**mygame.cpp**

CGameStateInit::CGameStateInit(CGame \*g): CGameState(g){}

void CGameStateInit::OnInit()

{

// 當圖很多時，OnInit載入所有的圖要花很多時間。為避免玩遊戲的人

// 等的不耐煩，遊戲會出現「Loading ...」，顯示Loading的進度。

ShowInitProgress(0); // 一開始的loading進度為0%

// 開始載入資料

GameScore::Instance()->Loading();

//logo.LoadBitmap(IDB\_BACKGROUND);

Sleep(300); // 放慢，以便看清楚進度，實際遊戲請刪除此Sleep

// 此OnInit動作會接到CGameStaterRun::OnInit()，所以進度還沒到100%

CAudio::Instance()->Load(AUDIO\_BEGIN, "sounds\\begin.mp3");

CAudio::Instance()->Load(AUDIO\_BUTTON, "sounds\\button.mp3");

pic[0].LoadBitmapA("Bitmaps\\BeginState\\1.bmp");

pic[1].LoadBitmapA("Bitmaps\\BeginState\\2.bmp");

pic[2].LoadBitmapA("Bitmaps\\BeginState\\3.bmp");

pic[3].LoadBitmapA("Bitmaps\\BeginState\\4.bmp");

}

void CGameStateInit::OnBeginState()

{

order=0;

}

void CGameStateInit::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)

{

const char KEY\_ESC = 27;

const char KEY\_ENTER = 13;

const char KEY\_LEFT = 0x25; // keyboard左箭頭

const char KEY\_RIGHT = 0x27; // keyboard右箭頭

if(nChar == KEY\_RIGHT){

if(order==0){

CAudio::Instance()->Play(AUDIO\_BUTTON);

order=1;

}

if(order==2){

CAudio::Instance()->Play(AUDIO\_BUTTON);

order=3;

}

}

if(nChar == KEY\_LEFT){

if(order==1){

CAudio::Instance()->Play(AUDIO\_BUTTON);

order=0;

}

if(order==3){

CAudio::Instance()->Play(AUDIO\_BUTTON);

order=2;

}

}

if(nChar == KEY\_ENTER){

CAudio::Instance()->Play(AUDIO\_BUTTON);

if(order==0)order=2;

else {

CAudio::Instance()->Stop(AUDIO\_BEGIN);

if(order==1){

CGameStateRun::multiplayer = 2;

GotoGameState(GAME\_STATE\_RUN);

}

else{

CGameStateRun::multiplayer = 1;

if(order==2)CGameStateRun::Chip\_Dale=0;

else CGameStateRun::Chip\_Dale=1;

GotoGameState(GAME\_STATE\_RUN);

}

}

}

}

void CGameStateInit::OnShow()

{

pic[order].ShowBitmap();

}

/////////////////////////////////////////////////////////////////////////////

// 這個class為遊戲的遊戲執行物件，主要的遊戲程式都在這裡

/////////////////////////////////////////////////////////////////////////////

int CGameStateRun::multiplayer=1;

int CGameStateRun::NowLevel=1;

int CGameStateRun::Chip\_Dale=0;

int CGameStateRun::TimeToBonus=0;

bool CGameStateRun::TimeToGo;

bool CGameStateRun::IfViewer=false;

bool CGameStateRun::IsBoss=false;

ChipDale\* CGameStateRun::player[2];

CGameStateRun::CGameStateRun(CGame \*g)

: CGameState(g)

{

IfViewer = false;

NowLevel=1;

show\_save = false;

show\_load = false;

EditMoveMode = 0;

Maps = NULL;

player[0] = NULL;

player[1] = NULL;

for(int i=0;i<CanTakeNum;i++) //這只需寫一次 初始化陣列

AllThrow[i]=NULL;

for(int i=0;i<ONE\_LEVEL\_MONSTER\_NUM;i++) //這只需寫一次 初始化陣列

AllMonster[i]=NULL;

ChipDale::CanThrow = AllThrow;

ChipDale::AllMonster = AllMonster;

}

CGameStateRun::~CGameStateRun()

{

if(Maps!=NULL) delete(Maps);

for(int i=0;i<ONE\_LEVEL\_MONSTER\_NUM && AllMonster[i]!=NULL;i++){

delete(AllMonster[i]);

}

for(int i=0;i<CanTakeNum;i++){

if(AllThrow[i]==NULL)continue;

delete(AllThrow[i]);

}

for(int i=0;i<2 && player[i]!=NULL;i++)

delete(player[i]);

}

void CGameStateRun::OnBeginState()

{

int initial\_Y;

TimeToGo=false;

EditPointer.SetWxWy(0,0);

Maps = new MapManage(NowLevel);

CAudio::Instance()->PlayLevelMusic(0,true);

CAudio::Instance()->PauseLevelMusic(0);

CAudio::Instance()->Play(AUDIO\_1\_1A);

ChipDale::Maps = Maps;

Maps->SetObstacle();

Maps->SetMonster(AllMonster);

for(int i=0;i<CanTakeNum;i++){

if(AllThrow[i]!=NULL){

delete AllThrow[i];

AllThrow[i]=NULL;

}

}

if(player[0]!=NULL)delete player[0];

if(player[1]!=NULL)delete player[1];

player[0] = new ChipDale(Chip\_Dale);

if(multiplayer==2){

player[1] = new ChipDale(1);

}

else

player[1]=NULL;

player[0]->SetPartner(player[1]);

if(multiplayer==2) player[1]->SetPartner(player[0]);

initial\_Y = (MapManage::Teleport>>7)/DesSize;//得到出生的高 , DesSize 表示 Des 最大的值(128)

TRACE("initial\_Y: %d\n",initial\_Y);

player[0]->InitialWidthHeight();

player[0]->SetWxWy(80,initial\_Y-player[0]->ReturnHeight());

if(multiplayer==2){

player[1]->InitialWidthHeight();

player[1]->SetWxWy(80+player[1]->ReturnWidth(),initial\_Y-player[1]->ReturnHeight());

}

}

void CGameStateRun::OnMove() // 移動遊戲元素

{

// 如果希望修改cursor的樣式，則將下面程式的commment取消即可

// SetCursor(AfxGetApp()->LoadCursor(IDC\_GAMECURSOR));

//TRACE("1\n");

int des,initial\_Y,fixX,fixY;

int toolCDC\_State = ToolCDC::ReturnStage();

static int music\_count=0;

if(music\_count<6\*33-5)music\_count++;

if(music\_count==6\*33-5){

CAudio::Instance()->PlayLevelMusic(NowLevel,true);

music\_count++;

}

if(GameScore::Instance()->Switch)return;

if(TimeToGo||TimeToBonus==1){

CAudio::Instance()->StopLevelMusic(NowLevel);

CAudio::Instance()->Stop(AUDIO\_EDIT);

if(TimeToGo){

player[0]->ResetScore();

if(player[1]!=NULL)player[1]->ResetScore();

TimeToGo=false;

GotoGameState(GAME\_STATE\_OVER);

}else{

TimeToBonus=false;

GotoGameState(GAME\_STATE\_BONUS);

}

}

if(TimeToBonus>0){

if(TimeToBonus==PassMusicTime)

{

CAudio::Instance()->StopLevelMusic(NowLevel);

CAudio::Instance()->Play(AUDIO\_1\_PASS);

}

TimeToBonus--;

}

if(toolCDC\_State==0){

if(!IfViewer){

//人物移動

if(!TimeToBonus){

player[0]->OnMove();

if(multiplayer==2)

player[1]->OnMove();

}

//取消注解此行可得人物Y軸

//static int lastNowY = 0;int tempNowY = //player[0]->ReturnWY()+player[0]->ReturnHeight();if(lastNowY!=tempNowY){TRACE("now\_Y: //%d\n",tempNowY);lastNowY=tempNowY;}

//人物修正

player[0]->FixXY(Maps);

if(multiplayer==2)

player[1]->FixXY(Maps);

for(int i=0;i<CanTakeNum;i++){

if(AllThrow[i]==NULL)continue;

if(AllThrow[i]->ReturnNowAct()==0){

delete(AllThrow[i]);

AllThrow[i]=NULL;

continue;

}

if(AllThrow[i]->ReturnNowAct()>=0)//有時候不想FIX修正位置 可以讓 NowAct 為負的!!!

AllThrow[i]->FixXY(Maps);

AllThrow[i]->OnMove(Maps);

}

//怪物移動+修正+碰撞測試

for(int i=0;i<ONE\_LEVEL\_MONSTER\_NUM && AllMonster[i]!=NULL;i++){

if(AllMonster[i]->OnMove(Maps,player)){

AllMonster[i]->CollisionChipDale(player[0]);

if(multiplayer==2)

AllMonster[i]->CollisionChipDale(player[1]);

}

}

//碰撞 物件 怪物 人物 測試

for(int i=0;i<CanTakeNum;i++){

if(AllThrow[i]==NULL)continue;

AllThrow[i]->CollisionMonster(AllMonster);

AllThrow[i]->CollisionChipDale(player[0]);

if(multiplayer==2)

AllThrow[i]->CollisionChipDale(player[1]);

}

}

else{

EditPointer.OnMove(Maps);

EditPointer.FixXY(Maps);

}

}

else if(toolCDC\_State==2){

des = (Maps->Teleport)%DesSize;//得到出生的高 , DesSize 表示 Des 最大的值(128)

initial\_Y = (Maps->GetRoute((des-1)%10,(des-1)/10)>>7)/DesSize;

TRACE("initial\_Y: %d\n",initial\_Y);

fixX = -Maps->ReturnNowX()+(des-1)%MAX\_XN\*MWIDTH;

fixY = -Maps->ReturnNowY()+(des-1)/MAX\_YN\*MHEIGHT;

Maps->SetMapXY((des-1)%MAX\_XN\*MWIDTH ,(des-1)/MAX\_YN\*MHEIGHT);

Maps->SetRecord();

for(int i=0;i<ONE\_LEVEL\_MONSTER\_NUM && AllMonster[i]!=NULL;i++){

AllMonster[i]->FixMapMove(fixX,fixY);

}

for(int i=0;i<CanTakeNum;i++){

if(AllThrow[i]==NULL)continue;

if(AllThrow[i]->ReturnNowAct()!=1){

delete(AllThrow[i]);

AllThrow[i]=NULL;

}

}

player[0]->Reset(80,initial\_Y-player[0]->ReturnHeight(),true,false);

if(multiplayer==2)

player[1]->Reset(125,initial\_Y-player[1]->ReturnHeight(),true,false);

ToolCDC::ToNextState();

}else if(toolCDC\_State==4){

if(Maps->GetRoute()&64){

IsBoss=true;

CAudio::Instance()->PlayLevelMusic(NowLevel,true);

}else{

IsBoss=false;

}

}

}

void CGameStateRun::OnInit()// 遊戲的初值及圖形設定

{

// 當圖很多時，OnInit載入所有的圖要花很多時間。為避免玩遊戲的人

// 等的不耐煩，遊戲會出現「Loading ...」，顯示Loading的進度。

ShowInitProgress(33); // 接個前一個狀態的進度，此處進度視為33%

//

// 開始載入資料

//

Health[0].LoadBitmapA("Bitmaps/StateRun/health\_1.bmp",PURPLE);

Health[1].LoadBitmapA("Bitmaps/StateRun/health\_2.bmp",PURPLE);

Health[2].LoadBitmapA("Bitmaps/StateRun/health\_3.bmp",PURPLE);

ShowWho[0].LoadBitmapA("Bitmaps/StateRun/chip\_L.bmp",PURPLE);

ShowWho[1].LoadBitmapA("Bitmaps/StateRun/dale\_L.bmp",PURPLE);

ShowWho[2].LoadBitmapA("Bitmaps/StateRun/dale\_R.bmp",PURPLE);

ShowInitProgress(40);

ChipDale::Loading();

EditPointer.Loading();

ToolCDC::Loading();

ShowInitProgress(45);

//怪物圖片loading

MachineDog::Loading();

Cactus::Loading();

Wire::Loading();

Mouse::Loading();

Wasp::Loading();

Centipede::Loading();

//End Loading

ShowInitProgress(50);

//物件圖片loading

Apple::Loading();

Stone::Loading();

Craft::Loading();

Star::Loading();

Explosion::Loading();

Angel::Loading();

ChipDale\_Dead::Loading();

ChipDale\_Resurrect::Loading();

ChipDale\_Taken::Loading();

Greedy::Loading();

Cheese::Loading();

Ball::Loading();

//End Loading

ShowInitProgress(75);

saved.LoadBitmapA("Bitmaps/object/save.bmp",0xffffff);

loading.LoadBitmapA("Bitmaps/object/loading.bmp",0xffffff);

// 完成部分Loading動作，提高進度

ShowInitProgress(80);

// 繼續載入其他資料

CAudio::Instance()->LevelMusicOnit();

CAudio::Instance()->Load(AUDIO\_JUMP, "sounds/jump.mp3");

CAudio::Instance()->Load(AUDIO\_EDIT, "sounds/edit.mp3");

CAudio::Instance()->Load(AUDIO\_DEAD, "sounds/dead.mp3");

CAudio::Instance()->Load(AUDIO\_1\_1, "sounds/state/1/level\_1B.mp3");

CAudio::Instance()->Load(AUDIO\_1\_BOSS,"sounds/state/1/boss\_1B.mp3");

CAudio::Instance()->Load(AUDIO\_1\_1A, "sounds/state/1/level\_1A.mp3");

CAudio::Instance()->Load(AUDIO\_1\_PASS,"sounds/state/1/level\_1pass.mp3");

// 此OnInit動作會接到CGameStaterOver::OnInit()，所以進度還沒到100%

}

void CGameStateRun::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)

{

//密技區 宣告

static char GodMode[]="GOD";static int GodModeI = 0;

static char EditMode[]="EDIT";static int EditModeI = 0;

const char KEY\_LEFT = 0x25; // keyboard左箭頭

const char KEY\_UP = 0x26; // keyboard上箭頭

const char KEY\_RIGHT = 0x27; // keyboard右箭頭

const char KEY\_DOWN = 0x28; // keyboard下箭頭

//密技區 實做

if(nChar==GodMode[GodModeI]){

GodModeI++;

if(GodMode[GodModeI]==0){

player[0]->GodMode();

GodModeI=0;

}

}

else GodModeI=0;

if(nChar==EditMode[EditModeI]){// CapsLock 切換人物與造物者模式

EditModeI++;

if(EditMode[EditModeI]==0){

if(player[0]->ReturnHealth()<=0||multiplayer==2){EditModeI=0;return;}

IfViewer = !IfViewer;

if(IfViewer){

CAudio::Instance()->PlayLevelMusic(0);

}else{

CAudio::Instance()->PlayLevelMusic(NowLevel);

}

player[0]->Reset(EditPointer.ReturnWX(),EditPointer.ReturnWY(),player[0]->ReturnLR(),false);

if(multiplayer==2){

player[1]->Reset(EditPointer.ReturnWX()+player[1]->ReturnWidth(),EditPointer.ReturnWY(),player[1]->ReturnLR(),false);

}

EditPointer.SetMove(false,true,true,true,true,EditMoveMode);

if(IfViewer){

Maps->LevelLoading(NowLevel,true);

Maps->SetMapXY(Maps->ReturnNowX()/ONEOBJX\*ONEOBJX,Maps->ReturnNowY()/ONEOBJY\*ONEOBJY);

}

else{

Maps->SetMapXY(Maps->ReturnNowX()/SPEED\*SPEED,Maps->ReturnNowY()/SPEED\*SPEED);

Maps->SetObstacle();

Maps->SetMonster(AllMonster);

player[0]->ReleaseNowTakeObj();

if(multiplayer==2)

player[1]->ReleaseNowTakeObj();

for(int i=0;i<CanTakeNum;i++){

if(AllThrow[i]==NULL)continue;

if(AllThrow[i]->ReturnNowAct()>=0){

delete(AllThrow[i]);

AllThrow[i]=NULL;

}

}

}

EditModeI = 0;

}

}else EditModeI = 0;

//一般按鍵觸發

if(!IfViewer){

if(multiplayer==1)

player[0]->setFlag(true,(nChar==KEY\_UP),(nChar == KEY\_DOWN),(nChar == KEY\_LEFT),(nChar == KEY\_RIGHT),(nChar == 'Z'),(nChar == 'X'));

if(multiplayer==2){

player[0]->setFlag(true,(nChar==KEY\_UP),(nChar == KEY\_DOWN),(nChar == KEY\_LEFT),(nChar == KEY\_RIGHT),(nChar == 'N'),(nChar == 'M'));

player[1]->setFlag(true,(nChar==87),(nChar == 83),(nChar == 65),(nChar == 68),(nChar == 90),(nChar == 88));

}

if(nChar==9)GameScore::Instance()->ShowScroe();

}

else{

show\_save = false;

EditPointer.SetMove(true,(nChar == KEY\_UP),(nChar == KEY\_DOWN),(nChar == KEY\_LEFT),(nChar == KEY\_RIGHT),EditMoveMode);

if(nChar=='Z') EditPointer.SetOrder(EditPointer.nowOrder()-1);

if(nChar=='X') EditPointer.SetOrder(EditPointer.nowOrder()+1);

if(nChar==13) EditPointer.SetObject(Maps); // Enter 放置

if(nChar==9) EditMoveMode = (EditMoveMode+1)%3; // TAB 切換模式

if(nChar==27) {// ESC 儲存

Maps->SaveObj(NowLevel);

show\_save=true;

}

}

}

void CGameStateRun::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)

{

//TRACE("3\n");

const char KEY\_LEFT = 0x25; // keyboard左箭頭

const char KEY\_UP = 0x26; // keyboard上箭頭

const char KEY\_RIGHT = 0x27; // keyboard右箭頭

const char KEY\_DOWN = 0x28; // keyboard下箭頭

if(!IfViewer){

if(multiplayer==1)

player[0]->setFlag(false,(nChar==KEY\_UP),(nChar == KEY\_DOWN),(nChar == KEY\_LEFT),(nChar == KEY\_RIGHT),(nChar == 'Z'),(nChar == 'X'));

if(multiplayer==2){

player[0]->setFlag(false,(nChar==KEY\_UP),(nChar == KEY\_DOWN),(nChar == KEY\_LEFT),(nChar == KEY\_RIGHT),(nChar == 'N'),(nChar == 'M'));

player[1]->setFlag(false,(nChar==87),(nChar == 83),(nChar == 65),(nChar == 68),(nChar == 90),(nChar == 88));

}

if(nChar==9)GameScore::Instance()->ShowScroe();

}

else{

EditPointer.SetMove(false,(nChar == KEY\_UP),(nChar == KEY\_DOWN),(nChar == KEY\_LEFT),(nChar == KEY\_RIGHT));

}

}

void CGameStateRun::OnShow()

{

// 注意：Show裡面千萬不要移動任何物件的座標，移動座標的工作應由Move做才對，

// 否則當視窗重新繪圖時(OnDraw)，物件就會移動，看起來會很怪。換個術語

// 說，Move負責MVC中的Model，Show負責View，而View不應更動Model。

// 貼上背景圖、撞擊數、球、擦子、彈跳的球

int toolCDC\_State = ToolCDC::ReturnStage();

if(GameScore::Instance()->Switch){

GameScore::Instance()->onShow();

return;

}

if(toolCDC\_State==0 || toolCDC\_State==1 || toolCDC\_State==4){

Maps->OnShow();

if(!IfViewer){

for(int i=0;i<CanTakeNum;i++){//物件顯示靜止 NowAct=3 顯是在怪物之後

if(AllThrow[i]==NULL)continue;

if(AllThrow[i]->ReturnNowAct()!=3)continue;

AllThrow[i]->OnShow(Maps);

}

for(int i=0;i<ONE\_LEVEL\_MONSTER\_NUM && AllMonster[i]!=NULL;i++){//怪物顯示

AllMonster[i]->OnShow(Maps);

}

for(int i=0;i<CanTakeNum;i++){//物件顯示靜止 NowAct=5 靜止在地上 顯是在怪物之前

if(AllThrow[i]==NULL)continue;

if(AllThrow[i]->ReturnNowAct()!=5)continue;

AllThrow[i]->OnShow(Maps);

}

Maps->OnShowObject();

//物件顯示在手中 NowAct=1

if(player[0]->ReturnNowTakeObj()!=NULL)

player[0]->ReturnNowTakeObj()->OnShow(Maps);

if(multiplayer==2 && player[1]->ReturnNowTakeObj()!=NULL){

player[1]->ReturnNowTakeObj()->OnShow(Maps);

}

player[0]->OnShow();//人物顯示

if(multiplayer==2)

player[1]->OnShow();

for(int i=0;i<CanTakeNum;i++){//物件顯示運動中 NowAct = 2 4

if(AllThrow[i]==NULL)continue;

if(AllThrow[i]->ReturnNowAct()!=2 && AllThrow[i]->ReturnNowAct()!=4)continue;

AllThrow[i]->OnShow(Maps);

}

ShowWho[Chip\_Dale].SetTopLeft(0,0);

