國立臺北科技大學

2022 Spring 資工系物件導向程式實習

期末報告

貓狗大戰



第6組

108590006 官齊笎

108590042 莊瞻

**目錄**

1. **簡介**
2. 動機 3
3. 分工 3
4. **遊戲介紹**
5. 遊戲說明 3
6. 遊戲圖形 4
7. 遊戲音效 8
8. **程式設計**
9. 程式架構 9
10. 程式類別 10
11. 程式技術 10
12. **結語**
13. 問題及解決方法 10
14. 時間表 11
15. 貢獻比例 11
16. 自我檢核表 12
17. 收獲 12
18. 心得、感想 12
19. 對於本課程的建議 13
20. **附錄**

**一、 簡介**

**1. 動機**

一開始我們想從較簡單的小遊戲選擇，然後我們想到小時候玩過的貓狗大戰，玩過之後我們覺得這個遊戲符合我們的目標。

**2. 分工**

我們這一組的分工方式是一人當Driver，另一人當Navigator。Driver由莊瞻同學擔任，負責根據Navigator的建議撰寫程式碼，還有擷取部分遊戲音效素材；Navigator由官齊笎同學擔任，負責指示Driver開發過程中的大方向以及提供程式邏輯，還有擷取所有遊戲圖片素材、部分音效素材。期末報告則由兩人共同撰寫完成。

**二、 遊戲介紹**

**1. 遊戲說明**

遊戲規則與玩法：

兩方可以在自身回合使用武器和技能攻擊，武器命中對手時會造成對方扣血，血量歸零者敗北。有單人和雙人遊戲可選擇。單人遊戲和電腦對戰時可以選擇難度；雙人遊戲則是玩家跟玩家進行對戰。只有玩家才可以使用四種技能。

玩家可使用的技能：

* 連擊技能：在一回合中連續攻擊兩次，若攻擊為miss，則會兩次都會miss。
* 重擊技能：攻擊範圍增大，傷害上升。
* 中毒技能：命中時會讓對方在之後的一定回合扣血。
* 治癒技能：回復自身血量，使用後換對方的回合。

密技：

* 按D鍵可以將狗的血量歸零。
* 按C鍵可以將貓的血量歸零。

**2. 遊戲圖形**



圖1：遊戲開頭畫面

一張含有 文字 的圖片

自動產生的描述

圖2：遊戲故事介紹畫面

一張含有 文字 的圖片

自動產生的描述

圖3：遊戲玩法介紹畫面



圖4：單人模式或雙人模式選擇畫面



圖5：單人遊戲模式難度選擇畫面



圖6：單人遊戲模式畫面



圖7：雙人遊戲模式畫面



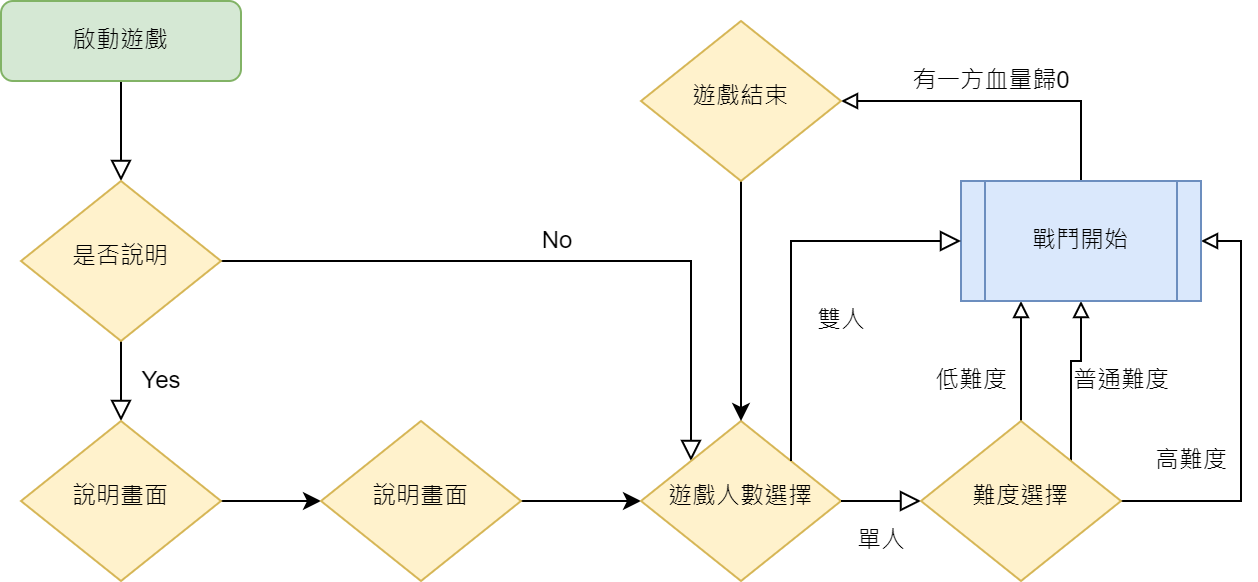
圖8：貓咪勝利遊戲結束畫面

**3. 遊戲音效**

|  |  |
| --- | --- |
| 音效名稱 | 音效說明 |
| 對戰-被精準命中 | 分成狗和貓兩種，攻擊準確命中時的音效 |
| 對戰-被攻擊擦到 | 分成狗和貓兩種，攻擊稍微打到時的音效 |
| 對戰-沒被攻擊到 | 分成狗和貓兩種，攻擊miss時的笑聲 |
| 對戰-技能發動 | 分成狗和貓兩種，技能發動時播放 |
| 對戰-武器發射 | 分成狗和貓兩種，武器射出時播放 |
| 對戰-治癒技能 | 回復時的滴滴聲 |
| 按鈕-叮一聲(短) | 其餘畫面的按鈕、對戰時回合輪到玩家的音效 |
| 按鈕-叮二聲 | 開頭畫面的按鈕、選擇單人遊戲按鈕 |
| 按鈕-叮二聲 | 遊戲結束Replay按鈕、選擇雙人遊戲按鈕 |
| 背景音效-進入遊戲 | 啟動遊戲畫時播放 |
| 背景音效-進入對戰 | 選完玩家人數和難度後進入對戰畫面時播放 |
| 背景音效-遊戲結束 | 其中一方血量歸零進到遊戲結束畫面時播放 |

**三、 程式設計**

**1. 程式架構**



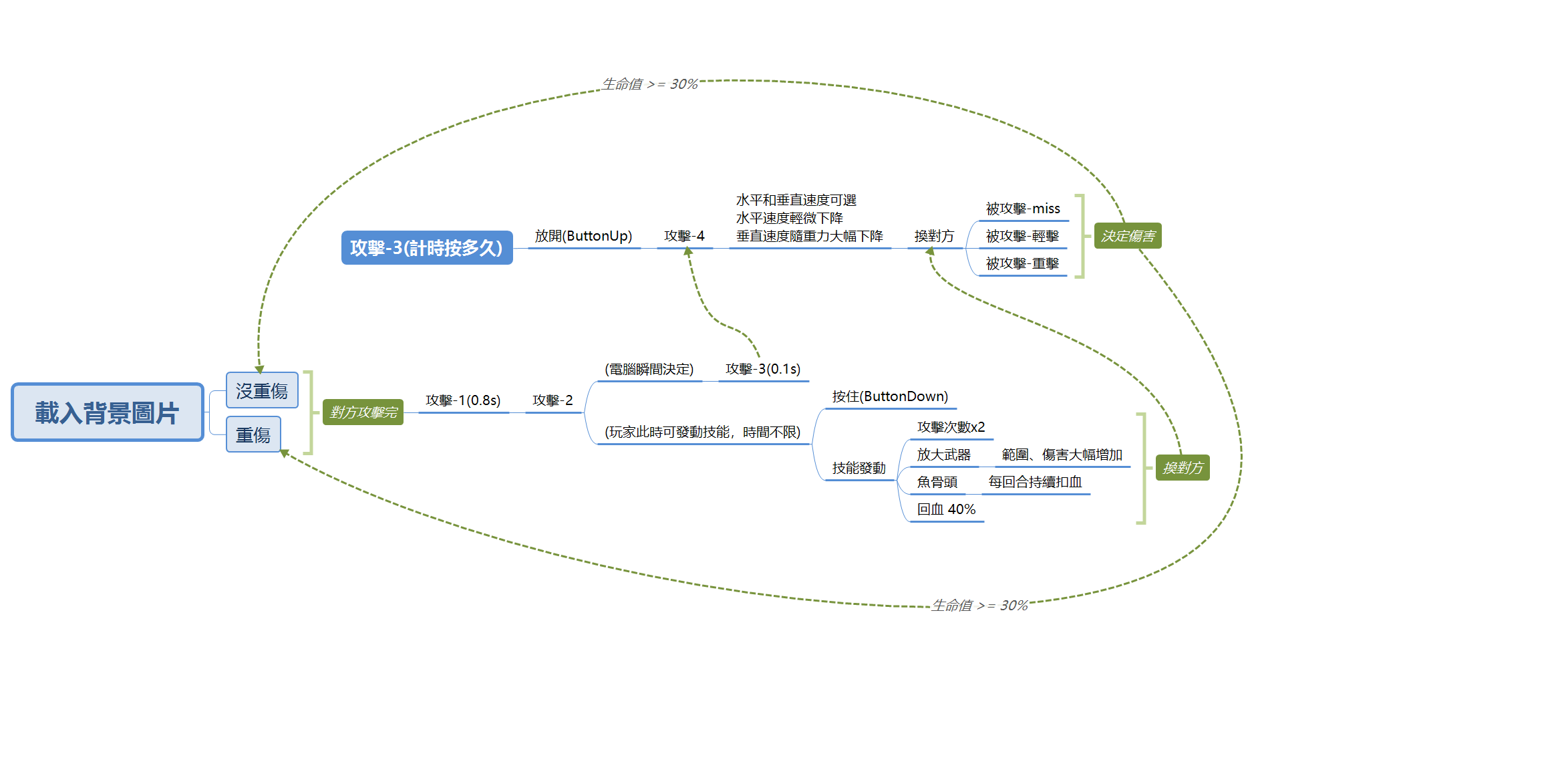
圖9：整體遊戲流程圖

圖10：戰鬥過程狀態流程圖

**2. 程式類別**

|  |  |  |  |
| --- | --- | --- | --- |
| 類別名稱 | .h 檔行數 | .cpp 檔行數 | 說明 |
| Cat | 29 | 85 | 控制貓的圖片及動畫 |
| CatHP | 25 | 75 | 記錄貓的血量及狀態 |
| Dog | 29 | 85 | 控制狗的圖片及動畫 |
| DogHP | 25 | 75 | 記錄狗的血量及狀態 |
| GameData | 11 | 0 | 紀錄要在各個視窗傳遞的全域變數 |
| GAMEID | 92 | 0 | 紀錄遊戲可以切換到的狀態 |
| Weapon | 45 | 203 | 控制武器的動畫 |
| 總行數 | 256 | 523 |  |

