南開大學
Nankai University

计算机学院

c++ 大作业报告

基于 QT 的 2048 小游戏

姓名：谢子涵

学号：2010507

专业：计算机科学与技术

2022 年 5 月 9 日

目录

1 开发软件

QT5.12

2 课题要求

2.1 作业要求

学生自选题目，使用 C++ 语言完成一个图形化的小程序。

- 图形化平台不限，可以是 MFC、QT 等。
- 程序内容主题不限，可以是小游戏、小工具等。

2.2 本实验要求

我选择制作 2048 小游戏，要求：

- 能实现 2048 游戏的核心逻辑，实现正常的游戏功能。
- 设计美观的界面
- 在功能上提升用户体验

3 实验流程

3.1 制作界面框架

设计两个类 Widget 和 GameWidget。Widget 类创建主窗口部件，包括两个标签（score 和 high score）、一个按钮（restart）、游戏部件 gameWidget（GameWidget 类的对象）、背景音乐。主要功能是：在构造函数中实现历史最高分 (high score) 的初始化和读取历史最高分；构造需要的标签、按钮和游戏部件对象，完成初始化并连接相应的信号和槽函数。在具体函数中实现：对分数增加、游戏失败、游戏胜利界面信号的相应（槽函数）；使主窗口位置和大小改变时，窗口中的组件能随之改变、缩放。(resizeEvent 函数)。

GameWidget 类是游戏部件类，要完成 2048 游戏界面的绘制（4*4 的方格），也控制着游戏的底层逻辑。游戏部件界面包括 4*4 个小方格，每个格子的数字存储在 board[4][4] 数组中。对 11 个数字设置一个颜色数组，依次保存着每个数字所代表的方格颜色。从而使不同的数字具有不同的颜色，更加美观。GameWidget 的构造函数同样要完成对数据的初始化和连接相应的信号和槽（如：连接手势移动信号和相应的槽函数、连接时钟信号和画板更新的槽）。并且会调用 init2Block 函数，在 board 数组的两个随机位置生成数字 2。GameWidget 类中同样有在主窗口位置和大小改变时，使窗口中的组件能随之改变、缩放的 resizeEvent 函数。而对于游戏界面的绘制主要通过 paintEvent 实现。如下为初始界面：

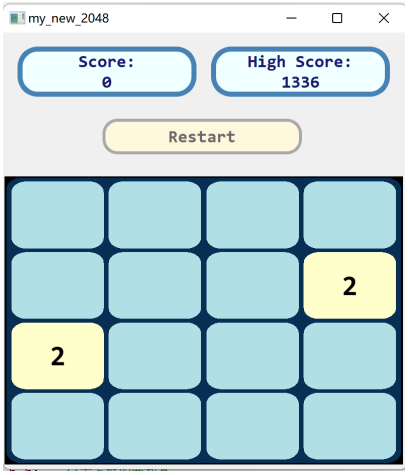

图 3.1: 初始界面

3.2 游戏核心逻辑

3.2.1 获取移动方向

我采用鼠标移动手势来判断移动方向。相比于用键盘控制方向，更具有沉浸感，再结合之后完成的方格移动动画和方格合并音效的实现，游戏体验感更佳。首先通过 mousePressEvent 获取鼠标按下的起点位置，再通过 mouseReleaseEvent 获取鼠标松开的终点位置，并结合起点位置判断移动方向。判断完后会发出该方向移动的信号（GestureMove），从而执行其对应的槽函数。

3.2.2 根据移动方向移动和改变方格

手势移动信号的槽函数 onGestureMove 对移动和合并格子事件进行处理。基本思想是：首先依据手势的不同方向执行相应的处理。如向左时，遍历每一行，通过 j、k 两个变量（初始化为 0）分别表示要交换的数字列号和交换到的位置的列号；通过 k 进行循环找到第一个不是 0 的数字对应的列号，并交换 j 和 k 对应位置的两个数字，从而完成移动的底层逻辑；如果交换后的数字与其前一列的数字相同，且还没有被合并，则与之合并。注意移动和合并的操作均需要记录起始点等动画的信息，并添加到动画列表之中，便于之后绘制动画；还需要注意增添当前有数字的格子数 digitCount，便于失败条件的判断。其余三个方向与向左类似。

完成移动和合并的底层逻辑后，还需要判断数字是否填满，如果没有填满，则随机生成一个行号和列号，如果该位置为 0，则在该位置填上 2 或 4，否则继续随机生成一个位置直到该位置为 0。同时，还需要判断是否需要播放合并时的音效、启动计时器 timer，令判断是否要绘制动画的 bool 值 isAnimating 为 true，便于之后动画的绘制。

