**飞机大战**

1. **组员**

陈丽怡 20182131053

1. **项目**

飞机大战

1. **设计**

在游戏中，“英雄机”发射子弹，我方可以通过移动“英雄机”，使子弹击中敌机，以达到消灭敌机的目的。

1.导入“地图”素材，实现地图的滚动；

2.导入“英雄飞机”素材和英雄飞机的“子弹”素材，使得鼠标可以在矩形框内拖动英雄飞机，同时英雄机发射子弹；

3.导入“敌机”素材，当英雄飞机的子弹击中敌机时，敌机消失。

1. **实现**
2. 创建一个项目“AircraftBattle”
3. 导入游戏所用到的图片素材资源



1. main.cpp

#include "mainscene.h"

#include <QApplication>

#include <QResource>

#include <config.h>

int main(int argc, char \*argv[])

{

QApplication a(argc, argv);

//注册二进制资源文件信息

QResource::registerResource(GAME\_RES\_PATH);

MainScene w;

w.show();

return a.exec();

}

1. config.h

#ifndef CONFIG\_H

#define CONFIG\_H

/\*\*\*\*\*\*\*\*\*\*\*\*\*游戏配置\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define GAME\_WIDTH 512//宽度

#define GAME\_HEIGHT 768//高度

#define GAME\_TITLE "飞机大战" //标题

#define GAME\_RES\_PATH "./fly.rcc"//rcc文件路径

#define GAME\_RATE 10 //定时器刷新（单位：毫秒）

/\*\*\*\*\*\*\*\*\*\*\*\*\*地图配置\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define MAP\_PATH ":/res/img\_bg\_level\_4.jpg"//地图路径

#define MAP\_SCROLL\_SPEED 2//滚动速度

/\*\*\*\*\*\*\*\*\*\*\*\*\*飞机配置数据\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define HERO\_PATH ":/res/hero.png" //英雄飞机图片

/\*\*\*\*\*\*\*\*\*\*\*\*\*子弹配置数据\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define BULLET\_PATH ":/res/bullet\_8.png" //英雄飞机的子弹图片

#define BULLET\_RATE 5 //子弹移动速度

#define BULLET\_NUM 30 //弹匣中子弹总数

#define BULLET\_INTERVAL 20 //发射子弹时间间隔

/\*\*\*\*\*\*\*\*\*\*\*\*\*敌机配置数据\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define ENEMY\_PATH ":/res/img-plane\_7.png" //敌机资源照片

#define ENEMY\_SPEED 5 //敌机移动速度

#define ENEMY\_MUN 20 //敌机总数

#define ENEMY\_INTERVAL 30 //敌机出场时间间隔

#endif // CONFIG\_H

1. mainscene主界面

（1）mainscene.h头文件

#ifndef MAINSCENE\_H

#define MAINSCENE\_H

#include <QWidget>

#include <QTimer>

#include "map.h"

#include "heroplane.h"

#include "bullet.h"

#include "enemyplane.h"

class MainScene : public QWidget

{

Q\_OBJECT

public:

MainScene(QWidget \*parent = 0);

~MainScene();

//初始化场景

void initScene();

//启动游戏

void playGame();

//更新所有游戏元素的坐标

void updatePosition();

//绘制到屏幕中 函数名称不可改

void paintEvent(QPaintEvent \*);

//重新鼠标移动事件 函数名称不可改

void mouseMoveEvent(QMouseEvent \*);

//敌机出场

void enemyToScene();

//敌机数组

Enemyplane m\_enemys[ENEMY\_MUN];

//敌机出场间隔记录

int m\_recorder;

//地图对象

Map m\_map;

//飞机对象

HeroPlane m\_hero;

//子弹对象

bullet temp\_Bullet;

void collisonDetection();

//定时器

QTimer m\_Timer;

};

#endif // MAINSCENE\_H

（2）mainscene.cpp源文件

#include "mainscene.h"

#include "config.h"

#include <QIcon>

#include<QPainter>

#include<QMouseEvent>

#include <ctime>

MainScene::MainScene(QWidget \*parent)