**3. 程式技術**

我們使用GAMEID.h裡面預設好的各種狀態來確定程式所在的畫面及內容，並配合OnMove()+鍵盤+滑鼠的操作切換畫面。

**四、 結語**

**1. 問題及解決方法**

|  |  |
| --- | --- |
| 遇到的問題 | 解決方法 |
| 圖片無法完全將背景透明，因為背景的顏色是漸層，整個素材沒有外框，看齊來就和漸層混在一起 | 將背景色強行抹上透明的顏色，並將不清楚的圖片外框加深加黑 |
| 擷取出來的音效有0.x秒的延遲，在播放的時候有點怪 | 不要在意音效被切到的問題，強行切，確保沒有無聲音的部分 |
| 在開始畫面的參數不知如何傳到戰鬥畫面 | 新增class GameData，將變數放入，並在mygame.cpp的程式上方初始化 |
| OnBeginState()無法播放音效 | 宣告一個bool值，用來表示是否播放過，並將播音效程式放在OnMove() |
| 一開始不知道要怎麼實現出戰鬥畫面上方血量條的效果 | 利用多個紅色小圖片排成一排來當作血量條，並且會隨者血量的減少，讓部分的圖片不再顯示，來達到血量條減少的視覺效果，並且在覆蓋一層血量條外框的圖片。 |
| 如何判斷現在是在哪個頁面按下特定按鈕，圖時顯示相對應畫面 | 在GAMEID.h裡面預先枚舉所有可能的狀態，根據當下的狀態顯示對應的畫面以及特定的程式功能。 |

**2. 時間表**

|  |  |  |  |
| --- | --- | --- | --- |
| 週次 | 組員-官齊笎(小時) | 組員-莊瞻(小時) | 說明 |
| 3 | 3 | 3 | 新增gitignore還有遊戲的原始文件。  修復錯誤  修改遊戲圖示檔  修改遊戲安裝檔屬性  修改專案屬性  增加icon不同尺寸的圖片，修改Setup裡的檔案名稱 |
| 4 | 6 | 6 | 剪輯圖片素材  遊戲起始畫面製作  遊戲Init完成，開始做GameRun |
| 5 | 5 | 5 | 將貓狗程式 Class 化  新增GAMEID戰鬥中狀態，並根據狀態顯示貓狗的圖片。 |
| 6 | 5 | 5 | 丟武器碰撞與標題畫面動畫 |
| 7 | 3 | 3 | 遊戲開始畫面的動畫+關卡難度可選(參數修正)+戰鬥中動畫 |
| 9 | 2 | 2 | 單人遊戲完成 |
| 10 | 3 | 3 | 遊戲結束素材處理與開始畫面補充動畫素材 |
| 11 | 3 | 3 | 技能動畫素材擷取+可閃爍 |
| 12 | 3 | 3 | 治癒、連擊技能 |
| 14 | 3 | 3 | 中毒技能、雙人遊戲、骨頭圖片修正 |
| 15 | 3 | 3 | 音樂及音效素材擷取 |
| 17 | 2 | 2 | 完成所有音效 |
| 總時數 | 41 | 41 |  |

**3. 貢獻比例**

組員-官齊笎：50%

組員-莊瞻：50%

**4. 自我檢核表**

|  |  |  |  |
| --- | --- | --- | --- |
| 項目 | 項目 | 完成否 | 無法完成原因 |
| 1 | 解決 Memory leak | █已完成 □未完成 |  |
| 2 | 自定遊戲 Icon | █已完成 □未完成 |  |
| 3 | 全螢幕啟動 | █已完成 □未完成 |  |
| 4 | 有 About 畫面 | █已完成 □未完成 |  |
| 5 | 初始畫面說明按鍵及滑鼠之用法與密技 | █已完成 □未完成 |  |
| 6 | 上傳 setup/apk/source 檔 | █已完成 □未完成 |  |
| 7 | setup 檔可正確執行 | █已完成 □未完成 |  |
| 8 | 報告字型、點數、對齊、行距、頁碼等格式正確 | █已完成 □未完成 |  |

**5. 收獲**

官齊笎：

這次開發方式較為固定，一人操作電腦，另一人指揮，最不太會出現工作分配不當或衝突，整個開發流程順利很多。整個開發流程順利進行，專案的開發也非常順利，讓我了解到整個團隊的開發模式和工作方式會大大影響整個專案的開發。

莊瞻：

我在這學期的開發中有學到一些特別的技巧，例如定義一個class用來幫助我們在不同game state中儲存資訊以在另一個game state使用，換做我的話可能想不到。還有了解如何使用debugger來查看出錯的地方，利用逐行執行來去檢查變數裡面的變化。

**6. 心得、感想**

官齊笎：

這次雖然選了一個架構較簡單的小遊戲做，但擷取素材時特別麻煩，要將各種小動畫的圖片擷取出來卻比想像中費事。因為這款小遊戲各種地方都有小動畫，有時沒注意到的部分像是眼睛在動這種就會被忽略。而且我不擅長擷取，用Xbox Game Bar的錄影出來的影片不太理想，會混有鼠標，解析度也很奇怪，最後就直接開遊戲一邊打，一邊截圖。

莊瞻：

我們這次開發是使用Pair programming的方式進行開發，一人當Driver，另一人當Navigator，這樣的開發方式讓我們彼此都清楚知道現在的進度在哪裡，整體開發過程滿順利的，這樣也讓我認識到不同的人是怎麼思考他們的程式邏輯，換成是我自己一個人的話可能就會因此卡住。還有我覺得這次遊戲開發雖然有用到一些OOP的概念，但是在實際上好像沒有用的很好。

