**一，我的分工：**

画地图，npc对话响应事件， 对战模式

**二，总结和收获：**

我一开始并不知道很了解Qt，一开始写大作业时总是感到无从下手，出现许多问题，为了解决问题，我就不断的上网搜索我想要的答案，网上的回答有很多，发现想要找到自己想要的回答并不是一件易事，在完成大作业的这段时间，我觉得我搜索问题的能力变得更强了，可以更好地找到自己想要的答案。

因为有许多不懂，我都是在一边学一边写，渐渐的我开始熟悉Qt的一些机构，它的原理大都是建立在不同的类上，我可以通过已经封装好的类去实现许多功能，例如加载动图释放动图，设置时间函数控制时间，设置音效，写动画等等强大的功能，我也可以继承已封装好的类去实现我想要的不同的功能。也就是说，如果我要是熟悉Qt里的不同的类，我就可以实现许多我想要的效果，如果不再封装里的效果，我也可以继承类将它扩展功能，这真的是一个十分有用的系统，如果能够熟练运用，这将会是一个强大的工具。这次大作业虽然写得很累，但受益匪浅。

**三，画地图：**

int j; //界面与游戏切换

int k;//esc键

Dialog \*dlg;

QPixmap \*pixmap;

Talk \_talk; //npc对话框

int plot; //故事情节

我设置了一个参数，运行程序时初始化参数进入界面，点击按钮在槽函数里改变参数的值从而画出不同的地图，这样就实现了不同地图之间的关联。

if(j==1) //不同参数对应不同地图

{

equipbutton->*setVisible*(false);

startbutton->*setVisible*(true);

continuebutton->*setVisible*(true);

quitbutton->*setVisible*(true);

backButton->*setVisible*(false);

BACK->*setVisible*(false);

savebutton->*setVisible*(false);

soundbutton->*setVisible*(false);

jindu1->*setVisible*(false);

jindu2->*setVisible*(false);

jindu3->*setVisible*(false);

sjindu1->*setVisible*(false);

sjindu2->*setVisible*(false);

sjindu3->*setVisible*(false);

pa->drawPixmap(rect(),QPixmap("C:\\Users\\67549\\Desktop\\Qt.map\\jiemian1.jpg"));

\_talk.show(pa);

}

else if(j==2)

{

// music1->play();

equipbutton->*setVisible*(true);

startbutton->*setVisible*(false);

continuebutton->*setVisible*(false);

quitbutton->*setVisible*(false);

backButton->*setVisible*(false);

jindu1->*setVisible*(false);

jindu2->*setVisible*(false);

jindu3->*setVisible*(false);

// this->\_game.showback(pa);

pa->drawPixmap(rect(),\*pixmap);

// pa->drawImage(QRect(30\*32,11\*32,230,110),QImage("C:/Users/67549/Desktop/Qt.map/npctalk/npc1-3.png"),QRect(0,0,230,110));

this->\_game.show(pa);

this->\_talk.show(pa);

}

else if(j==3)

{

equipbutton->*setVisible*(true);

jindu1->*setVisible*(false);

jindu2->*setVisible*(false);

jindu3->*setVisible*(false);

bool m1 = dlg->getenemy1();

this->\_game.changeenemy11(m1);

if(m1==false)

{

plot=3;

}

bool m2 = dlg->getenemy2();

this->\_game.changeenemy22(m2);

if(m2==false)

{

plot=7;

}

bool m3 = dlg->getenemy3();

this->\_game.changeenemy33(m3);

if(m3==false)

{

plot=9;

}

pa->drawPixmap(rect(),QPixmap("C:\\Users\\67549\\Desktop\\Qt.map\\map2.png"));

this->\_game.showfight(pa); //调用\_game里得show函数画地图

}

对人物的位置进行判断，如果人物的位置走过地图的边际就切换到另一副地图，从而实现地图的连接，再在不同的地图上画出相应的东西就可以了。

else if(e->key() == Qt::Key\_D)

{

if(\_game.\_player.getPosX()>54\*32&&j==2)

{

\_game.\_player.changei(-1);

\_game.\_player.setPosX(42\*32);

\_game.\_player.setPosY(28\*32);

changej(3);

playlist->next();

}

this->\_game.\_player.changedirect(4);

this->\_game.handlePlayerMove1(4,16);

}

**四，npc对话响应事件：**

我利用鼠标点击事件，当人物走到npc周围用鼠标点击npc时才会有相应的对话框弹出，在点击时改变相应参数从而弹出不同的对话框，这样就实现了npc对话。

if(j==2&&e->x()>=20\*32&&e->x()<=24\*32&&e->y()>=15\*32&&e->y()<=19\*32)

