**找乐子组作业报告**

**一、程序功能介绍**

程序：Fireboy & Watergirl

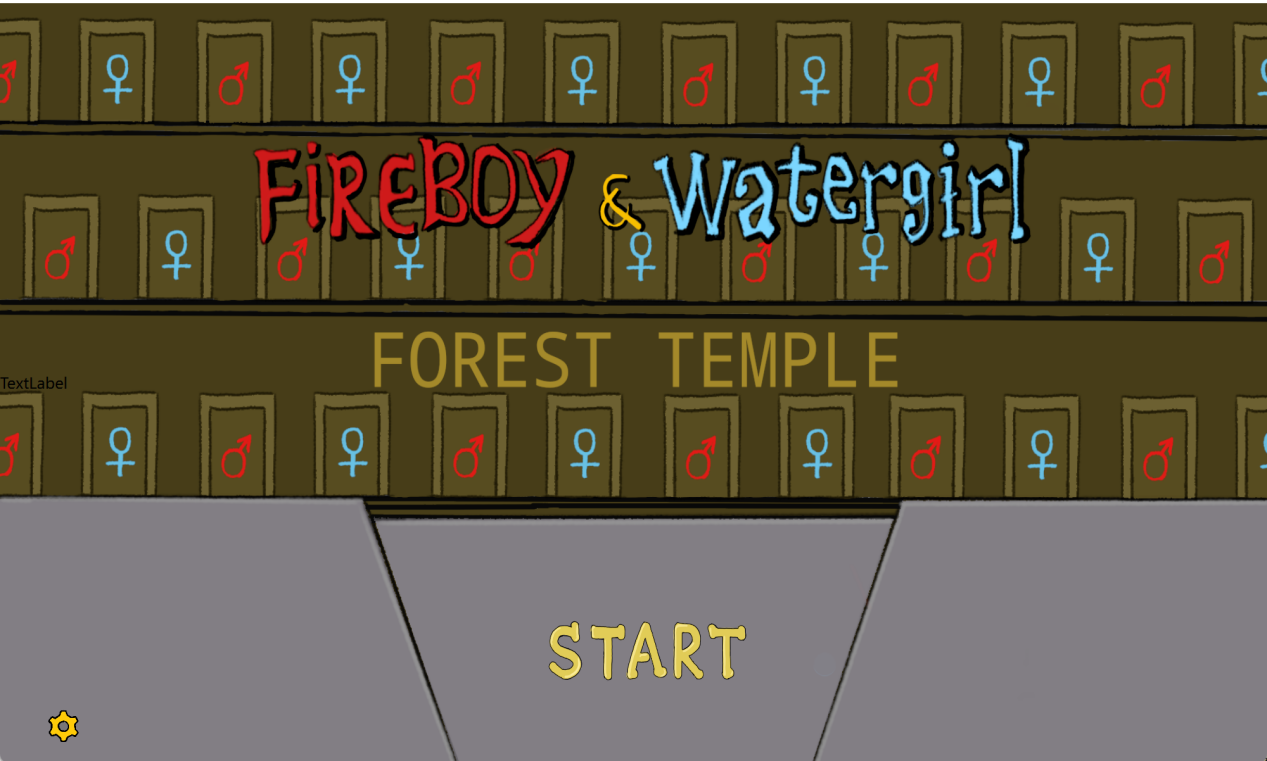
受到森林冰火人游戏的启发，我们组的大作业主要实现森林冰火人第一关的游戏。

所有的图片和按钮立绘均为手绘。

1、主界面及相关代码

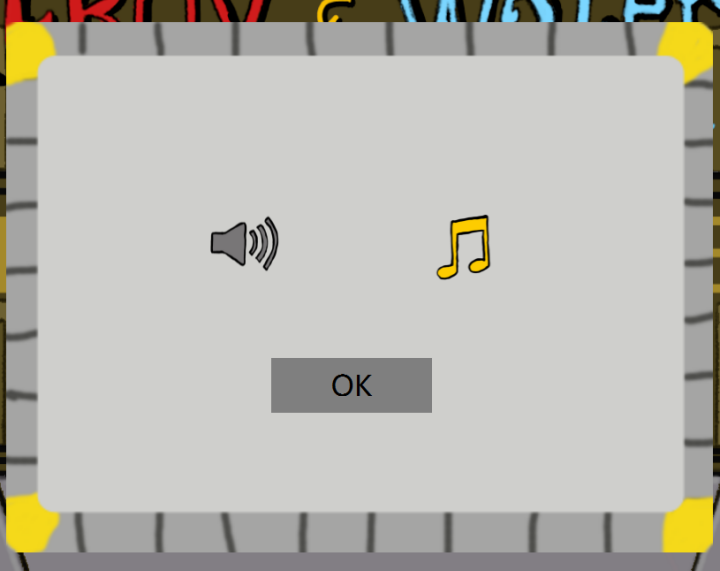
主界面按照steam平台发布的游戏，我们组绘制了森林圣殿主题的开场主界面，截图如下：

主界面按钮：设置按钮和开始按钮

2、设置界面及相关代码

在主界面中，我们实现了设置按钮和开始游戏按钮，设置按钮（左下角）按下后，弹出设置界面，如下：



两个按钮分别可以实现音乐、音效的开启、关闭，默认情况为开启，其中按钮效果如下：

实现代码（部分）：

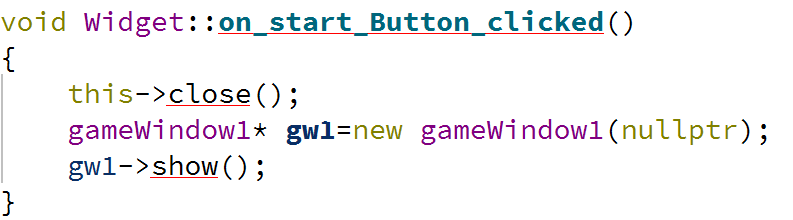
在settingBeginning.cpp中，我们设置了音效和音乐的按钮图片更改，随着鼠标点击，进行图片切换和音乐效果更改。