: QWidget(parent)

{

//调用初始化场景

initScene();

//启动游戏

playGame();

}

MainScene::~MainScene()

{

}

void MainScene::initScene()

{

//设置窗口固定尺寸

setFixedSize(GAME\_WIDTH,GAME\_HEIGHT);

//设置标题

setWindowTitle(GAME\_TITLE);

//定时器设置

m\_Timer.setInterval(GAME\_RATE);

//敌机出场时间间隔初始化

m\_recorder = 0 ;

//随机数种子

srand((unsigned int) time(NULL));

}

void MainScene::playGame()

{

//启动定时器

m\_Timer.start();

//监听定时器发送的信号

connect(&m\_Timer,&QTimer::timeout,[=](){

//敌机出场

enemyToScene();

//更新游戏中的游戏坐标

updatePosition();

//绘制到屏幕中

update();

//碰撞检测

collisonDetection();

});

}

void MainScene::updatePosition()

{

//更新地图的坐标

m\_map.mapPosition();

//发射子弹

m\_hero.shoot();

//计算所有非空闲子弹的当前坐标

for (int i=0;i<BULLET\_NUM;i++) {

//如果非空闲，计算发射位置

if(!m\_hero.m\_bullets[i].m\_Free)

{

m\_hero.m\_bullets[i].updatePosition();

}

}

//敌机出场

for (int i=0;i<ENEMY\_MUN;i++) {

if(!m\_enemys[i].m\_Free)

{

m\_enemys[i].updatePosition();

}

}

//测试子弹

// temp\_Bullet.m\_Free = false;

// temp\_Bullet.updatePosition();

}

void MainScene::paintEvent(QPaintEvent \*)

{

QPainter painter(this);

//绘制地图

painter.drawPixmap(0,m\_map.m\_map1\_posY,m\_map.m\_map1);

painter.drawPixmap(0,m\_map.m\_map2\_posY,m\_map.m\_map2);

//绘制英雄飞机

painter.drawPixmap(m\_hero.m\_X,m\_hero.m\_Y,m\_hero.m\_Plane);

//绘制子弹

for (int i=0;i<BULLET\_NUM;i++) {

//如果非空闲，绘制

if(!m\_hero.m\_bullets[i].m\_Free)

{

painter.drawPixmap(m\_hero.m\_bullets[i].m\_X,m\_hero.m\_bullets[i].m\_Y,m\_hero.m\_bullets[i].m\_Bullet);

}

}

//绘制敌机

for (int i=0;i<ENEMY\_MUN;i++) {

if(!m\_enemys[i].m\_Free)

{

painter.drawPixmap(m\_enemys[i].m\_X,m\_enemys[i].m\_Y,m\_enemys[i].m\_enemy);

}

}

//测试子弹

//painter.drawPixmap(temp\_Bullet.m\_X,temp\_Bullet.m\_Y,temp\_Bullet.m\_Bullet);

}

void MainScene::mouseMoveEvent(QMouseEvent \* event)

{

int x=event->x()-m\_hero.m\_Rect.width()/2;

int y=event->y()-m\_hero.m\_Rect.height()/2;

//边界检测

if(x<=0)

{

x=0;

}

if(x>=GAME\_WIDTH-m\_hero.m\_Rect.width())

{

x=GAME\_WIDTH-m\_hero.m\_Rect.width();

}

if(y<=0)

{

y=0;

}

if(y>=GAME\_HEIGHT-m\_hero.m\_Rect.height())

{

y=GAME\_HEIGHT-m\_hero.m\_Rect.height();

}

m\_hero.setPosition(x,y);

}

void MainScene::enemyToScene()

{

//未达到出场间隔，直接return

m\_recorder++;

if(m\_recorder<ENEMY\_INTERVAL)

{

return;

}

m\_recorder=0;

for (int i=0;i<ENEMY\_MUN;i++) {

//如果是空闲敌机，出场

if(m\_enemys[i].m\_Free)

{

m\_enemys[i].m\_Free=false;