**7. 對於本課程的建議**

要是期末報告能有一個word的範本直接改就很方便，也不太需要擔心字型和各種格式的問題。

**五、 附錄**

|  |
| --- |
| Cat.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <ddraw.h>  #include "gamelib.h"  #include "GAMEID.h"  #include "Cat.h"  namespace game\_framework {  /////////////////////////////////////////////////////////////////////////////  // Cat: Cat class  /////////////////////////////////////////////////////////////////////////////  Cat::Cat()  {  CatPositionX = 0;  CatPositionY = 559;  }  void Cat::Initialize()  {  Cat\_Normal.SetTopLeft(CatPositionX, CatPositionY);  Cat\_Normal\_BadlyHurt.SetTopLeft(CatPositionX, CatPositionY);  Cat\_Attack\_1.SetTopLeft(CatPositionX, CatPositionY);  Cat\_Attack\_2.SetTopLeft(CatPositionX, CatPositionY);  Cat\_Attack\_3.SetTopLeft(CatPositionX, CatPositionY);  Cat\_Attack\_4.SetTopLeft(CatPositionX, CatPositionY);  Cat\_BeingAttacked\_Big.SetTopLeft(CatPositionX, CatPositionY);  Cat\_BeingAttacked\_Little.SetTopLeft(CatPositionX, CatPositionY);  Cat\_BeingAttacked\_Miss.SetTopLeft(CatPositionX, CatPositionY);  }  void Cat::LoadBitmap()  {  Cat\_Normal.LoadBitmap("GamePicture/GameRun/Cat/Normal.bmp");  Cat\_Normal\_BadlyHurt.LoadBitmap("GamePicture/GameRun/Cat/Normal\_BadlyHurt.bmp");  Cat\_Attack\_1.LoadBitmap("GamePicture/GameRun/Cat/Attack\_1.bmp");  Cat\_Attack\_2.LoadBitmap("GamePicture/GameRun/Cat/Attack\_2.bmp");  Cat\_Attack\_3.LoadBitmap("GamePicture/GameRun/Cat/Attack\_3.bmp");  Cat\_Attack\_4.LoadBitmap("GamePicture/GameRun/Cat/Attack\_4.bmp");  Cat\_BeingAttacked\_Big.LoadBitmap("GamePicture/GameRun/Cat/BeingAttacked\_Big.bmp");  Cat\_BeingAttacked\_Little.LoadBitmap("GamePicture/GameRun/Cat/BeingAttacked\_Little.bmp");  Cat\_BeingAttacked\_Miss.LoadBitmap("GamePicture/GameRun/Cat/BeingAttacked\_Miss.bmp");  }  void Cat::OnMove(GAME\_RUN\_ID runId)  {  // 根據目前的條件改變狀態  }  void Cat::OnShow(PICTURE\_STATUS pictureStatus)  {  switch (pictureStatus)  {  case Normal:  Cat\_Normal.ShowBitmap();  break;  case Normal\_BadlyHurt:  Cat\_Normal\_BadlyHurt.ShowBitmap();  break;  case Attack\_1:  Cat\_Attack\_1.ShowBitmap();  break;  case Attack\_2:  Cat\_Attack\_2.ShowBitmap();  break;  case Attack\_3:  Cat\_Attack\_3.ShowBitmap();  break;  case Attack\_4:  Cat\_Attack\_4.ShowBitmap();  break;  case BeingAttacked\_Big:  Cat\_BeingAttacked\_Big.ShowBitmap();  break;  case BeingAttacked\_Little:  Cat\_BeingAttacked\_Little.ShowBitmap();  break;  case BeingAttacked\_Miss:  Cat\_BeingAttacked\_Miss.ShowBitmap();  break;  default:  break;  }  }  } |
| Cat.h |
| #pragma once  namespace game\_framework {  class Cat  {  public:  Cat();  void OnMove(GAME\_RUN\_ID runId); // 根據目前的條件改變狀態  void OnShow(PICTURE\_STATUS pictureStatus); // 根據目前的狀態貼上貓貓圖片  void Initialize(); // 初始化設定  void LoadBitmap();  protected:  int CatPositionX;  int CatPositionY;  // 貓貓的圖片  CMovingBitmap Cat\_Normal;  CMovingBitmap Cat\_Normal\_BadlyHurt;  CMovingBitmap Cat\_Attack\_1;  CMovingBitmap Cat\_Attack\_2;  CMovingBitmap Cat\_Attack\_3;  CMovingBitmap Cat\_Attack\_4;  CMovingBitmap Cat\_BeingAttacked\_Big;  CMovingBitmap Cat\_BeingAttacked\_Little;  CMovingBitmap Cat\_BeingAttacked\_Miss;  };  } |
| CatHP.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <ddraw.h>  #include "gamelib.h"  #include "GAMEID.h"  #include "CatHP.h"  namespace game\_framework {  /////////////////////////////////////////////////////////////////////////////  // CatHP: CatHP class  /////////////////////////////////////////////////////////////////////////////  CatHP::CatHP()  {  HealthPoint = 100;  poisoningRound = 0;  }  void CatHP::Initialize()  {  HealthPoint = 100;  for (int i = 0; i < 469; i++) {  CatHealthPointPicture[i].SetTopLeft(561 - i, 57);  }  }  void CatHP::LoadBitmap()  {  for (int i = 0; i < 469; i++) {  CatHealthPointPicture[i].LoadBitmap("GamePicture/GameRun/Cat/HP\_semicircle.bmp", RGB(180, 0, 255));  }  }  void CatHP::OnShow()  {  for (int i = 0; i < HealthPoint \* 469/100; i++) {  CatHealthPointPicture[i].ShowBitmap();  }  }  void CatHP::AddHP(int hp)  {  HealthPoint += hp;  if (HealthPoint > 100) {  HealthPoint = 100;  }  }  void CatHP::SubHP(int hp)  {  HealthPoint -= hp;  if (HealthPoint < 0) {  HealthPoint = 0;  }  }  bool CatHP::isDead()  {  return HealthPoint == 0;  }  void CatHP::Poisoning()  {  if (poisoningRound > 0)  {  SubHP(10);  poisoningRound = poisoningRound - 1;  }  }  void CatHP::SetPoisoning()  {  poisoningRound = 3;  }  } |
| CatHP.h |
| #pragma once  namespace game\_framework {  class CatHP  {  public:  CatHP();  void Initialize();  void LoadBitmap();  void OnShow();  void AddHP(int hp);  void SubHP(int hp);  bool isDead();  void Poisoning();  void SetPoisoning();  protected:  int HealthPoint;  int poisoningRound; // 紀錄中毒剩餘回合數，0代表沒有中毒  // 貓咪血量條圖片  CMovingBitmap CatHealthPointPicture[469];  };  } |
| Dog.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <ddraw.h>  #include "gamelib.h"  #include "GAMEID.h"  #include "Dog.h"  namespace game\_framework {  /////////////////////////////////////////////////////////////////////////////  // Dog: Dog class  /////////////////////////////////////////////////////////////////////////////  Dog::Dog()  {  DogPositionX = 1032;  DogPositionY = 634;  }  void Dog::Initialize()  {  Dog\_Normal.SetTopLeft(DogPositionX, DogPositionY);  Dog\_Normal\_BadlyHurt.SetTopLeft(DogPositionX, DogPositionY);  Dog\_Attack\_1.SetTopLeft(DogPositionX, DogPositionY);  Dog\_Attack\_2.SetTopLeft(DogPositionX, DogPositionY);  Dog\_Attack\_3.SetTopLeft(DogPositionX, DogPositionY);  Dog\_Attack\_4.SetTopLeft(DogPositionX, DogPositionY);  Dog\_BeingAttacked\_Big.SetTopLeft(DogPositionX, DogPositionY);  Dog\_BeingAttacked\_Little.SetTopLeft(DogPositionX, DogPositionY);  Dog\_BeingAttacked\_Miss.SetTopLeft(DogPositionX, DogPositionY);  }  void Dog::LoadBitmap()  {  Dog\_Normal.LoadBitmap("GamePicture/GameRun/Dog/Normal.bmp");  Dog\_Normal\_BadlyHurt.LoadBitmap("GamePicture/GameRun/Dog/Normal\_BadlyHurt.bmp");  Dog\_Attack\_1.LoadBitmap("GamePicture/GameRun/Dog/Attack\_1.bmp");  Dog\_Attack\_2.LoadBitmap("GamePicture/GameRun/Dog/Attack\_2.bmp");  Dog\_Attack\_3.LoadBitmap("GamePicture/GameRun/Dog/Attack\_3.bmp");  Dog\_Attack\_4.LoadBitmap("GamePicture/GameRun/Dog/Attack\_4.bmp");  Dog\_BeingAttacked\_Big.LoadBitmap("GamePicture/GameRun/Dog/BeingAttacked\_Big.bmp");  Dog\_BeingAttacked\_Little.LoadBitmap("GamePicture/GameRun/Dog/BeingAttacked\_Little.bmp");  Dog\_BeingAttacked\_Miss.LoadBitmap("GamePicture/GameRun/Dog/BeingAttacked\_Miss.bmp");  }  void Dog::OnMove(GAME\_RUN\_ID runId)  {  // 根據目前的條件改變狀態  }  void Dog::OnShow(PICTURE\_STATUS pictureStatus)  {  switch (pictureStatus)  {  case Normal:  Dog\_Normal.ShowBitmap();  break;  case Normal\_BadlyHurt:  Dog\_Normal\_BadlyHurt.ShowBitmap();  break;  case Attack\_1:  Dog\_Attack\_1.ShowBitmap();  break;  case Attack\_2:  Dog\_Attack\_2.ShowBitmap();  break;  case Attack\_3:  Dog\_Attack\_3.ShowBitmap();  break;  case Attack\_4:  Dog\_Attack\_4.ShowBitmap();  break;  case BeingAttacked\_Big:  Dog\_BeingAttacked\_Big.ShowBitmap();  break;  case BeingAttacked\_Little:  Dog\_BeingAttacked\_Little.ShowBitmap();  break;  case BeingAttacked\_Miss:  Dog\_BeingAttacked\_Miss.ShowBitmap();  break;  default:  break;  }  }  } |
| Dog.h |
| #pragma once  namespace game\_framework {  class Dog  {  public:  Dog();  void OnMove(GAME\_RUN\_ID runId); // 根據目前的條件改變狀態  void OnShow(PICTURE\_STATUS pictureStatus); // 根據目前的狀態貼上狗勾圖片  void Initialize(); // 初始化設定  void LoadBitmap();  protected:  int DogPositionX;  int DogPositionY;  // 貓貓的圖片  CMovingBitmap Dog\_Normal;  CMovingBitmap Dog\_Normal\_BadlyHurt;  CMovingBitmap Dog\_Attack\_1;  CMovingBitmap Dog\_Attack\_2;  CMovingBitmap Dog\_Attack\_3;  CMovingBitmap Dog\_Attack\_4;  CMovingBitmap Dog\_BeingAttacked\_Big;  CMovingBitmap Dog\_BeingAttacked\_Little;  CMovingBitmap Dog\_BeingAttacked\_Miss;  };  } |