3.2.3 获胜和失败判断

在 paintEvent 中，绘制动画完成后，会停止计时器并清除动画列表中的所有动画，再通过当前数字格数 digitCount 进行判断是否失败。如果数字格数为 16，则会调用 checkGameOver 函数，循环检测是否含有相邻的相同数码，若有则没有结束，return false；反之 return true，并在 paintEvent 函数中发出游戏失败的信号 GameOver，Widget 类中的槽函数会对此做出处理，弹出游戏失败的对话框。同样也会检测游戏是否胜利，从而判断是否要发出游戏胜利的信号弹出对应对话框。游戏胜利通过 checkWin 函数循环检测是否某个方格的数字为 2048 来实现。

3.3 功能优化

3.3.1 动画效果

区别于单调、直接地显示当前方格的数字，绘制出方格移动过程、新的方格的出现过程能使游戏体验更佳。动画效果分为方格移动动画和方格出现动画。具体的动画的类型、方向、起始点终止点坐标、数字等信息通过 GameWidget 类中定义的动画结构体实现。绘制动画效果的基本思想是：在 paintEvent 函数中先通过 bool 值 isAnimating 判断是否要播放动画，如果需要，则调用 drawAnimation 函数绘制动画效果（此时已经在 onGestureMove 中启动了计时器，方便之后绘制每一帧动画）。而在 drawAnimation 函数中，则是根据动画列表中存储的要播放的动画的信息，循环绘制游戏面板小方格，并且循环播放每个方格动画，直到当前动画列表中所有的动画播放完。而播放动画则是通过 playAnimation 函数实现。在 playAnimation 函数中，首先判断要播放的动画的类型是方格移动还是方格出现，再去执行具体的绘制。方格的移动和逐渐出现效果主要是通过一个记录了每个方向增量的数组来实现。

3.3.2 音乐播放

在游戏过程中有舒缓的背景音，这通过 Widget 类中的 QSound 对象 backgroundSound 来实现。在有方格合并时会出现轻快的音效，通过 GameWidget 类中的 QSound 对象 combineSound 来实现，具体来说是在 onGestureMove 函数中判断有方格合并后播放。

3.3.3 分数展示和记录

在 onGestureMove 函数中，如果产生方格合并，则会发出相应的分数增加的信号，使 Widget 类中对应的槽函数 onScoreInc 进行处理。具体处理为：更新分数显示并判断是否需要更新历史最高分，如果需要则存储并更新显示历史最高分。

3.3.4 放大和缩小界面

在主窗口位置和大小改变时，使窗口中的组件能够随之改变、缩放。这通过两个类中的 resizeEvent 函数实现，基本思想是通过计算宽度和高度的缩放比例来重新设置字体、部件、方格的大小和位置。