ShowWho[Chip\_Dale].ShowBitmap();

switch(player[0]->ReturnHealth()){//血量顯示

case 1 : Health[0].SetTopLeft(0,0);Health[0].ShowBitmap();break;

case 2 : Health[1].SetTopLeft(0,0);Health[1].ShowBitmap();break;

case 3 : Health[2].SetTopLeft(0,0);Health[2].ShowBitmap();break;

}

if(multiplayer==2){

ShowWho[2].SetTopLeft(MWIDTH-ShowWho[2].Width(),0);

ShowWho[2].ShowBitmap();

switch(player[1]->ReturnHealth()){//血量顯示

case 1 : Health[0].SetTopLeft(MWIDTH-ShowWho[2].Width(),0);Health[0].ShowBitmap();break;

case 2 : Health[1].SetTopLeft(MWIDTH-ShowWho[2].Width(),0);Health[1].ShowBitmap();break;

case 3 : Health[2].SetTopLeft(MWIDTH-ShowWho[2].Width(),0);Health[2].ShowBitmap();break;

}

}

}

else

EditPointer.OnShow(Maps);

//ClearisMoveMap 一定要放在所有要 show 物件的最後面

Maps->ClearisMoveMap(IfViewer);

if(show\_save)

saved.ShowBitmap();

if(toolCDC\_State==1||toolCDC\_State==4){

ToolCDC::SaveCDC();

ToolCDC::ToNextState();

if(toolCDC\_State==4) ToolCDC::CDDraw();

}

}

else{

ToolCDC::CDDraw();

ToolCDC::ToNextState();

}

}

/////////////////////////////////////////////////////////////////////////////

// 這個class為遊戲的結束狀態(Game Bonus)

/////////////////////////////////////////////////////////////////////////////

CGameStateBonus::CGameStateBonus(CGame\* g)

: CGameState(g)

{

TRACE("BonusConstruct\n");

}

CGameStateBonus::~CGameStateBonus(){

TRACE("~CGameStateBonus()\n");

}

void CGameStateBonus::OnInit(){

TRACE("BonusInit\n");

}

void CGameStateBonus::OnBeginState(){

TRACE("BonusOnBeginState\n");

}

void CGameStateBonus::OnMove(){}

void CGameStateBonus::OnShow(){}

/////////////////////////////////////////////////////////////////////////////

// 這個class為遊戲的結束狀態(Game Over)

/////////////////////////////////////////////////////////////////////////////

CGameStateOver::CGameStateOver(CGame \*g): CGameState(g){}

void CGameStateOver::OnMove()

{

counter--;

if (counter < 0){

CAudio::Instance()->Play(AUDIO\_BEGIN,true);

GotoGameState(GAME\_STATE\_INIT);

}

}

void CGameStateOver::OnBeginState()

{

counter = 30 \* 5; // 5 seconds

CAudio::Instance()->Play(AUDIO\_GAMEOVER);

}

void CGameStateOver::OnInit()

{

// 當圖很多時，OnInit載入所有的圖要花很多時間。為避免玩遊戲的人

// 等的不耐煩，遊戲會出現「Loading ...」，顯示Loading的進度。

ShowInitProgress(90); // 接個前一個狀態的進度，此處進度視為66%

// 開始載入資料

//Sleep(300); // 放慢，以便看清楚進度，實際遊戲請刪除此Sleep

// 最終進度為100%

ShowInitProgress(100);

CAudio::Instance()->Load(AUDIO\_GAMEOVER,"sounds\\gameover.mp3");

CAudio::Instance()->Play(AUDIO\_BEGIN,true);

}

void CGameStateOver::OnShow()

{

CDC \*pDC = CDDraw::GetBackCDC(); // 取得 Back Plain 的 CDC

CFont f,\*fp;

f.CreatePointFont(160,"Times New Roman"); // 產生 font f; 160表示16 point的字

fp=pDC->SelectObject(&f); // 選用 font f

pDC->SetBkColor(RGB(0,0,0));

pDC->SetTextColor(RGB(255,255,0));

char str[80]; // Demo 數字對字串的轉換

sprintf(str, "Game Over ! (%d)", counter / 30);

pDC->TextOut(240,210,str);

pDC->SelectObject(fp); // 放掉 font f (千萬不要漏了放掉)

CDDraw::ReleaseBackCDC(); // 放掉 Back Plain 的 CDC

}

void CAudio::Resume\_for\_CGame()

{

int state=CGame::Instance()->WhichState();

if(state==GAME\_STATE\_RUN){

if(CGameStateRun::IfViewer)

PlayLevelMusic(0);

else

{

PlayLevelMusic(CGameStateRun::NowLevel);

}

}else

Resume();

}

void CAudio::PlayLevelMusic(int level,bool stop\_and\_play){

if (!Instance()->isOpened)

return;

int Notboss;

if(CGameStateRun::IsBoss)Notboss=0;

else Notboss=1;

if(level==0)Notboss=0;

char command[MAX\_MCI\_COMMAND\_SIZE];

if(stop\_and\_play)

sprintf(command, "play device%d from 0 repeat", LevelMusic.at(level\*2-Notboss));

else

sprintf(command, "resume device%d", LevelMusic.at(level\*2-Notboss));

Instance()->SendMciCommand(command);

if(level!=0){

Instance()->PauseById(LevelMusic[0]);

if(Notboss==0)Instance()->PauseById(LevelMusic[level\*2-1]);

}

else{

Instance()->PauseById(LevelMusic[CGameStateRun::NowLevel\*2-1]);

Instance()->PauseById(LevelMusic[CGameStateRun::NowLevel\*2]);

}

}

void CAudio::PauseLevelMusic(int level){

if (!Instance()->isOpened)

return;

int Notboss;

if(CGameStateRun::IsBoss)Notboss=0;

else Notboss=1;

if(level==0)Notboss=0;

char command[MAX\_MCI\_COMMAND\_SIZE];

sprintf(command, "pause device%d wait", LevelMusic.at(level\*2-Notboss));

Instance()->SendMciCommand(command);

}

void CAudio::StopLevelMusic(int level){

if (!Instance()->isOpened)

return;

char command[MAX\_MCI\_COMMAND\_SIZE];

sprintf(command, "stop device%d", LevelMusic.at(level\*2-1));

Instance()->SendMciCommand(command);

sprintf(command, "stop device%d", LevelMusic.at(level\*2));

Instance()->SendMciCommand(command);

}

void CAudio::LevelMusicOnit(){

static bool IsExecuted=false;

if(IsExecuted)return;

LevelMusic.insert(pair<int,int>(0,AUDIO\_EDIT));

LevelMusic.insert(pair<int,int>(1,AUDIO\_1\_1));

LevelMusic.insert(pair<int,int>(2,AUDIO\_1\_BOSS));

IsExecuted=true;

}

void CAudio::PauseById(int id)

{

int j=0;

if (!isOpened)

return;

map<int, Info>::iterator i;

i=info.find(id);

if (i->second.isGood) {

char command[MAX\_MCI\_COMMAND\_SIZE];

sprintf(command, "pause device%d wait", i->first);

SendMciCommand(command);

}

}

void CAudio::SetSpeedByID(unsigned int id,int speed){

if (!isOpened)

return;

GAME\_ASSERT(info.find(id) != info.end(), "Can not play back audio: incorrect Audio ID!");

if (!info[id].isGood)

return;

char command[400];

sprintf(command, "set device%d speed %d", id,speed);

SendMciCommand(command);

}

/////////////////////////////////////////////////////////////////////////////

// ToolCDC

/////////////////////////////////////////////////////////////////////////////

COLORREF ToolCDC::color[MHEIGHT][MWIDTH];

CAnimation ToolCDC::MovePointer;

CMovingBitmap ToolCDC::loading\_BG,ToolCDC::loading\_Bar,ToolCDC::loading\_Mask;

CMovingBitmap ToolCDC::letter[LEVELNUM],ToolCDC::level;

int ToolCDC::NowState=0;

double ToolCDC::alpha=0.0;

ToolCDC::ToolCDC(){}

void ToolCDC::Loading(){

char name[100];

MovePointer.SetDelayCount(20);

MovePointer.AddBitmap("Bitmaps/action/","Chip/","run\_1R.bmp",PURPLE);

MovePointer.AddBitmap("Bitmaps/action/","Chip/","run\_2R.bmp",PURPLE);

MovePointer.AddBitmap("Bitmaps/action/","Chip/","run\_3R.bmp",PURPLE);

MovePointer.AddBitmap("Bitmaps/action/","Chip/","run\_4R.bmp",PURPLE);

loading\_BG.LoadBitmapA("Bitmaps/BeginState/loadingBG.bmp",PURPLE);

loading\_Bar.LoadBitmapA("Bitmaps/BeginState/loadingBar.bmp",PURPLE);

loading\_Mask.LoadBitmapA("Bitmaps/BeginState/loadingMask.bmp",PURPLE);

level.LoadBitmapA("Bitmaps/BeginState/Level.bmp",PURPLE);

for(int i=0;i<LEVELNUM;i++){

sprintf(name,"Bitmaps/BeginState/Letter%d.bmp",i+1);

letter[i].LoadBitmapA(name,PURPLE);

}

}

void ToolCDC::CDDraw()

{

CDC \*pDC = CDDraw::GetBackCDC(); // 取得 Back Plain 的 CDC

for(int i=0;i<MWIDTH;i+=1){

for(int j=0;j<MHEIGHT;j+=1){

pDC->SetPixel(i,j,ALPHA(alpha,i,j));

}

}

CDDraw::ReleaseBackCDC(); // 放掉 Back Plain 的 CDC

}

void ToolCDC::SaveCDC()

{

CDC \*pDC = CDDraw::GetBackCDC(); // 取得 Back Plain 的 CDC

for(int i=0;i<MWIDTH;i+=1){

for(int j=0;j<MHEIGHT;j+=1){

if(i%2==0&&j%2==0)

color[j][i] =pDC->GetPixel(i,j);

else

color[j][i] = color[j/2\*2][i/2\*2];

}

}

CDDraw::ReleaseBackCDC(); // 放掉 Back Plain 的 CDC

if(NowState<=2) alpha = 0.75;

if(NowState>=3) alpha = 0.25;

}

void ToolCDC::ToNextState()

{

switch(NowState){

case 1:case 2:case 4:

NowState += 1;

break;

case 3:

if(alpha!=0.0){

alpha -= 0.25;

if(alpha<0.0)alpha=0.0;

}

else{

NowState += 1;

}

break;

case 5:

if(alpha!=1.0){

alpha += 0.25;

if(alpha>1.0)alpha=1.0;

}

else{

NowState = 0;

}

break;

}

}

void ToolCDC::ShowProgressBar(int WhichLevel,int percent){

int width = MovePointer.Width(),height = MovePointer.Height();

if(percent>100)percent = 100;

CDDraw::BltBackColor(DEFAULT\_BG\_COLOR); // 將 Back Plain 塗上預設的顏色

loading\_Mask.SetTopLeft(95,275);

loading\_Mask.ShowBitmap();

loading\_Bar.SetTopLeft(95-450\*(100-percent)/100,275);

loading\_Bar.ShowBitmap();

loading\_BG.SetTopLeft(0,0);

loading\_BG.ShowBitmap();

MovePointer.SetBottomLeft(95+450\*percent/100-(width)/2,275-(height-30)/2,height);

MovePointer.OnShow();

MovePointer.OnMove();

level.SetTopLeft(160,140);

level.ShowBitmap();

letter[WhichLevel-1].SetTopLeft(415,140);

letter[WhichLevel-1].ShowBitmap();

CDDraw::BltBackToPrimary(); // 將 Back Plain 貼到螢幕

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//GameMapEdit實作 \*/

// \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GameMapEdit::GameMapEdit(){

Order = 0;

wx = wy = 0;

for(int i=0;i<4;i++)

move[i]=false;

ContinueMode = 0;

}

void GameMapEdit::Loading(){

char temp[100];

frame\_pic.LoadBitmapA("Bitmaps/object/Pointer.bmp");

for(int i=0;i<OrderSize;i++){

sprintf(temp,"Bitmaps/object/%d.bmp",i);

frame\_background[i].LoadBitmapA(temp,PURPLE);

}

for(int i=-1;i>=-MonsterOrderSize;i--){

sprintf(temp,"Bitmaps/monster/%d.bmp",-i);

frame\_background[-i+OrderSize-1].LoadBitmapA(temp,PURPLE);

}

}

void GameMapEdit::OnShow(MapManage \*map){

int temp;

//顯示 目前螢幕上的OBJ

for(int i=0;i<=(MWIDTH+2\*WinShowBuffer)/ONEOBJX;i++)

for(int j=0;j<=(MHEIGHT+2\*WinShowBuffer)/ONEOBJY;j++)

map->ShowObstacle(i,j);

//顯示 目前位置

temp = (Order<0)\*(-Order+OrderSize-1) + (Order>=0)\*(Order);

if(Order==0)

frame\_background[0].SetTopLeft(wx-16,wy-16);

else

frame\_background[temp].SetTopLeft(wx,wy);

frame\_background[temp].ShowBitmap();

frame\_pic.SetTopLeft(wx,wy);

frame\_pic.ShowBitmap();

}

void GameMapEdit::SetOrder(int value){

if (value < OrderSize && value >= -MonsterOrderSize)

Order = value;

if (value < -MonsterOrderSize)

Order = OrderSize-1;

if (value >=OrderSize)

Order = -MonsterOrderSize;

}

void GameMapEdit::OnMove(MapManage \*map)

{

int width,height,temp; //目前圖片高度和寬度

int timesX,timesY;

if(ContinueMode==0) {timesX=1; timesY=3; }

else if(ContinueMode==1){timesX=1; timesY=1; }

else if(ContinueMode==2){timesX=-1;timesY=-1;}

temp = (Order<0)\*(-Order+OrderSize-1) + (Order>=0)\*(Order);

width = frame\_background[temp].Width();

height = frame\_background[temp].Height();

if(timesX==-1||timesY==-1){

timesX = width/ONEOBJX;

timesY = height/ONEOBJY;

}

static int limitX = (MWIDTH + width)/2;

static int limitY = (MHEIGHT + height)/2;

if(move[0]){

wy -= ONEOBJY\*timesY;

if(wy < 0){

wy += ONEOBJY\*timesY;

}

if(wy < limitY){

if((map->GetRoute()&14) && ((map->ReturnNowY())%MHEIGHT)<ONEOBJY\*timesY && ((map->ReturnNowY())%MHEIGHT)>0)

timesY =((map->ReturnNowY())%MHEIGHT)/ONEOBJY;

map->MoveMap(1,ONEOBJX\*timesX,ONEOBJY\*timesY);

}

ThisMoveTimesX=timesX;

ThisMoveTimesY=timesY;

}

if(move[1]){

wy += ONEOBJY\*timesY;

if(wy > MHEIGHT-height){

wy -= ONEOBJY\*timesY;

}

if(wy > limitY){

if((map->GetRoute()&13) && MHEIGHT-((map->ReturnNowY())%MHEIGHT)<ONEOBJY\*timesY && MHEIGHT-((map->ReturnNowY())%MHEIGHT)>0)

timesY =(MHEIGHT-((map->ReturnNowY())%MHEIGHT))/ONEOBJY;

map->MoveMap(2,ONEOBJX\*timesX,ONEOBJY\*timesY);

}

ThisMoveTimesX=timesX;

ThisMoveTimesY=timesY;

}

if(move[2]){

wx -= ONEOBJX\*timesX;

if(wx < 0){

wx += ONEOBJX\*timesX;

}

if(wx < limitX)

map->MoveMap(4,ONEOBJX\*timesX,ONEOBJY\*timesY);

ThisMoveTimesX=timesX;

ThisMoveTimesY=timesY;

}

if(move[3]){

wx += ONEOBJX\*timesX;

if(wx > MWIDTH-width){

wx -= ONEOBJX\*timesX;

}

if(wx > limitX)

map->MoveMap(8,ONEOBJX\*timesX,ONEOBJY\*timesY);

ThisMoveTimesX=timesX;

ThisMoveTimesY=timesY;

}

if(ContinueMode>0){

move[0] = move[1] = move[2] = move[3] = false;

ContinueMode =0;

}

}

void GameMapEdit::SetWxWy(int setWx,int setWy)

{

wx = setWx;

wy = setWy;

}

void GameMapEdit::SetMove(bool flag,bool IsUp,bool IsDown,bool IsLeft,bool IsRight,int mode)

{

if(IsUp) move[0]=flag;

if(IsDown) move[1]=flag;

if(IsLeft) move[2]=flag;

if(IsRight) move[3]=flag;

if(mode) ContinueMode = mode;

}

void GameMapEdit::FixXY(MapManage \*map)

{

int temp = map->MoveMap(0);

if(temp&1)

wy += ONEOBJY\*ThisMoveTimesY;

if(temp&2)

wy -= ONEOBJY\*ThisMoveTimesY;

if(temp&4)

wx += ONEOBJX\*ThisMoveTimesX;

if(temp&8)

wx -= ONEOBJX\*ThisMoveTimesX;

ThisMoveTimesX=0;

ThisMoveTimesY=0;

}

void GameMapEdit::SetObject(MapManage \*map)

{

map->SetObj(Order,wx,wy);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//GameScore實作 \*/

// \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

GameScore GameScore::Game\_Score;

GameScore::GameScore(){

Switch=false;

score\_p=new CInteger(2);

}

GameScore::~GameScore(){

delete(score\_p);

TRACE("~GameScore()\n");

}

GameScore\* GameScore::Instance()

{

return &Game\_Score;

}

void GameScore::Loading(){

Head[0].LoadBitmapA("Bitmaps/score/chiphead.bmp");

Head[1].LoadBitmapA("Bitmaps/score/dalehead.bmp");

F.LoadBitmapA("Bitmaps/score/F.bmp",PURPLE);

S.LoadBitmapA("Bitmaps/score/S.bmp",PURPLE);

L\_C.LoadBitmapA("Bitmaps/score/L\_C.bmp",PURPLE);

L\_D.LoadBitmapA("Bitmaps/score/L\_D.bmp",PURPLE);

score\_p->LoadBitmap();

}

void GameScore::ShowScroe(){

if(Switch==true)Switch=false;

else Switch=true;

}

void GameScore::onShow(){

if(Switch==false)return;

if(CGameStateRun::player[0]->ReturnIsDale()==0)

{

Head[0].SetTopLeft(MWIDTH/2-100-Head[0].Width(),100);

Head[0].ShowBitmap();

F.SetTopLeft(MWIDTH/2-100-Head[0].Width(),240);

S.SetTopLeft(MWIDTH/2-100-Head[0].Width(),300);

L\_C.SetTopLeft(MWIDTH/2-100-Head[0].Width(),360);

F.ShowBitmap();

S.ShowBitmap();

L\_C.ShowBitmap();

score\_p->SetInteger(CGameStateRun::player[0]->Score\_Flower);

score\_p->SetTopLeft(MWIDTH/2-50-Head[0].Width(),250);

score\_p->ShowBitmap();

score\_p->SetInteger(CGameStateRun::player[0]->Score\_Star);

score\_p->SetTopLeft(MWIDTH/2-50-Head[0].Width(),310);

score\_p->ShowBitmap();

score\_p->SetInteger(CGameStateRun::player[0]->ReturnLife());

score\_p->SetTopLeft(MWIDTH/2-50-Head[0].Width(),370);

score\_p->ShowBitmap();

}

if(CGameStateRun::multiplayer==2||CGameStateRun::player[0]->ReturnIsDale())

{

ChipDale\* p=CGameStateRun::player[0];

if(CGameStateRun::multiplayer==2)p=CGameStateRun::player[1];

Head[1].SetTopLeft(MWIDTH/2+100,100);

Head[1].ShowBitmap();

F.SetTopLeft(MWIDTH/2+100,240);

S.SetTopLeft(MWIDTH/2+100,300);

L\_D.SetTopLeft(MWIDTH/2+100,360);

F.ShowBitmap();

S.ShowBitmap();

L\_D.ShowBitmap();

score\_p->SetInteger(p->Score\_Flower);

score\_p->SetTopLeft(MWIDTH/2+150,250);

score\_p->ShowBitmap();

score\_p->SetInteger(p->Score\_Star);

score\_p->SetTopLeft(MWIDTH/2+150,310);

score\_p->ShowBitmap();

score\_p->SetInteger(p->ReturnLife());

score\_p->SetTopLeft(MWIDTH/2+150,370);

score\_p->ShowBitmap();

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//CFrame實作 \*/

// \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int CFrame::Jump\_Fix = 0;

CFrame::CFrame()

{

wx=wy=0;

for(int i=0;i<6;i++)

flag[i]=false;

LRflag=1;

}

void CFrame::SetWxWy(int setX,int setY,bool shiftmode)

{

if(shiftmode){

if((wx+width+setX)<=MWIDTH&&(wx+setX)>=0) wx += setX;

if((wy+setY)<MHEIGHT) wy += setY;

}

else{

if(setX>=0&&(width+setX)<MWIDTH) wx = setX;

if(setY<MHEIGHT) wy = setY;

}

}

void CFrame::SetWidthHeight(int tempW,int tempH)

{

width = tempW;

height = tempH;

}

void CFrame::FixXY(MapManage \*map)