| DogHP.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <ddraw.h>  #include "gamelib.h"  #include "GAMEID.h"  #include "DogHP.h"  namespace game\_framework {  /////////////////////////////////////////////////////////////////////////////  // DogHP: DogHP class  /////////////////////////////////////////////////////////////////////////////  DogHP::DogHP()  {  HealthPoint = 100;  poisoningRound = 0;  }  void DogHP::Initialize()  {  HealthPoint = 100;  for (int i = 0; i < 467; i++) {  DogHealthPointPicture[i].SetTopLeft(766 + i, 55);  }  }  void DogHP::LoadBitmap()  {  for (int i = 0; i < 467; i++) {  DogHealthPointPicture[i].LoadBitmap("GamePicture/GameRun/Dog/HP\_semicircle.bmp", RGB(180, 0, 255));  }  }  void DogHP::OnShow()  {  for (int i = 0; i < HealthPoint \* 467/100; i++) {  DogHealthPointPicture[i].ShowBitmap();  }  }  void DogHP::AddHP(int hp)  {  HealthPoint += hp;  if (HealthPoint > 100) {  HealthPoint = 100;  }  }  void DogHP::SubHP(int hp)  {  HealthPoint -= hp;  if (HealthPoint < 0) {  HealthPoint = 0;  }  }  bool DogHP::isDead()  {  return HealthPoint == 0;  }  void DogHP::Poisoning()  {  if (poisoningRound > 0)  {  SubHP(10);  poisoningRound = poisoningRound - 1;  }  }  void DogHP::SetPoisoning()  {  poisoningRound = 3;  }  } |
| DogHP.h |
| #pragma once  namespace game\_framework {  class DogHP  {  public:  DogHP();  void Initialize();  void LoadBitmap();  void OnShow();  void AddHP(int hp);  void SubHP(int hp);  bool isDead();  void Poisoning();  void SetPoisoning();  protected:  int HealthPoint;  int poisoningRound; // 紀錄中毒剩餘回合數，0代表沒有中毒  // 貓咪血量條圖片  CMovingBitmap DogHealthPointPicture[467];  };  } |
| GameData.h |
| #pragma once  namespace game\_framework {  class GameData  {  public:  static GAME\_LEVEL GameLevel;  static bool isDogWin;  static bool is2P;  };  } |
| GAMEID.h |
| #pragma once  namespace game\_framework {  /////////////////////////////////////////////////////////////////////////////  // Constants  /////////////////////////////////////////////////////////////////////////////  // 定義畫面的狀態  enum GAME\_PAGE\_ID {  ABOUT\_PICTURE,  START\_BUTTON\_PAGE\_NO\_HOVER,  START\_BUTTON\_PAGE\_HOVER,  FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER,  FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_HOW\_TO\_PLAY,  FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_LETS\_PLAY,  HOW\_TO\_PLAY\_PAGE\_NO\_HOVER,  HOW\_TO\_PLAY\_PAGE\_HOVER,  CHOOSE\_GAME\_PAGE\_NO\_HOVER,  CHOOSE\_GAME\_PAGE\_HOVER\_1PLAYER,  CHOOSE\_GAME\_PAGE\_HOVER\_2PLAYER,  SELECT\_YOUR\_LEVEL\_PAGE\_NO\_HOVER,  SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BEGINNER,  SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_AVERAGE,  SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BRING\_IT\_ON  };  // 定義遊玩中，現在是貓還是狗攻擊的狀態  enum GAME\_RUN\_ID {  CAT\_PREPARE,  CAT\_ATTACK\_TAKE,  CAT\_ATTACK\_BEGIN,  CAT\_ATTACK\_CHARGE,  CAT\_ATTACK\_FIRE,  CAT\_BEGIN\_ATTACKED\_MISS,  CAT\_BEGIN\_ATTACKED\_LITTLE,  CAT\_BEGIN\_ATTACKED\_BIG,  DOG\_PREPARE,  DOG\_ATTACK\_TAKE,  DOG\_ATTACK\_BEGIN,  DOG\_ATTACK\_CHARGE,  DOG\_ATTACK\_FIRE,  DOG\_BEGIN\_ATTACKED\_MISS,  DOG\_BEGIN\_ATTACKED\_LITTLE,  DOG\_BEGIN\_ATTACKED\_BIG  };  enum PICTURE\_STATUS {  Normal,  Normal\_BadlyHurt,  Attack\_1,  Attack\_2,  Attack\_3,  Attack\_4,  BeingAttacked\_Big,  BeingAttacked\_Little,  BeingAttacked\_Miss  };  enum class GAME\_LEVEL {  ImaBeginner,  ImAverage,  BringItOn  };  // 定義技能的狀態  enum SKILL\_STATUS {  IS\_NOT\_HOVER,  IS\_HOVER,  USED,  USING  };  // 定義各種音效的編號 0~15  enum AUDIO\_ID {  AUDIO\_SINGLE, // GameInit 開頭按鈕, 單人遊戲  AUDIO\_SINGLE\_SHORT, // GameInit 其他畫面按鈕, GameRun 玩家開始  AUDIO\_DOUBLE, // GameInit 雙人遊戲, GameOver 重完按鈕  AUDIO\_INIT\_MUSIC, // GameInit 開頭畫面音樂  AUDIO\_OVER\_MUSIC, // GameOver 遊戲結束音樂  AUDIO\_RUNSTART\_MUSIC, // GameRun 遊戲開始音樂  AUDIO\_COM\_SKILL\_HEAL, // 共用: 治癒技能  AUDIO\_CAT\_SKILL, // 貓: 技能  AUDIO\_CAT\_ATTACK, // 發射武器  AUDIO\_CAT\_ATTACKED\_BIG, // 受到重擊  AUDIO\_CAT\_ATTACKED\_LITTLE, // 受到輕擊  AUDIO\_CAT\_ATTACKED\_MISS, // 沒有擊中  AUDIO\_DOG\_SKILL, // 狗: 技能  AUDIO\_DOG\_ATTACK, // 發射武器  AUDIO\_DOG\_ATTACKED\_BIG, // 受到重擊  AUDIO\_DOG\_ATTACKED\_LITTLE, // 受到輕擊  AUDIO\_DOG\_ATTACKED\_MISS, // 沒有擊中  };  } |
| mygame.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <mmsystem.h>  #include <ddraw.h>  #include <cstdlib>  #include "audio.h"  #include "gamelib.h"  #include "mygame.h"  namespace game\_framework {  // 初始化遊戲難度  GAME\_LEVEL GameData::GameLevel = GAME\_LEVEL::ImaBeginner;  // 初始化狗狗輸贏  bool GameData::isDogWin = false;  // 初始化遊戲 1P 2P  bool GameData::is2P = false;  /////////////////////////////////////////////////////////////////////////////  // 這個class為遊戲的遊戲開頭畫面物件  /////////////////////////////////////////////////////////////////////////////  CGameStateInit::CGameStateInit(CGame\* g)  : CGameState(g)  {  }  void CGameStateInit::OnInit()  {  char buffer[100];  // 遊戲開始畫面音樂  gameStartMusicDidNotPlay = false;  // 初始化遊戲狀態，一開始進入到About畫面  CurrentPage = ABOUT\_PICTURE;  // 加載遊戲音效  CAudio::Instance()->Load(AUDIO\_SINGLE, "GameSFX/GameButtonSingle.mp3");  CAudio::Instance()->Load(AUDIO\_SINGLE\_SHORT, "GameSFX/GameButtonSingleShort.mp3");  CAudio::Instance()->Load(AUDIO\_DOUBLE, "GameSFX/GameButtonDouble.mp3");  CAudio::Instance()->Load(AUDIO\_INIT\_MUSIC, "GameSFX/GameInitMusic.mp3");  // 加載About介紹畫面  AboutPicture.LoadBitmap("GamePicture/About.bmp");  // 加載遊戲開始畫面圖片  // 遊戲開始畫面動畫  StartButtonBackground.SetDelayCount(4);  for (int i = 1; i <= 8; i++) {  std::string PicturePath = string("GamePicture/GameInit/StartButton/") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  StartButtonBackground.AddBitmap(buffer);  }  // 遊戲開始畫面  StartButton\_noHover.LoadBitmap("GamePicture/GameInit/StartButton\_noHover.bmp");  StartButton\_hover.LoadBitmap("GamePicture/GameInit/StartButton\_hover.bmp");  // 遊戲大綱畫面  FleabagVsMutt\_noHover.LoadBitmap("GamePicture/GameInit/FleabagVsMutt\_noHover.bmp");  FleabagVsMutt\_hoverHowToPlay.LoadBitmap("GamePicture/GameInit/FleabagVsMutt\_hoverHowToPlay.bmp");  FleabagVsMutt\_hoverLetsPlay.LoadBitmap("GamePicture/GameInit/FleabagVsMutt\_hoverLetsPlay.bmp");  FleabagVsMutt\_titleCat.SetDelayCount(4);  for (int i = 1; i <= 8; i++) {  std::string PicturePath = string("GamePicture/GameInit/FleabagVsMutt\_titleCat/") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  FleabagVsMutt\_titleCat.AddBitmap(buffer);  }  // 遊戲說明畫面  HowToPlay\_noHover.LoadBitmap("GamePicture/GameInit/HowToPlay\_noHover.bmp");  HowToPlay\_hover.LoadBitmap("GamePicture/GameInit/HowToPlay\_hover.bmp");  // 玩家人數選擇畫面  ChooseGame\_noHover.LoadBitmap("GamePicture/GameInit/ChooseGame\_noHover.bmp");  ChooseGame\_hover1P.LoadBitmap("GamePicture/GameInit/ChooseGame\_hover1P.bmp");  ChooseGame\_hover2P.LoadBitmap("GamePicture/GameInit/ChooseGame\_hover2P.bmp");  // 選擇遊戲難度頁面各難度動畫  SelectYourLevel\_hoverBeginnerAnimation.SetDelayCount(3);  SelectYourLevel\_hoverAverageAnimation.SetDelayCount(3);  SelectYourLevel\_hoverBringItOnAnimation.SetDelayCount(3);  for (int i = 1; i <= 4; i++) {  std::string PicturePath = string("GamePicture/GameInit/SelectYourLevel/1-") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  SelectYourLevel\_hoverBeginnerAnimation.AddBitmap(buffer);  PicturePath = string("GamePicture/GameInit/SelectYourLevel/2-") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  SelectYourLevel\_hoverAverageAnimation.AddBitmap(buffer);  }  for (int i = 1; i <= 7; i++) {  std::string PicturePath = string("GamePicture/GameInit/SelectYourLevel/3-") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  