//鼠标点击位置

{

if(\_game.\_player.getPosX()>19\*32&&\_game.\_player.getPosX()<22\*32&&\_game.\_player.getPosY()<19\*32) //人物位置

{

if(plot<5) //情节和对话得判断

{

\_talk.changek3(0);

this->repaint();

}

else if(plot==5)

{

if(\_talk.getk3()<7)

{

int ii = \_talk.getk3();

ii++;

\_talk.changek3(ii);

this->repaint();

}

else if(\_talk.getk3()==7)

{

\_talk.changek3(8);

this->repaint();

changeplot(6);

}

}

else if(plot==7)

{

if(\_talk.getk3()<13)

{

int ii = \_talk.getk3();

ii++;

\_talk.changek3(ii);

this->repaint();

}

else if(\_talk.getk3()==13)

{

\_talk.changek3(14);

this->repaint();

changeplot(8);

}

}

else if(plot==9)

{

if(\_talk.getk3()<16)

{

int ii = \_talk.getk3();

ii++;

\_talk.changek3(ii);

this->repaint();

}

else if(\_talk.getk3()==16)

{

\_talk.changek3(17);

this->repaint();

changeplot(10);

}

}

}

}

**五，实现作战：**

首先我先继承RPGObj类创建一个fightactor类，在这个类里添加新的属性：人物的血量，攻击力和防御力，以及一个判断角色存活的bool变量。

private:

int blood, attract, defence; //人物属性

bool live; //判断人物是否存活

构造函数：

Fightactor::Fightactor()

{

blood=1000;

attract = 100;

defence = 100;

live = true;

}

紧接着我有继承fightactor类创建fightworld类，然后再里面定义fightactor对象，这个fightworld类是为了再作战页面显示人物及各种状态。

private:

Fightactor evil1, evil14, evil15, evil2, evil3, evil21,evil22; //敌人

Fightactor actor1,actor2; //玩家

QImage die; //死亡图片

void Fightworld::show1(QPainter \*painter)

{

if(actor1.getlive() == true) //判断是否存活

this->actor1.show(painter);

else

painter->drawImage(360,500,die); //角色死亡图片

if(evil1.getlive() == true)

this->evil1.show(painter);

else

painter->drawImage(1160,500,die);

if(evil14.getlive() == true)

this->evil14.show(painter);

else

painter->drawImage(1260,400,die);

if(evil15.getlive() == true)

this->evil15.show(painter);

else

painter->drawImage(1260,600,die);

}

void Fightworld::show2(QPainter \*painter) //不同得怪物调用不同得show函数

{

if(actor1.getlive() == true)

this->actor1.show(painter);

else

painter->drawImage(360,500,die);

if(evil2.getlive() == true)

this->evil2.show(painter);

else

painter->drawImage(1160,500,die);

if(evil21.getlive() == true)

this->evil21.show(painter);

else

painter->drawImage(1260,400,die);

if(evil22.getlive() == true)

this->evil22.show(painter);

else

painter->drawImage(1260,600,die);

}

void Fightworld::show3(QPainter \*painter)

{

if(actor1.getlive() == true)

this->actor1.show(painter);

else

painter->drawImage(360,500,die);

if(evil3.getlive() == true)

this->evil3.show(painter);

else

painter->drawImage(1160,500,die);

}

然后我主要是在新建的窗口dialog里画作战界面，利用鼠标点击事件mouseEvent来确定鼠标点击的怪物从而来绘制不同的敌人，然后绘制角色图像，血条。

再创建不同的按钮及相应的槽函数来匹配不同的技能。技能我用QMovie来加载gif动图，在根据不同的技能播放不同的动图。为了可以控制动图的播放时间，我用QTimer设置了时间及其槽函数，技能放完后再设置一个时间来控制怪物的反击，这样就完成了一个回合的对战。怪物反击同样是利用gif动图展示出来，同理，我也是设置了一个函数来控制怪物反击技能的停止。

void Dialog::stop11()

{

if(skill==1) //判断在放的是哪个动图

{

movie11->stop();

movie110->stop();

}

if(skill==2)

{

movie12->stop();

movie120->stop();

}

if(skill==3)

{

movie13->stop();

movie130->stop();

}

ui->label->hide();

ui->label\_3->hide();

time->setSingleShot(true); //控制时间

time->start();

time->setInterval(2000);

}

void Dialog::stop22()

{

skill1->*setVisible*(true);

skill2->*setVisible*(true);

skill3->*setVisible*(true);

if(skill0==1)

{

movie0->stop();

}

if(skill0==2)

{

movie21->stop();

movie210->stop();

}

if(skill0==3)

