Qt-森林冰火人作业报告

一、 程序功能介绍

Ui 设计上，我们设计了包括初始界面、融合米哈游风味的选关页面、死亡页面、以及根据收集钻石数量来判定等级的结算页面。

人物设计上，我们设计了火人奔跑、冰人跳跃和死亡瞬间。

道具设计上，我们设计了流动的水（三种颜色）、上下移动的挡板、发灯的机关、升降平台、可收缩的平台、可推动的箱子、悬挂的木头、吹风机以及钻石收集。

玩家在通过一关后可进入下一关的界面，碰到毒水后会立即死亡，通关后将根据钻石数进行结算。

二、 项目各模块及类设计细节

level 类：表示关卡，同时包括了结算页面，死亡页面，开始页面，选关页面。

通过传递不同的参数可以调取不同的页面出来，在通关了第一关后才可以选取第二关，同时通关时会根据玩家收集的钻石数量来进行不同的评级，同时通关后，根据评级的不同，选关页面也将呈现出不同的贴图，用以表示玩家在此关曾取得的最高成绩。

icegirl & fireboy 类：

表示玩家可操作的人物，内置了碰撞体积，以及运动状态判定，内含速度向量，输入的 WASD 按键可以改变 icegirl / fireboy 的速度向量，从而改变人物的运动状态与图片。

```
level::level(QWidget *parent, int type, int n1, int n2) :
    QWidget(parent), ui(new Ui::level)
{
    ui->setupUi(this);
    setFixedSize(GAME_WIDTH, GAME_HEIGHT) ;
    setWindowTitle(GAME_TITLE) ;
    setAttribute(Qt::WA_DeleteOnClose);
    if(type == 0) {

        keyRespondTimer = new QTimer(this);
        connect(keyRespondTimer, &QTimer::timeout, this, &level::slotTimeOut);//设置按键的计时器
        _Show = new QTimer(this) ;
        connect(_Show, &QTimer::timeout, this, &level::toShow) ;
        _Show -> start(14) ; //每 14 间隔刷新人物图像

        _mechanism1 = new mechanism1(map_block) ;//mechanism2 的初始位置
        _mechanism1 -> setParent(this) ;
        _mechanism1 -> Stay(51, 370, map_block) ;

        _mechanismbegin1 = new mechanismbegin1(map_block) ;//mechanism1 的初始位置
        _mechanismbegin1 -> setParent(this) ;
        _mechanismbegin1 -> Stay(254, 505, map_block) ;

        _mechanism2 = new mechanism2(map_block) ;//mechanism2 的初始位置
        _mechanism2 -> setParent(this) ;
        _mechanism2 -> Stay(880, 285, map_block) ;


        _mechanismbegin2 = new mechanismbegin2(map_block) ;//mechanism2 的初始位置
        _mechanismbegin2 -> setParent(this) ;
        _mechanismbegin2 -> Stay(750, 264, map_block) ;

        _mechanismbegin02 = new mechanismbegin3(map_block) ;//mechanism2 的初始位置
        _mechanismbegin02 -> setParent(this) ;
        _mechanismbegin02 -> Stay(320, 353, map_block) ;

        _box = new box(map_block) ; //设置box的初始位置
        _box -> setParent(this) ;
        _box -> Stay(300, 237, map_block) ;

        _ice_door=new ice_door; //设置icedoor的初始位置
        _ice_door->setParent(this);
        _ice_door->Stay(860,95);

        _fire_door=new fire_door;
        _fire_door->setParent(this);
        _fire_door->Stay(940,95);
```

```
class icegirl:public QLabel
{

public:
    icegirl();
    int _x, _y ;
    int speed_x, speed_y, _count ; //速度向量
    //动作包括：站立，上跳，下跳，左跑，右跑，左跳，右跳 7 类
    //依次用 0, 1, 2, 3, 4, 5, 6, 7 表示
    int State, alive ;
    int dq_y[8], dq_x[8] ; //对齐参数
    int lx() ;
    int ly() ;
    int if_stay_sky(const Block &map_block) ;
    void Resize(int x, int y) ;
    void Stay(int x, int y) ;
    void Stay(const Block &x) ;
    void Change_S(QString gif, int x, int y, int if_hair) ;
    void Change_State() ;
    void Stop_x(int x) ;
    void Jump(const Block &map_block) ;
    void Run(int x) ;
    void Change(int x, int y, const Block &map_block) ;
    //碰撞体积
    //_x+dq_x[State], _x+_dy
    //_y, _y+_dy 为所围对象
    int _dx, _dy, _hair ; //是否要判断穿模的头发
    int check(int x) ;
    void check(const Block &x) ;
    void check_r(const Block &x) ;
    void check_l(const Block &x) ;
    //void check_d(const Block &x) ;
    void check_u(const Block &x) ;
    int check_box(const Block &x) ;
    int check_mechanism2(const Block &x) ;
    int check_mechanism1(const Block &x) ;
    int check_water(const Block &x, int c) ;
    int check_diamond(const Block &x) ;
    int _get_diamond ;
    void dead() ;
    //void paintEvent(QPaintEvent *event);
    QMovie *movie = nullptr ;
```

