课程设计三 贪吃蛇GUI程序

181870290 周心瑜

一、主要内容与目标

主要内容：

本次课程设计需要通过程序编写，完成经典小游戏“贪吃蛇”的GUI版本。贪吃蛇游戏是一款经典的益智游戏，通过控制蛇头方向吃食物，从而使蛇越变越长。而当蛇身越长时，玩的难度就越大，贪吃蛇行进过程中不能碰墙壁，也不能咬到自己的身体和尾巴。

设计目标：

模拟贪吃蛇的游戏运行过程和最终游戏结束的分数计算显示。

二、设计思路

总体设计思路：通过MVC架构，展示贪吃蛇小游戏。其中视图为贪吃蛇游戏的地图，模型含有贪吃蛇、食物与地图中障碍物（围墙）。通过游戏控制器实现模型间交互，并更新个模型信息。

三、主要类的设计

Mainwindow类

主窗口，负责初始化：

```
19      void initScene();
20      void initSceneBackground();
21      void adjustViewSize();
```

Gamecontroller类

通过控制器实现各个类之间信息的交互，接口信息如下：

```
22      void snakeAteFood(Food *food);                  //处理蛇吃食物的信息
23      void snakeHitWall(Snake *snake, Wall *wall);    //处理触碰障碍物的信息
24      void snakeAteItself();                          //处理贪吃蛇碰到自己的信息
25      void countingDown();                            //处理限时食物计时的信息
26      void changeTime(int t);                         //处理限时食物计时的信息
27
28  public slots:
29      void gameOver();
30
31  protected:
32      bool eventFilter(QObject *object, QEvent *event);
33
34  private:
35      void listen_key_board(QKeyEvent *event);        //监听键盘
36      void addNewFood();
37      void addNewSnake();
38      void addNewWall();
39      void addLimitFood();                            //产生新的item
40      void setScore();
41      void setTime();
42      void changeScore();                             //改变游戏信息输出
43
44
45      int eat_food_num;
46      int game_score;
47      QTimer timer;                                   //通过timer与各个item的计时链接
```

Snake类

贪吃蛇的实现，接口规定如下：

```
    void change_direction(Direction direction); //改变移动


protected:
    void advance(int step);

private:
    void moveLeft();
    void moveRight();
    void moveUp();
    void moveDown();        //贪吃蛇前进方向的控制
    void moveFast();
    void moveSlow();        //贪吃蛇前进速度的调节

    void state_of_snake();  //贪吃蛇状态的判断

    bool           isAilve;
    QPointF        head;            //蛇头
    int            is_eat_food;
    int            speed;           //速度（缓冲量大小）
    QList<QPointF> tail;            //蛇身
    int            load;            //前进的当前缓冲
    Direction      m_direction;     //当前前进方向
    GameController &controller;     //MVC中控制器
```

Food类

食物实现，接口如下：

```
class Food : public QGraphicsItem
{
protected:
    int load;                                   //食物缓冲计时
    static const int limitTime = 150;           //限时食物的时限
    void advance(int phase);
public:
    qreal coord_x;
    qreal coord_y;
    enum food_t {
        Limit,
        Unlimit
    };

    food_t cate;                                //食物种类
```

Wall类

与食物类相似，只是在setobjectdata时设为GO_Wall表示item信息为障碍物。

四、程序的功能特点

一、地图初始化

我们首先设计场景scene左上角坐标为(wide_edge_min, height_edge_min) 右下角坐标为(wide_edge_max - wide_edge_min + TILE_SIZE, height_edge_max - height_edge_min + TILE_SIZE). 接下来我们来绘制一个灰色底黑色条纹的方格纸。此时创建一个QPixmap类对象bg大小为TILE_SIZE，此时bg默认为黑色。用QPainter类创建的对象p在bg中坐标(0，0)处填充长度为TILE_SIZE出的灰色矩形，则bg的下底与右侧边为黑线。此时使用bg将view的背景填充，便可得到灰色底黑色方格线的地图背景。

二、对于代码的复用的思考

在处理食物随机生成、限时食物的时间计算、贪吃蛇的前进与贪吃蛇状态判断时，程序设计逻辑与控制台版本的贪吃蛇设计较为相同。在实现中，由于Graphics View架构使得，每一个对象作为一个item，对比控制台版本的贪吃蛇，调整了对于Food类的设计。先前版本的Food类在游戏运行中，只创建两个对象，表示普通食物和限时食物只是通过改变坐标来体现食物的变化，这样游戏最多同时生成两个食物；修改之后的版本，每次产生新Food时候都会创建新的对象若要产生多个食物可以快速更改实现代码复用。

在显示和判断障碍物时，需要重新调整代码，将障碍物也作为一个场景中的item，比较直接判断地图坐标，可以实现更多的地图边界实现。

三、游戏结束与分数结算

通过游戏结束时，Qmessagebox::information消息框来获取是否进行新游戏的请求。

五、程序的操作说明

1. 按任意方向键进入游戏，贪吃蛇将以初始速度前进；

2. 游戏左下角将显示当前获的分数，右下角将显示显示食物的时限；（如果无限时食物则为0）

3. 当贪吃蛇碰到围墙或者是蛇身则游戏结束，显示是否开始新游戏的界面。如果选择开始新游戏则游戏更新，否则退出程序；

4. 按下1键贪吃蛇将加速前进，按下2键贪吃蛇将减速前进。

六、遇到的问题和解决方案

如何实现蛇、食物与地图的交互

使用Qt中的Graphics View结构，Graphics View提供了一种接口用于管理2D图形元素。Graphics View是一个基于元素（item）的模型-视图架构，其中元素、场景和视图为Graphics View的主要构成。Graphics View提供了QGraphicsScene作为场景，这构成了贪吃蛇游戏的地图scene；QGraphicsItem作为图形元件，将蛇、食物和墙都作为图形添加到场景中去。通过MVC架构进行软件设计，设计gamecontroller类来处理游戏中用户交互的部分。

贪吃蛇的前进方式

在实现贪吃蛇的数据结构为蛇头为单独记录的QPointF类对象head，其余部分放入tail队列中使用QList存储。每次前进前先判断是否吃到食物，如吃到食物，则不删除，未吃到食物则删除尾部。之后将蛇头加入尾部并生成新的蛇头。在初始化蛇时，若初始化未吃到食物，则蛇在第一次前进时尾部为空，删除尾部时出错。可解决的方法有：改变程序顺序，先将蛇头添加进队尾并生成新的蛇头，再判断是否删除尾部；或者在初始化时将蛇的长度定为2. 设计中，选择第二种方案，解决初始化问题。

在贪吃蛇触碰墙壁导致游戏结束时，蛇身会覆盖墙壁

在判断蛇与墙壁信息交互的时候，gamecontroller是无论如何先触发event，将键盘信息传递给snake并使其向前行进。在行进之后，才会在Snake类中调用state_of_snake来判断蛇是否撞击到障碍物。当蛇collidingItem产生的队列中，food对象的标记为墙，才会判断游戏结束。此时，由于蛇头覆盖了墙壁则会使的墙壁显示不全。可以解决的方法有：将蛇头从场景中移出；或者改变蛇头颜色。设计中，选择在游戏结束时，将整条蛇的颜色都做修改。将蛇修改成黑色，来表示游戏结束蛇触碰到障碍物或自身。