SelectYourLevel\_hoverBringItOnAnimation.AddBitmap(buffer);  }  // 選擇遊戲難度頁面  SelectYourLevelBackground.LoadBitmap("GamePicture/GameInit/SelectYourLevelBackground.bmp");  SelectYourLevel\_hoverBeginner.LoadBitmap("GamePicture/GameInit/SelectYourLevel\_hoverBeginner.bmp");  SelectYourLevel\_hoverAverage.LoadBitmap("GamePicture/GameInit/SelectYourLevel\_hoverAverage.bmp");  SelectYourLevel\_hoverBringItOn.LoadBitmap("GamePicture/GameInit/SelectYourLevel\_hoverBringItOn.bmp");  // 初始化所有圖片位置，因為都是整個畫面所以設置在(0, 0)位置  AboutPicture.SetTopLeft(0, 0);  StartButtonBackground.SetTopLeft(0, 0);  StartButton\_noHover.SetTopLeft(572, 778);  StartButton\_hover.SetTopLeft(572, 778);  FleabagVsMutt\_noHover.SetTopLeft(0, 0);  FleabagVsMutt\_hoverHowToPlay.SetTopLeft(0, 0);  FleabagVsMutt\_hoverLetsPlay.SetTopLeft(0, 0);  FleabagVsMutt\_titleCat.SetTopLeft(220, 52);  HowToPlay\_noHover.SetTopLeft(0, 0);  HowToPlay\_hover.SetTopLeft(0, 0);  ChooseGame\_noHover.SetTopLeft(0, 0);  ChooseGame\_hover1P.SetTopLeft(0, 0);  ChooseGame\_hover2P.SetTopLeft(0, 0);  SelectYourLevelBackground.SetTopLeft(0, 0);  SelectYourLevel\_hoverBeginner.SetTopLeft(0, 0);  SelectYourLevel\_hoverAverage.SetTopLeft(0, 0);  SelectYourLevel\_hoverBringItOn.SetTopLeft(0, 0);  SelectYourLevel\_hoverBeginnerAnimation.SetTopLeft(429, 298);  SelectYourLevel\_hoverAverageAnimation.SetTopLeft(423, 418);  SelectYourLevel\_hoverBringItOnAnimation.SetTopLeft(435, 548);  // 所有按鈕的CPoint 左上(LT)與右下(RB)  StartPage\_StartButtonLT = CPoint(436, 771);  StartPage\_StartButtonRB = CPoint(912, 839);  FleabagVsMuttPage\_HowToPlayButtonLT = CPoint(403, 559);  FleabagVsMuttPage\_HowToPlayButtonRB = CPoint(915, 635);  FleabagVsMuttPage\_LetsPlayButtonLT = CPoint(400, 709);  FleabagVsMuttPage\_LetsPlayButtonRB = CPoint(911, 788);  HowToPlayPage\_LetsPlayButtonLT = CPoint(400, 788);  HowToPlayPage\_LetsPlayButtonRB = CPoint(912, 866);  ChooseGamePage\_1PlayerButtonLT = CPoint(398, 350);  ChooseGamePage\_1PlayerButtonRB = CPoint(597, 563);  ChooseGamePage\_2PlayerButtonLT = CPoint(678, 364);  ChooseGamePage\_2PlayerButtonRB = CPoint(912, 563);  SelectYourLevelPage\_ImBeginnerButtonLT = CPoint(429, 305);  SelectYourLevelPage\_ImBeginnerButtonRB = CPoint(929, 409);  SelectYourLevelPage\_ImAverageButtonLT = CPoint(433, 426);  SelectYourLevelPage\_ImAverageButtonRB = CPoint(893, 539);  SelectYourLevelPage\_BringItOnButtonLT = CPoint(436, 555);  SelectYourLevelPage\_BringItOnButtonRB = CPoint(896, 647);  }  void CGameStateInit::OnMouseMove(UINT nFlags, CPoint point)  {  switch (CurrentPage)  {  case START\_BUTTON\_PAGE\_NO\_HOVER:  case START\_BUTTON\_PAGE\_HOVER:  if (point.x > StartPage\_StartButtonLT.x && point.x < StartPage\_StartButtonRB.x && point.y > StartPage\_StartButtonLT.y && point.y < StartPage\_StartButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE, false); // Start按鈕hover音效  CurrentPage = START\_BUTTON\_PAGE\_HOVER;  }  else  CurrentPage = START\_BUTTON\_PAGE\_NO\_HOVER;  break;  case FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER:  case FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_HOW\_TO\_PLAY:  case FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_LETS\_PLAY:  if (point.x > FleabagVsMuttPage\_HowToPlayButtonLT.x && point.x < FleabagVsMuttPage\_HowToPlayButtonRB.x && point.y > FleabagVsMuttPage\_HowToPlayButtonLT.y && point.y < FleabagVsMuttPage\_HowToPlayButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE\_SHORT, false); // How To Play按鈕hover音效  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_HOW\_TO\_PLAY;  }  else if (point.x > FleabagVsMuttPage\_LetsPlayButtonLT.x && point.x < FleabagVsMuttPage\_LetsPlayButtonRB.x && point.y > FleabagVsMuttPage\_LetsPlayButtonLT.y && point.y < FleabagVsMuttPage\_LetsPlayButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE\_SHORT, false); // Let's Play按鈕hover音效  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_LETS\_PLAY;  }  else  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER;  break;  case HOW\_TO\_PLAY\_PAGE\_NO\_HOVER:  case HOW\_TO\_PLAY\_PAGE\_HOVER:  if (point.x > HowToPlayPage\_LetsPlayButtonLT.x && point.x < HowToPlayPage\_LetsPlayButtonRB.x && point.y > HowToPlayPage\_LetsPlayButtonLT.y && point.y < HowToPlayPage\_LetsPlayButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE\_SHORT, false); // Let's Play按鈕hover音效  CurrentPage = HOW\_TO\_PLAY\_PAGE\_HOVER;  }  else  CurrentPage = HOW\_TO\_PLAY\_PAGE\_NO\_HOVER;  break;  case CHOOSE\_GAME\_PAGE\_NO\_HOVER:  case CHOOSE\_GAME\_PAGE\_HOVER\_1PLAYER:  case CHOOSE\_GAME\_PAGE\_HOVER\_2PLAYER:  if (point.x > ChooseGamePage\_1PlayerButtonLT.x && point.x < ChooseGamePage\_1PlayerButtonRB.x && point.y > ChooseGamePage\_1PlayerButtonLT.y && point.y < ChooseGamePage\_1PlayerButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE, false); // 1P按鈕hover音效  CurrentPage = CHOOSE\_GAME\_PAGE\_HOVER\_1PLAYER;  }  else if (point.x > ChooseGamePage\_2PlayerButtonLT.x && point.x < ChooseGamePage\_2PlayerButtonRB.x && point.y > ChooseGamePage\_2PlayerButtonLT.y && point.y < ChooseGamePage\_2PlayerButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_DOUBLE, false); // 2P按鈕hover音效  CurrentPage = CHOOSE\_GAME\_PAGE\_HOVER\_2PLAYER;  }  else  CurrentPage = CHOOSE\_GAME\_PAGE\_NO\_HOVER;  break;  case SELECT\_YOUR\_LEVEL\_PAGE\_NO\_HOVER:  case SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BEGINNER:  case SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_AVERAGE:  case SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BRING\_IT\_ON:  if (point.x > SelectYourLevelPage\_ImBeginnerButtonLT.x && point.x < SelectYourLevelPage\_ImBeginnerButtonRB.x && point.y > SelectYourLevelPage\_ImBeginnerButtonLT.y && point.y < SelectYourLevelPage\_ImBeginnerButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE, false); // 按鈕hover音效  CurrentPage = SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BEGINNER;  }  else if (point.x > SelectYourLevelPage\_ImAverageButtonLT.x && point.x < SelectYourLevelPage\_ImAverageButtonRB.x && point.y > SelectYourLevelPage\_ImAverageButtonLT.y && point.y < SelectYourLevelPage\_ImAverageButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE, false); // 按鈕hover音效  CurrentPage = SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_AVERAGE;  }  else if (point.x > SelectYourLevelPage\_BringItOnButtonLT.x && point.x < SelectYourLevelPage\_BringItOnButtonRB.x && point.y > SelectYourLevelPage\_BringItOnButtonLT.y && point.y < SelectYourLevelPage\_BringItOnButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_SINGLE, false); // 按鈕hover音效  CurrentPage = SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BRING\_IT\_ON;  }  else  CurrentPage = SELECT\_YOUR\_LEVEL\_PAGE\_NO\_HOVER;  break;  default:  break;  }  }  void CGameStateInit::OnLButtonUp(UINT nFlags, CPoint point)  {  switch (CurrentPage)  {  case game\_framework::ABOUT\_PICTURE: // About畫面  CurrentPage = START\_BUTTON\_PAGE\_NO\_HOVER;  gameStartMusicDidNotPlay = true; // 等一下進入遊戲開始畫面才會播放音樂  break;  case START\_BUTTON\_PAGE\_HOVER:  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER;  break;  case FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_HOW\_TO\_PLAY:  CurrentPage = HOW\_TO\_PLAY\_PAGE\_NO\_HOVER;  break;  case FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_LETS\_PLAY:  case HOW\_TO\_PLAY\_PAGE\_HOVER:  CurrentPage = CHOOSE\_GAME\_PAGE\_NO\_HOVER;  break;  case CHOOSE\_GAME\_PAGE\_HOVER\_1PLAYER:  GameData::is2P = false;  CurrentPage = SELECT\_YOUR\_LEVEL\_PAGE\_NO\_HOVER;  break;  case CHOOSE\_GAME\_PAGE\_HOVER\_2PLAYER:  GameData::is2P = true;  // 重置CurrentPage到FLEABAG\_VS\_MUTT頁面，遊戲結束按下reply會回到這個頁面。  