{

int temp = map->MoveMap(0);

if(temp&1)

wy += Jump\_Fix;

if(temp&2)

wy -= Jump\_Fix;

if(temp&4)

wx += MapManage::LRMargin;

if(temp&8)

wx -= MapManage::LRMargin;

}

void CFrame::ReFixXY(MapManage \*map)

{

int temp = map->MoveMap(0);

if(temp&1)

wy -= Jump\_Fix;

if(temp&2)

wy += SPEED;

if(temp&4)

wx -= MapManage::LRMargin;

if(temp&8)

wx += MapManage::LRMargin;

}

void CFrame::FixMapMove(int fixX,int fixY)

{

wx -= fixX;

wy -= fixY;

}

int CFrame::IfCollision(int twx,int twy,int twidth,int theight)

{

//左上 1

int cx,cy;

for(int i=1;i<=4;i++){

switch(i){

case 1:

cx = twx;

cy = twy;

break;

case 2:

cx = twx+twidth;

cy = twy;

break;

case 3:

cx = twx+twidth;

cy = twy+theight;

break;

case 4:

cx = twx;

cy = twy+theight;

break;

}

if(wx<=cx && wx+width >= cx && wy<=cy && wy+height >= cy) return i;

}

if(twx<=wx && twx+twidth >= wx && twy<=wy && twy+theight >= wy) return 5;

return 0;

}

int CFrame::IfCollision(int Direct,int passSpeed){

MapManage \*map = ChipDale::Maps;

passSpeed = abs(passSpeed);

switch(Direct){

case 1: return map->IfCollision(wx,wy-passSpeed,width,passSpeed,true);

case 4:

if(wx-passSpeed < 0) return 1;

return map->IfCollision(wx-passSpeed,wy,passSpeed,height,true);

case 8:

if(wx+width+passSpeed > MWIDTH) return 1;

return map->IfCollision(wx+width,wy,passSpeed,height,true);

}

return 0;

}

**mapmanage.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//MapManage實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define SET\_INITIAL(Level,LOCATE,INITIAL\_Y) limit[(Level-1)\*MAX\_YN+LOCATE/10][(Level-1)\*MAX\_YN+LOCATE%MAX\_XN-1] += ((INITIAL\_Y\*DesSize)<<7);

#define DELIVER(Level,LOCATE,DES) limit[(Level-1)\*MAX\_YN+LOCATE/10][(Level-1)\*MAX\_YN+LOCATE%MAX\_XN-1] += (DES<<7);

int MapManage::isMoveMap = 0;

int MapManage::Teleport = 0;

int MapManage::LRMargin = 0;

MapManage::MapManage(int WhichLevel)

{

LevelLoading(WhichLevel);

}

void MapManage::LevelLoading(int WhichLevel,bool NotResetAll){

char temp[100];

int count=0,OnePercent = (MAX\_YN\*MAX\_XN\*2 + OrderSize + MonsterOrderSize + 1)/100;

FILE \*file;

static int limit[MAX\_YN\*2][MAX\_XN] = {//第一關

{15, 0,15+64, 0, 0, 0, 0, 0, 0, 0},

{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{ 8, 8, 8, 8, 8, 8, 0, 0, 0, 0},

{ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0},

{ 9, 8, 8, 8, 8, 8, 8+16, 0, 0, 0},

{ 1, 0, 0, 0, 0, 0 ,0 ,0, 0, 0},

{ 5, 4, 4, 4, 4, 5, 0, 0, 0, 0},

{ 0, 0, 0, 0, 0, 1, 0, 0, 0, 0},

{ 0, 0, 0, 0, 0, 1, 0, 0, 0, 0},

{ 8, 8, 8, 8, 8, 9, 0, 0, 0, 0},

//第二關...(待開發)

},IfSetLimit=0;

//initial\_Y 是要輸入 地板的左上角y位置 不是人物的左上角y位置

if(!IfSetLimit){//防止跑兩次以上 因為 limit 是 static!!!! (有修正過)

//第一關

SET\_INITIAL(1,91,135);

SET\_INITIAL(1,21,326);

SET\_INITIAL(1,1,395);

SET\_INITIAL(1,3,395);

DELIVER(1,47,21);

DELIVER(1,26,1);

DELIVER(1,1,3);

//第二關...(待開發)

//關卡設定結束

IfSetLimit=1;

}

if(!NotResetAll){

x=0;y=0;

ToolCDC::ShowProgressBar(WhichLevel,count/OnePercent);

for(int i=0;i<MAX\_XN;i++){

for(int j=0;j<MAX\_YN;j++){

Route[j][i] = limit[(WhichLevel-1)\*MAX\_YN+j][i];

if(Route[j][i]>0){

sprintf(temp,"Bitmaps/0x%d/%d.bmp",WhichLevel,j\*10+i+1);

MapSP[j][i].LoadBitmap(temp);

if(Route[j][i]&16){

x = i\*MWIDTH;

y = j\*MHEIGHT;

recordx=x;

recordy=y;

Teleport = Route[j][i];

}

}

count++;

ToolCDC::ShowProgressBar(WhichLevel,count/OnePercent);

}

}

}

/\* NotResetAll 有跟無差在底下SET \*/

sprintf(temp,"ObjInfo/0x%d.ChipDale",WhichLevel);

file = fopen(temp,"rb");

for(int j=0;j<MAX\_OY;j++)for(int i=0;i<MAX\_OX;i++)MapObjXY[j][i]=0;

if(file){

for(int j=0;j<MAX\_YN;j++){

for(int i=0;i<MAX\_XN;i++){

if(Route[j][i]){

for(int k=0;k<MAX\_OY/MAX\_YN;k++){

fread(&MapObjXY[j\*MAX\_OY/MAX\_YN+k][i\*MAX\_OX/MAX\_XN],MAX\_OX/MAX\_XN,sizeof(int),file);

}

}

if(!NotResetAll){

count++;

ToolCDC::ShowProgressBar(WhichLevel,count/OnePercent);

}

}

}

fclose(file);

}

else{

SaveObj(WhichLevel);

if(!NotResetAll){

count += MAX\_YN\*MAX\_XN;

ToolCDC::ShowProgressBar(WhichLevel,count/OnePercent);

}

}

if(!NotResetAll){

for(int i=0;i<OrderSize;i++){

sprintf(temp,"Bitmaps/object/%d.bmp",i);

frame\_obstacle[i].LoadBitmapA(temp,PURPLE);

count++;

ToolCDC::ShowProgressBar(WhichLevel,count/OnePercent);

}

for(int i=-1;i>=-MonsterOrderSize;i--){

sprintf(temp,"Bitmaps/monster/%d.bmp",-i);

frame\_obstacle[-i+OrderSize-1].LoadBitmapA(temp,PURPLE);

count++;

ToolCDC::ShowProgressBar(WhichLevel,count/OnePercent);

}

frame\_obstacle[OrderSize+MonsterOrderSize + 0].LoadBitmapA("Bitmaps/object/StoneUnVisible.bmp",PURPLE);

count++;

ToolCDC::ShowProgressBar(WhichLevel,100);

}

}

void MapManage::OnShow(){

int gx = x/MWIDTH,gy = y/MHEIGHT; //地圖左上角轉換成螢幕格座標

int ox = x/ONEOBJX,oy = y/ONEOBJY;

for(int i=0;i<2;i++){

for(int j=0;j<2;j++){

if(gx+i>=0 && gx+i<MAX\_XN && gy+j>=0 && gy+j<MAX\_YN && Route[gy+j][gx+i]>0){

MapSP[gy+j][gx+i].SetTopLeft((gx+i)\*MWIDTH-x,(gy+j)\*MHEIGHT-y);

MapSP[gy+j][gx+i].ShowBitmap();

}

}

}

}

void MapManage::OnShowObject(){

int gx = x/MWIDTH,gy = y/MHEIGHT; //地圖左上角轉換成螢幕格座標

int ox = x/ONEOBJX,oy = y/ONEOBJY;

for(int j=-(WinShowBuffer/ONEOBJY);j<MHEIGHT/ONEOBJY;j++){

if(oy+j<0)j=-oy;

for(int i=-(WinShowBuffer/ONEOBJX);i<MWIDTH/ONEOBJX;i++){

if(ox+i<0)i=-ox;

if(MapObjXY[oy+j][ox+i]>=OrderSize||MapObjXY[oy+j][ox+i]<=2)

continue;

frame\_obstacle[MapObjXY[oy+j][ox+i]].SetTopLeft( (ox+i)\*ONEOBJX-x , (oy+j)\*ONEOBJY-y );

frame\_obstacle[MapObjXY[oy+j][ox+i]].ShowBitmap();

}

}

}

void MapManage::ShowObstacle(int i,int j){

int ox = (x-WinShowBuffer)/ONEOBJX,oy = (y-WinShowBuffer)/ONEOBJY,temp=0;

if(ox+i>=0 && ox+i<MAX\_OX && oy+j>=0 && oy+j<MAX\_OY && MapObjXY[oy+j][ox+i]!=0){

if(MapObjXY[oy+j][ox+i]>=OrderSize || MapObjXY[oy+j][ox+i] < -MonsterOrderSize)

return;

temp = (MapObjXY[oy+j][ox+i]<0)\*(-MapObjXY[oy+j][ox+i]+OrderSize-1) + (MapObjXY[oy+j][ox+i]>=0)\*(MapObjXY[oy+j][ox+i]);

frame\_obstacle[ temp ].SetTopLeft((ox+i)\*ONEOBJX-x,(oy+j)\*ONEOBJY-y);

frame\_obstacle[ temp ].ShowBitmap();

}

}

int MapManage::MoveMap(int move,int MoveOneX,int MoveOneY){

if((move&4) || (move&8)){

LRMargin = MoveOneX;

}

if((move&1) || (move&2)){

CFrame::Jump\_Fix = MoveOneY;

}

if( (move&1) && !(isMoveMap&1)){

if(x%MWIDTH == 0 && ( (Route[y/MHEIGHT][x/MWIDTH]&1) || ((Route[y/MHEIGHT][x/MWIDTH]&2)&&(isMoveMap&16)))){

y -=MoveOneY;

isMoveMap |= 1;

if(y<0 || Route[y/MHEIGHT][x/MWIDTH]==0){

y +=MoveOneY;

isMoveMap &= 30;

}

}

else if(x%MWIDTH <= SPEED && (Route[y/MHEIGHT][x/MWIDTH]&1)){

y -=MoveOneY;

x -= x%MWIDTH;

isMoveMap |= (1+4);

LRMargin = x%MWIDTH;

if(y<0 || Route[y/MHEIGHT][x/MWIDTH]==0){

y +=MoveOneY;

isMoveMap &= 30;

}

}

else if((x+SPEED)%MWIDTH <= SPEED && (Route[y/MHEIGHT][(x+SPEED)/MWIDTH]&1)){

y -=MoveOneY;

x += (x+SPEED)/MWIDTH\*MWIDTH - x;

isMoveMap |= (1+8);

LRMargin = (x+SPEED)/MWIDTH\*MWIDTH - x;

if(y<0 || Route[y/MHEIGHT][x/MWIDTH]==0){

y +=MoveOneY;

isMoveMap &= 30;

}

}

}

if( (move&2) && !(isMoveMap&2)){

if(x%MWIDTH == 0 && ( (Route[y/MHEIGHT][x/MWIDTH]&2) || ((Route[y/MHEIGHT][x/MWIDTH]&1)&&(isMoveMap&16)))){

y +=MoveOneY;

isMoveMap |= 2;

if(y+MHEIGHT-1>MHEIGHT\*MAX\_YN || Route[(y+MHEIGHT-1)/MHEIGHT][x/MWIDTH]==0){

y -=MoveOneY;

isMoveMap &= 29;

}

}

}

if( (move&4) && !(isMoveMap&4)){

if(y%MHEIGHT == 0 && ( (Route[y/MHEIGHT][x/MWIDTH]&4) || ((Route[y/MHEIGHT][x/MWIDTH]&8)&&(isMoveMap&16)))){

x -=MoveOneX;

isMoveMap |= 4;

if(x<0 || Route[y/MHEIGHT][x/MWIDTH]==0){

x +=MoveOneX;

isMoveMap &= 27;

}

}

}

if( (move&8) && !(isMoveMap&8)){

if(y%MHEIGHT == 0 && ( (Route[y/MHEIGHT][x/MWIDTH]&8) || ((Route[y/MHEIGHT][x/MWIDTH]&4)&&(isMoveMap&16)))){

x +=MoveOneX;

isMoveMap |= 8;

if(x+MWIDTH-1>MWIDTH\*MAX\_XN || Route[y/MHEIGHT][(x+MWIDTH-1)/MWIDTH]==0){

x -=MoveOneX;

isMoveMap &= 23;

}

}

}

return isMoveMap;

}

void MapManage::SetMapXY(int setX,int setY){

x = setX;

y = setY;

}

void MapManage::Set\_toRecord(){

Object \*\*AllThrow = ChipDale::CanThrow;

LevelLoading(CGameStateRun::NowLevel,true);

SetMapXY(recordx,recordy);

SetObstacle();

SetMonster(ChipDale::AllMonster);

for(int i=0;i<CanTakeNum;i++){

if(AllThrow[i]!=NULL)

AllThrow[i]->SetNowAct(0);

}

}

void MapManage::SetObj(int value,int wx,int wy){

MapObjXY[(wy+y)/ONEOBJY][(wx+x)/ONEOBJX] = value;

}

void MapManage::SaveObj(int WhichLevel){

char temp[100];

FILE \*file;

sprintf(temp,"ObjInfo/0x%d.ChipDale",WhichLevel);

file = fopen(temp,"wb");

for(int j=0;j<MAX\_YN;j++){

for(int i=0;i<MAX\_XN;i++){

if(Route[j][i]){

for(int k=0;k<MAX\_OY/MAX\_YN;k++){

fwrite(&MapObjXY[j\*MAX\_OY/MAX\_YN+k][i\*MAX\_OX/MAX\_XN],MAX\_OX/MAX\_XN,sizeof(int),file);

}

}

}

}

fclose(file);

}

void MapManage::ClearisMoveMap(bool WatchMode)

{

if(WatchMode)isMoveMap = 16;

else isMoveMap=0;

}

int MapManage::IfCollision(int wx,int wy,int PicWidth,int PicHeight,bool ignore\_obj\_2,bool ignore\_eat,bool findNext,bool Reverse)

{

int start\_ox,start\_oy,end\_ox,end\_oy,tj;

if(wy<0){

PicHeight += wy;

wy=0;

}

start\_ox = (x+wx)/ONEOBJX;

start\_oy = (y+wy)/ONEOBJY;

if(start\_ox<0)start\_ox = 0;

if(start\_oy<0)start\_oy = 0;

end\_ox = (x+wx+PicWidth)/ONEOBJX;

end\_oy = (y+wy+PicHeight)/ONEOBJY;

if(end\_ox>=MAX\_OX)end\_ox = MAX\_OX-1;

if(end\_oy>=MAX\_OY)end\_oy = MAX\_OY-1;

for(int j=end\_oy-1;j>=start\_oy;j--){

if(Reverse) tj = (start\_oy+end\_oy-1)-j;

else tj = j;

for(int i=start\_ox;i<end\_ox;i++){

if(!ignore\_eat && ( Obstacle[tj][i]/E8==4 || Obstacle[tj][i]/E8>=7 && Obstacle[tj][i]/E8<=13 || Obstacle[tj][i]/E8==17) )//花

continue;

if(ignore\_obj\_2 && Obstacle[tj][i]==2)

continue;

if(Obstacle[tj][i]>0){

if(Obstacle[tj][i]!=1&&Obstacle[tj][i]!=2)

if(findNext==true){

j=(Obstacle[tj][i]%E8)/E4-1;

findNext=false;

break;

}

return Obstacle[tj][i];

}

}

}

return 0;

}

void MapManage::SetObstacle()

{

for(int i=0;i<MAX\_OX;i++)

for(int j=0;j<MAX\_OY;j++)

Obstacle[j][i]=0;

int ox\_end,oy\_end,ox\_start,oy\_start;

for(int i=0;i<MAX\_OX;i++)

{

for(int j=0;j<MAX\_OY;j++)

{

if(MapObjXY[j][i] < OrderSize && MapObjXY[j][i] > 0){

ox\_start=i;

oy\_start=j;

switch(MapObjXY[j][i]){

case 1:

case 3:case 4:case 5:case 6:case 7:case 8:

case 12:case 13:case 14:case 15:case 16:case 17:case 18:case BALL:

ox\_end = ox\_start + frame\_obstacle[MapObjXY[j][i]].Width()/ONEOBJX;

oy\_end = oy\_start + frame\_obstacle[MapObjXY[j][i]].Height()/ONEOBJY;

break;

case 9:case 10:case 11:

ox\_end = ox\_start + frame\_obstacle[MapObjXY[j][i]].Width()/ONEOBJX;

oy\_end = oy\_start + frame\_obstacle[MapObjXY[j][i]].Height()/ONEOBJY;

oy\_start += 5;

ox\_start += 2;

ox\_end -= 2;

break;

case 2:

ox\_end = ox\_start + frame\_obstacle[MapObjXY[j][i]].Width()/ONEOBJX;

oy\_end = oy\_start + 1;

break;

default:

ox\_end = ox\_start;oy\_end = oy\_start;

break;

}

for(int l=oy\_start;l<oy\_end;l++){

for(int k=ox\_start;k<ox\_end;k++){

if(MapObjXY[j][i]==1||MapObjXY[j][i]==2)

Obstacle[l][k] = MapObjXY[j][i];

else

Obstacle[l][k] = MapObjXY[j][i]\*E8+(j+1)\*E4+(i+1);

}

}

switch(MapObjXY[j][i]){

case 9:case 10:case 11:

MapObjXY[j][i]=18;

break;

case 14:case 15:case 16:

MapObjXY[j][i]=3;

break;

case 13:case 17:

MapObjXY[j][i]=0;

break;

}

}

}

}

}

void MapManage::SetMonster(Monster \*monster[])

{

int lengthO,k=0;

for(int i=k;i<ONE\_LEVEL\_MONSTER\_NUM;i++){

if(monster[i]!=NULL){

delete(monster[i]);

}

}

for(int j=0;j<MAX\_OY;j++){

for(int i=0;i<MAX\_OX;i++){

switch(MapObjXY[j][i]){

case -1:

monster[k] = new MachineDog(this,i,j);

k++;break;

case -2:

monster[k] = new Cactus(this,i,j);

k++;break;

case -3:

for(lengthO=i+1;lengthO<i+50;lengthO++){

if(MapObjXY[j][lengthO] == -3){

MapObjXY[j][lengthO] = 0;

break;

}

}

monster[k] = new Wire(this,i,j,lengthO-i);

k++;break;

case -4:

monster[k] = new Mouse(this,i,j);

k++;break;

case -5:

monster[k] = new Wasp(this,i,j);

k++;break;

case -6:

monster[k] = new Centipede(this,i,j);

k++;break;

}

}

}

if(k>=ONE\_LEVEL\_MONSTER\_NUM){

TRACE("SetMonster Overflow ( %d )!!!\n",k);

Sleep(10000);

}

else{

for(int i=k;i<ONE\_LEVEL\_MONSTER\_NUM;i++)

monster[i] = NULL;

}

}

void MapManage::ClearObstacle(int Value)

{

int ox\_start,oy\_start;

int ox\_end,oy\_end;

int ObjetID;

ox\_start = Value%E4-1; Value /= E4;

oy\_start = Value%E4-1; Value /= E4;

ObjetID = Value;

if(Value!=0){

ox\_end = ox\_start + frame\_obstacle[ObjetID].Width()/ONEOBJX;

oy\_end = oy\_start + frame\_obstacle[ObjetID].Height()/ONEOBJY;

for(int k=ox\_start;k<ox\_end;k++)

for(int l=oy\_start;l<oy\_end;l++)

Obstacle[l][k] = 0;

MapObjXY[oy\_start][ox\_start]=0;

if(Value==5)

MapObjXY[oy\_start+1][ox\_start]=0;

}

}

int MapManage::FillObstacle(int Value,int setWx,int setWy,bool Visible)

{

int minusX=0,minusY=0;

int ox\_start,oy\_start;

int ox\_end,oy\_end;

int ObjetID;

ox\_start = (setWx + x)/ONEOBJX;

oy\_start = (setWy + y)/ONEOBJY;

ObjetID = Value;

if(Value!=0){

ox\_end = ox\_start + frame\_obstacle[ObjetID].Width()/ONEOBJX;

oy\_end = oy\_start + frame\_obstacle[ObjetID].Height()/ONEOBJY;

switch(ObjetID){

case 4:case 7:case 8:

minusX = 1;

minusY = 1;

break;

}

for(int l=oy\_start+minusY;l<oy\_end-minusY;l++)

for(int k=ox\_start+minusX;k<ox\_end-minusX;k++)

Obstacle[l][k] = Value\*E8+(oy\_start+1)\*E4+(ox\_start+1);

if(Visible)

MapObjXY[oy\_start][ox\_start] = Value;

}

return Value\*E8+(oy\_start+1)\*E4+(ox\_start+1);