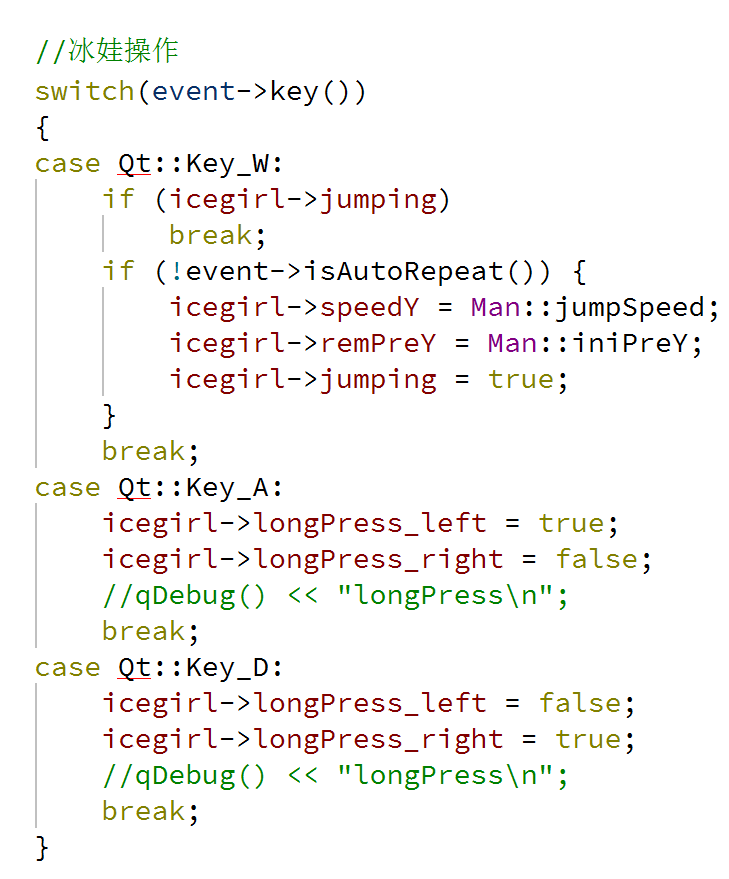
1. 开始按钮及代码

开始按钮（START）按下后，游戏开始。玩家可以通过“WASD”和“↑↓←→”键j盘控制fireboy和watergirl进行游戏，游戏界面如下：

在游戏中，我们设置了water和fire以及greengoo等障碍，角色需要通过跳跃或者道具来进行躲避和通关。同时我们在路途中设置红蓝钻石，角色需要在通关过程中获取所有的钻石，最终到达各自的门前，通过关卡。