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER;  GotoGameState(GAME\_STATE\_RUN);  break;  case SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BEGINNER:  // 重置CurrentPage到FLEABAG\_VS\_MUTT頁面，遊戲結束按下reply會回到這個頁面。  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER;  // 紀錄難度  GameData::GameLevel = GAME\_LEVEL::ImaBeginner;  GotoGameState(GAME\_STATE\_RUN);  break;  case SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_AVERAGE:  // 重置CurrentPage到FLEABAG\_VS\_MUTT頁面，遊戲結束按下reply會回到這個頁面。  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER;  // 紀錄難度  GameData::GameLevel = GAME\_LEVEL::ImAverage;  GotoGameState(GAME\_STATE\_RUN);  break;  case SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BRING\_IT\_ON:  // 重置CurrentPage到FLEABAG\_VS\_MUTT頁面，遊戲結束按下reply會回到這個頁面。  CurrentPage = FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER;  // 紀錄難度  GameData::GameLevel = GAME\_LEVEL::BringItOn;  GotoGameState(GAME\_STATE\_RUN);  break;  default:  break;  }  }  void CGameStateInit::OnMove()  {  // 遊戲開始畫面音樂  if (gameStartMusicDidNotPlay)  {  CAudio::Instance()->Play(AUDIO\_INIT\_MUSIC, false); // 遊戲開頭音樂  gameStartMusicDidNotPlay = false;  }  // 遊戲開始畫面動畫  switch (CurrentPage)  {  case game\_framework::START\_BUTTON\_PAGE\_NO\_HOVER:  case game\_framework::START\_BUTTON\_PAGE\_HOVER:  StartButtonBackground.OnMove();  break;  case game\_framework::FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER:  case game\_framework::FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_HOW\_TO\_PLAY:  case game\_framework::FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_LETS\_PLAY:  case game\_framework::HOW\_TO\_PLAY\_PAGE\_NO\_HOVER:  case game\_framework::HOW\_TO\_PLAY\_PAGE\_HOVER:  FleabagVsMutt\_titleCat.OnMove();  break;  case game\_framework::SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BEGINNER:  SelectYourLevel\_hoverBeginnerAnimation.OnMove();  break;  case game\_framework::SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_AVERAGE:  SelectYourLevel\_hoverAverageAnimation.OnMove();  break;  case game\_framework::SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BRING\_IT\_ON:  SelectYourLevel\_hoverBringItOnAnimation.OnMove();  break;  default:  break;  }  }  void CGameStateInit::OnShow()  {  switch (CurrentPage)  {  case game\_framework::ABOUT\_PICTURE: // About畫面  AboutPicture.ShowBitmap();  break;  case game\_framework::START\_BUTTON\_PAGE\_NO\_HOVER: // 遊戲開始畫面  StartButtonBackground.OnShow();  StartButton\_noHover.ShowBitmap();  break;  case game\_framework::START\_BUTTON\_PAGE\_HOVER:  StartButtonBackground.OnShow();  StartButton\_hover.ShowBitmap();  break;  case game\_framework::FLEABAG\_VS\_MUTT\_PAGE\_NO\_HOVER: // 遊戲大綱畫面  FleabagVsMutt\_noHover.ShowBitmap();  FleabagVsMutt\_titleCat.OnShow();  break;  case game\_framework::FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_HOW\_TO\_PLAY:  FleabagVsMutt\_hoverHowToPlay.ShowBitmap();  FleabagVsMutt\_titleCat.OnShow();  break;  case game\_framework::FLEABAG\_VS\_MUTT\_PAGE\_HOVER\_LETS\_PLAY:  FleabagVsMutt\_hoverLetsPlay.ShowBitmap();  FleabagVsMutt\_titleCat.OnShow();  break;  case game\_framework::HOW\_TO\_PLAY\_PAGE\_NO\_HOVER: // 遊戲說明畫面  HowToPlay\_noHover.ShowBitmap();  FleabagVsMutt\_titleCat.OnShow();  break;  case game\_framework::HOW\_TO\_PLAY\_PAGE\_HOVER:  HowToPlay\_hover.ShowBitmap();  FleabagVsMutt\_titleCat.OnShow();  break;  case game\_framework::CHOOSE\_GAME\_PAGE\_NO\_HOVER: // 玩家人數選擇畫面  ChooseGame\_noHover.ShowBitmap();  break;  case game\_framework::CHOOSE\_GAME\_PAGE\_HOVER\_1PLAYER:  ChooseGame\_hover1P.ShowBitmap();  break;  case game\_framework::CHOOSE\_GAME\_PAGE\_HOVER\_2PLAYER:  ChooseGame\_hover2P.ShowBitmap();  break;  case game\_framework::SELECT\_YOUR\_LEVEL\_PAGE\_NO\_HOVER: // 選擇遊戲難度頁面  SelectYourLevelBackground.ShowBitmap();  break;  case game\_framework::SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BEGINNER:  SelectYourLevel\_hoverBeginner.ShowBitmap();  SelectYourLevel\_hoverBeginnerAnimation.OnShow();  break;  case game\_framework::SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_AVERAGE:  SelectYourLevel\_hoverAverage.ShowBitmap();  SelectYourLevel\_hoverAverageAnimation.OnShow();  break;  case game\_framework::SELECT\_YOUR\_LEVEL\_PAGE\_HOVER\_BRING\_IT\_ON:  SelectYourLevel\_hoverBringItOn.ShowBitmap();  SelectYourLevel\_hoverBringItOnAnimation.OnShow();  break;  default:  break;  }  }  /////////////////////////////////////////////////////////////////////////////  // 這個class為遊戲的結束狀態(Game Over)  /////////////////////////////////////////////////////////////////////////////  void CGameStateOver::OnMove()  {  // 撥放音樂  if (gameOverMusicHadNotPlay)  {  CAudio::Instance()->Play(AUDIO\_OVER\_MUSIC, false); // 遊戲結束音樂  gameOverMusicHadNotPlay = false;  }  if (GameData::isDogWin) {  DogWin\_catAnimation.OnMove();  DogWin\_dogAnimation.OnMove();  }  else {  CatWin\_titleAnimation.OnMove();  CatWin\_catAnimation.OnMove();  CatWin\_dogAnimation.OnMove();  }  }  void CGameStateOver::OnBeginState()  {  // 遊戲開始畫面音樂  gameOverMusicHadNotPlay = true;  }  void CGameStateOver::OnInit()  {  char buffer[100];  isHoverReplay = false;  // 遊戲開始畫面音樂  gameOverMusicHadNotPlay = true;  // 加載遊戲音效  CAudio::Instance()->Load(AUDIO\_OVER\_MUSIC, "GameSFX/GameOverMusic.mp3");  // 貓咪贏  CatWin\_background.LoadBitmap("GamePicture/GameOver/CatWin/background.bmp");  CatWin\_replayHover.LoadBitmap("GamePicture/GameOver/CatWin/replayHover.bmp");  // 載入遊戲結束動畫  CatWin\_titleAnimation.SetDelayCount(3);  CatWin\_catAnimation.SetDelayCount(2);  CatWin\_dogAnimation.SetDelayCount(3);  for (int i = 1; i <= 6; i++) {  std::string PicturePath = string("GamePicture/GameOver/CatWin/Title/") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  CatWin\_titleAnimation.AddBitmap(buffer);  }  for (int i = 1; i <= 10; i++) {  std::string PicturePath = string("GamePicture/GameOver/CatWin/Cat/") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  CatWin\_catAnimation.AddBitmap(buffer);  }  for (int i = 1; i <= 4; i++) {  std::string PicturePath = string("GamePicture/GameOver/CatWin/Dog/") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  CatWin\_dogAnimation.AddBitmap(buffer);  }  // 狗溝贏  DogWin\_background.LoadBitmap("GamePicture/GameOver/DogWin/background.bmp");  DogWin\_replayHover.LoadBitmap("GamePicture/GameOver/DogWin/replayHover.bmp");  // 載入遊戲結束動畫  DogWin\_catAnimation.SetDelayCount(3);  DogWin\_dogAnimation.SetDelayCount(4);  for (int i = 1; i <= 10; i++) {  std::string PicturePath = string("GamePicture/GameOver/DogWin/Cat/") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  DogWin\_catAnimation.AddBitmap(buffer);  }  for (int i = 1; i <= 6; i++) {  std::string PicturePath = string("GamePicture/GameOver/DogWin/Dog/") + std::to\_string(i) + ".bmp";  std::sprintf(buffer, "%s", (PicturePath.c\_str()));  DogWin\_dogAnimation.AddBitmap(buffer);  }  // 初始化所有圖片位置  CatWin\_background.SetTopLeft(0, 0);  CatWin\_replayHover.SetTopLeft(585, 455);  CatWin\_titleAnimation.SetTopLeft(417, 310);  CatWin\_catAnimation.SetTopLeft(0, 501);  CatWin\_dogAnimation.SetTopLeft(999, 669);  DogWin\_background.SetTopLeft(0, 0);  DogWin\_replayHover.SetTopLeft(584, 449);  DogWin\_catAnimation.SetTopLeft(0, 612);  DogWin\_dogAnimation.SetTopLeft(1030, 620);  // replay按鈕的CPoint 左上與右下  CatWin\_ReplayButtonLT = CPoint(585, 455);  CatWin\_ReplayButtonRB = CPoint(769, 507);  DogWin\_ReplayButtonLT = CPoint(584, 449);  DogWin\_ReplayButtonRB = CPoint(767, 505);  }  void CGameStateOver::OnMouseMove(UINT nFlags, CPoint point)  {  if (GameData::isDogWin) {  if (point.x > DogWin\_ReplayButtonLT.x && point.x < DogWin\_ReplayButtonRB.x && point.y > DogWin\_ReplayButtonLT.y && point.y < DogWin\_ReplayButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_DOUBLE, false); // Replay按鈕hover音效  isHoverReplay = true;  }  else  isHoverReplay = false;  }  else {  if (point.x > CatWin\_ReplayButtonLT.x && point.x < CatWin\_ReplayButtonRB.x && point.y > CatWin\_ReplayButtonLT.y && point.y < CatWin\_ReplayButtonRB.y)  {  CAudio::Instance()->Play(AUDIO\_DOUBLE, false); // Replay按鈕hover音效  isHoverReplay = true;  }  else  isHoverReplay = false;  }  }  void CGameStateOver::OnLButtonUp(UINT nFlags, CPoint point)  {  if (isHoverReplay)  {  CAudio::Instance()->Stop(AUDIO\_OVER\_MUSIC); // 停止遊戲結束音樂  GotoGameState(GAME\_STATE\_INIT);  }  }  void CGameStateOver::OnShow()  {  if (GameData::isDogWin) {  DogWin\_background.ShowBitmap();  DogWin\_catAnimation.OnShow();  DogWin\_dogAnimation.OnShow();  if (isHoverReplay)  DogWin\_replayHover.ShowBitmap();  }  else {  CatWin\_background.ShowBitmap();  CatWin\_titleAnimation.OnShow();  CatWin\_catAnimation.OnShow();  CatWin\_dogAnimation.OnShow();  if (isHoverReplay)  CatWin\_replayHover.