//坐标

m\_enemys[i].m\_X=rand()%(GAME\_WIDTH-m\_enemys[i].m\_Rect.width());

m\_enemys[i].m\_Y=-m\_enemys[i].m\_Rect.height();

break;

}

}

}

void MainScene::collisonDetection()

{

//遍历所有非空闲的敌机

for (int i=0;i<ENEMY\_MUN;i++) {

//如果是空闲的飞机，执行下一次循环

if(m\_enemys[i].m\_Free)

{

continue;

}

//遍历所有非空闲的子弹

for (int j=0;j<BULLET\_NUM;j++) {

//如果是空闲的子弹，执行下一次循环

if(m\_hero.m\_bullets[j].m\_Free)

{

continue;

}

//如果子弹和飞机相交，发射碰撞，同时将飞机和子弹空闲状态设置为真

if(m\_enemys[i].m\_Rect.intersects(m\_hero.m\_bullets[j].m\_Rect))

{

m\_enemys[i].m\_Free = true;

m\_hero.m\_bullets[j].m\_Free = true;

}

}

}

}

6. map地图

（1）map.h头文件

#ifndef MAP\_H

#define MAP\_H

#include<QPixmap>

class Map

{

public:

//构造函数

Map();

//地图滚动坐标计算

void mapPosition();

//地图图片对象

QPixmap m\_map1;

QPixmap m\_map2;

//地图Y坐标

int m\_map1\_posY;

int m\_map2\_posY;

//地图滚动幅度

int m\_scroll\_speed;

};

#endif // MAP\_H

（2）map.cpp源文件

#include "map.h"

#include "config.h"

Map::Map()

{

//初始化加载地图对象

m\_map1.load(MAP\_PATH);

m\_map2.load(MAP\_PATH);

//初始化外轴坐标

m\_map1\_posY=-GAME\_HEIGHT;

m\_map2\_posY=0;

//地图滚动速度

m\_scroll\_speed = MAP\_SCROLL\_SPEED;

}

void Map::mapPosition()

{

//处理第一张图片滚动速度

m\_map1\_posY +=m\_scroll\_speed;

if(m\_map1\_posY>=0)

{

m\_map1\_posY=-GAME\_HEIGHT;

}

//处理第二张图片滚动速度

m\_map2\_posY+=m\_scroll\_speed;

if(m\_map2\_posY>=GAME\_HEIGHT)

{

m\_map2\_posY=0;

}

}

7. heroplane英雄机

（1）heroplane.h头文件

#ifndef HEROPLANE\_H

#define HEROPLANE\_H

#include<QPixmap>

#include<QRect>

#include"config.h"

#include"bullet.h"

class HeroPlane

{

public:

HeroPlane();

//射击子弹

void shoot();

//设置飞机位置

void setPosition(int x,int y);

//int randomIntPlane();

//飞机资源 对象

QPixmap m\_Plane;

//飞机坐标

int m\_X;

int m\_Y;

//飞机的矩形边框

QRect m\_Rect;

//弹匣

bullet m\_bullets[BULLET\_NUM];

//发射间隔记录

int m\_recorder;

};

#endif // HEROPLANE\_H

（2）heroplane.cpp源文件

#include "heroplane.h"

#include "config.h"

HeroPlane::HeroPlane()

{

//初始化加载飞机图片

m\_Plane.load(HERO\_PATH);

//初始化飞机坐标

m\_X=(GAME\_WIDTH-m\_Plane.width())/2;

m\_Y=GAME\_HEIGHT-m\_Plane.height();

//初始化矩形边框

m\_Rect.setWidth(m\_Plane.width());

m\_Rect.setHeight(m\_Plane.height());

m\_Rect.moveTo(m\_X,m\_Y);

m\_recorder=0;

}

void HeroPlane::setPosition(int x, int y)

{

m\_X=x;

m\_Y=y;

m\_Rect.moveTo(m\_X,m\_Y);

}

void HeroPlane::shoot()

{

//累加时间间隔记录变量

m\_recorder++;

//如果记录数字未达到子弹发射时间间隔，直接return，不发射子弹

if(m\_recorder<BULLET\_INTERVAL)