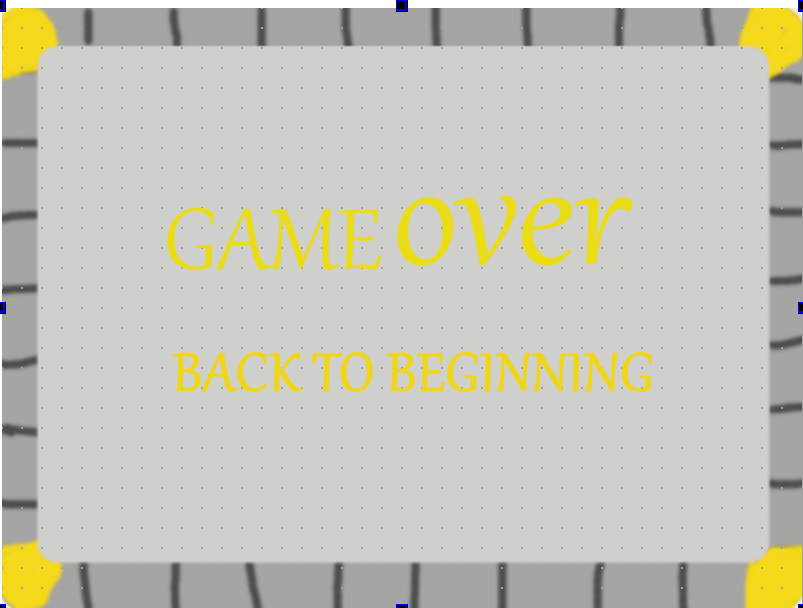
部分代码：

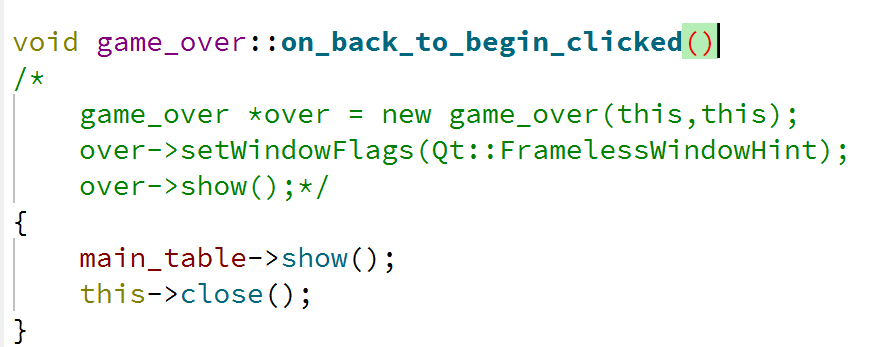
游戏界面切换：

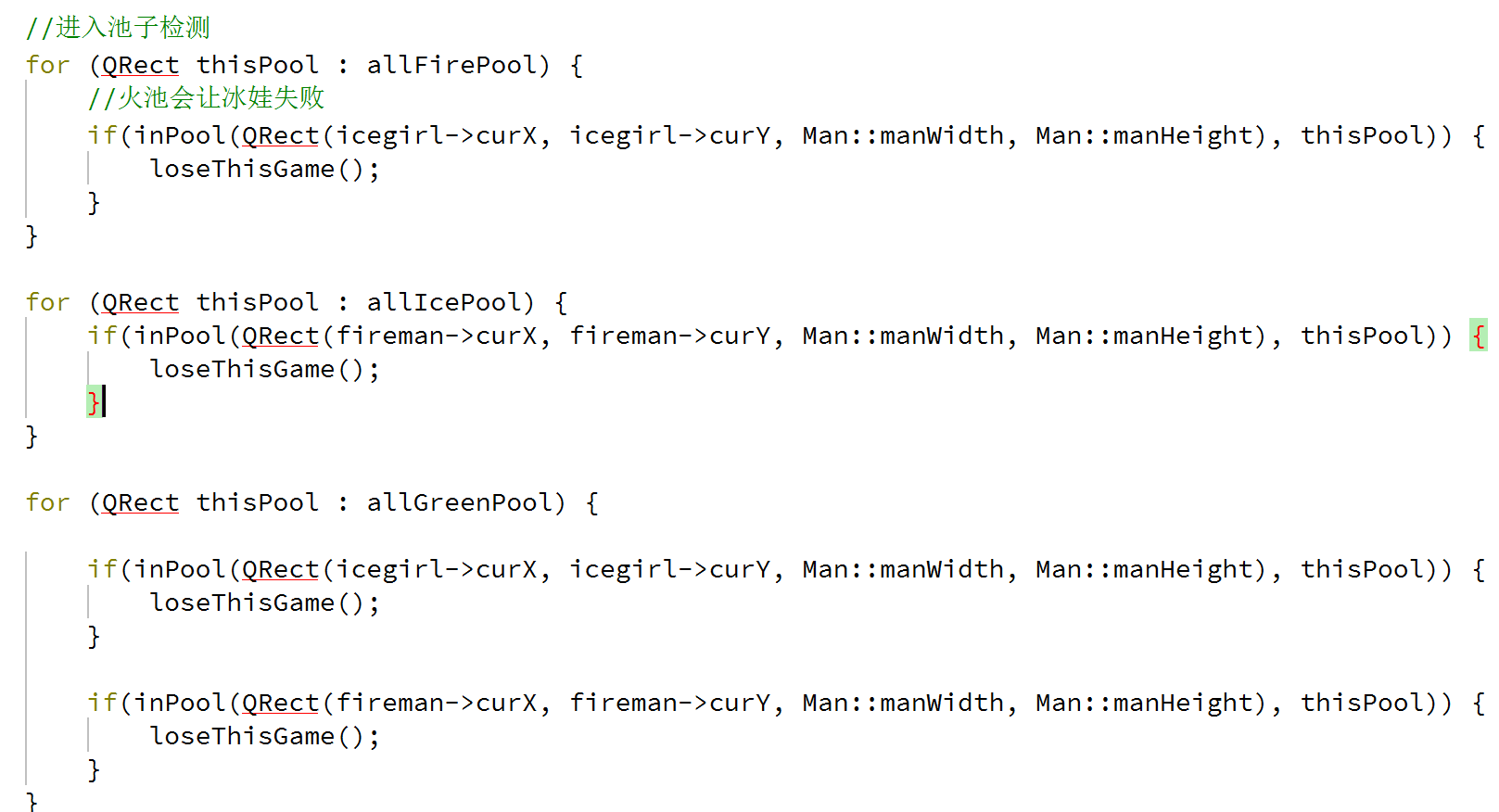
键盘控制：

4、死亡界面及代码

如果fireboy进入了water和greengoo，或者watergirl进入了fire和greengoo均会触发游戏结束界面，界面如下：