lava & poison & water 类：表示毒水、岩浆，冰池，玩家操作任意角色进入毒水都会导致角色死亡，冰人进入岩浆与火人进入冰池则分别导致对象死亡。

mechanism1 & mechanism2 类：

表示玩家可以通过机关触发的升降梯，内置了速度向量与碰撞体积，可以带动人物模型在地图上移动。

mechanismbegin 类：

对应前文所描述的升降梯的机关，当此类被触发时将启用 mechanism1/2 类。内置碰撞体积。

box 类：

这是一个可以被推动的箱子，玩家可以操作冰人与火人推动箱子，碰到边界时停止，推动箱子时玩家的移动速度会更为缓慢，同时停止推动箱子时由于惯性箱子会稍微往前移动一定的距离。内置碰撞体积。

diamond 类：

可以被冰人/火人收集的钻石，收集的钻石数目会影响通关时的等级评定。

map 类：

对应背景地图的图片。

block 类：

对应地图上的单位，通过绘制函数来确定每个像素具体的对象。

Wind 类：

对应地图上的吹风机，吹风机会从底部提供一个风场，越靠近底部加速度越大，同时风场导致冰人/火人随机的上下浮游从而提供一种上升平台的通道。

对于类别的撰写中，diamond 类，mechanism，mechanismbegin，door，wind 等类型，都被撰写为 object 类的派生类。

```
class mechanism : public object
{
public:
    int speed_x,speed_y,_count;
    int _limit ;
    mechanism() ;
};

class mechanism1 : public mechanism
{
public:
    mechanism1() ;
    mechanism1(Block &x) ;
    void init() ;
    void Stay(Block &map_block, icegirl &_icegirl, fireboy &_fireboy) ;
};

class mechanism2 : public mechanism
{
public:
    int Speed_x, Speed_y, LS, RS ;
    QString S1, S2 ;
    mechanism2() ;
    mechanism2(Block &x) ;
    mechanism2(Block &x, int dx, int dy, int BSx, int BSy, int _LS, int
    void init() ;
    void Stay(Block &map_block, icegirl &_icegirl, fireboy &_fireboy) ;
};
```

```
class object : public QLabel
{
public:
    int _x,_y;
    int _dx,_dy;//碰撞体积
    int _del, _ins ;
    QPixmap img ;
    friend class icegirl ;
    friend class fireboy ;
    friend class level;
    object();
    void Stay2(int x, int y, Block &map_block) ;
    void Resize(int x, int y, Block &map_block) ;
    void del(Block &x) ;
    void ins(Block &x) ;
};


class diamond : public object
{
public:
    int _flag ;
    diamond() ;
};
class ice_diamond : public diamond
{
public:
    ice_diamond() ;
    ice_diamond(Block &map_block) ;
    void Stay(Block &map_block, icegirl &_icegirl, fireboy &_fireboy) ;
};
```