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  /////////////////////////////////////////////////////////////////////////////  void CGameStateRun::OnBeginState()  {  // 每次進到CGameStateRun都會跑一次  // 攻擊方音效  catAttackIsNotPlaying = true;  dogAttackIsNotPlaying = true;  gameRunMusicHadNotPlay = true;  // 初始化貓咪跟狗狗武器物件  CatObject.Initialize();  DogObject.Initialize();  WeaponObject.Initialize();  // 初始化貓咪跟狗狗血量為100%  CatHealthPointBar.Initialize();  DogHealthPointBar.Initialize();  // 一開始由狗先行攻擊  runId = DOG\_PREPARE;  // 初始化計時器  Timer = 0;  // 初始化技能狀態  // 如果是1P 貓咪設定成沒有技能可用  if (GameData::is2P)  {  catSkillDoubleAttackStatus = IS\_NOT\_HOVER;  catSkillPowerAttackStatus = IS\_NOT\_HOVER;  catSkillPoisonGasStatus = IS\_NOT\_HOVER;  catSkillOKBandStatus = IS\_NOT\_HOVER;  }  else  {  catSkillDoubleAttackStatus = USED;  catSkillPowerAttackStatus = USED;  catSkillPoisonGasStatus = USED;  catSkillOKBandStatus = USED;  }  dogSkillDoubleAttackStatus = IS\_NOT\_HOVER;  dogSkillPowerAttackStatus = IS\_NOT\_HOVER;  dogSkillPoisonGasStatus = IS\_NOT\_HOVER;  dogSkillOKBandStatus = IS\_NOT\_HOVER;  }  void CGameStateRun::OnMove() // 移動遊戲元素  {  // 撥放音樂  if (gameRunMusicHadNotPlay)  {  CAudio::Instance()->Play(AUDIO\_RUNSTART\_MUSIC, false); // 遊戲回合開始音樂  gameRunMusicHadNotPlay = false;  }  switch (runId)  {  case CAT\_PREPARE:  Timer++;  if (Timer >= 30) {  CatHealthPointBar.Poisoning(); // 處理中毒  runId = CAT\_ATTACK\_TAKE;  Timer = 0;  }  break;  case CAT\_ATTACK\_TAKE:  Timer++;  if (Timer >= 25) {  runId = CAT\_ATTACK\_BEGIN;  Timer = 0;  }  break;  case CAT\_ATTACK\_BEGIN:  if (catAttackIsNotPlaying)  {  CAudio::Instance()->Play(AUDIO\_SINGLE, false); // 輪到攻擊方音效  catAttackIsNotPlaying = false;  }  if (GameData::is2P)  {  CatAttackInvertedTriangle.OnMove();  }  else  {  Timer ++;  if (Timer >= 20) {  runId = CAT\_ATTACK\_CHARGE;  Timer = 0;  }  }  break;  case CAT\_ATTACK\_CHARGE:  catAttackIsNotPlaying = true; // 重置音效  Timer++;  // 1P的時候 貓咪由電腦產生力道  if (Timer >= 20 && GameData::is2P == false) {  int max = 0;  int min = 0;  // 難度每升一級就將攻擊範圍縮小12  switch (GameData::GameLevel)  {  case GAME\_LEVEL::ImaBeginner:  // 命中率 3/25 == 12%  max = 27;  min = 3;  break;  case GAME\_LEVEL::ImAverage:  // 命中率 3/15 == 20%  max = 22;  min = 8;  break;  case GAME\_LEVEL::BringItOn:  // 命中率 3/5 == 60%  max = 17;  min = 13;  break;  default:  break;  }  int rand\_power = rand() % (max - min + 1) + min;  WeaponObject.CatFire(rand\_power);  runId = CAT\_ATTACK\_FIRE;  Timer = 0;  }  break;  case CAT\_ATTACK\_FIRE:  if (WeaponObject.isHitDog() == 1) {  CAudio::Instance()->Play(AUDIO\_DOG\_ATTACKED\_LITTLE, false); // 狗被輕擊音效  runId = DOG\_BEGIN\_ATTACKED\_LITTLE;  Timer = 0;  if (catSkillPoisonGasStatus == USING)  {  DogHealthPointBar.SetPoisoning();  DogHealthPointBar.SubHP(5);  }  else  {  DogHealthPointBar.SubHP(10);  }  }  else if (WeaponObject.isHitDog() == 2) {  CAudio::Instance()->Play(AUDIO\_DOG\_ATTACKED\_BIG, false); // 狗被重擊音效  runId = DOG\_BEGIN\_ATTACKED\_BIG;  Timer = 0;  if (catSkillPoisonGasStatus == USING)  {  DogHealthPointBar.SetPoisoning();  DogHealthPointBar.SubHP(15);  }  else  {  DogHealthPointBar.SubHP(25);  }  }  else if (WeaponObject.isHitGround()) {  CAudio::Instance()->Play(AUDIO\_DOG\_ATTACKED\_MISS, false); // 狗沒被擊中音效  runId = DOG\_BEGIN\_ATTACKED\_MISS;  Timer = 0;  }  if (DogHealthPointBar.isDead())  {  GameData::isDogWin = false;  GotoGameState(GAME\_STATE\_OVER);  }  break;  case CAT\_BEGIN\_ATTACKED\_LITTLE:  case CAT\_BEGIN\_ATTACKED\_BIG:  case CAT\_BEGIN\_ATTACKED\_MISS:  Timer++;  if (Timer > 30)  {  // 連擊  if (dogSkillDoubleAttackStatus == USING)  {  runId = DOG\_ATTACK\_FIRE;  WeaponObject.DogFire(lastPower);  dogSkillDoubleAttackStatus = USED;  }  else  runId = CAT\_PREPARE;  // 強力攻擊  if (dogSkillPowerAttackStatus == USING)  {  WeaponObject.setSize(1);  dogSkillPowerAttackStatus = USED;  }  // 中毒  if (dogSkillPoisonGasStatus == USING)  {  WeaponObject.setNormalWeapon(true);  dogSkillPoisonGasStatus = USED;  }  }  break;  case DOG\_PREPARE:  Timer++;  if (Timer >= 30) {  DogHealthPointBar.Poisoning(); // 處理中毒  runId = DOG\_ATTACK\_TAKE;  Timer = 0;  }  break;  case DOG\_ATTACK\_TAKE:  Timer++;  if (Timer >= 25) {  runId = DOG\_ATTACK\_BEGIN;  Timer = 0;  }  break;  case DOG\_ATTACK\_BEGIN:  if (dogAttackIsNotPlaying)  {  CAudio::Instance()->Play(AUDIO\_SINGLE, false); // 輪到攻擊方音效  dogAttackIsNotPlaying = false;  }  DogAttackInvertedTriangle.OnMove();  break;  case DOG\_ATTACK\_CHARGE:  dogAttackIsNotPlaying = true; // 重置音效  Timer++;  break;  case DOG\_ATTACK\_FIRE:  if (WeaponObject.isHitCat() == 1) {  CAudio::Instance()->Play(AUDIO\_CAT\_ATTACKED\_LITTLE, false); // 貓被輕擊音效  runId = CAT\_BEGIN\_ATTACKED\_LITTLE;  Timer = 0;  if (dogSkillPoisonGasStatus == USING)  {  CatHealthPointBar.SetPoisoning();  CatHealthPointBar.SubHP(5);  }  else  {  CatHealthPointBar.SubHP(10);  }  }  else if (WeaponObject.isHitCat() == 2) {  CAudio::Instance()->Play(AUDIO\_CAT\_ATTACKED\_BIG, false); // 貓被重擊音效  runId = CAT\_BEGIN\_ATTACKED\_BIG;  Timer = 0;  if (dogSkillPoisonGasStatus == USING)  {  CatHealthPointBar.SetPoisoning();  CatHealthPointBar.SubHP(15);  }  else  {  CatHealthPointBar.SubHP(25);  }  }  else if (WeaponObject.isHitGround()) {  CAudio::Instance()->Play(AUDIO\_CAT\_ATTACKED\_MISS, false); // 狗沒被擊中音效  runId = CAT\_BEGIN\_ATTACKED\_MISS;  Timer = 0;  }  if (CatHealthPointBar.isDead())  {  GameData::isDogWin = true;  GotoGameState(GAME\_STATE\_OVER);  }  break;  case DOG\_BEGIN\_ATTACKED\_LITTLE:  case DOG\_BEGIN\_ATTACKED\_BIG:  case DOG\_BEGIN\_ATTACKED\_MISS:  Timer++;  if (Timer > 30)  {  // 連擊  if (catSkillDoubleAttackStatus == USING)  {  runId = CAT\_ATTACK\_FIRE;  WeaponObject.CatFire(lastPower);  catSkillDoubleAttackStatus = USED;  }  else  runId = DOG\_PREPARE;  // 強力攻擊  if (catSkillPowerAttackStatus == USING)  {  WeaponObject.setSize(1);  catSkillPowerAttackStatus = USED;  }  // 中毒  if (catSkillPoisonGasStatus == USING)  {  WeaponObject.setNormalWeapon(true);  catSkillPoisonGasStatus = USED;  }  }  break;  default:  break;  }  // Weapon位移運算  WeaponObject.OnMove(runId);  // 技能hover動畫撥放  if (catSkillPowerAttackStatus == IS\_HOVER)  // 正常播放動畫  catSkillPowerAttackHoverAnimation.OnMove();  else if (catSkillPowerAttackStatus == IS\_NOT\_HOVER)  // 停止撥放動畫  catSkillPowerAttackHoverAnimation.Reset();  // 毒氣  if (catSkillPoisonGasStatus == IS\_HOVER)  catSkillPoisonGasHoverAnimation.OnMove();  else if (catSkillPoisonGasStatus == IS\_NOT\_HOVER)  catSkillPoisonGasHoverAnimation.Reset();  // 治癒  if (catSkillOKBandStatus == IS\_HOVER)  catSkillOKBandHoverAnimation.OnMove();  else if (catSkillOKBandStatus == IS\_NOT\_HOVER)  catSkillOKBandHoverAnimation.Reset();  // Dog skill  // 重擊  if (dogSkillPowerAttackStatus == IS\_HOVER)  