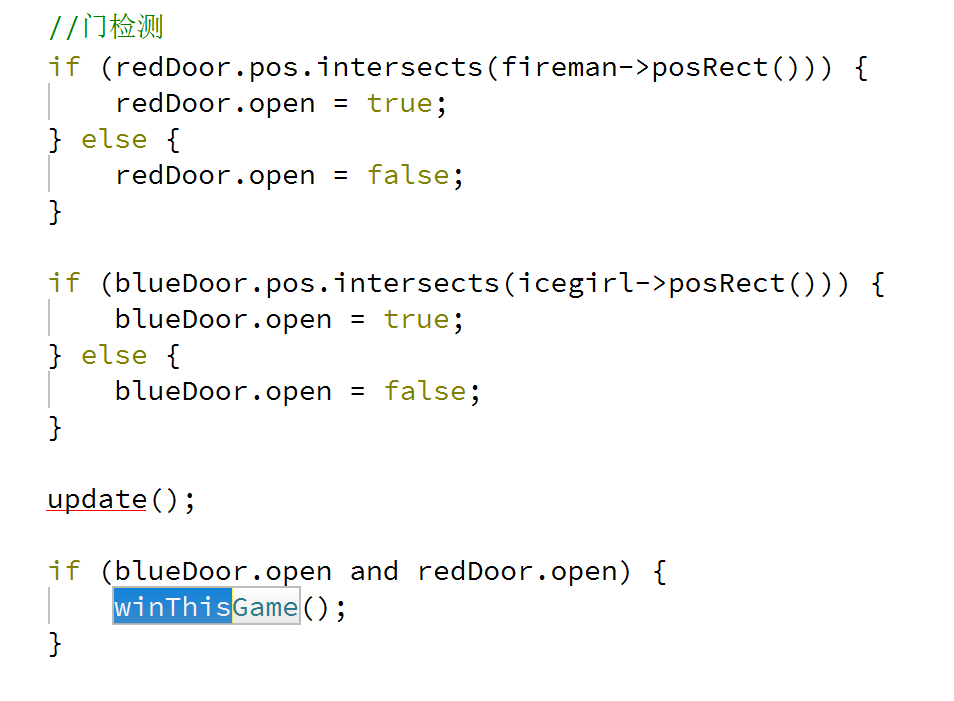
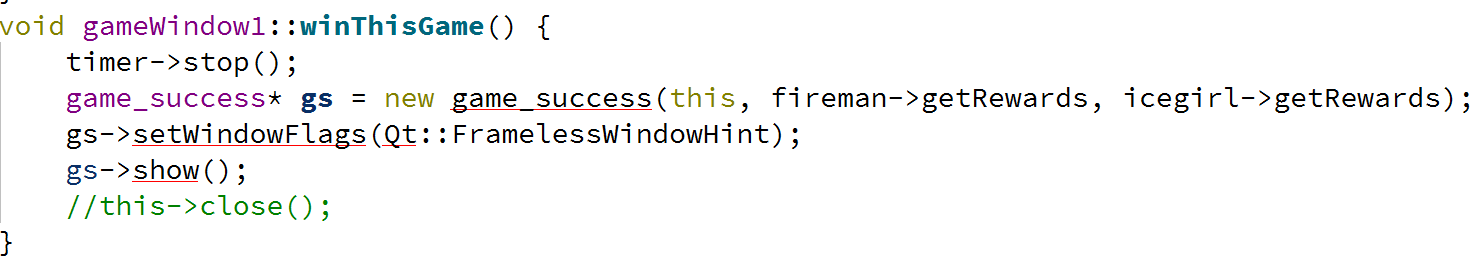


界面实现了按钮“BACK TO BEGINNING”可以返回主界面，重新进行游戏

相应部分代码：

4、成功界面

当角色均成功到达对应的门前时，触发游戏成功的界面，界面如下：

相关设定代码：

以上为我们主要的界面和部分涉及到的代码，下为具体代码功能。

**二、项目功能模块**

（1）角色模块：man.h与man.cpp相关介绍

本游戏的角色主要包括了火娃和冰娃。两个角色共用一个Man类，该类的另一在man.h中，详细的功能完成在man.cpp中。

man类有诸多静态变量，在man.h中完成定义，在man.cpp中完成初始化。在游戏设计中，我们认为冰娃和火娃需要完成左右移动和上下跳动的动作。经过观察，我们发现角色竖直方向运动类似竖直上抛和自由落体运动，水平方向运动类似匀加速后的匀速直线运动。因此我们用这些静态变量定义了相关参数。此外这些静态变量还控制了角色的尺寸。以下是参数解释：