上述代码均被封装在 object.h 与 object.cpp 内。

设计上的细节：

对于 level 类，主要通过构造函数传入不同参数的形式，来区别不同的页面，详见下图：

```cpp
level::level(QWidget *parent, int type, int n1, int n2, int A, int B, int C) :
    QWidget(parent)
{
    setFixedSize(GAME_WIDTH, GAME_HEIGHT) ;
    setWindowTitle(GAME_TITLE) ;
    setAttribute(Qt::WA_DeleteOnClose);
    this -> rkA = A, this -> rkB = B, this -> rkC = C, this->rank = 0 ;
    if(type == 1) { //关卡1
        nowlevel = 1 ;
        level1() ;
    }
    else if(type == 2) {
        m_map.m_map2.load(MAP3_PATH) ;
        update() ;
        nowlevel = 2 ;
        level2() ;
    }
    else if(type == 3) {
        m_map.m_map2.load(MAP4_PATH) ;
        update() ;
        nowlevel = 3 ;
        level3() ;
    }
    else if(type < 0) { // 0 表示初始页面，10 表示选关页面，1x 表示结算页面，-1 表示失败页面。
        m_map.m_map2.load(GAMEOVER) ;
        update() ;
        button = new QLabel_C ;
        button -> setParent(this) ;
        button -> move(375, 360) ;
        button -> resize(280, 50) ;


        beflevel = -type ;
        connect(button,&QLabel_C::clicked,this,[&](){
            level *1 = new level(nullptr, this->beflevel, 0, 0, this->rkA, this->rkB, this->rkC) ;
            1->show() ;
            this->close() ;
        }) ;
        button2 = new QLabel_C ;
        button2 -> setParent(this) ;
        button2 -> move(425, 455) ;
        button2 -> resize(170, 40) ;
        connect(button2,&QLabel_C::clicked,this,[&](){
            level *1 = new level(nullptr, 0, 0, 0, this->rkA, this->rkB, this->rkC) ;
            1->show() ;
            this->close() ;
        }) ;
```


除此之外，内部还有 lambda 表示式撰写的按钮类，与内置计时器。

内置计时器有两个，其一用于读取玩家对键盘进行的按压/松开操作，对应着人物模型运动的改变，其二则每间隔一定根据当前人物与物品 object 的交互提供行为反馈并带来图像上的改变。

对于 fireboy/icegirl 类，由于我们制作的 gif 图像的大小不统一，同时考虑到人物偏向时头发往往是瞬间从左边移动到右边的情形，我们进行了一定的对齐工作，用于保证人物运动的流畅与顺利。

同时因为像素格大小为 1024 * 720，如果采取 1/2gt^2 来拟合跳跃曲线会导致人物行动过快，因而采取了内置计数器改变速度的逻辑来切合流畅的游戏体验。

对于 object 类，主题部分并无什么过多细节可言，主要是设置了碰撞体积以及速度向量，较为麻烦的是可移动的机关类的设置，以及吹风机的设置。

前者由于这类机关很多，个体的大小，速度，方向，位置都不一，最后在构造函数内传入了大量参数来确保统一以及方便设置，后者则采取在对应的位置上随机分配不同的加速度，越靠近底部加速度越大来模拟真实的风场现象。

