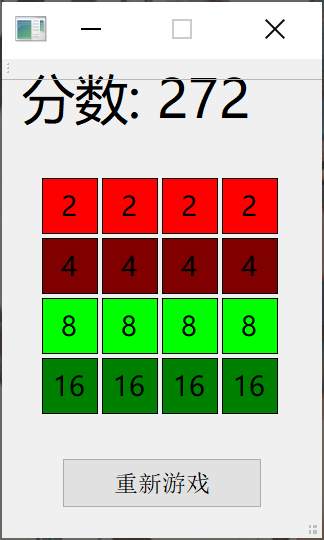
 

图 4成果展示图

# 5.总结思考

通过使用Qt应用框架实现了2048小游戏，采用Qt的QPaintEvent事件解决了游戏页面刷新布局问题，用keyPressEvent实现按键的功能绑定设置。该程序是一个集继承、图形界面、事件处理等面向对象编程知识的综合应用的实例程序。