int Man::updateTime = 10; 刷新时间，一方面控制角色速度和位置的刷新，一方面控制游戏全局的刷新

double Man::accX\_Press = 1; 水平方向在按下左键或右键时的加速度

double Man::accX\_noPress = 10; 水平方向在没按键时减速为0的加速度

double Man::maxSpeedX = 5; 水平方向加速的最大速度

double Man::accY = 0.2; 竖直方向加速度

double Man::jumpSpeed = -6; 竖直起跳的初始速度，因为是向上的所以为负值。

double Man::manHeight = 40;

double Man::manWidth = 30; 角色的尺寸

int Man::iniPreY = 20; 用来做上跳保护，防止在发出上跳指令后由于还接触地面速度被置为0

在Man类中有角色的当前位置、当前速度、按键窗框与相对按钮的状况等参数。在构造函数中我们对这些参数进行了初始化。

pullButton()函数针对的是游戏中角色对于扳手开关的操作。扳手开关操作有左扳和右扳，不能通过简单的碰撞检测来实现。在这个函数中，我们首先算出了角色横坐标中间值相对于开关左右情况，如果与之前保存的左右情况不同，说明扳动了扳手，需要进行更新。此外，在一次函数运行后还要将当前的左右情况保存到之前的左右情况中，方便下一次程序的使用。

manMove()函数控制角色的移动，在gameWindow1运行过程中每次刷新时调用，依据上边已经说明的规则，对X方向速度和Y方向速度做一定的加速和减速处理，然后根据速度更新坐标位置。

（2）游戏模块：gameWindow1.h与gameWindow1.cpp介绍

在gameWindow1中，我们主要实现了一下功能。

首先，在元素方面，我们实现了冰娃和火娃的运动，在运动过程中需要和环境互动，不能穿越环境中的屏障；我们实现了冰池和火池和绿池，在火娃碰到冰池、冰娃碰到火池或冰娃和火娃碰到绿池时，都会触发游戏失败。此外，我们实现了钻石奖励，角色可以在运动过程中收集和自己颜色相应的奖励，这会显示在最终游戏成绩当中。

我们还实现了按钮、扳手和电梯功能。当角色按下或松开按钮，或扳动扳手的时候，相应的电梯可以上下左右移动，角色可以站在电梯上移动。我们还实现了计时器，显示在游戏界面的顶部。

对于屏障，我们选择使用QRect类描绘屏障的边缘，并分为floorBarrier、ceilingBarrier、rightBarrier、leftBarrier四种边缘障碍。当角色触碰到相应障碍的时候，相应方向的速度会变为0，坐标也会被改变到刚好贴近障碍边缘的位置。

我们将火池、冰池、绿池开关和奖励位置都包装进了QRect中，一方面方便后续绘图，另一方面，QRect类中的intersects()函数可以方便地检测是否与角色相碰。电梯的上下边界也有QRect包装，并添加到相应的障碍中，随着电梯位置的变化，随时更新这个障碍，来起到辅助角色上下运动的功能。

在gameWindow1构造函数中，我们初始化了所有的屏障位置、道具位置和角色的初始位置，并初始化了定时器。这个定时器的主要作用是在每过一定时间（游戏设置为10ms）触发timeout()函数，实现角色位置的更新、碰撞检测和画面刷新。

在函数winThisGame和loseThisGame中，我们实现了链接到相应的界面展示成功与否的情况。

在函数inPool中，我们实现了角色是否位于池子中的检测。单独实现这个函数而不单纯地用intersects()碰撞检测的原因是我们希望达到角色有多余一半的身体进入异色池子中才算游戏失败，以增加游戏可玩性。

paintEvent()是本窗口的绘制函数，继承并覆盖了QWidget相关函数，在每次timeout()时调用，相当于界面的一次刷新。

timeout()是一个槽函数，承接timer每10s中一次的timeout信号。在承接到这个信号之后，程序开始执行按钮按下检测、扳手扳动检测、电梯运动、冰娃和火娃触及屏障检测及运动、进入池子、接受奖励和进入们的检测等等。在完成了所有的这些检测之后，我们执行update()函数来刷新画面。

keyPressEvent()式本窗口的键盘控制函数，继承并覆盖了QWidget的相关函数。在这里我们需要检测左右键和上键是否被按下。左右键的按下情况被记录在角色对象中，以控制加速减速；上键按下后我们直接改变角色竖直方向速度。此外，对于上键，我们避免了连续上跳，增加游戏可玩性。