{

return;

}

//达到发射时间

m\_recorder=0;

//发射子弹

for (int i=0;i<BULLET\_NUM;i++) {

//如果是空闲的子弹

if(m\_bullets[i].m\_Free)

{

//将空闲状态改为假

m\_bullets[i].m\_Free=false;

//设置子弹位置

m\_bullets[i].m\_X=m\_X+m\_Rect.width()/2-10;

m\_bullets[i].m\_Y=m\_Y-25;

break;

}

}

}

8. bullet子弹

（1）bullet.h头文件

#ifndef BULLET\_H

#define BULLET\_H

#include<QPixmap>

#include<QRect>

#include"config.h"

class bullet

{

public:

bullet();

//更新子弹坐标

void updatePosition();

//int randomIntBullet();

//子弹资源对象

QPixmap m\_Bullet;

//子弹坐标

int m\_X;

int m\_Y;

//子弹移动速度

int m\_Speed;

//子弹是否闲置

bool m\_Free;

//子弹的矩形边框（用于碰撞检测）

QRect m\_Rect;

};

#endif // BULLET\_H

（2）bullet.cpp源文件

#include "bullet.h"

#include "config.h"

bullet::bullet()

{

//加载子弹资源

m\_Bullet.load(BULLET\_PATH);

//子弹坐标

m\_X=GAME\_WIDTH/2-m\_Bullet.width()/2;

m\_Y=GAME\_HEIGHT;

//子弹状态 默认空闲

m\_Free=true;

//子弹速度

m\_Speed=BULLET\_RATE;

//子弹边框

m\_Rect.setWidth(m\_Bullet.width());

m\_Rect.setHeight(m\_Bullet.height());

m\_Rect.moveTo(m\_X,m\_Y);

}

void bullet::updatePosition()

{

//空闲状态下的子弹，不需要计算坐标

if (m\_Free)

{

return;

}

//子弹向上移动

m\_Y-=m\_Speed;

m\_Rect.moveTo(m\_X,m\_Y);

//子弹位置超出屏幕，重新变为空闲状态

if(m\_Y<=-m\_Rect.height())

{

m\_Free=true;

}

}

9. enemyplane敌机

（1）enemyplane.h头文件

#ifndef ENEMYPLANE\_H

#define ENEMYPLANE\_H

#include"config.h"

#include"QPixmap"

#include"QRect"

class Enemyplane

{

public:

Enemyplane();

//更新坐标

void updatePosition();

//int randomIntEnemy();

//敌机资源对象

QPixmap m\_enemy;

//位置

int m\_X;

int m\_Y;

//速度

int m\_Speed;

//状态

bool m\_Free;//璁剧疆鏁屾満闂茬疆

//敌机的矩形边框

QRect m\_Rect;//鏁屾満妗?

};

#endif // ENEMYPLANE\_H

（2）enemyplane.cpp源文件

#include "enemyplane.h"

#include<QTime>

#include "config.h"

//#include<qDebug>

Enemyplane::Enemyplane()

{

//敌机资源加载

m\_enemy.load(ENEMY\_PATH);

// qDebug()<<pixmap\_url;

//敌机位置

m\_X=0;

m\_Y=0;

//敌机状态

m\_Free=true;

//敌机状态

m\_Speed=ENEMY\_SPEED;

//敌机矩形框（碰撞检测）

m\_Rect.setWidth(m\_enemy.width());

m\_Rect.setHeight(m\_enemy.height());

m\_Rect.moveTo(m\_X,m\_Y);

}

void Enemyplane::updatePosition()

{

//空闲状态敌机，不计算坐标

if(m\_Free)

{

return;

}

m\_Y+=m\_Speed;

m\_Rect.moveTo(m\_X,m\_Y);

//如果超出屏幕，重置空闲状态

if(m\_Y>=GAME\_HEIGHT+m\_Rect.height())

{

m\_Free=true;

}

}

1. **测试**



1. **安装**

需要安装QT。

我所用的版本是Qt Creator 4.9.1 (Enterprise)