Project1 实验报告　　201220180 李全昊

一．整体设计

主窗口设置为 QMainwindows 类，里面含有成员 scene 和 view （分别是 QGraphicsscene 和 QGraphicsView 类）。还有一个 QTimer 用来定时发出信号

```
class MainWindow:public QMainWindow
{
    Q_OBJECT
public:
    MainWindow(QWidget* parent = nullptr);
    ~MainWindow()override;
    void mousePressEvent(QMouseEvent *event)override;
    void addPlant();
    void addPaths();
    void addShop();
    void addCard();
public slots:
    void addZombie();
    void check();
private:
    QTimer* timer;
    QGraphicsScene* scene;
    QGraphicsView* view;
    static int ZOMBIE_GEN_TIMES_NOW;
    int life;
};
```

植物和僵尸各为一个类，且都是 QGraphicsItem 的子类（shop 和 card 也都是该类的子类），这样可以方便在 scene 中添加子类，用来在 view 中显示。

```
class Plant:public QGraphicsItem
{
public:
    int hp;
    int hurt;
    int cost;
    int state;//攻击还是静止
    void attack();
    Plant();
    ~Plant()override;
    void advance(int phase) override;
    enum{Type = UserType + 1};
    int type() const override
    {
        // Enable the use of qgraphicsitem_cast with this item.
        return Type;
    }
    bool collidesWithItem(const QGraphicsItem *other, Qt::ItemSelectionMode mode = Qt::IntersectsItemShape) const override;
};

class Zombie:public QGraphicsItem
{
public:
    int hp;
    int hurt;
    double speed;
    int state;//在走还是在攻击
    int path_num;
    int nextpos_x;//下一个节点的坐标
    int nextpos_y;
    void advance(int phase) override;
    Zombie();
    ~Zombie()override;
    enum{Type = UserType + 2};
    int type() const override
    {
        // Enable the use of qgraphicsitem_cast with this item.
        return Type;
    }
    bool collidesWithItem(const QGraphicsItem *other, Qt::ItemSelectionMode mode = Qt::IntersectsItemShape) const override;
};
```

僵尸行走的实现：利用 scene 的槽函数 advance()，他会定时给每一个在其中的 item 发射信号，所以每个僵尸都可以接收到定时的信号，开始往前走。

二．路径的实现

首先定义了一个 Pathpoint 类，是路径的每个拐点。它含有指向下一个拐点的指针。

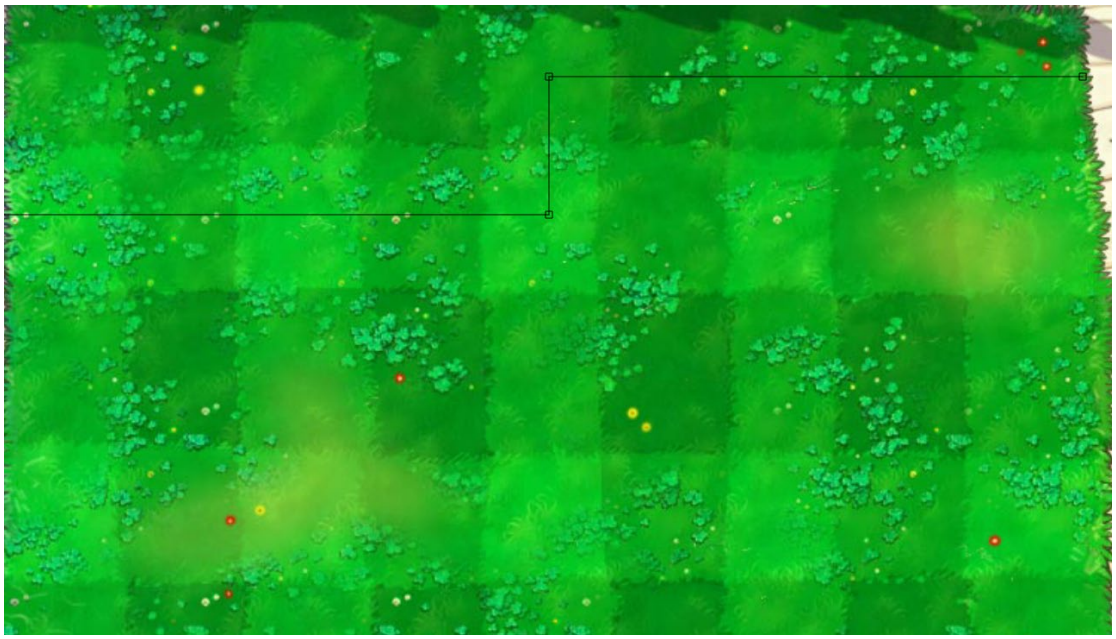
```
class Pathpoint
{
public:
    QPoint pos;
    Pathpoint* nextpoint;//指向下一个point
    Pathpoint(QPoint pos);
    ~Pathpoint();
};
extern vector<Pathpoint*> start_points;
#endif // PATHPOINT_H
```

所有的路径的起点存放在一个 vector 中

```
vector<Pathpoint*> start_points;//存储一系列开始的节点
```

这样就可以得到一条条的路径，在 mainwindows 的成员函数中再去进行添加这些路径的端点即可。

```
void MainWindow::addPaths()
{
    Pathpoint* point1 = new Pathpoint(QPoint(1540,230));
    Pathpoint* point2 = new Pathpoint(QPoint(920,230));
    Pathpoint* point3 = new Pathpoint(QPoint(920,390));
    Pathpoint* point4 = new Pathpoint(QPoint(150,390));
    point1->nextpoint = point2;
    point2->nextpoint = point3;
    point3->nextpoint = point4;
    point4->nextpoint = nullptr;
    start_points.push_back(point1);
    int x1 = point1->pos.x()+150;int x2 = point2->pos.x()+150; int y1 = point1->pos.y();int y2 = point2->pos.y();
    int x3 = point3->pos.x()+150; int y3 = point3->pos.y();
    int x4 = point4->pos.x()+150; int y4 = point4->pos.y();
    scene->addLine(x1,y1,x2,y2);
    scene->addLine(x2,y2,x3,y3);
    scene->addLine(x3,y3,x4,y4);
    scene->addRect(x1-4,y1-4,8,8);
    scene->addRect(x2-4,y2-4,8,8);
    scene->addRect(x3-4,y3-4,8,8);
    scene->addRect(x4-4,y4-4,8,8);
}
```



上面是路径的效果图，僵尸会在路径上行走。

三．整体的效果

目前的效果是做了一种近战的植物和一种普通的僵尸，然后僵尸会懂，两者也可以打架，僵尸也可以沿着路径行走。还搞了一个 shop，其中有植物的卡片，点击卡片再点击屏幕中对应的位置即可放置植物。

同时也做了失败和胜利的界面。