（3）道具模块：elem.h与elem.cpp介绍

这个模块主要实现了reward、gameButton、lift、door类。

reward是游戏中的奖励，类中记录了奖励的位置、奖励的颜色以及是否被取得。

gameButton是游戏中的按钮和扳手类，类中记录了按钮（扳手）的位置、种类以及是否被按下（扳动）。

door类是游戏中的冰门和火门。类中记录了门的位置以及门是否开启。

lift是电梯类，记录了电梯在按钮按下（扳动）或不按下（不扳动）时以及当前的位置。类中包括两个成员函数liftMoveVert()和liftMoveHori()，控制电梯的上下或左右移动。

此外，这几个类的静态变量都记录了该道具的长宽尺寸。

（4）视觉界面及按钮图片切换部分:widget.cpp,gameWindow1.cpp,settingbeginning.cpp等

我们设计了界面之间的切换，在游戏中统一使用了无菜单框界面，重写了showEvent函数，使其出现方式和模式得到统一。

在settingBeginning中，我们使用了图片切换作为按钮的显示，能够实现视频中设置界面切换的效果。

（5）计时器模块：mytimer.h 与 mytimer.cpp 介绍

计时器模块实现的功能是在游戏界面的上方显示游戏进行的时间，并在通关后显示通关所用时长。实现方法大致是创建一个每秒发出一次信号的QTimer，利用发出的信号更新游戏进行的秒数，将秒书转化为分钟与秒钟。在MyTimer类中含有加载了对应不同数字图片的QPixmap。游戏界面的顶部为几个QLabel，在timeout的槽函数中对图片进行更新，从而达到类似于动画的时间切换效果。

（6）多媒体模块：mysoundeffect.h、mainbgm.h、gamebgm.h与mysoundeffect.cpp、mainbgm.cpp、gamebgm.cpp介绍

多媒体模块中，MainBgm与GameBgm是主界面与游戏界面的背景音乐实现，主要用到了QMediaPlayer与QMediaPlayList。此外，还在其他类中修改了相关槽函数，实现了设置界面点击按钮可以控制背景音乐的播放与停止。

MySoundEffect用于添加游戏的音效，主要用到了QMediaPlayer。也修改了相关槽函数以实现了设置界面控制音效开关，在特定游戏事件发生时触发音效播放。在编写相关代码时，用到了QMap类，实现了在添加音效时赋予其唯一的id名称，在需要播放音效的位置调用MySoundEffect::play(const QString &id) 即可，使得代码简单易懂。

**三、人员分工安排**

郑童：游戏模块，游戏角色及道具代码

朱冉：游戏界面，计时器及多媒体代码

刘若琳：美工设计、界面设计、界面及按钮切换代码及报告撰写

**四、项目总结与反思**

在本次Qt大作业中，我们最重要的收获是学习了Qt面向对象图形界面代码的编写，掌握了类及类的继承、虚函数与多态在图形界面实现的重要作用，理解了图形界面的更新与刷新逻辑。我们学会了如何在大的项目中组合代码，如何分配不同的头文件与代码文件，同时还提升了代码团队协作的能力。

从编写到调试，这次大作业的编写经历都让我们受益匪浅。但由于时间精力等的限制，最后的项目还有诸多遗憾，很多功能都未能优化到最佳状态。我们相信之后能够更加合理地安排项目的进度，把控总体的时间和框架，做出更好的项目。

正所谓“工欲善其事，必先利其器”能实现的功能的复杂程度与细腻程度不仅取决于代码设计还取决于对所用工具的熟悉程度。因为我们对Qt中很多模块还处于比较生疏的状态，无法用更高级、更简洁的代码逻辑实现更丰富更细腻的功能。我详细在未来我们水平进一步长进之后，能更好地完成类似的任务。