4 单元测试

4.1 初始化两个格子

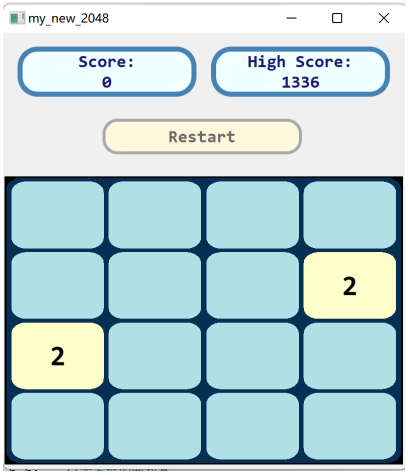

图 4.2: 游戏开始时初始化两个格子

如图，能够成功初始化两个格子。
点击 restart，观察是否重新进行了初始化两个格子。

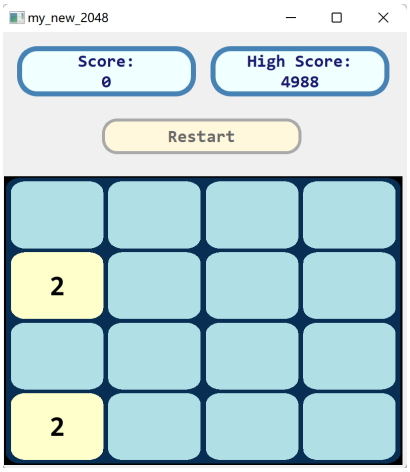

图 4.3: 点击 restart 后初始化两个格子

如图，格子位置发生变化，重新进行了初始化，功能正确。

4.2 放大和缩小界面

如下图所示，成功进行了界面的放大和缩小，功能正确。

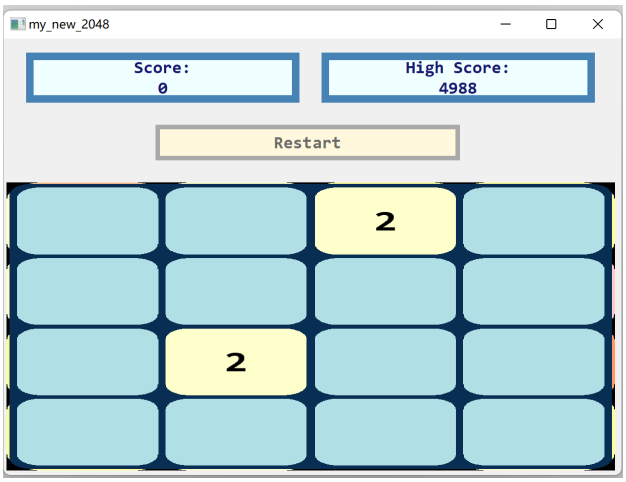

图 4.4: 放大界面

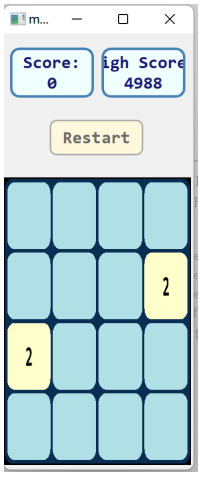

图 4.5: 缩小界面

4.3 颜色和界面

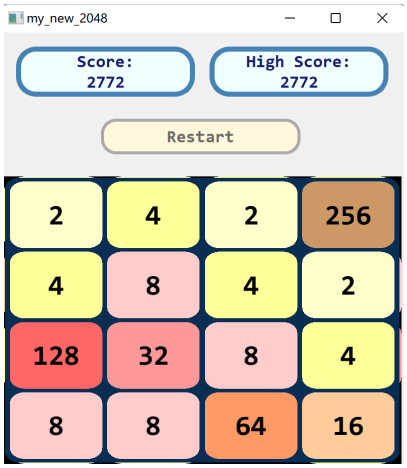

图 4.6: 游戏过程

如图4.6所示，上面为当前局分数和历史最高分数，其下为 restart 按钮，最下面为游戏主界面，界面分布合理而美观。观察游戏界面中的小方格，不同的数字具有不同颜色，比较美观。

4.4 游戏失败和胜利判断

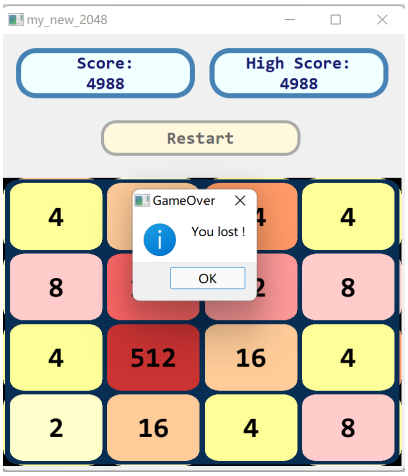

图 4.7: 游戏失败

如图4.7所示，16 个格子被填满并且没有相邻的相同数字格子，说明游戏无法继续进行，游戏失败。弹出了 GameOver 的对话框，显示” You lost! “，表明游戏失败，功能正确。

而对于游戏胜利的功能同理。由于本人不太擅长这个游戏，暂时还没有打出 2048 胜利条件，如图已经是我的最佳成绩了。但这个功能是类似游戏失败的，我还更改过胜利判断的条件，将 2048 改为了 64，最后成功弹出了如图4.8中的 Congratulation 对话框，显示”You win!”，表明该功能正常。

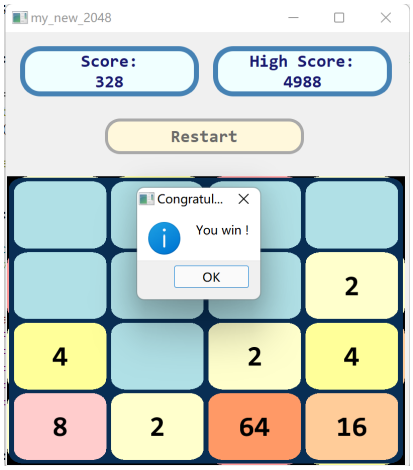

图 4.8: 游戏胜利

4.5 鼠标手势操作和动画效果展示

难以在报告中体现，具体实现见视频。可以看到鼠标手势的判断正确而灵敏，动画效果的显示也很流畅。但是由于用腾讯会议录制的视频本身存在一定卡顿，可能会在一定程度上造成影响。

5 收获

通过完成这个基于 QT 的 2048 小游戏，我对可视化技术的消息的传递和响应机制的理解更加深刻，对 C++ 面向对象的特点理解更加深刻，也对 QT 的 API 更加熟悉了。在完成作业的过程中也遇到了一些困难，比如：

最初没有在 paintEvent 函数中判断是否是正在播放动画的时候，仅能显示一次的移动，之后就不会移动了。而更改了这部分的处理，使得游戏一直能够产生移动动画和出现动画。

在编写 paintEvent 函数时，对于组件坐标的确定比较麻烦，也是经过多次修改才合适的。

总之，这次作业让我的代码能力得到了提高，对于 c++ 和可视化技术的理解也更加深刻了，自己写出一个小游戏也很有成就感。

6 相关链接

- **b 站视频** 链接
- **Github** 链接