}

**instruction.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//AI\_Instructions實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

AI\_Instructions::AI\_Instructions()

{

UpSpeed=0;

IsJump = false;

LR\_Space = 0;

NoIgnore\_2 = true;

NoIgnore\_eat = false;

}

bool AI\_Instructions::MoveUp(MapManage\* map){

if(NoCollision || !map->IfCollision(wx,wy-MoveSPEED,width,height,true)){

wy -= MoveSPEED;

return true;

}

return false;

}

bool AI\_Instructions::MoveDown(MapManage\* map){

if(NoCollision || !map->IfCollision(wx,wy+MoveSPEED,width,height,true)){

wy += MoveSPEED;

return true;

}

return false;

}

bool AI\_Instructions::MoveLeft(MapManage\* map){

if(NoCollision || !map->IfCollision(wx-MoveSPEED,wy,MoveSPEED,height,true)){

wx -= MoveSPEED;

return true;

}

return false;

}

bool AI\_Instructions::MoveRight(MapManage\* map){

if(NoCollision || !map->IfCollision(wx+width,wy,MoveSPEED,height,true)){

wx += MoveSPEED;

return true;

}

return false;

}

void AI\_Instructions::Jump()

{

UpSpeed = JumpSPEED;

}

int AI\_Instructions::FallingDown(MapManage\* map)

{

int i=0;

int fixY=0,fixX=0;

int IsLand=false;

//修正落下時左右兩邊應該空多少不判斷

wx += LR\_Space;

width -= 2\*LR\_Space;

//最後面記得要回復原本設定

//落下過程的FLAG初始化

if(IsJump || !map->IfCollision(wx+fixX,wy+height+fixY,width,1,false,NoIgnore\_eat/\*ONEOBJY\*/)){

UpSpeed -= GRAVITY;

IsJump = true;

}

///////////////////////////////

//以下部分為操作速度的實作過程

///////////////////////////////

if(UpSpeed<0 && map->IfCollision(wx+fixX,wy+height+fixY,width,abs(UpSpeed),false,NoIgnore\_eat)){

//落地動作

while(!map->IfCollision(wx+fixX,wy+height+i+fixY,width,1,false,NoIgnore\_eat))i++;/\* 碰撞的修正 \*/

wy += i;

UpSpeed = 0;

IsJump = false;

IsLand=true;

}

else if(UpSpeed>0 && map->IfCollision(wx+fixX,wy+fixY-abs(UpSpeed),width,abs(UpSpeed),NoIgnore\_2)){

/\*向上跳過程遇到障礙物\*/

i=0;while(!map->IfCollision(wx+fixX,wy-i+fixY-1,width,1,NoIgnore\_2))i++;// 碰撞的修正

wy -= i;

UpSpeed = 0;

}

else{

//跳躍過程

wy -= UpSpeed;

}

if(map->IfCollision(wx+fixX,wy+height+fixY,width,1,false,NoIgnore\_eat/\*ONEOBJY\*/)){

IsLand=true;

IsJump = false;

}

//修正落下時左右兩邊應該空多少不判斷 回復原本設定

wx -= LR\_Space;

width += 2\*LR\_Space;

return IsLand;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//BasicInstructions實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

bool BasicInstructions::canMoveMapX = true;

bool BasicInstructions::canMoveMapY = true;

BasicInstructions::BasicInstructions()

{

UpSpeed=0;

time\_jump=0;

IsJump = false;

IsRun = false;

IsLessCollision = false;

ani\_jump\_count=TIMEFOR\_ANI\_JUMP+1;

Reduce\_UP\_VELOCITY = 0;

}

bool BasicInstructions::MoveLeft(MapManage\* map){

int i,limitX = (MWIDTH + width)/2,temp,testTemp;

int UnVisibleStone = OrderSize+MonsterOrderSize +0;

TriggerObj(3);

if(ToolCDC::ReturnStage()!=0)return false; //防止已經存圖又移動

//抓人

if(Instance()->Partner!=NULL&&CollisionChipDale(4,2,0)&&!Instance()->ReturnIsTaken()&&Instance()->A\_FLAG

&&!Instance()->Partner->ReturnIsTaken()&&!Instance()->Partner->ReturnInvincible()){

Object\*\* CanThrow=ChipDale::CanThrow;

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL){

//if(Partner->Alive)break; //註解掉此行會有殘影效果

CanThrow[i]=new ChipDale\_Taken(Instance());

Instance()->NowTakeObj=CanThrow[i];

Instance()->NowTaken=ChipDale\_taken;

Instance()->A\_FLAG = false;

break;

}

}

}

//拿東西

if(A\_FLAG && !SetNowTaken(-1)){

temp = map->IfCollision(wx-SPEED-ONEOBJX,wy,ONEOBJX,height,true,false);

testTemp = map->IfCollision(wx-SPEED-ONEOBJX,wy,ONEOBJX,height,true,false,true);

if(testTemp/E8==5 || temp/E8==5 && testTemp/E8!=5 && testTemp/E8!=0) temp = testTemp;

if(testTemp/E8==UnVisibleStone || temp/E8==UnVisibleStone && testTemp/E8!=UnVisibleStone && testTemp/E8!=0) temp = testTemp;

if(temp!=1&&temp!=2&&temp!=0&&temp/E8!=4){

map->ClearObstacle(temp);

SetNowTaken(temp);

A\_FLAG = false;

}

}

temp = CollisionChipDale(4,SPEED,0);

//移動

if(!map->IfCollision(wx-SPEED,wy,SPEED,height,true) && !temp){

SetWxWy(-SPEED,0,true);

if(canMoveMapX && wx < limitX)

map->MoveMap(4);

return true;

}

else if(temp) {

//FIRST 腳色完全貼近

i=SPEED-1;while(i>=0 && CollisionChipDale(4,i,0))i--;

SetWxWy(-i,0,true);

//移動剩餘額度 SPEED-i

//貼近之後推擠對方

if(!getPartner()->ReturnHideComplete()&&!IfCollision(4,1))

CollisionChipDale(4,1);

//帶動螢幕

if(canMoveMapX && wx < limitX)

map->MoveMap(4,1);

return true;

}

else {

//撞到地圖障礙物修正

i=0;while(!map->IfCollision(wx-i,wy,width,height,true))i++;

wx -=(i!=0)\*(i-1)+(i==0)\*(-SPEED);

}

return false;

}

bool BasicInstructions::MoveRight(MapManage\* map){

int i,limitX = (MWIDTH + width)/2,temp,testTemp;

int UnVisibleStone = OrderSize+MonsterOrderSize +0;

TriggerObj(4);

if(ToolCDC::ReturnStage()!=0)return false; //防止已經存圖又移動

//抓人

if(Instance()->Partner!=NULL&&CollisionChipDale(8,2,0)&&!Instance()->ReturnIsTaken()&&Instance()->A\_FLAG

&&!Instance()->Partner->ReturnIsTaken()&&!Instance()->Partner->ReturnInvincible()){

Object\*\* CanThrow=ChipDale::CanThrow;

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL){

CanThrow[i]=new ChipDale\_Taken(Instance());

Instance()->NowTakeObj=CanThrow[i];

Instance()->NowTaken=ChipDale\_taken;

Instance()->A\_FLAG = false;

break;

}

}

}

//拿東西

if(A\_FLAG && !SetNowTaken(-1)){

temp = map->IfCollision(wx+width+SPEED,wy,ONEOBJX,height,true,false);

testTemp = map->IfCollision(wx+width+SPEED,wy,ONEOBJX,height,true,false,true);

if(testTemp/E8==5 || temp/E8==5 && testTemp/E8!=5 && testTemp/E8!=0) temp = testTemp;

if(testTemp/E8== UnVisibleStone|| temp/E8==UnVisibleStone && testTemp/E8!=UnVisibleStone && testTemp/E8!=0) temp = testTemp;

if(temp!=1&&temp!=2&&temp!=0&&temp/E8!=4){//箱子...

map->ClearObstacle(temp);

SetNowTaken(temp);

A\_FLAG = false;

}

}

//移動

temp = CollisionChipDale(8,SPEED,0);

if(!map->IfCollision(wx+width,wy,SPEED,height,true) && !temp){

SetWxWy(SPEED,0,true);

if(canMoveMapX && wx > limitX)

map->MoveMap(8);

return true;

}

else if(temp){

//FIRST 腳色完全貼近

i=SPEED-1;while(i>=0 && CollisionChipDale(8,i,0))i--;

SetWxWy(i,0,true);

//移動剩餘額度 SPEED-i

//貼近之後推擠對方

if(!getPartner()->ReturnHideComplete()&&!IfCollision(8,1))

CollisionChipDale(8,1);

//帶動螢幕

if(canMoveMapX && wx > limitX)

map->MoveMap(8,1);

return true;

}

else{

i=0;while(!map->IfCollision(wx+i,wy,width,height,true))i++;

wx +=(i!=0)\*(i-1)+(i==0)\*(-SPEED);

}

return false;

}

bool BasicInstructions::Jump()

{

if(ToolCDC::ReturnStage()!=0)return false; //防止已經存圖又移動

if(time\_jump < TIMEFORJUMP && !IsLessCollision){

if(B\_FLAG){

UpSpeed = UP\_VELOCITY - (time\_jump-1)\*GRAVITY -Reduce\_UP\_VELOCITY;

time\_jump++;

}

else{

time\_jump = TIMEFORJUMP;

UpSpeed = UP\_VELOCITY - (TIMEFORJUMP-1)\*GRAVITY -Reduce\_UP\_VELOCITY;

}

return true;

}

return false;

}

int BasicInstructions::FallingDown(MapManage\* map)

{

int i=0,limitY = (MHEIGHT )/2- height;

int If\_Reset\_State=false,tempFixY=0,tempFixX=0;

if(ToolCDC::ReturnStage()!=0)return If\_Reset\_State; //防止已經存圖又移動

//暫時性修正　讓遊戲在未修正前也能判斷正常

if(map->MoveMap(0)&1)tempFixY = Jump\_Fix;

if(map->MoveMap(0)&4)tempFixX = SPEED;

if(map->MoveMap(0)&8)tempFixX = -SPEED;

//落下過程的FLAG初始化 // 預設是不進行跳躍

if(!IsLessCollision && !map->IfCollision(wx+tempFixX,wy+height+tempFixY,width,1/\*ONEOBJY\*/)

&&!CollisionChipDale(2,1,0)){

UpSpeed -= GRAVITY;

IsJump = true;

If\_Reset\_State = true;

}

//如果下方為藍色地板以外的物件則不能下跳

if(IsLessCollision && map->IfCollision(wx+tempFixX,wy+height+tempFixY,width,1/\*ONEOBJY\*/,true )){

IsLessCollision= false;

IsJump=false;

B\_FLAG = false;

UpSpeed=0;

}

//等到MoveRL修好了再啟用

if(CollisionChipDale(2,0,0)){

i=0;

while(CollisionChipDale(2,1,0)){

wy--;

i++;/\* 碰撞的修正 \*/

}

}

///////////////////////////////

//以下部分為操作速度的實作過程

///////////////////////////////

//吃花

if(UpSpeed<=0)

TriggerObj(1);

else

TriggerObj(2);

if(IsLessCollision){

//下跳穿越地板

UpSpeed -= GRAVITY;

wy -= UpSpeed;

if(!map->IfCollision(wx+tempFixX,wy+height+tempFixY,width,1))

IsLessCollision= false;

IsJump = true;

If\_Reset\_State = true;

}

else if(UpSpeed<0 && map->IfCollision(wx+tempFixX,wy+height+tempFixY,width,abs(UpSpeed))){

//落地動作

while(!map->IfCollision(wx+tempFixX,wy+height+i+tempFixY,width,1))i++;/\* 碰撞的修正 \*/

wy += i;

UpSpeed = 0;

time\_jump = 0;

If\_Reset\_State = true;

IsJump = false;

B\_FLAG = false;

ani\_jump\_count=0;

}

else if(UpSpeed<0 && CollisionChipDale(2,UpSpeed,0)){

i=0;

while(CollisionChipDale(2,i,0)!=1){

i++;/\* 碰撞的修正 \*/

}

wy += i-1;

UpSpeed = 0;

time\_jump = 0;

If\_Reset\_State = true;

IsJump = false;

B\_FLAG = false;

ani\_jump\_count=0;

}

else if(UpSpeed>0 && map->IfCollision(wx+tempFixX,wy+tempFixY-abs(UpSpeed),width,abs(UpSpeed),true)){

/\*向上跳過程遇到障礙物\*/

i=0;while(!map->IfCollision(wx+tempFixX,wy-i+tempFixY-1,width,1,true))i++;// 碰撞的修正

wy -= i;

UpSpeed = 0;

time\_jump = TIMEFORJUMP;

}

else{//跳躍過程,持續呼叫

//2P碰撞

if(UpSpeed>0)CollisionChipDale(1,UpSpeed,1);

wy -= UpSpeed;

}

/\*跳躍帶動螢幕\*/

if(canMoveMapY && wy < limitY && UpSpeed>0 && !(map->MoveMap(0)&1)){

map->MoveMap(1,SPEED,UpSpeed/SPEED\*SPEED);

Jump\_Fix =UpSpeed/SPEED\*SPEED;

}

return If\_Reset\_State;

}

**ChipDale.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ChipDale實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Object \*\*ChipDale::CanThrow;

MapManage \*ChipDale::Maps;

Monster \*\*ChipDale::AllMonster;

CAnimation ChipDale::animation[2][2][ACTION\_NUM][2]; // 動作~方向

CAnimation ChipDale::ani\_sweat[2];

CAnimation ChipDale::ani\_dizzy;

CAnimation ChipDale::ani\_god[2];

ChipDale::ChipDale(int isDale){

IsDale = isDale;

Life=3;

Reset(10,10);//必須先有Life才能reset

IsGod=false;

ResetScore();

}

void ChipDale::Reset(int Wx,int Wy,bool LR,bool FullHealth){

if(Life<=0)return;

wx=Wx;

wy=Wy;

LRflag=LR;

if(FullHealth)Health=3;

LOCK=false;

time\_jump = 0;

NowAct=LastAct=0;

NowTaken=0;

freeze=0;

Reduce\_UP\_VELOCITY = 0;

Invincible = IsFaint=0;

Hurt=false;

NowTakeObj=NULL;

ani\_sweat\_count=0;

ani\_Hide\_freeze\_jump=false;

setFlag(false,true,true,true,true,true,true);

Alive=true;

UpSpeed=0;

time\_jump=0;

IsJump = false;

IsRun = false;

IsLessCollision = false;

ani\_jump\_count=TIMEFOR\_ANI\_JUMP+1;

Reduce\_UP\_VELOCITY = 0;

};

void ChipDale::Loading(){

char name[6]="chip/";

for(int i=0;i<2;i++){

animation[i][0][0][0].AddBitmap("Bitmaps/action/",name,"stand\_L.bmp",PURPLE);

animation[i][0][0][1].AddBitmap("Bitmaps/action/",name,"stand\_R.bmp",PURPLE);

animation[i][0][1][0].AddBitmap("Bitmaps/action/",name,"run\_1L.bmp",PURPLE);

animation[i][0][1][0].AddBitmap("Bitmaps/action/",name,"run\_2L.bmp",PURPLE);

animation[i][0][1][0].AddBitmap("Bitmaps/action/",name,"run\_3L.bmp",PURPLE);

animation[i][0][1][0].AddBitmap("Bitmaps/action/",name,"run\_4L.bmp",PURPLE);

animation[i][0][1][1].AddBitmap("Bitmaps/action/",name,"run\_1R.bmp",PURPLE);

animation[i][0][1][1].AddBitmap("Bitmaps/action/",name,"run\_2R.bmp",PURPLE);

animation[i][0][1][1].AddBitmap("Bitmaps/action/",name,"run\_3R.bmp",PURPLE);

animation[i][0][1][1].AddBitmap("Bitmaps/action/",name,"run\_4R.bmp",PURPLE);

animation[i][0][2][0].AddBitmap("Bitmaps/action/",name,"jump\_1L.bmp",PURPLE);

animation[i][0][2][1].AddBitmap("Bitmaps/action/",name,"jump\_1R.bmp",PURPLE);

animation[i][0][3][0].AddBitmap("Bitmaps/action/",name,"throw\_L.bmp",PURPLE);

animation[i][0][3][1].AddBitmap("Bitmaps/action/",name,"throw\_R.bmp",PURPLE);

animation[i][0][4][0].AddBitmap("Bitmaps/action/",name,"Squat\_1L.bmp",PURPLE);

animation[i][0][4][1].AddBitmap("Bitmaps/action/",name,"Squat\_1R.bmp",PURPLE);

animation[i][0][5][0].AddBitmap("Bitmaps/action/",name,"Hurt\_L.bmp",PURPLE);

animation[i][0][5][1].AddBitmap("Bitmaps/action/",name,"Hurt\_R.bmp",PURPLE);

animation[i][0][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_1L.bmp",PURPLE);

animation[i][0][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_2L.bmp",PURPLE);

animation[i][0][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_3L.bmp",PURPLE);

animation[i][0][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_4L.bmp",PURPLE);

animation[i][0][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_1R.bmp",PURPLE);

animation[i][0][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_2R.bmp",PURPLE);

animation[i][0][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_3R.bmp",PURPLE);

animation[i][0][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_4R.bmp",PURPLE);

animation[i][1][0][0].AddBitmap("Bitmaps/action/",name,"Take\_L.bmp",PURPLE);

animation[i][1][0][1].AddBitmap("Bitmaps/action/",name,"Take\_R.bmp",PURPLE);

animation[i][1][1][0].AddBitmap("Bitmaps/action/",name,"Take\_Run\_1L.bmp",PURPLE);

animation[i][1][1][0].AddBitmap("Bitmaps/action/",name,"Take\_Run\_2L.bmp",PURPLE);

animation[i][1][1][1].AddBitmap("Bitmaps/action/",name,"Take\_Run\_1R.bmp",PURPLE);

animation[i][1][1][1].AddBitmap("Bitmaps/action/",name,"Take\_Run\_2R.bmp",PURPLE);

animation[i][1][2][0].AddBitmap("Bitmaps/action/",name,"Take\_Jump\_L.bmp",PURPLE);

animation[i][1][2][1].AddBitmap("Bitmaps/action/",name,"Take\_Jump\_R.bmp",PURPLE);

animation[i][1][3][0].AddBitmap("Bitmaps/action/",name,"throw\_L.bmp",PURPLE);

animation[i][1][3][1].AddBitmap("Bitmaps/action/",name,"throw\_R.bmp",PURPLE);

animation[i][1][4][0].SetDelayCount(2);

animation[i][1][4][0].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_1L.bmp",PURPLE);

animation[i][1][4][0].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_2L.bmp",PURPLE);

animation[i][1][4][0].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_3L.bmp",PURPLE);

animation[i][1][4][0].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_4L.bmp",PURPLE);

animation[i][1][4][0].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_5L.bmp",PURPLE);

animation[i][1][4][0].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_6L.bmp",PURPLE);

animation[i][1][4][1].SetDelayCount(2);

animation[i][1][4][1].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_1R.bmp",PURPLE);

animation[i][1][4][1].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_2R.bmp",PURPLE);

animation[i][1][4][1].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_3R.bmp",PURPLE);

animation[i][1][4][1].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_4R.bmp",PURPLE);

animation[i][1][4][1].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_5R.bmp",PURPLE);

animation[i][1][4][1].AddBitmap("Bitmaps/action/",name,"Hide\_crate\_6R.bmp",PURPLE);

animation[i][1][5][0].AddBitmap("Bitmaps/action/",name,"Hurt\_L.bmp",PURPLE);

animation[i][1][5][1].AddBitmap("Bitmaps/action/",name,"Hurt\_R.bmp",PURPLE);

animation[i][1][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_1L.bmp",PURPLE);

animation[i][1][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_2L.bmp",PURPLE);

animation[i][1][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_3L.bmp",PURPLE);

animation[i][1][6][0].AddBitmap("Bitmaps/action/",name,"Faint\_4L.bmp",PURPLE);

animation[i][1][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_1R.bmp",PURPLE);

animation[i][1][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_2R.bmp",PURPLE);

animation[i][1][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_3R.bmp",PURPLE);

animation[i][1][6][1].AddBitmap("Bitmaps/action/",name,"Faint\_4R.bmp",PURPLE);

sprintf(name,"%s","dale/");

}

ani\_sweat[0].AddBitmap("Bitmaps/action/ANI\_ChipDale/sweat\_L.bmp",PURPLE);

ani\_sweat[1].AddBitmap("Bitmaps/action/ANI\_ChipDale/sweat\_R.bmp",PURPLE);

ani\_dizzy.SetDelayCount(7);

ani\_dizzy.AddBitmap("Bitmaps/action/ANI\_ChipDale/Dizzy1.bmp",PURPLE);

ani\_dizzy.AddBitmap("Bitmaps/action/ANI\_ChipDale/Dizzy2.bmp",PURPLE);

ani\_dizzy.AddBitmap("Bitmaps/action/ANI\_ChipDale/Dizzy3.bmp",PURPLE);

ani\_dizzy.AddBitmap("Bitmaps/action/ANI\_ChipDale/Dizzy4.bmp",PURPLE);

ani\_god[0].AddBitmap("Bitmaps/action/ANI\_ChipDale/god\_L.bmp",PURPLE);

ani\_god[1].AddBitmap("Bitmaps/action/ANI\_ChipDale/god\_R.bmp",PURPLE);