详见下方代码：

```cpp
mechanism2::mechanism2(Block &x, int dx, int dy, int BSx, int BSy, int _LS, int _RS, QString _S1, QString _S2) {
    S1 = _S1, S2 = _S2 ;
    img.load(S1) ;
    _x = _y = _count = 0 ; _limit = 0 ;
    LS = _LS, RS = _RS ;
    speed_x = 0, speed_y = 0 ;
    Speed_x = BSx, Speed_y = BSy ;
    Resize(dx, dy, x) ;
}
void mechanism2::init() {
    img.load(S1) ;
    speed_x = Speed_x, speed_y = Speed_y ;
}
void mechanism2::Stay(Block &map_block, icegirl &_icegirl, fireboy &_fireboy) { //每次行动，刷新当前图像，被计时器 _Show 绑定
    del(map_block) ;
    if(speed_x < 0 && _count > LS) _x += speed_x, _count += speed_x ;
    if(speed_x > 0 && _count < RS) _x += speed_x, _count += speed_x ;
    if(speed_y < 0 && _count > LS) _y += speed_y, _count += speed_y ;
    if(speed_y > 0 && _count < RS) _y += speed_y, _count += speed_y ;
    ins(map_block) ;
    move(_x, _y) ;

    this->setPixmap(img);
    this->setScaledContents(true);
    this->show() ;
}
void wind::Stay(int x, int y) {
    _x = x, _y = y ;
    move(x, y) ;
    this->setMovie(movie) ;
    movie->start() ;
    this->show() ;
}
void wind::Ins(Block &x) {
    //50 对应中值点
    //
    for(int j = _y; j < _y + _dy; ++ j) for(int i = _x; i < _x + _dx; ++ i) {
            int u = j - _y ;
            u /= 40 ;
            if(u == 0) x.block[i][j] = 20 ;
            else x.block[i][j] = 20 + ((u + 1) / 2) - rand() % 2 ;
    }
}

```

三、 小组成员分工情况

组长陈永志：完成主要类的书写，包括人物跳跃和机关的设置。

王鹤霖：完成人物的抠图和 gif 制作，以及地图 2 和 3 的绘制。

冯子康：完成地图 1 的绘制及部分文职工作。

四、 项目总结与反思

在《程序设计实习》这门课上，我们通过利用 Qt 软件完成了游戏《森林冰火人》的设计。这份项目总结和反思将回顾我们项目的整体过程，介绍我们所取得的成果以及面临的挑战，并对我们的团队合作和个人能力进行反思和总结。

1、项目概述

项目名称：森林冰火人游戏设计

使用的软件工具：Qt 软件

目标：设计一个具有吸引力和娱乐性的游戏，实现冰火人在森林中的冒险之旅

2、工作分配

我们的团队由 3 个成员组成，每个成员负责不同的任务，包括游戏逻辑、图形界面设计、关卡设计等。

每个成员根据自己的专长和兴趣领域进行任务分配，确保项目能够平衡分工和协作。

3、成果与亮点

我们成功完成了《森林冰火人》游戏的设计和实现。

游戏具有吸引力的图形界面和流畅的操作体验，能够吸引玩家的注意力，提供愉悦的游戏体验。

我们设计了多个有趣的游戏关卡和挑战，增加了游戏的趣味性和可玩性。

我们通过合理的游戏逻辑和玩家反馈机制，使游戏更具挑战性和可持续性。

4、面临的挑战

学习曲线：使用 Qt 软件进行游戏设计对我们来说是一个新的挑战，需要学习和掌握新的技术和工具。我们花费了一些时间来熟悉 Qt 的各种功能。

时间管理：完成这个项目需要一定的时间，我们需要合理安排时间，确保项目进展顺利，并保证其他学习任务和课程的平衡。

团队协作：合作是一个团队项目的重要方面。我们需要确保有效的沟通和协作，以保持项目的进度和质量。

反思与总结

团队合作：我们团队在这个项目中展示了良好的合作精神和团队协作能力。我们定期开会讨论工作进展，并及时解决遇到的问题。这有助于保持项目的进展和合作的顺利进行。

技术学习：通过这个项目，我们学习了如何使用 Qt 软件进行游戏设计，并提高了我们的编程技能和解决问题的能力。

时间管理：在项目中，我们意识到时间管理的重要性。合理规划时间并设定里程碑，有助于保持项目进度，并避免最后时刻的压力和冲突。

不断改进：项目的完成并不是终点，我们应该采取持续改进的态度。我们可以从用户反馈中收集信息，并在未来的项目中应用这些经验教训。

通过这个项目，我们不仅学到了编程技术和团队合作的能力，还提高了解决问题和创造性思维的能力。我们对自己在这个项目中的表现感到自豪，并期待在将来的项目中继续发展和应用所学到的知识和经验。