{

movie31->stop();

movie310->stop();

}

if(skill0==4)

{

movie51->stop();

movie510->stop();

}

if(skill0==5)

{

movie52->stop();

movie520->stop();

}

ui->label\_2->hide();

ui->label\_4->hide();

}

为了怪物的AI，我对怪物反击的函数时会检测怪物的血量，根据血量来设置随机数，从而控制不同不同技能的反击，怪物的血量越低，则怪物释放高伤害技能的可能性就越高，因此并不是每次都能够将怪物杀死，从而实现AI。

if(fightworld.evil1.getblood()<800&&fightworld.evil1.getblood()>300)

//判断血量

{

if (2!=rand()%4) //设置随机数来决定释放技能的几率

{

ui->label\_2->setMovie(movie21);

movie21->start();

ui->label\_2->show();

ui->label\_4->setMovie(movie210);

movie210->start();

ui->label\_4->show();

skill0=2;

time22->setSingleShot(true); //时间控制技能的释放时间

time22->start();

time22->setInterval(2200);

this->fightworld.actor1.changeblood(fightworld.actor1.getblood()-2\*fightworld.evil1.getattract());

k=1;

}

}

else if(fightworld.evil1.getblood()<=300)

{

if (4!=rand()%5)

{

ui->label\_2->setMovie(movie21);

movie21->start();

skill0=2;

ui->label\_2->show();

ui->label\_4->setMovie(movie210);

movie210->start();

ui->label\_4->show();

time22->setSingleShot(true);

time22->start();

time22->setInterval(2200);

this->fightworld.actor1.changeblood(fightworld.actor1.getblood()-3\*fightworld.evil1.getattract());

k=1;

}

}

if(k==0) //如果没有释放技能，则用普通攻击

{

ui->label\_2->setMovie(movie0);

movie0->start();

skill0=1;

ui->label\_2->show();

time22->setSingleShot(true);

time22->start();

time22->setInterval(1600);

this->fightworld.actor1.changeblood(fightworld.actor1.getblood()-fightworld.evil1.getattract());

}

在每次攻击后改变血量，再画出不同的血条，从而实现显示人物的血量。

int xue110; //得到角色血量

xue110=fightworld.evil1.getblood();

xue110=xue110/100;

if(xue110<=0)

xue110=0;

int xue120;

xue120=fightworld.evil14.getblood();

xue120=xue120/100;

if(xue120<=0)

xue120=0;

int xue130;

xue130=fightworld.evil15.getblood();

xue130=xue130/100;

if(xue130<=0)

xue130=0;

int xue210;

xue210=fightworld.actor1.getblood();

xue210=xue210/100;

if(xue210<=0)

xue210=0;

pa->drawImage(QRect (280,450,160,50),blood0,QRect (0,0,200,50)); //绘制血量条

pa->drawImage(QRect (440-16\*(10-xue210),450,16\*(10-xue210),50),blood1,QRect (0,0,200,50));

pa->drawImage(QRect (1180,350,160,50),blood0,QRect (0,0,200,50));

pa->drawImage(QRect (1180,350,32\*(5-xue120),50),blood1,QRect (0,0,200,50));

pa->drawImage(QRect (1080,450,160,50),blood0,QRect (0,0,200,50));

pa->drawImage(QRect (1080,450,16\*(10-xue110),50),blood1,QRect (0,0,200,50));

pa->drawImage(QRect (1180,550,160,50),blood0,QRect (0,0,200,50));

pa->drawImage(QRect (1180,550,32\*(5-xue130),50),blood1,QRect (0,0,200,50));

if(xue110<=0)

{

fightworld.evil1.changelive(false);

}

if(xue120<=0)

{

fightworld.evil14.changelive(false);

}

if(xue130<=0)

{

fightworld.evil15.changelive(false);

}

if(xue210<=0)

{

fightworld.actor1.changelive(false);

}

作战胜利则升级改变角色的参数，作战失败则重置各个角色的状态，这样就可以进行下一次作战。

if(fightworld.evil1.getlive()==false)

{

pa->drawImage(600,100,victory);

enemy1 = false;

fightworld.actor1.changeblood(2000); //胜利后升级改变角色属性

}

else if(fightworld.actor1.getlive()==false)

{

pa->drawImage(700,100,lose);

fightworld.actor1.changeblood(1000); //失败后重置角色血量等属性

fightworld.actor1.changelive(true);

fightworld.evil1.changeblood(1000);

fightworld.evil2.changelive(true);

fightworld.evil14.changeblood(500);

fightworld.evil21.changelive(true);

fightworld.evil15.changeblood(500);

fightworld.evil22.changelive(true);

}