}

void ChipDale::OnShow()

{

int Last\_ani\_height,fix\_hieght;

if(!Alive&&!IsGod)return;

//特定動畫重設循環

if(NowAct!=4){

if(animation[IsDale][1][4][0].GetCurrentBitmapNumber()>=4){

animation[IsDale][1][4][0].SetDelayCount(2);/\*SET完記得RESET，不然第一張圖會用到舊的DELAY\_COUNTER\*/

animation[IsDale][1][4][0].Reset();

}

if(animation[IsDale][1][4][1].GetCurrentBitmapNumber()>=4){

animation[IsDale][1][4][1].SetDelayCount(2);

animation[IsDale][1][4][1].Reset();

}

}

if(LastAct==2){

animation[IsDale][0][2][0].Reset();

animation[IsDale][0][2][1].Reset();

}

if(ani\_Hide\_freeze\_jump==0)ani\_Hide\_freeze\_jump=1;

//動畫切9=換 及 重設height wxwy

if(!ReturnHideComplete()

Last\_ani\_height=height;

if(NowAct!=2)

animation[IsDale][TAKEN\_FLAG][NowAct][LRflag].OnMove();

if(Last\_ani\_height!=animation[IsDale][TAKEN\_FLAG][NowAct][LRflag].Height()){

height=animation[IsDale][TAKEN\_FLAG][NowAct][LRflag].Height();

fix\_hieght=Last\_ani\_height-height;

SetWxWy(wx,wy+fix\_hieght);

}

}else {

ani\_Hide\_freeze\_jump=0;

if(TAKEN\_FLAG&&NowAct==4){

if(!animation[IsDale][1][4][LRflag].IsFinalBitmap()){

animation[IsDale][1][4][LRflag].SetDelayCount(5);

animation[IsDale][1][4][LRflag].OnMove();

}else{

animation[IsDale][1][4][LRflag].SetDelayCount(100);

animation[IsDale][1][4][LRflag].OnMoveToNum(4);

}

}

}

//顯示人物

//人物無敵效果

if((Invincible && (Invincible %5==2||Invincible %5==3)) || !Invincible || (NowAct==4&&TAKEN\_FLAG) ){

if(IsGod){

IsJump=false;

if(LRflag)

ani\_god[LRflag].SetBottomLeft(wx+(-ani\_god[LRflag].Width()+15),wy,ani\_god[LRflag].Height());

else

ani\_god[LRflag].SetBottomLeft(wx+width-20,wy,ani\_god[LRflag].Height());

ani\_god[LRflag].OnShow();

}

if(ani\_jump\_count<TIMEFOR\_ANI\_JUMP){

//跳轉蹲動畫

Last\_ani\_height=height;

height=animation[IsDale][0][4][LRflag].Height();

fix\_hieght=Last\_ani\_height-height;

SetWxWy(wx,wy+fix\_hieght);

animation[IsDale][0][4][LRflag].SetBottomLeft(wx,wy,height);

animation[IsDale][0][4][LRflag].OnShow();

}else{

//一般狀態動畫顯示

animation[IsDale][TAKEN\_FLAG][NowAct][LRflag].SetBottomLeft(wx,wy,height);

animation[IsDale][TAKEN\_FLAG][NowAct][LRflag].OnShow();

}

//流汗動畫顯示

if(ani\_sweat\_count>0){

int tempx=ani\_sweat\_x+10-Maps->ReturnNowX();/\*-((Maps->MoveMap(0)&8)>0)\*SPEED\*ani\_sweat\_count\*/;

if(!LRflag)tempx=ani\_sweat\_x+width-20-Maps->ReturnNowX();/\*+((Maps->MoveMap(0)&4)>0)\*SPEED\*ani\_sweat\_count\*/;

ani\_sweat[LRflag].SetBottomLeft(tempx,ani\_sweat\_wy+ani\_sweat\_count\*7/\*汗的位移\*/,ani\_sweat[LRflag].Height());

ani\_sweat[LRflag].OnShow();

}

}

//暈眩顯示

if(NowAct==6){

ani\_dizzy.SetBottomLeft(wx,wy-ani\_dizzy.Height(),ani\_dizzy.Height());

ani\_dizzy.OnShow();

ani\_dizzy.OnMove();

}

}

void ChipDale::Dead(){

Object \*\*CanThrow = ChipDale::CanThrow;

Life--;

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]!=NULL)continue;

CanThrow[i] = new ChipDale\_Dead(this);

break;

}

ReleaseNowTakeObj();

Alive=false;

Health=0;

}

void ChipDale::GetHurt(){

if(!Alive||IsGod)return;

Hurt=true;

if(UpSpeed>=0)UpSpeed = 40;

else UpSpeed+=40;

SetState();

Health--;

if(Health<=0){

Dead();

}

}

void ChipDale::Faint(){

if(!Alive)return;

IsFaint=1;

SetState();

}

int ChipDale::SetNowTaken(int Value){

if(!Alive)return 0;

if(Value==-1)

return NowTaken!=0;

else{

NowTaken = Value/E8;

if(NowTaken == OrderSize + MonsterOrderSize + 0){

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL)continue;

if(CanThrow[i]->ReturnObjValue() == Value){

NowTakeObj = CanThrow[i];

NowTakeObj->RecoverObj(this);

break;

}

if(i+1==CanTakeNum){TRACE("SetNowTaken: No Find In CanThrow\n");Sleep(100000);}

}

}

else{

switch(NowTaken){

case 14:

NowTaken = 3;

Maps->FillObstacle(7,(Value%E4-1)\*ONEOBJX-Maps->ReturnNowX(),(Value%E8/E4-1)\*ONEOBJY-Maps->ReturnNowY());

break;

case 15:

NowTaken = 3;

Maps->FillObstacle(8,(Value%E4-1)\*ONEOBJX-Maps->ReturnNowX(),(Value%E8/E4-1)\*ONEOBJY-Maps->ReturnNowY());

break;

case 16:

NowTaken = 3;

Maps->FillObstacle(4,(Value%E4-1)\*ONEOBJX-Maps->ReturnNowX(),(Value%E8/E4-1)\*ONEOBJY-Maps->ReturnNowY());

break;

}

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL){

switch(NowTaken){

case 3:

CanThrow[i] = new Craft(this);

break;

case 5:

CanThrow[i] = new Stone(this);

break;

case 6:

CanThrow[i] = new Apple(this);

break;

case BALL:

CanThrow[i] = new Ball(this);

break;

}

NowTakeObj = CanThrow[i];

break;

}

if(i+1==CanTakeNum){TRACE("SetNowTaken: No Find In CanThrow\n");Sleep(100000);}

}

}

return Value;

}

}

void ChipDale::FixSelf\_Onto\_ObjectTop(){

int i=0;

if(!Alive)return;

while(Maps->IfCollision(wx,wy+i,width,height,true))i--;

wy+=i;

if(i<0)TRACE("fix\n");

}

void ChipDale::SetState(){

int LastHeight=height;

LastAct = NowAct;

//設定LRflag

if(!(LastAct==4)||(Last\_flag[1]&&!DOWN\_FLAG)){

//蹲下不可以變換方向

if(LEFT\_FLAG&&!RIGHT\_FLAG) LRflag=0;

if(!LEFT\_FLAG&&RIGHT\_FLAG) LRflag=1;

}

//state切換

if(LEFT\_FLAG==RIGHT\_FLAG){ NowAct=0;IsRun=false;} //左右同時按 或都不按 Act=站

else{NowAct=1;if(!DOWN\_FLAG||(NowTaken==6))IsRun=true;}

if(IsJump) NowAct=2;

if(DOWN\_FLAG&&NowAct!=2){

if(NowTaken!=APPLE&&NowTaken!=ChipDale\_taken){

NowAct=4;

IsRun=0;

}

}

if(A\_FLAG && (TAKEN\_FLAG)) NowAct=3;

if(Hurt)NowAct=5;

if(IsFaint)NowAct=6;

//重設wx wy

if((NowAct==4/\*||LastAct==4\*/)&&TAKEN\_FLAG){

SetWxWy(wx,wy-34+height);

}

else

SetWxWy(wx,wy-animation[IsDale][TAKEN\_FLAG][NowAct][LRflag].Height()+height);

//重設height

if(NowAct==4&&TAKEN\_FLAG&&animation[IsDale][0][4][LRflag].GetCurrentBitmapNumber()==0)height=34;

else SetWidthHeight(45,animation[IsDale][TAKEN\_FLAG][NowAct][LRflag].Height());

//蹲下放開需要調整2P位置

if(LastAct==4&&Partner!=NULL){

if(CollisionChipDale(0,0,0))

Partner->SetWxWy(0,LastHeight-height,true);

}

}

void ChipDale::setFlag(bool value,bool up,bool down,bool left,bool right,bool A,bool B){

if(LOCK) return; //!Alive必須能控制 ChipDale\_

if(IsFaint&&value) return;

if(up) UP\_FLAG = value;

if(down) DOWN\_FLAG = value;

if(left) LEFT\_FLAG = value;

if(right) RIGHT\_FLAG = value;

if(A) A\_FLAG = value;

if(B){

if(!B\_FLAG && time\_jump==0 && value && !DOWN\_FLAG){

B\_FLAG = true;

IsJump = true;

}

if(!value)

B\_FLAG = false;

if(DOWN\_FLAG && value && !IsJump ){

if((TAKEN\_FLAG&&(!ani\_Hide\_freeze\_jump||NowTaken==APPLE||NowTaken==ChipDale\_taken))||!TAKEN\_FLAG)//HIDE動畫冷卻時間

IsLessCollision = true;

}

}

SetState();

//設定LAST\_FLAG

for(int i=0;i<6;i++){

Last\_flag[i]=flag[i];

}

};

void ChipDale::GodMode(){

//GodMode不支援2P

if(CGameStateRun::multiplayer!=2)

IsGod=!IsGod;

}

void ChipDale::GodMove(){

int limitX = (MWIDTH + width)/2;

int limitY = (MHEIGHT + height)/2;

if(LEFT\_FLAG&&wx-SPEED\*2>=0){

SetWxWy(-SPEED\*2,0,true);

if(wx < limitX)Maps->MoveMap(4,20);

}else if(RIGHT\_FLAG&&wx+width+SPEED\*2<=MWIDTH){

SetWxWy(SPEED\*2,0,true);

if(wx > limitX)Maps->MoveMap(8,20);

}

if(UP\_FLAG&&wy-SPEED\*2>=0){

SetWxWy(0,-SPEED\*2,true);

if(wy < limitY){

Maps->MoveMap(1,10,20);

}

}else if(DOWN\_FLAG&&wy+height+SPEED\*2<=MHEIGHT){

SetWxWy(0,SPEED\*2,true);

if(wy > limitY){

Maps->MoveMap(2,10,20);

}

}

}

void ChipDale::OnMove(){

int temp,tDerict,LRMargin=MapManage::LRMargin;

if(!Alive)return;

if(wy>MHEIGHT+50){Dead();return;}

if(freeze){

HOLD(3,freeze){freeze=0;}; //HOLD值 2~3為佳 (1為沒效果)

}

TriggerObj(0);

//運動

if(!IsGod){

if((!freeze&&IsRun&&!LRflag&&!Hurt||Hurt&&LRflag))MoveLeft(Maps);

if(!freeze&&IsRun&& LRflag&&!Hurt||Hurt&&!LRflag)MoveRight(Maps);

if(IsJump)Jump();

if(FallingDown(Maps)){SetState();}

}else{

GodMove();

}

if(NowTaken==6) Reduce\_UP\_VELOCITY=10;//增加重力加速度

//丟的冷卻時間

if(NowTaken < 0)NowTaken++;

//丟實作

if(NowAct==3 && NowTakeObj!=NULL){

tDerict = 1\*(UP\_FLAG)+4\*(!LRflag)+8\*(LRflag);

if(UP\_FLAG && !IsJump) UpSpeed += 20;//上丟會稍微往上跳

NowTakeObj->Throw(tDerict);

NowTakeObj=NULL;

if(NowTaken==6)Reduce\_UP\_VELOCITY=0;//改成正常重力加速度

NowTaken = -6;

}

//丟得僵直時間

if(NowTaken==-3){setFlag(0,0,0,0,0,1,0);freeze=1;} //決定丟動畫的延遲時間 NowTaken判斷可為-1 ~-5

//跳轉蹲動畫切換變數設定

if(ani\_jump\_count<TIMEFOR\_ANI\_JUMP){

if(ani\_jump\_count < 1){

//clock\_t t=clock();

CAudio::Instance()->Play(AUDIO\_JUMP);

//TRACE("CAudio %d\n",clock()-t);

}

if(ani\_jump\_count==2)SetState();

//TRACE("%d wy=%d height=%d \n",ani\_jump\_count,wy,height);

ani\_jump\_count++;

}

//受傷及無敵的變數設定

const int Hurt\_time=10;

const int Hurt\_Shine\_time=50;

const int invincibility\_time=240;

const int Hurt\_start=invincibility\_time-Hurt\_Shine\_time;

const int Hurt\_end=Hurt\_start+Hurt\_time;

if(Invincible)Invincible++;

if(Hurt&&Invincible<Hurt\_start)Invincible=Hurt\_start;

if(Invincible>invincibility\_time){

SetInvincible(0);

if(Partner==NULL||!Partner->ReturnInvincible())

CAudio::Instance()->SetSpeedByID(AUDIO\_1\_1,1000);

}

//受傷回復設定

if(Invincible==Hurt\_end){Hurt=0;SetState();};

//暈眩變數設定

if(IsFaint){

if(NowTakeObj!=NULL)ReleaseNowTakeObj();

//石頭暈眩修正

if(IsFaint<7){//暈眩圖片寬度不一，此條件防止連續修正

int i=0,j=0;

while(Maps->IfCollision(wx,wy+i,width,height,true))i--;//先修正WY 如修正超過一個物件高則修正WX

if(-i>34){

i=0;

while(Maps->IfCollision(wx-LRflag\*j+!LRflag\*j,wy,width,height,true))j++;//根據人物面對方向往反方向修正

if(j>45){

j=0;

//反方向修正超過一個物件寬則朝相同方向修正

while(Maps->IfCollision(wx-LRflag\*j+!LRflag\*j,wy,width,height,true))j--;

if(-j>45)j=0;//三種方向修正都超過一個物件長寬則不修正(卡在石頭裡)

}

wx-=(LRflag\*j-!LRflag\*j);

}

wy+=i;

}

IsFaint++;

if(IsFaint>50){IsFaint=0;SetState();}

setFlag(0,1,1,1,1,1,1);

}

//流汗動畫顯示變數設定

if(NowTaken==APPLE)ani\_sweat\_count++;

else ani\_sweat\_count=0;

if(ani\_sweat\_count>5/\*週期\*/)ani\_sweat\_count=1; //更改週期亦需改變每顆汗流下的位移(onshow)

if(ani\_sweat\_count==1){

ani\_sweat\_x=wx+Maps->ReturnNowX();

ani\_sweat\_wy=wy;

}

//2P移動鎖螢幕

canMoveMapX=true;

canMoveMapY=true;

temp = Maps->GetRoute();

if((temp&8) && wx<=LRMargin || (temp&4) && (wx>=MWIDTH-width-LRMargin))

canMoveMapX = false;

if((temp&2) && wy<=Jump\_Fix || (temp&1) && wy>=(MHEIGHT-2\*height-Jump\_Fix))

canMoveMapY = false;

}

void ChipDale::InitialWidthHeight(){

width = animation[IsDale][0][0][0].Width();

height = animation[IsDale][0][0][0].Height();

}

int ChipDale::CollisionChipDale(int Direct,int passSpeed,int mode){

int temp=0,ObjFix=0;

const int LRspace=5;

if(Partner == NULL||!Partner->Alive)return 0;

passSpeed = abs(passSpeed);

if(Partner->ReturnNowTakeObj()!=NULL && !Partner->ReturnHideComplete()){

ObjFix = Partner->ReturnNowTakeObj()->ReturnHeight();

wy += ObjFix;

}

if(Direct==0) temp = Partner->IfCollision(wx+LRspace,wy,width-2\*LRspace,height);

else if(Direct==1) temp = Partner->IfCollision(wx+LRspace,wy-passSpeed,width-2\*LRspace,passSpeed);

else if(Direct==2) temp = Partner->IfCollision(wx+LRspace,wy+height,width-2\*LRspace,passSpeed);

else if(Direct==4) temp = Partner->IfCollision(wx+LRspace-passSpeed,wy,passSpeed,height);

else if(Direct==8) temp = Partner->IfCollision(wx-LRspace+width,wy,passSpeed,height);

if(ObjFix!=0)

wy -= ObjFix;

if(temp>0){

if(!Partner->IfCollision(Direct,passSpeed)){

if(mode){

if(Direct==1) Partner->SetWxWy(0,-passSpeed,true);

else if(Direct==4) Partner->SetWxWy(-passSpeed,0,true);

else if(Direct==8) Partner->SetWxWy(passSpeed,0,true);

}

return 1;//發生碰撞，推擠2P

}

return 2;//發生碰撞，不能推擠2P

}

return 0;//無發生碰撞

}

void ChipDale::TriggerObj(int Derict){

int temp=0,fixX=0,fixY=0,ox,oy;

if(!Alive)return;

if(Maps->MoveMap(0)&1)fixY = Jump\_Fix;

if(Maps->MoveMap(0)&4)fixX = SPEED;

if(Maps->MoveMap(0)&8)fixX = -SPEED;

while(true){

if(Derict==0) temp = Maps->IfCollision(wx+fixX,wy+fixY,width,height,true,true);

else if(Derict==1) temp = Maps->IfCollision(wx+fixX,wy+fixY+height,width,abs(UpSpeed),false,true,true);

else if(Derict==2) temp = Maps->IfCollision(wx+fixX,wy+fixY-UpSpeed,width,abs(UpSpeed),true,true);

else if(Derict==4) temp = Maps->IfCollision(wx+width+fixX,wy+fixY,ONEOBJX,height,true,true);

else if(Derict==3) temp = Maps->IfCollision(wx-ONEOBJX+fixX,wy+fixY,ONEOBJX,height,true,true);

switch(temp/E8){

case FLOWER:

Score\_Flower++;

Maps->ClearObstacle(temp);

break;

case STAR:

Score\_Star++;

Maps->ClearObstacle(temp);

break;

case 8:

Health=3;

Maps->ClearObstacle(temp);

break;

case 9:

CAudio::Instance()->SetSpeedByID(AUDIO\_1\_1,1500);

Maps->ClearObstacle(temp);

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL){

CanThrow[i] = new Explosion(temp%E4-1,temp%E8/E4-1,9,this);

break;

}

if(i+1==CanTakeNum){TRACE("SetNowTaken: No Find In CanThrow\n");Sleep(100000);}

}

break;

case 10://Cheese Box

Maps->ClearObstacle(temp);

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL){

CanThrow[i] = new Explosion(temp%E4-1,temp%E8/E4-1,CHEESE,this);

break;

}

if(i+1==CanTakeNum){TRACE("SetNowTaken: No Find In CanThrow\n");Sleep(100000);}

}

break;

case 11:

Maps->ClearObstacle(temp);

break;

case 12:

Maps->ClearObstacle(temp);

break;

case DOOR:

Lock();

if(Partner!=NULL)Partner->Lock();

ToolCDC::Fadeout();

ox = temp%E4-1;

oy = temp%E8/E4-1;

Maps->Teleport = Maps->GetRoute(ox\*ONEOBJX/MWIDTH,oy\*ONEOBJY/MHEIGHT)>>7;

Lock(0);

if(Partner!=NULL)Partner->Lock(0);

return;

case 17:

Maps->ClearObstacle(temp);

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL){

CanThrow[i] = new Explosion(temp%E4-1,temp%E8/E4-1,7,this);

CanThrow[i]->ReFixXY(Maps);

break;

}

if(i+1==CanTakeNum){TRACE("SetNowTaken: No Find In CanThrow\n");Sleep(100000);}

}

return;

default:

return ;

}

}

}

void ChipDale::ReleaseNowTakeObj(){

if(NowTakeObj==NULL||!Alive) return;

if(NowTaken==STONE || NowTaken==MonsterOrderSize+OrderSize+0)

NowTakeObj->Throw(0);

else

NowTakeObj->Throw(1);

NowTakeObj=NULL;

NowTaken=-6;

}

**object.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Object實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Object::Object()

{

NowAct = 1; //石頭拿起

ThisObjValue = 0;

wx = wy = 0;

CanAttackMode = 1;

Owener = NULL;

Direct = 0;

}

void Object::RecoverObj(ChipDale \*player){

Owener = player;//記住自己的主人

NowAct = 1;

ThisObjValue = 0;

}

void Object::CollisionMonster(Monster \*\*monster)

{

int temp;

if(NowAct==0 || Owener==NULL || CanAttackMode==2) return; //NowAct==0 準備投胎 , 沒有主人不行 , CanAttackMode==2 不攻擊任何人

if(NowAct==1 && Owener->ReturnHideComplete()){//NowAct==1 且主人蹲著 須要先修正wx wy (目前與主人關西)

wx = Owener->ReturnWX();

wy = Owener->ReturnWY()+Owener->ReturnHeight()-height;

}

else if(NowAct!=2)return; //NowAct==2 為運動狀態才能攻擊人 , 但如果NowAct==1 有例外 主人蹲著 要判斷碰撞

for(int i=0;i<ONE\_LEVEL\_MONSTER\_NUM && monster[i]!=NULL;i++){

if(monster[i]->ReturnNowAct()<=0) continue; //怪物運動狀態為 0 表示尚未遇到 , -1 已死亡

temp = monster[i]->IfCollision(wx,wy,width,height);//測試碰撞

switch(temp){

case 1:case 4://箱子的左上左下碰到怪物

if(monster[i]->KillMonster(4)){

CollisionReact(temp,monster[i]);

if(NowAct==1 && Owener->ReturnHideComplete()) Owener->ReleaseNowTakeObj();//打完怪物丟出此物品

NowAct=4;//表示已經丟出去打到怪物

i=ONE\_LEVEL\_MONSTER\_NUM;//跳出此迴圈 以防double kill

}

break;

case 2:case 3://箱子的右上右下碰到怪物

if(monster[i]->KillMonster(8)){

CollisionReact(temp,monster[i]);

if(NowAct==1 && Owener->ReturnHideComplete())Owener->ReleaseNowTakeObj();//打完怪物丟出此物品

NowAct=4;//表示已經丟出去打到怪物

i=ONE\_LEVEL\_MONSTER\_NUM;//跳出此迴圈 以防double kill

}

break;

case 5://箱子整個在怪物裡面 幾乎不可能發生

if(monster[i]->KillMonster(1)){

CollisionReact(temp,monster[i]);

if(NowAct==1 && Owener->ReturnHideComplete())Owener->ReleaseNowTakeObj();//打完怪物丟出此物品

NowAct=4;//表示已經丟出去打到怪物

i=ONE\_LEVEL\_MONSTER\_NUM;//跳出此迴圈 以防double kill

}

break;

}

}

}

void Object::CollisionChipDale(ChipDale \*player){

if(Owener==NULL)return;//沒有主人的物件不攻擊人

if(CanAttackMode==2 || CanAttackMode==3)return;//CanAttackMode = 2 表示此物件不具攻擊性) , 3 表示此物件只攻擊怪物

if(player->ReturnInvincible())return;//此玩家為無敵 不攻擊

if(NowAct==2||NowAct==4){//NowAct==2 表示此物件在運動的過程 且只有運動過程才會打傷人

if(Owener==player){

//TRACE("Self\n");

if(CanAttackMode==1)return;//CanAttackMode == 1 表示此物件不會攻擊自己的主人

if(CanAttackMode==0 && Owener->ReturnNowAct()>3 && Owener->ReturnNowAct()!=4

&& !Owener->ReturnHideComplete()&&!(Direct&1))return;

if((Direct&32)&&Owener->ReturnNowAct()==4)return;

//當CanAttackMode==0時 且箱子自己的主人的NowAct>=3 則不打它主人

}

int ThrowFix=(Owener==player)\*(4\*(LRflag)-4\*(!LRflag));

if(player->IfCollision(wx+ThrowFix,wy,width,height)){

player->Faint();

}

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ChipDale\_Dead實作 \*/

// \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap ChipDale\_Dead::frame\_pic[2][2];

ChipDale\_Dead::ChipDale\_Dead(ChipDale\* chip){

isDale =chip->IsDale;

LRflag =chip->LRflag;

wx =chip->wx;

wy =chip->wy;

Owener =chip;

CanAttackMode=2;

height=frame\_pic[0][0].Height();

NowAct=2; //運動

NoCollision = true;

UpSpeed = 0;

JumpSPEED = 30;

MoveSPEED = SPEED/2;

Wait\_A\_Minute=-1;

CAudio::Instance()->PauseLevelMusic(CGameStateRun::NowLevel);

CAudio::Instance()->Play(AUDIO\_DEAD);

}

ChipDale\_Dead::~ChipDale\_Dead(){

CAudio::Instance()->PlayLevelMusic(CGameStateRun::NowLevel);

}

void ChipDale\_Dead::Loading(){

frame\_pic[0][0].LoadBitmapA("Bitmaps/action/chip/Hurt\_L.bmp",PURPLE);

frame\_pic[0][1].LoadBitmapA("Bitmaps/action/chip/Hurt\_R.bmp",PURPLE);

frame\_pic[1][0].LoadBitmapA("Bitmaps/action/dale/Hurt\_L.bmp",PURPLE);

frame\_pic[1][1].LoadBitmapA("Bitmaps/action/dale/Hurt\_R.bmp",PURPLE);

}

void ChipDale\_Dead::OnMove(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

//死亡之等待時間

if(Wait\_A\_Minute>=0){

if(Wait\_A\_Minute>0)Wait\_A\_Minute--;

else {

NowAct=0;

CAudio::Instance()->PlayLevelMusic(CGameStateRun::NowLevel);

if((Owener->Partner==NULL||Owener->Partner->Life<=0)&&Owener->Life>0){

//回紀錄點

map->Set\_toRecord();

Owener->Reset(10,10);

}else if(Owener->Life<=0&&(Owener->Partner==NULL||Owener->Partner->Life<=0)){

//GameOver

CGameStateRun::TimeToGo=true;

}

}

return;

}

//Dead物件生命週期結束設定

if(wy+frame\_pic[isDale][LRflag].Height()>MHEIGHT+100 ){

Wait\_A\_Minute=33;

if(Owener->Life>0&&Owener->Partner!=NULL&&Owener->Partner->Life>0){

//復活

for(int i=0;i<CanTakeNum;i++){

if(ChipDale::CanThrow[i]!=NULL)continue;

ChipDale::CanThrow[i] = new ChipDale\_Resurrect(this);

break;

}

}else if((Owener->Partner==NULL||Owener->Partner->Life<=0)&&Owener->Life>0){

//回紀錄點

}else if(Owener->Life<=0&&(Owener->Partner==NULL||Owener->Partner->Life<=0))

{

//GameOver

}

}

if(!IsJump && wy+height<MHEIGHT){

IsJump = true;

Jump();

}

if(!LRflag&&wx+width+MoveSPEED<MWIDTH) MoveRight(map);

else if(LRflag&&wx-MoveSPEED>0) MoveLeft(map);

FallingDown(map);

}

void ChipDale\_Dead::OnShow(MapManage\* map){

frame\_pic[isDale][LRflag].SetTopLeft(wx,wy);

frame\_pic[isDale][LRflag].ShowBitmap();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ChipDale\_Resurrect實作 \*/

// \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap ChipDale\_Resurrect::frame\_pic[2][2];

ChipDale\_Resurrect::ChipDale\_Resurrect(ChipDale\_Dead\* chip){

isDale =chip->isDale;

LRflag =chip->LRflag;

wx =chip->wx;

wy =chip->wy;

Owener =chip->Owener;

timeCount=0;

CanAttackMode=2;

height=frame\_pic[isDale][LRflag].Height();

width=frame\_pic[isDale][LRflag].Width();

Direct =1;//上升

NowAct=2; //運動

NoCollision = true;

MoveSPEED = SPEED;

if(wx<0)wx=0;

if(wx+width>MWIDTH)wx=MWIDTH-width;

}

void ChipDale\_Resurrect::Loading(){

frame\_pic[0][0].LoadBitmapA("Bitmaps/action/chip/Resurrect\_L.bmp",PURPLE);

frame\_pic[0][1].LoadBitmapA("Bitmaps/action/chip/Resurrect\_R.bmp",PURPLE);

frame\_pic[1][0].LoadBitmapA("Bitmaps/action/dale/Resurrect\_L.bmp",PURPLE);

frame\_pic[1][1].LoadBitmapA("Bitmaps/action/dale/Resurrect\_R.bmp",PURPLE);

}

void ChipDale\_Resurrect::OnMove(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

if(Owener->RIGHT\_FLAG){

LRflag=1;

if(wx+width+MoveSPEED<MWIDTH)MoveRight(map);

}

else if(Owener->LEFT\_FLAG){

LRflag=0;

if(wx-MoveSPEED>0 )MoveLeft(map);

}

//自動上下

if(Direct==1)MoveUp(map);

else if(Direct==2)MoveDown(map);

if(wy<=0)Direct=2;

else if(wy+height>MHEIGHT) Direct=1;

//召喚(1秒後按A手動召喚or10秒過後自動召喚)

if(((Owener->A\_FLAG&&timeCount>33\*1)||timeCount>33\*10)&&(wy+height<MHEIGHT)){

Owener->Reset(wx,wy,LRflag==1);

//Owener->SetWxWy(wx,wy);

NowAct=0;

}else timeCount++;

//防止被推出場

if(wx<0)wx=0;

if(wx+width>MWIDTH)wx=MWIDTH-width;

}

void ChipDale\_Resurrect::OnShow(MapManage\* map){

frame\_pic[isDale][LRflag].SetTopLeft(wx,wy);

//1秒前有閃爍效果

if(timeCount>33||(timeCount%5==2||timeCount%5==3))

frame\_pic[isDale][LRflag].ShowBitmap();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//ChipDale\_Taken實作 \*/

//Note::此物件是以chip的NowTakeObj創造 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CAnimation ChipDale\_Taken::animation[2][2];

ChipDale\_Taken::ChipDale\_Taken(ChipDale\* chip){

isDale =chip->Partner->IsDale;

LRflag =chip->LRflag;

height=animation[0][0].Height();

width=animation[0][0].Width();

wx =chip->wx;

wy =chip->wy-height;

Owener =chip;

Owener->Partner->Alive=false;

CanAttackMode=2;

NowAct=1; //拿在手上

NoCollision = true;

UpSpeed = 0;

JumpSPEED = 30;

MoveSPEED = SPEED;

}

void ChipDale\_Taken::Loading(){

animation[0][0].SetDelayCount(5);

animation[0][1].SetDelayCount(5);

animation[1][0].SetDelayCount(5);

animation[1][1].SetDelayCount(5);

animation[0][0].AddBitmap("Bitmaps/action/chip/taken\_1L.bmp",PURPLE);

animation[0][0].AddBitmap("Bitmaps/action/chip/taken\_2L.bmp",PURPLE);

animation[0][0].AddBitmap("Bitmaps/action/chip/taken\_3L.bmp",PURPLE);

animation[0][1].AddBitmap("Bitmaps/action/chip/taken\_1R.bmp",PURPLE);

animation[0][1].AddBitmap("Bitmaps/action/chip/taken\_2R.bmp",PURPLE);

animation[0][1].AddBitmap("Bitmaps/action/chip/taken\_3R.bmp",PURPLE);

animation[1][0].AddBitmap("Bitmaps/action/dale/taken\_1L.bmp",PURPLE);

animation[1][0].AddBitmap("Bitmaps/action/dale/taken\_2L.bmp",PURPLE);

animation[1][0].AddBitmap("Bitmaps/action/dale/taken\_3L.bmp",PURPLE);

animation[1][1].AddBitmap("Bitmaps/action/dale/taken\_1R.bmp",PURPLE);

animation[1][1].AddBitmap("Bitmaps/action/dale/taken\_2R.bmp",PURPLE);

animation[1][1].AddBitmap("Bitmaps/action/dale/taken\_3R.bmp",PURPLE);

}

void ChipDale\_Taken::Throw(int setDirect){

LRflag=Owener->LRflag;

NowAct=2;

Jump();

}

void ChipDale\_Taken::OnMove(MapManage\* map){

if(NowAct==1){

wx=Owener->wx;

wy=Owener->wy-height;

LRflag=Owener->LRflag;

}

else if(NowAct==2){

if(LRflag&&wx+width<MWIDTH)

MoveRight(map);

else if(!LRflag&&wx>0)

MoveLeft(map);

IsJump = true;

FallingDown(map);

if(UpSpeed<=0){

NowAct=0;

Owener->Partner->Reset(wx,wy,LRflag==1,false);

}

}

}

void ChipDale\_Taken::OnShow(MapManage\* map){

animation[isDale][LRflag].SetBottomLeft(wx,wy,height);

animation[isDale][LRflag].OnMove();

animation[isDale][LRflag].OnShow();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Apple實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap Apple::frame\_pic[2];

Apple::Apple(ChipDale \*player){

width = frame\_pic[0].Width();

height = frame\_pic[0].Height();

Owener = player;//記住自己的主人

}

void Apple::Loading(){

frame\_pic[0].LoadBitmapA("Bitmaps/object/Apple\_L.bmp",PURPLE);

frame\_pic[1].LoadBitmapA("Bitmaps/object/Apple\_R.bmp",PURPLE);

}

void Apple::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

else if(NowAct==1){//拿在手上

frame\_pic[Owener->ReturnLR()].SetTopLeft(Owener->ReturnWX()-(width-Owener->ReturnWidth())/2,Owener->ReturnWY()-height\*(!Owener->ReturnHideComplete()));

frame\_pic[Owener->ReturnLR()].ShowBitmap();

}

else{//丟出與落地

frame\_pic[LRflag].SetTopLeft(wx,wy);

frame\_pic[LRflag].ShowBitmap();

}

}

void Apple::OnMove(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

if(NowAct==1) return;//還在人物手上

if(wx < -WinShowBuffer || wx > MWIDTH || wy < -WinShowBuffer){

NowAct = 0;

return;

}

if(Direct&1) MoveUp(map);

if(Direct&4) MoveLeft(map);

if(Direct&8) MoveRight(map);

}

void Apple::Throw(int setDirect){

NowAct=2;

LRflag = Owener->ReturnLR();

Direct = setDirect;

JumpSPEED = 0;

MoveSPEED = SPEED \*2;

NoCollision = true;

IsJump = false;

if(Direct&1)Direct=1;

else Direct = Direct&12;

wx = Owener->ReturnWX()-(width-Owener->ReturnWidth())/2;

wy = Owener->ReturnWY()+Owener->ReturnHeight()-height\*(!Owener->ReturnHideComplete())-40;//離地面 40 公分

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Angel實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CAnimation Angel::frame\_pic[2];

Monster \*\*Angel::monster;

Angel::Angel(int setWx,int setWy,ChipDale \*player):TheBeeMaxSpeed(SPEED\*2){//Angel X Y 方向的移動速度最高為 SPEED\*2

width = frame\_pic[0].Width();

height = frame\_pic[0].Height();

Owener = player;//記住自己的主人

wx = setWx;

wy = setWy;

NowAttack=-1;

CanAttackMode = 3;

LRflag=0;

NoCollision = true;

}

void Angel::Loading(){

frame\_pic[0].AddBitmap("Bitmaps/action/Angel/bee\_1R.bmp",PURPLE);

frame\_pic[1].AddBitmap("Bitmaps/action/Angel/bee\_1L.bmp",PURPLE);

monster = ChipDale::AllMonster;

}

void Angel::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

frame\_pic[LRflag].SetBottomLeft(wx,wy,height);

frame\_pic[LRflag].OnShow();

}

void Angel::OnMove(MapManage\* map){

int tempX,tempY,tempT;

if(NowAct==0){

return;//死亡準備投胎

}

NowAct=2;

if(!Owener->ReturnInvincible()){

if(wx < -WinShowBuffer || wx > MWIDTH || wy < -WinShowBuffer){

NowAct = 0;

return;

}

MoveSPEED = TheBeeMaxSpeed;

if(((ChipDale::Maps->GetRoute())&8)==8) MoveLeft(map);

else MoveRight(map);

MoveUp(map);

NowAct=4;

return;

}

if(NowAttack!=-1 && (monster[NowAttack]->ReturnNowAct()<=0 || monster[NowAttack]->ReturnNowAct()>=100)) NowAttack=-1;//Look if Monster is Alive?

if(NowAttack==-1){//不知道要去攻擊哪一個怪物 搜尋目標

for(int i=0;i<ONE\_LEVEL\_MONSTER\_NUM && monster[i]!=NULL;i++){

if(monster[i]->ReturnNowAct()>=1 && monster[i]->ReturnNowAct()<100 && monster[i]->ReturnCanTrace()){

NowAttack = i;

break;

}

}

}

if(NowAttack==-1){

LRflag = !LRflag;

if(LRflag) tempX = (Owener->ReturnWX()+(Owener->ReturnWidth()-width)/2-5)-wx;//離人物中心偏左5px

else tempX = (Owener->ReturnWX()+(Owener->ReturnWidth()-width)/2+5)-wx;//離人物中心偏左5px

tempY = (Owener->ReturnWY()-height)-wy;

}

else{

tempX = monster[NowAttack]->ReturnWX()-wx;

tempY = monster[NowAttack]->ReturnWY()-wy;

if(tempX<0) LRflag = 1;

else LRflag = 0;

}

//華麗移動(新版)

if(tempX>tempY){

tempT = tempX/TheBeeMaxSpeed;

if(abs(tempX)>=TheBeeMaxSpeed) MoveSPEED=TheBeeMaxSpeed;

else MoveSPEED=abs(tempX);

if(tempX<0) MoveLeft(map);

else if(tempX>0) MoveRight(map);

if(abs(tempT)==0)tempT=1;

if(abs(tempY)>=TheBeeMaxSpeed) MoveSPEED=TheBeeMaxSpeed;

else MoveSPEED=abs(tempY/tempT);

if(tempY<0) MoveUp(map);

else if(tempY>0) MoveDown(map);

}

else{

tempT = tempY/TheBeeMaxSpeed;

if(abs(tempY)>=TheBeeMaxSpeed) MoveSPEED=TheBeeMaxSpeed;

else MoveSPEED=abs(tempY);

if(tempY<0) MoveUp(map);

else if(tempY>0) MoveDown(map);

if(abs(tempT)==0)tempT=1;

if(abs(tempX)>=TheBeeMaxSpeed) MoveSPEED=TheBeeMaxSpeed;

else MoveSPEED=abs(tempX/tempT);

if(tempX<0) MoveLeft(map);

else if(tempX>0) MoveRight(map);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Ball實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap Ball::frame\_pic[2];

Ball::Ball(ChipDale \*player){

width = frame\_pic[0].Width();

height = frame\_pic[0].Height();

Owener = player;//記住自己的主人

CanAttackMode = 1;

LRflag=0;

NowAct=1;

Rebound\_times=0;

IsJump=false;

NoCollision=false;

MoveSPEED=SPEED\*2;

UpSpeed=0;

JumpSPEED=40;

}

void Ball::Loading()

{

frame\_pic[0].LoadBitmapA("Bitmaps/object/Ball\_L.bmp",PURPLE);

frame\_pic[1].LoadBitmapA("Bitmaps/object/Ball\_R.bmp",PURPLE);

}

void Ball::OnMove(MapManage\* map)

{

if(NowAct==0) return;//死亡準備投胎

if(NowAct==1) return;//還在人物手上

if(NowAct==3) {

CanAttackMode=1;

return; //在地上

}

if(wx < -WinShowBuffer || wx > MWIDTH || wy < -WinShowBuffer){

NowAct = 0;

return;

}

if(NowAct==2||NowAct==4) //運動狀態

{

if(Direct&4){

MoveLeft(map);

LRflag=0;

if(wx<=0){

wx=0;

Direct=8;

Rebound\_times--;

}

}

else if(Direct&8){

MoveRight(map);

LRflag=1;

if(wx+width>=MWIDTH){

wx=MWIDTH-width;

Direct=4;

Rebound\_times--;

}

}else if(Direct&1){

MoveUp(map);

if(wy<0){

Direct=2;

Rebound\_times--;

}

}else if(!MoveDown(map)){

Rebound\_times--;

}

if(Rebound\_times<=0){

//Direct=0;

IsJump=true;

FallingDown(map);

CanAttackMode=0;

if(IsJump==false)

{

ThisObjValue = map->FillObstacle(OrderSize+MonsterOrderSize + 0,wx,wy);

wx = (ThisObjValue%E4-1)\*ONEOBJX-map->ReturnNowX();

wy = (ThisObjValue%E8/E4-1)\*ONEOBJY-map->ReturnNowY();

NowAct=5;

}

}

else if(Rebound\_times==1)CanAttackMode=0;

}

}

void Ball::OnShow(MapManage\* map)

{

if(NowAct==0) return;//死亡準備投胎

else if(NowAct==1){//拿在手上

frame\_pic[Owener->ReturnLR()].SetTopLeft(Owener->ReturnWX()-(width-Owener->ReturnWidth())/2,Owener->ReturnWY()-height\*(!Owener->ReturnHideComplete()));

frame\_pic[Owener->ReturnLR()].ShowBitmap();

}

else{//丟出與落地

frame\_pic[LRflag].SetTopLeft(wx,wy);

frame\_pic[LRflag].ShowBitmap();

}

}

void Ball::Throw(int setdirect)

{

NowAct=2;

CanAttackMode=1;

Rebound\_times=2;

wx = Owener->ReturnWX()+(Owener->ReturnWidth()-width)/2;

if(Owener->ReturnLastAct()==4)

wy = Owener->ReturnWY()+Owener->ReturnHeight()-height;

else

wy = Owener->ReturnWY();

if(setdirect&1)

Direct=1;

else if(setdirect&4)

Direct=4;

else if(setdirect&8)

Direct=8;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Craft實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap Craft::frame\_pic[2];

Craft::Craft(ChipDale \*player){

width = frame\_pic[0].Width();

height = frame\_pic[0].Height();

Owener = player;//記住自己的主人

}

void Craft::Loading(){

frame\_pic[0].LoadBitmapA("Bitmaps/object/Craft\_L.bmp",PURPLE);

frame\_pic[1].LoadBitmapA("Bitmaps/object/Craft\_R.bmp",PURPLE);

}

void Craft::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

else if(NowAct==1){//拿在手上

frame\_pic[Owener->ReturnLR()].SetTopLeft(Owener->ReturnWX()-(width-Owener->ReturnWidth())/2,Owener->ReturnWY()-height\*(!Owener->ReturnHideComplete()));

frame\_pic[Owener->ReturnLR()].ShowBitmap();

}

else{//丟出與落地

frame\_pic[LRflag].SetTopLeft(wx,wy);

frame\_pic[LRflag].ShowBitmap();

}

}

void Craft::OnMove(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

if(NowAct==1) return;//還在人物手上

if(wx < -WinShowBuffer || wx > MWIDTH || wy < -WinShowBuffer){

NowAct = 0;

return;

}

if(Direct&1) MoveUp(map);

if(Direct&4) MoveLeft(map);

if(Direct&8) MoveRight(map);

}

void Craft::Throw(int setDirect){

NowAct=2;

LRflag = Owener->ReturnLR();

Direct = setDirect;

JumpSPEED = 0;

MoveSPEED = SPEED \*2;

NoCollision = true;

IsJump = false;

if(Direct&1)Direct=1;

else Direct = Direct&12;

wx = Owener->ReturnWX()-(width-Owener->ReturnWidth())/2;

if(Owener->ReturnLastAct()==4)

wy = Owener->ReturnWY()+Owener->ReturnHeight()-height;

else

wy = Owener->ReturnWY()+Owener->ReturnHeight()-height\*(!Owener->ReturnHideComplete())-40;//離地面 40 公分

}

void Craft::CollisionReact(int setDirect,CFrame \*which){

switch(setDirect){

case 1:case 4:

Direct = 9;

break;

case 2:case 3:

Direct = 5;

break;

case 5:

Direct = 1;

break;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Explosion實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CAnimation Explosion::frame\_pic;

Explosion::Explosion(int setOx,int setOy,int setChangeWhat,ChipDale \*player){

MapManage \*map = ChipDale::Maps;

width = frame\_pic.Width();

height = frame\_pic.Height();

ChangeWhat = setChangeWhat;

wx = setOx\*ONEOBJX-map->ReturnNowX();

wy = setOy\*ONEOBJY-map->ReturnNowY();

frame\_pic.Reset();

CanAttackMode = 2;

NowAct=2;

Owener = player;

IfNeedReFix = false;

}

void Explosion::Loading(){

frame\_pic.SetDelayCount(2);

frame\_pic.AddBitmap("Bitmaps/action/Explosion/Explosion\_1.bmp",PURPLE);

frame\_pic.AddBitmap("Bitmaps/action/Explosion/Explosion\_2.bmp",PURPLE);

frame\_pic.AddBitmap("Bitmaps/action/Explosion/Explosion\_3.bmp",PURPLE);

frame\_pic.AddBitmap("Bitmaps/action/Explosion/Explosion\_4.bmp",PURPLE);

}

void Explosion::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

frame\_pic.SetBottomLeft(wx,wy,height);

frame\_pic.OnShow();

}

void Explosion::OnMove(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

Object \*\*CanThrow = ChipDale::CanThrow;

bool IfNeedReFix=false;

if(frame\_pic.IsFinalBitmap()){

switch(ChangeWhat){ //改變成甚麼要寫在這邊

case STAR:

for(int i=0,IfNeedReFix=false;i<CanTakeNum;i++){

if(CanThrow[i]==this) IfNeedReFix=true;

if(CanThrow[i]==NULL){

CanThrow[i] = new Star(wx,wy);

if(IfNeedReFix) CanThrow[i]->ReFixXY(map);

break;

}

if(i+1==CanTakeNum){

TRACE("CanThrow OverFlow!!!\n");

Sleep(10000);

}

}

break;

case BOX\_ANGLE:

for(int i=0,IfNeedReFix=false;i<CanTakeNum;i++){

if(CanThrow[i]==this) IfNeedReFix=true;

if(CanThrow[i]==NULL){

CanThrow[i] = new Angel(wx,wy,Owener);

break;

}

if(i+1==CanTakeNum){

TRACE("CanThrow OverFlow!!!\n");

Sleep(10000);

}

}

Owener->SetInvincible(1);

break;

case CHEESE:

for(int i=0,IfNeedReFix=false;i<CanTakeNum;i++){

if(CanThrow[i]==this) IfNeedReFix=true;

if(CanThrow[i]==NULL){

CanThrow[i] = new Cheese(wx,wy);

if(IfNeedReFix) CanThrow[i]->ReFixXY(map);

break;

}

if(i+1==CanTakeNum){

TRACE("CanThrow OverFlow!!!\n");

Sleep(10000);

}

}

}

NowAct=0;

}

frame\_pic.OnMove();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Star實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap Star::frame\_pic;

Star::Star(int setWx,int setWy){

width = frame\_pic.Width();

height = frame\_pic.Height();

wx = setWx;

wy = setWy;

CanAttackMode = 2;

NowAct=2;

UpSpeed=30;

IsJump = false;

}

void Star::Loading(){

frame\_pic.LoadBitmapA("Bitmaps/object/7.bmp",PURPLE);

}

void Star::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

frame\_pic.SetTopLeft(wx,wy);

frame\_pic.ShowBitmap();

}

void Star::OnMove(MapManage\* map){

if(NowAct==-1){

NowAct=2;

return;

}

if(NowAct==0) return;//死亡準備投胎

lastUpSpeed=UpSpeed;

FallingDown(map);

if(!IsJump && lastUpSpeed==0){

map->FillObstacle(7,wx,wy);

NowAct=0;

}

else if(!IsJump){

UpSpeed = abs(lastUpSpeed)\*2/3;//落地彈跳 2/3為系數

IsJump=true;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Cheese實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap Cheese::frame\_pic;

Cheese::Cheese(int setWx,int setWy){

width = frame\_pic.Width();

height = frame\_pic.Height();

wx = setWx;

wy = setWy;

CanAttackMode = 2;

NowAct=2;

UpSpeed=30;

IsJump = false;

}

void Cheese::Loading(){

frame\_pic.LoadBitmapA("Bitmaps/object/Cheese.bmp",PURPLE);

}

void Cheese::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

frame\_pic.SetTopLeft(wx,wy);

frame\_pic.ShowBitmap();

}

void Cheese::OnMove(MapManage\* map){

Object \*\*CanThrow = ChipDale::CanThrow;

if(NowAct==-1){

NowAct=2;

return;

}

if(NowAct==0) return;//死亡準備投胎

lastUpSpeed=UpSpeed;

FallingDown(map);

if(!IsJump && lastUpSpeed==0&&NowAct!=3){

//map->FillObstacle(7,wx,wy);

//NowAct=0;

for(int i=0;i<CanTakeNum;i++){

if(CanThrow[i]==NULL){

CanThrow[i] = new Greedy(0,wy+height,this);

TRACE("Greedy: %d %d\n",wx,wy);

break;

}

if(i+1==CanTakeNum){

TRACE("CanThrow OverFlow!!!\n");

Sleep(10000);

}

}

NowAct=3;

}

else if(!IsJump){

UpSpeed = abs(lastUpSpeed)\*2/3;//落地彈跳 2/3為系數

IsJump=true;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Stone實作 \*/

//此Class中的Direct編碼 => 1上 , 2下 , 4左 , 8右 , \*/

// 16跑一次 , 32ReFix \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap Stone::frame\_pic[2];

Stone::Stone(ChipDale \*player){

width = frame\_pic[0].Width();

height = frame\_pic[0].Height();

Owener = player;//記住自己的主人

CanAttackMode = 0;

HideThrow=false;

ReboundLR=0;

ReboundSpeed=0;

NoIgnore\_eat = true;

}

void Stone::Loading(){

frame\_pic[0].LoadBitmapA("Bitmaps/object/Stone\_L.bmp",PURPLE);

frame\_pic[1].LoadBitmapA("Bitmaps/object/Stone\_R.bmp",PURPLE);

}

void Stone::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

else if(NowAct==1){//拿在手上

frame\_pic[Owener->ReturnLR()].SetTopLeft(Owener->ReturnWX()-(width-Owener->ReturnWidth())/2,Owener->ReturnWY()-height\*(!Owener->ReturnHideComplete()));

frame\_pic[Owener->ReturnLR()].ShowBitmap();

}

else{//丟出與落地

frame\_pic[LRflag].SetTopLeft(wx,wy);

frame\_pic[LRflag].ShowBitmap();

//TRACE("show!!\n");

}

}

bool Stone::MoveLeft(MapManage\* map){

if(!map->IfCollision(wx-MoveSPEED,wy,MoveSPEED,height,true)){

wx -= MoveSPEED;

return true;

}else{

int i=0;

while(!map->IfCollision(wx-i,wy,MoveSPEED,height,true))i++;

wx-=i;

Jump();

ReboundLR=1;

ReboundSpeed=(MoveSPEED-i)/4;

}

return false;

}

bool Stone::MoveRight(MapManage\* map){

if(!map->IfCollision(wx+width,wy,MoveSPEED,height,true)){

wx += MoveSPEED;

return true;

}else{

int i=0;

while(!map->IfCollision(wx+width,wy,i,height,true))i++;

wx+=i;

Jump();

ReboundLR=-1;

ReboundSpeed=(MoveSPEED-i)/4;

}

return false;

}

void Stone::ReBound(MapManage\* map){

if(!IsJump){

ReboundLR=0;

ReboundSpeed=0;

}

if(ReboundLR==-1){

if(!map->IfCollision(wx-MoveSPEED,wy,MoveSPEED,height,true))

wx-=ReboundSpeed;

}

else if(ReboundLR==1){

if(!map->IfCollision(wx+width,wy,MoveSPEED,height,true))

wx+=ReboundSpeed;

}

}

void Stone::OnMove(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

if(NowAct==1) return;//還在人物手上

if(wx < -WinShowBuffer || wx > MWIDTH || wy < -WinShowBuffer){

if(NowAct==3){

map->ClearObstacle(ThisObjValue);

TRACE("clear!!!\n");

}

NowAct = 0;

return;

}

if(NowAct==3) return;

if(!IsJump){

IsJump = false;

ThisObjValue = map->FillObstacle(OrderSize+MonsterOrderSize + 0,wx/\*+(Direct&4)/4\*(5)\*(!HideThrow)+(Direct&8)/8\*(-5)\*(!HideThrow)\*/,wy); //(Direct&4)/4\*(5)+(Direct&8)/8\*(-5) 丟出落地後往回拉 不然一開始不丟遠 會打到自己

wx = (ThisObjValue%E4-1)\*ONEOBJX-map->ReturnNowX();

wy = (ThisObjValue%E8/E4-1)\*ONEOBJY-map->ReturnNowY();

if(Direct&32){//有蹲下砸到怪物 會修正人物位置 用 Direct 32 當編碼

Owener->FixSelf\_Onto\_ObjectTop();

Direct &= 31;

}

NowAct = 3;

return;

}

if(Direct&4 && !(Direct&16)) {MoveLeft(map);Direct|=16;}

if(Direct&8 && !(Direct&16)) {MoveRight(map);Direct|=16;}

ReBound(map);

if(FallingDown(map))ReboundLR=0;;

}

void Stone::Throw(int setDirect){

NowAct=2;

LRflag = Owener->ReturnLR();

Direct = setDirect | (Direct&32);

if(Direct&1)Direct=1;

UpSpeed = 0;

MoveSPEED = 45-5;

JumpSPEED = 30;

NoCollision = false;

IsJump = true;

if(Direct&1) Jump();

//TRACE("Direct: %d\n",Direct);

wx = Owener->ReturnWX()+(Owener->ReturnWidth()-width)/2;

if(Owener->ReturnLastAct()==4)

wy = Owener->ReturnWY()+Owener->ReturnHeight()-height;

else

wy = Owener->ReturnWY()-height\*(!Owener->ReturnHideComplete());

}

void Stone::CollisionReact(int setDirect,CFrame \*which){

if(NowAct==1) Direct |= 32;//有蹲下砸到怪物 會修正人物位置 用 Direct 32 當編碼

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Greedy實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CAnimation Greedy::frame\_pic[2];

CMovingBitmap Greedy::frame\_hole;

Greedy::Greedy(int setWx,int setBottonWy,Object \*setCheeseObj){

width = frame\_pic[1].Width();

height = frame\_pic[1].Height();

wx = setWx;

wy = setBottonWy-height;

Owener = NULL;//記住自己的主人

CanAttackMode = 2;

if(setWx<=MWIDTH/2){

LRflag=1;

wx -= width;

}

else

LRflag=0;

NowAct = 2;

MoveSPEED = SPEED\*4/5;

CheeseObj = setCheeseObj;

NoCollision = false;

showMouse=true;

}

void Greedy::Loading(){

frame\_pic[0].SetDelayCount(5);

frame\_pic[0].AddBitmap("Bitmaps/action/Greedy/Greedy\_1L.bmp",PURPLE);

frame\_pic[0].AddBitmap("Bitmaps/action/Greedy/Greedy\_2L.bmp",PURPLE);

frame\_pic[1].SetDelayCount(5);

frame\_pic[1].AddBitmap("Bitmaps/action/Greedy/Greedy\_1R.bmp",PURPLE);

frame\_pic[1].AddBitmap("Bitmaps/action/Greedy/Greedy\_2R.bmp",PURPLE);

frame\_hole.LoadBitmapA("Bitmaps/object/hole.bmp",PURPLE);

}

void Greedy::OnShow(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

if(showMouse && NowAct>0){//拿在手上

frame\_pic[LRflag].SetBottomLeft(wx,wy,height);

frame\_pic[LRflag].OnShow();

frame\_pic[LRflag].OnMove();

}

else if(!showMouse){

frame\_hole.SetTopLeft(wx,wy);

frame\_hole.ShowBitmap();

}

}

void Greedy::OnMove(MapManage\* map){

if(NowAct==0) return;//死亡準備投胎

if(NowAct==1) return;//還在人物手上

if(!showMouse) return;//變成洞了

if(wx>=-width && wx<=width+MWIDTH){

if(!(!LRflag&&MoveLeft(map)) && !(LRflag&&MoveRight(map))){

if(!NoCollision){

NoCollision = true;

if(!LRflag){

PassWx = wx;

MoveLeft(map);

}

else{

PassWx = wx+width;

MoveRight(map);

}

}

}

else if(CheeseObj!=NULL && CheeseObj->IfCollision(wx+(width/2)\*(!LRflag),wy,width/2,height)){//只運算老鼠的一半寬度 這樣看起來比較像有吃到

CheeseObj->SetNowAct(0);

CheeseObj=NULL;

}

else if(NoCollision && (!LRflag&&wx<=PassWx || LRflag&&wx>=PassWx)){

showMouse = false;

map->FillObstacle(13,PassWx,wy-11,false);//11為寫死的值 兩個圖片(Greedy & hole)高的差值

map->FillObstacle(13,PassWx,wy+height-34,false);//34為寫死的 過關門的高度

wx = PassWx;

wy -= 11;

}

}

else //自動消失 (離開顯示螢幕)

NowAct=0;

}

**monster.cpp**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Monster 實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

Monster::Monster(){

CanTrace = true;

wait=0;

MoveSPEED = SPEED;

JumpSPEED=0;

NoCollision=false;

LR\_Space = 0;

wait = 0;

UpSpeed = 0;

LR\_flag = 0;

NowAct = 0;

IsJump = false;

Health = 1;

}

void Monster::CollisionChipDale(ChipDale \*player){

if(NowAct>=1&&NowAct<100){

if(player->ReturnInvincible())return;

if(player->ReturnHideComplete())return;

if(player->IfCollision(wx,wy,width,height)){

player->GetHurt();

}

}

}

ChipDale\* Monster::Detect(ChipDale \*\*player,int \*WLength,int \*HLength,int WRange,int HRange,bool IfTraceInvincible){

ChipDale \*result = NULL;

int twx,twy,twidth,theight,tWLength,tHLength;

MapManage \*Maps = ChipDale::Maps;

for(int i=0;i<2;i++){

if(player[i]==NULL || player[i]->ReturnTakenByPartner()) continue;

if(player[i]->ReturnHealth()<=0)continue;

if(!IfTraceInvincible && player[i]->ReturnInvincible())continue;

twidth = 0;

theight = 0;

twx = wx + width/2;

twy = wy;

if(WRange<0){

twx -= MWIDTH;

twidth = 2\*MWIDTH + width;

}

else{

twidth = WRange;

if(!LRflag)

twx -= WRange;

}

if(HRange<0){

twy = (Maps->ReturnNowY()+twy)/MHEIGHT\*MHEIGHT-Maps->ReturnNowY()-WinShowBuffer;

theight = MHEIGHT+WinShowBuffer;

}

else{

twy -= HRange;

theight = 2\*HRange + height;

}

if(player[i]->IfCollision(twx,twy,twidth,theight)){

if(result!=NULL){

tWLength = player[i]->ReturnWX() - wx;

tHLength = player[i]->ReturnWY() - wy;

if(tWLength\*tWLength+tHLength\*tHLength < (\*WLength)\*(\*WLength)+(\*HLength)\*(\*HLength)){

\*WLength = tWLength;

\*HLength = tHLength;

result = player[i];

}

}

else{

\*WLength = player[i]->ReturnWX() - wx;

\*HLength = player[i]->ReturnWY() - wy;

result = player[i];

}

}

}

return result;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//MachineDog 實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CAnimation MachineDog::frame\_monster[2][2];

MachineDog::MachineDog(MapManage \*map,int SetOx,int SetOy){

MoveSPEED = SPEED\*3/5;

JumpSPEED=30;

NoCollision=false;

LR\_Space = 15;

wait = 15;

width = frame\_monster[0][0].Width();

height = frame\_monster[0][0].Height();

wx = SetOx\*ONEOBJX - map->ReturnNowX();

wy = SetOy\*ONEOBJY - map->ReturnNowY();

lastWy = wy;

UpSpeed = 0;

LR\_flag = 0;

NowAct = 0;

IsJump = false;

Health = 1;

}

void MachineDog::Loading(){

frame\_monster[0][0].SetDelayCount(5);

frame\_monster[0][0].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_stand\_1L.bmp",PURPLE);

frame\_monster[0][0].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_stand\_2L.bmp",PURPLE);

frame\_monster[0][1].SetDelayCount(5);

frame\_monster[0][1].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_stand\_1R.bmp",PURPLE);

frame\_monster[0][1].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_stand\_2R.bmp",PURPLE);

frame\_monster[1][0].SetDelayCount(5);

frame\_monster[1][0].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_run\_1L.bmp",PURPLE);

frame\_monster[1][0].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_run\_2L.bmp",PURPLE);

frame\_monster[1][1].SetDelayCount(5);

frame\_monster[1][1].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_run\_1R.bmp",PURPLE);

frame\_monster[1][1].AddBitmap("Bitmaps/action/","MachineDog/","mechine\_dog\_run\_2R.bmp",PURPLE);

}

void MachineDog::OnShow(MapManage \*map){

if(NowAct>0){

frame\_monster[NowAct>wait][LR\_flag].SetBottomLeft(wx,wy,height);

frame\_monster[NowAct>wait][LR\_flag].OnShow();

frame\_monster[NowAct>wait][LR\_flag].OnMove();

}

}

bool MachineDog::OnMove(MapManage \*map,ChipDale \*\*player){

int MLength,HLength;

if(NowAct<0)return false;

FixXY(map);

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0){

if(Detect(player,&MLength,&HLength,-1,-1,true)!=NULL){

LR\_flag = MLength >= 0;

NowAct=1;//啟動

}

}

else if(NowAct==100){

MoveLeft(map);

MoveUp(map);

}

else if(NowAct==101){

MoveRight(map);

MoveUp(map);

}

else if(NowAct>=1){

if(NowAct <= wait){

NowAct++;

}

else if(NowAct >= wait+1){

if(!IsJump && NowAct >= wait+4){

LR\_flag = !LR\_flag;

NowAct = wait+1;

}

if(!IsJump && wy!=lastWy){

NowAct=wait+1;

lastWy = wy;

LR\_Space = 15;

}

if((LR\_flag==0&&!MoveLeft(map)||LR\_flag==1&&!MoveRight(map))&&!IsJump){

IsJump = true;

LR\_Space = 0;

Jump();

NowAct++;

}

}

FallingDown(map);

}

return true;

}

else if(NowAct>0)

NowAct=-1;//表示活起來後又離開螢幕死掉

return true;

}

bool MachineDog::KillMonster(int Direct){

if(Direct&4)

NowAct=100;

else

NowAct=101;

IsJump=false;

MoveSPEED = SPEED\*2;

NoCollision=true;

Health = 0;

return true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Cactus 實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CMovingBitmap Cactus::frame\_monster;

Cactus::Cactus(MapManage \*map,int SetOx,int SetOy){

wx = SetOx\*ONEOBJX - map->ReturnNowX();

wy = SetOy\*ONEOBJY - map->ReturnNowY();

width = frame\_monster.Width();

height = frame\_monster.Height();

NowAct = 0;

CanTrace=false;

}

void Cactus::Loading(){

frame\_monster.LoadBitmapA("Bitmaps/action/Cactus/Cactus.bmp",PURPLE);

}

bool Cactus::OnMove(MapManage \*map,ChipDale \*\*player){

if(NowAct<0)return false;

FixXY(map);

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0){

NowAct=1;//啟動

}

}

else if(NowAct>0)

NowAct=-1;//表示活起來後又離開螢幕死掉

return true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Electric & Wire 實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

///////////////////////////////////////////////////////////

//Electric 屬於 Wire 架框的實際怪物

CMovingBitmap Electric::frame\_monster[2];

Electric::Electric(MapManage \*map,int SetWx,int SetWy,int SetWireLength):ChangeSpeed(4),MaxMoveSpeed((int)(SPEED\*1.5)){

MoveLength=0;//移動距離 和 顯示時使用

NoCollision=true;

width = frame\_monster[0].Width();

height = frame\_monster[0].Height();

wx = SetWx;

wy = SetWy;

NowAct = 1;

WireLength = SetWireLength;

Health = 1;

LRflag = 1;//電流的流動方向

CanTrace=false;

}

void Electric::Loading(){

frame\_monster[0].LoadBitmapA("Bitmaps/action/Electric/Electric1.bmp",PURPLE);

frame\_monster[1].LoadBitmapA("Bitmaps/action/Electric/Electric2.bmp",PURPLE);

}

void Electric::OnShow(MapManage \*map){

if(NowAct>0){

frame\_monster[(MoveLength%(ChangeSpeed\*2))>=ChangeSpeed].SetTopLeft(wx,wy);

frame\_monster[(MoveLength%(ChangeSpeed\*2))>=ChangeSpeed].ShowBitmap();

}

}

bool Electric::OnMove(MapManage \*map,ChipDale \*\*player){

if(NowAct<0)return false;

if(WireLength == MoveLength){

LRflag = !LRflag;

MoveLength = 0;

}

if(WireLength-MoveLength >= MaxMoveSpeed) MoveSPEED = MaxMoveSpeed;

else MoveSPEED = WireLength-MoveLength;

MoveLength += MoveSPEED;

if(!LRflag) MoveLeft(map);

else MoveRight(map);

return true;

}

///////////////////////////////////////////////////////////

//Wire

Wire::Wire(MapManage \*map,int SetOx,int SetOy,int SetWireOLength){

NoCollision=true;

WireLength = SetWireOLength\*ONEOBJX;

wx = SetOx\*ONEOBJX - map->ReturnNowX();

wy = SetOy\*ONEOBJY - map->ReturnNowY();

NowAct = 0;

Health = 1;

CanTrace=false;

real\_monster = new Electric(map,wx,wy,WireLength);

}

void Wire::Loading(){

Electric::Loading();

}

void Wire::OnShow(MapManage \*map){

if(NowAct>0){

real\_monster->OnShow(map);

}

}

bool Wire::OnMove(MapManage \*map,ChipDale \*\*player){

if(NowAct<0)return false;

FixXY(map);

real\_monster->FixXY(map);

if(wx > -WinShowBuffer-WireLength && wx < MWIDTH+WireLength && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0){

NowAct=1;//啟動

}

if(NowAct==1){

real\_monster->OnMove(map,player);

}

return true;

}

else if(NowAct>0){

NowAct=-1;//表示活起來後又離開螢幕死掉

}

return true;

}

void Wire::CollisionChipDale(ChipDale \*player){

if(NowAct>=1&&NowAct<100){

if(player->ReturnInvincible())return;

if(real\_monster->ElectricCollision(player)){

player->GetHurt();

}

}

}

void Wire::FixMapMove(int fixX,int fixY)

{

wx -= fixX;

wy -= fixY;

real\_monster->FixMapMove(fixX,fixY);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Mouse 實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CAnimation Mouse::frame\_monster[3][2];

Mouse::Mouse(MapManage \*map,int SetOx,int SetOy){

MoveSPEED = SPEED\*3/5;

NoCollision=false;

LR\_Space = 0;

width = frame\_monster[0][0].Width();

height = frame\_monster[0][0].Height();

wx = SetOx\*ONEOBJX - map->ReturnNowX();

wy = SetOy\*ONEOBJY - map->ReturnNowY();

UpSpeed = 0;

LR\_flag = 0;

NowAct = 0;

IsJump = false;

Health = 1;

}

void Mouse::Loading(){

frame\_monster[0][0].SetDelayCount(5);

frame\_monster[0][0].AddBitmap("Bitmaps/action/","Mouse/","mouse\_stand\_L.bmp",PURPLE);

frame\_monster[0][1].SetDelayCount(5);

frame\_monster[0][1].AddBitmap("Bitmaps/action/","Mouse/","mouse\_stand\_R.bmp",PURPLE);

frame\_monster[1][0].SetDelayCount(5);

frame\_monster[1][0].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_1L.bmp",PURPLE);

frame\_monster[1][0].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_2L.bmp",PURPLE);

frame\_monster[1][0].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_3L.bmp",PURPLE);

frame\_monster[1][0].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_4L.bmp",PURPLE);

frame\_monster[1][1].SetDelayCount(5);

frame\_monster[1][1].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_1R.bmp",PURPLE);

frame\_monster[1][1].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_2R.bmp",PURPLE);

frame\_monster[1][1].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_3R.bmp",PURPLE);

frame\_monster[1][1].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Run\_4R.bmp",PURPLE);

frame\_monster[2][0].SetDelayCount(5);

frame\_monster[2][0].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Jump\_L.bmp",PURPLE);

frame\_monster[2][1].SetDelayCount(5);

frame\_monster[2][1].AddBitmap("Bitmaps/action/","Mouse/","mouse\_Jump\_R.bmp",PURPLE);

}

void Mouse::OnShow(MapManage \*map){

int showAct;

if(NowAct>0){

if(!IsJump) showAct=0;

else showAct=2;

if(NowAct>=50) showAct=1;

frame\_monster[showAct][LR\_flag].SetBottomLeft(wx,wy,height);

frame\_monster[showAct][LR\_flag].OnShow();

frame\_monster[showAct][LR\_flag].OnMove();

}

}

bool Mouse::OnMove(MapManage \*map,ChipDale \*\*player){

int MLength,HLength;

if(NowAct<0)return false;

FixXY(map);

if(wx > -WinShowBuffer && wx < MWIDTH+WinShowBuffer && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0 && wx > 0 && wx < MWIDTH && wy > 0 && wy < MHEIGHT){

if(Detect(player,&MLength,&HLength,-1,-1,true)!=NULL){

LR\_flag = MLength >= 0;

NowAct=1;//啟動

}

NowAct=1;//啟動

}

else if(NowAct==100){

MoveLeft(map);

MoveUp(map);

}

else if(NowAct==101){

MoveRight(map);

MoveUp(map);

}

else if(NowAct>=1){

tracePlayer = Detect(player,&MLength,&HLength,-1,0);

if(tracePlayer!=NULL){

LR\_flag = MLength >= 0;

NowAct = 50;

}

if(NowAct <= wait){

NowAct++;

}

else if(NowAct >= wait+1 && NowAct < 50){

if(!IsJump){

if(NowAct >= wait+5\*4){

LR\_flag = !LR\_flag;

NowAct = wait+1;

}

if((NowAct-wait)%4==1){

IsJump = true;

JumpSPEED=30;

Jump();

}

NowAct++;

}

else{

if(LR\_flag==0){

if(!MoveLeft(map)){

LR\_flag = !LR\_flag;

NowAct = wait+2;

}

}

else if(LR\_flag==1){

if(!MoveRight(map)){

LR\_flag = !LR\_flag;

NowAct = wait+2;

}

}

}

}

else if(NowAct >= 50){

if((LR\_flag==0&&!MoveLeft(map)||LR\_flag==1&&!MoveRight(map)) && !IsJump){

IsJump = true;

JumpSPEED=30;

Jump();

if(NowAct==52){

LR\_flag = !LR\_flag;

NowAct = 50;

}

NowAct++;

}

if(Detect(player,&MLength,&HLength,-1,height)==NULL){

NowAct = wait+1;

//LR\_flag = !LR\_flag;

}

}

FallingDown(map);

}

return true;

}

else if(NowAct>0)

NowAct=-1;//表示活起來後又離開螢幕死掉

return true;

}

bool Mouse::KillMonster(int Direct){

if(Direct&4)

NowAct=100;

else

NowAct=101;

IsJump=false;

MoveSPEED = SPEED\*2;

NoCollision=true;

Health = 0;

return true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Wasp 實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CAnimation Wasp::frame\_monster[2];

Wasp::Wasp(MapManage \*map,int SetOx,int SetOy){

NoCollision=true;

width = frame\_monster[0].Width();

height = frame\_monster[0].Height();

wx = SetOx\*ONEOBJX - map->ReturnNowX();

wy = SetOy\*ONEOBJY - map->ReturnNowY();

NowAct = 0;

Health = 1;

MoveSPEED = 3;

}

void Wasp::Loading(){

frame\_monster[0].SetDelayCount(5);

frame\_monster[0].AddBitmap("Bitmaps/action/","Wasp/","Wasp\_1L.bmp",PURPLE);

frame\_monster[0].AddBitmap("Bitmaps/action/","Wasp/","Wasp\_2L.bmp",PURPLE);

frame\_monster[1].SetDelayCount(5);

frame\_monster[1].AddBitmap("Bitmaps/action/","Wasp/","Wasp\_1R.bmp",PURPLE);

frame\_monster[1].AddBitmap("Bitmaps/action/","Wasp/","Wasp\_2R.bmp",PURPLE);

}

void Wasp::OnShow(MapManage \*map){

if(NowAct>0){

frame\_monster[LR\_flag].SetBottomLeft(wx,wy,height);

frame\_monster[LR\_flag].OnShow();

frame\_monster[LR\_flag].OnMove();

}

}

bool Wasp::OnMove(MapManage \*map,ChipDale \*\*player){

int MLength,HLength;

if(NowAct<0)return false;

FixXY(map);

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0){

if(Detect(player,&MLength,&HLength,-1,-1,true)!=NULL){

LR\_flag = MLength >= 0;

NowAct=1;//啟動

}

}

else if(NowAct==100){

MoveLeft(map);

MoveUp(map);

}

else if(NowAct==101){

MoveRight(map);

MoveUp(map);

}

else if(NowAct>=1){

if(NowAct <= wait){

NowAct++;

}

else if(NowAct >= wait+1){

if(NowAct == wait+1){

MoveDown(map);

if(Detect(player,&MLength,&HLength,-1,height/2)){

NowAct++;

}

}

else{

MoveSPEED = 1;

MoveDown(map);

MoveSPEED = SPEED;

if(LR\_flag==0) MoveLeft(map);

if(LR\_flag==1) MoveRight(map);

}

}

}

return true;

}

else if(NowAct>0)

NowAct=-1;//表示活起來後又離開螢幕死掉

return true;

}

bool Wasp::KillMonster(int Direct){

if(Direct&4)

NowAct=100;

else

NowAct=101;

IsJump=false;

MoveSPEED = SPEED\*2;

NoCollision=true;

Health = 0;

return true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//bullet 實作 怪物所有用的子彈 \*/

// 需要另外做onshow \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

bullet::bullet(){

NoCollision=true;

NowAct = 1;

Health = 1;

LRflag = 1;

CanTrace=false;

}

bool bullet::OnMove(MapManage \*map){

if(NowAct<0)return false;

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0){

NowAct=1;//啟動

}

if(NowAct==1){

MoveSPEED = abs(MoveLR);

if(MoveLR<0) MoveLeft(map);

if(MoveLR>0) MoveRight(map);

MoveSPEED = abs(MoveUD);

if(MoveUD<0) MoveUp(map);

if(MoveUD>0) MoveDown(map);

}

return true;

}

else if(NowAct>0){

NowAct=-1;//表示活起來後又離開螢幕死掉

}

return true;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Centipede & Centipedelimbs 實作 \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

///////////////////////////////////////////////////////////

//Lighting

CMovingBitmap Lighting::frame\_monster;

Lighting::Lighting(MapManage \*map,int SetWx,int SetWy,int LR,int UD){

wx = SetWx;

wy = SetWy;

MoveLR=LR;

MoveUD=UD;

}

void Lighting::Loading(){

frame\_monster.LoadBitmapA("Bitmaps/action/Centipede/Lightning.bmp",PURPLE);

}

void Lighting::OnShow(MapManage \*map){

if(NowAct>0){

frame\_monster.SetTopLeft(wx,wy);

frame\_monster.ShowBitmap();

}

}

///////////////////////////////////////////////////////////

//Centipedelimbs

CAnimation Centipedelimbs::frame\_monster[3];

Centipedelimbs::Centipedelimbs(MapManage \*map,int SetWx,int SetWy,int SetSelect):ReleaseBullTime(15),TotalBullTime(50),BulletMaxSpeed(10),WRandBullet(100){

NoCollision=true;

wx = SetWx;

wy = SetWy;

NowAct = 0;

Health = 1;

CanTrace=false;

Select = SetSelect;

width = frame\_monster[Select].Width();

height = frame\_monster[Select].Height();

for(int i=0;i<CentipedeLightingNum;i++)

bullets[i] = NULL;

}

Centipedelimbs::~Centipedelimbs(){

for(int i=0;i<CentipedeLightingNum;i++)

if(bullets[i]!=NULL)

delete(bullets[i]);

}

void Centipedelimbs::Loading(){

frame\_monster[0].SetDelayCount(2);

frame\_monster[0].AddBitmap("Bitmaps/action/","Centipede/","head.bmp",PURPLE);

frame\_monster[0].AddBitmap("Bitmaps/action/","Centipede/","head\_die.bmp",PURPLE);

frame\_monster[1].SetDelayCount(1);

frame\_monster[1].AddBitmap("Bitmaps/action/","Centipede/","L\_Fist.bmp",PURPLE);

frame\_monster[1].AddBitmap("Bitmaps/action/","Centipede/","L\_Cloth.bmp",PURPLE);

frame\_monster[2].SetDelayCount(1);

frame\_monster[2].AddBitmap("Bitmaps/action/","Centipede/","R\_Fist.bmp",PURPLE);

frame\_monster[2].AddBitmap("Bitmaps/action/","Centipede/","R\_Cloth.bmp",PURPLE);

}

void Centipedelimbs::OnShow(MapManage \*map,int countFlicker){

if(NowAct<0)return;

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

switch(Select){

case 0:

if(countFlicker==0)

frame\_monster[0].Reset();

frame\_monster[0].SetBottomLeft(wx,wy,height);

frame\_monster[0].OnShow();

frame\_monster[0].OnMove();

break;

case 1:case 2:

if(TotalBullTime-NowAct<=ReleaseBullTime) frame\_monster[Select].OnMoveToNum(1);

else frame\_monster[Select].OnMoveToNum(0);

frame\_monster[Select].SetBottomLeft(wx,wy,height);

frame\_monster[Select].OnShow();

for(int i=0;i<CentipedeLightingNum;i++){

if(bullets[i]!=NULL)

bullets[i]->OnShow(map);

}

break;

}

}

}

bool Centipedelimbs::OnMove(MapManage \*map,ChipDale \*\*player){

int WLength,HLength,UD,LR;

if(NowAct<0)return false;

for(int i=0;i<CentipedeLightingNum;i++){

if(bullets[i]!=NULL)

bullets[i]->FixXY(map);

}

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0){

NowAct=1;//啟動

}

if(NowAct>=1){

if(Select>=1){//Select 1 & 2 左手跟右手 , 0 是頭

if(TotalBullTime-NowAct==ReleaseBullTime){

for(int i=0;i<CentipedeLightingNum;i++){

if(bullets[i]!=NULL){

if(bullets[i]->ReturnNowAct()<0){

delete(bullets[i]);

bullets[i]=NULL;

}

continue;

}

else{

Detect(player,&WLength,&HLength,-1,-1,true);

WLength += rand()%(WRandBullet\*2)-WRandBullet;

if(WLength==0&&HLength==0) continue;

if(abs(WLength)>abs(HLength)){

LR = WLength>0 ? BulletMaxSpeed:-BulletMaxSpeed;

UD = HLength/(WLength/LR);

}

else{

UD = HLength>0 ? BulletMaxSpeed:-BulletMaxSpeed;

LR = WLength/(HLength/UD);

}

}

if(Select==1){

bullets[i] = new Lighting(map,wx+36,wy+32,LR,UD);

break;

}

else if(Select==2){

bullets[i] = new Lighting(map,wx+10,wy+23,LR,UD);

break;

}

}

}

NowAct++;

if(NowAct>50)NowAct=1;

}

for(int i=0;i<CentipedeLightingNum;i++){

if(bullets[i]!=NULL)

bullets[i]->OnMove(map);

}

}

return true;

}

else if(NowAct>0){

NowAct=-1;//表示活起來後又離開螢幕死掉

}

return true;

}

int Centipedelimbs::CentipedelimbsCollision(ChipDale \*player){

if(player->IfCollision(wx,wy,width,height))

return true;

for(int i=0;i<CentipedeLightingNum;i++){

if(bullets[i]!=NULL && bullets[i]->Collision(player))

return true;

}

return false;

}

///////////////////////////////////////////////////////////

//Centipede

CAnimation Centipede::frame\_monster;

Centipede::Centipede(MapManage \*map,int SetOx,int SetOy){

NoCollision=true;

wx = SetOx\*ONEOBJX - map->ReturnNowX();

wy = SetOy\*ONEOBJY - map->ReturnNowY();

width = frame\_monster.Width();

height = frame\_monster.Height();

TRACE("Centipede!!! wx %d , wy %d\n",wx,wy);

NowAct = 0;

Health = 5;

CanTrace=false;

real\_monster[0] = new Centipedelimbs(map,wx+180,wy-32,0);

real\_monster[1] = new Centipedelimbs(map,wx-22,wy+52,1);

real\_monster[2] = new Centipedelimbs(map,wx+321,wy-24,2);

countFlicker=0;

}

Centipede::~Centipede(){

for(int i=0;i<3;i++)

delete(real\_monster[i]);

}

void Centipede::Loading(){

frame\_monster.SetDelayCount(2);

frame\_monster.AddBitmap("Bitmaps/action/","Centipede/","Centipede\_body\_normal.bmp",PURPLE);

frame\_monster.AddBitmap("Bitmaps/action/","Centipede/","Centipede\_body\_die.bmp",PURPLE);

Lighting::Loading();

Centipedelimbs::Loading();

}

void Centipede::OnShow(MapManage \*map){

if(NowAct<0)return;

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

if(countFlicker==0)

frame\_monster.Reset();

else

countFlicker--;

frame\_monster.SetBottomLeft(wx,wy,height);

frame\_monster.OnShow();

frame\_monster.OnMove();

for(int i=0;i<3;i++)

real\_monster[i]->OnShow(map,countFlicker);

}

}

bool Centipede::OnMove(MapManage \*map,ChipDale \*\*player){

if(NowAct<0)return false;

FixXY(map);

real\_monster[0]->FixXY(map);

real\_monster[1]->FixXY(map);

real\_monster[2]->FixXY(map);

if(wx > -WinShowBuffer && wx < MWIDTH && wy > -WinShowBuffer && wy < MHEIGHT){

if(NowAct==0){

NowAct=1;//啟動

}

if(NowAct==1){

for(int i=1;i<3;i++)//real\_monster[0] 基本上不會有OnMove

real\_monster[i]->OnMove(map,player);

}

return true;

}

else if(NowAct>0){

NowAct=-1;//表示活起來後又離開螢幕死掉

}

return true;

}

void Centipede::CollisionChipDale(ChipDale \*player){

if(NowAct>=1&&NowAct<100){

if(player->ReturnInvincible())return;

if(real\_monster[1]->CentipedelimbsCollision(player)||real\_monster[2]->CentipedelimbsCollision(player)){

player->GetHurt();

}

}

}

void Centipede::FixMapMove(int fixX,int fixY)

{

wx -= fixX;

wy -= fixY;

real\_monster[0]->FixMapMove(fixX,fixY);

real\_monster[1]->FixMapMove(fixX,fixY);

real\_monster[2]->FixMapMove(fixX,fixY);

}

bool Centipede::KillMonster(int Direct){

if(countFlicker!=0) return false;

Health--;

if(Health>0){

countFlicker = 50;

return false;

}

else{

CGameStateRun::ToBonus();

NowAct=-1;

return true;

}

}