dogSkillPowerAttackHoverAnimation.OnMove();  else if (dogSkillPowerAttackStatus == IS\_NOT\_HOVER)  dogSkillPowerAttackHoverAnimation.Reset();  // 毒氣  if (dogSkillPoisonGasStatus == IS\_HOVER)  dogSkillPoisonGasHoverAnimation.OnMove();  else if (dogSkillPoisonGasStatus == IS\_NOT\_HOVER)  dogSkillPoisonGasHoverAnimation.Reset();  // 治癒  if (dogSkillOKBandStatus == IS\_HOVER)  dogSkillOKBandHoverAnimation.OnMove();  else if (dogSkillOKBandStatus == IS\_NOT\_HOVER)  dogSkillOKBandHoverAnimation.Reset();  }  void CGameStateRun::OnInit()  {  char num\_char[100];  // 攻擊方音效  catAttackIsNotPlaying = true;  dogAttackIsNotPlaying = true;  gameRunMusicHadNotPlay = true;  // 加載遊戲音效  CAudio::Instance()->Load(AUDIO\_RUNSTART\_MUSIC, "GameSFX/GameRunStartMusic.mp3");  CAudio::Instance()->Load(AUDIO\_COM\_SKILL\_HEAL, "GameSFX/SkillHeal.mp3");  CAudio::Instance()->Load(AUDIO\_CAT\_SKILL, "GameSFX/Cat/Skill.mp3");  CAudio::Instance()->Load(AUDIO\_CAT\_ATTACK, "GameSFX/Cat/Attack.mp3");  CAudio::Instance()->Load(AUDIO\_CAT\_ATTACKED\_BIG, "GameSFX/Cat/BeingAttacked\_Big.mp3");  CAudio::Instance()->Load(AUDIO\_CAT\_ATTACKED\_LITTLE, "GameSFX/Cat/BeingAttacked\_Little.mp3");  CAudio::Instance()->Load(AUDIO\_CAT\_ATTACKED\_MISS, "GameSFX/Cat/BeingAttacked\_Miss.mp3");  CAudio::Instance()->Load(AUDIO\_DOG\_SKILL, "GameSFX/Dog/Skill.mp3");  CAudio::Instance()->Load(AUDIO\_DOG\_ATTACK, "GameSFX/Dog/Attack.mp3");  CAudio::Instance()->Load(AUDIO\_DOG\_ATTACKED\_BIG, "GameSFX/Dog/BeingAttacked\_Big.mp3");  CAudio::Instance()->Load(AUDIO\_DOG\_ATTACKED\_LITTLE, "GameSFX/Dog/BeingAttacked\_Little.mp3");  CAudio::Instance()->Load(AUDIO\_DOG\_ATTACKED\_MISS, "GameSFX/Dog/BeingAttacked\_Miss.mp3");  // 啟動遊戲時進行遊戲圖形載入  // 貓狗攻擊時頭上的倒三角形動畫載入  CatAttackInvertedTriangle.SetDelayCount(4);  for (int i = 1; i <= 4; i++) {  std::string PicturePath = string("GamePicture/GameRun/Cat/AttackInvertedTriangle\_") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  CatAttackInvertedTriangle.AddBitmap(num\_char);  }  DogAttackInvertedTriangle.SetDelayCount(4);  for (int i = 1; i <= 4; i++) {  std::string PicturePath = string("GamePicture/GameRun/Dog/AttackInvertedTriangle\_") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  DogAttackInvertedTriangle.AddBitmap(num\_char);  }  // 載入所有戰鬥中的圖片  // 背景  Background.LoadBitmap("GamePicture/GameRun/Background.bmp");  CatObject.LoadBitmap();  DogObject.LoadBitmap();  WeaponObject.LoadBitmap();  // 血量條  CatHealthPointBar.LoadBitmap();  DogHealthPointBar.LoadBitmap();  // 技能按鈕圖片  catSkillDoubleAttackButton.LoadBitmap("GamePicture/GameRun/Cat/Skill/DoubleAttack.bmp");  dogSkillDoubleAttackButton.LoadBitmap("GamePicture/GameRun/Dog/Skill/DoubleAttack.bmp");  // 技能按鈕hover動畫  // 貓技能  catSkillPowerAttackHoverAnimation.SetDelayCount(4);  for (int i = 1; i <= 6; i++) {  std::string PicturePath = string("GamePicture/GameRun/Cat/Skill/PowerAttack-") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  catSkillPowerAttackHoverAnimation.AddBitmap(num\_char);  }  catSkillPoisonGasHoverAnimation.SetDelayCount(4);  for (int i = 1; i <= 6; i++) {  std::string PicturePath = string("GamePicture/GameRun/Cat/Skill/PoisonGas-") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  catSkillPoisonGasHoverAnimation.AddBitmap(num\_char);  }  catSkillOKBandHoverAnimation.SetDelayCount(4);  for (int i = 1; i <= 4; i++) {  std::string PicturePath = string("GamePicture/GameRun/Cat/Skill/OKBand-") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  catSkillOKBandHoverAnimation.AddBitmap(num\_char);  }  // 狗技能  dogSkillPowerAttackHoverAnimation.SetDelayCount(4);  for (int i = 1; i <= 4; i++) {  std::string PicturePath = string("GamePicture/GameRun/Dog/Skill/PowerAttack-") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  dogSkillPowerAttackHoverAnimation.AddBitmap(num\_char);  }  dogSkillPoisonGasHoverAnimation.SetDelayCount(4);  for (int i = 1; i <= 4; i++) {  std::string PicturePath = string("GamePicture/GameRun/Dog/Skill/PoisonGas-") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  dogSkillPoisonGasHoverAnimation.AddBitmap(num\_char);  }  dogSkillOKBandHoverAnimation.SetDelayCount(4);  for (int i = 1; i <= 6; i++) {  std::string PicturePath = string("GamePicture/GameRun/Dog/Skill/OKBand-") + std::to\_string(i) + ".bmp";  std::sprintf(num\_char, "%s", (PicturePath.c\_str()));  dogSkillOKBandHoverAnimation.AddBitmap(num\_char);  }  // 初始化所有圖片位置  // 攻擊倒三角  CatAttackInvertedTriangle.SetTopLeft(158, 474);  DogAttackInvertedTriangle.SetTopLeft(1175, 546);  // 技能按鈕  catSkillDoubleAttackButton.SetTopLeft(123, 94);  catSkillPowerAttackHoverAnimation.SetTopLeft(217, 97);  catSkillPoisonGasHoverAnimation.SetTopLeft(310, 95);  catSkillOKBandHoverAnimation.SetTopLeft(401, 95);  dogSkillDoubleAttackButton.SetTopLeft(846, 97);  dogSkillPowerAttackHoverAnimation.SetTopLeft(939, 96);  dogSkillPoisonGasHoverAnimation.SetTopLeft(1033, 96);  dogSkillOKBandHoverAnimation.SetTopLeft(1122, 97);  // 所有技能按鈕的CPoint 左上(LT)與右下(RB)  catSkillDoubleAttackButtonLT = CPoint(123, 94);  catSkillDoubleAttackButtonRB = CPoint(204, 172);  catSkillPowerAttackButtonLT = CPoint(217, 97);  catSkillPowerAttackButtonRB = CPoint(298, 169);  catSkillPoisonGasButtonLT = CPoint(310, 95);  catSkillPoisonGasButtonRB = CPoint(388, 170);  catSkillOKBandButtonLT = CPoint(401, 95);  catSkillOKBandButtonRB = CPoint(479, 169);  dogSkillDoubleAttackButtonLT = CPoint(846, 97);  dogSkillDoubleAttackButtonRB = CPoint(926, 169);  dogSkillPowerAttackButtonLT = CPoint(939, 96);  dogSkillPowerAttackButtonRB = CPoint(1021, 170);  dogSkillPoisonGasButtonLT = CPoint(1033, 96);  dogSkillPoisonGasButtonRB = CPoint(1114, 173);  dogSkillOKBandButtonLT = CPoint(1122, 97);  dogSkillOKBandButtonRB = CPoint(1200, 172);  }  void CGameStateRun::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)  {  const char KEY\_C = 0x43;  const char KEY\_D = 0x44;  // debug專用 按C, D可以扣貓狗一滴血，  if (nChar == KEY\_C)  {  CatHealthPointBar.SubHP(100);  }  if (nChar == KEY\_D)  {  DogHealthPointBar.SubHP(100);  }  }  void CGameStateRun::OnLButtonDown(UINT nFlags, CPoint point) // 處理滑鼠的動作  {  // 技能點擊  if (runId == CAT\_ATTACK\_BEGIN)  {  bool isNotUsingSKill = (catSkillDoubleAttackStatus != USING && catSkillPowerAttackStatus != USING && catSkillPoisonGasStatus != USING && catSkillOKBandStatus != USING);  // 連擊  if (catSkillDoubleAttackStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_CAT\_SKILL, false); // Skill按鈕按下音效  catSkillDoubleAttackStatus = USING;  }  // 重擊  else if (catSkillPowerAttackStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_CAT\_SKILL, false); // Skill按鈕按下音效  WeaponObject.setSize(2);  catSkillPowerAttackStatus = USING;  }  // 毒氣  else if (catSkillPoisonGasStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_CAT\_SKILL, false); // Skill按鈕按下音效  WeaponObject.setNormalWeapon(false);  catSkillPoisonGasStatus = USING;  }  // 治癒  else if (catSkillOKBandStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_COM\_SKILL\_HEAL, false); // Skill治癒按鈕按下音效  CatHealthPointBar.AddHP(30);  catSkillOKBandStatus = USING;  runId = DOG\_PREPARE;  }  // 正常攻擊蓄力  else if (GameData::is2P)  {  runId = CAT\_ATTACK\_CHARGE;  Timer = 0;  }  }  else if (runId == DOG\_ATTACK\_BEGIN)  {  bool isNotUsingSKill = (dogSkillDoubleAttackStatus != USING && dogSkillPowerAttackStatus != USING && dogSkillPoisonGasStatus != USING && dogSkillOKBandStatus != USING);  // 連擊  if (dogSkillDoubleAttackStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_DOG\_SKILL, false); // Skill按鈕按下音效  dogSkillDoubleAttackStatus = USING;  }  // 重擊  else if (dogSkillPowerAttackStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_DOG\_SKILL, false); // Skill按鈕按下音效  WeaponObject.setSize(2);  dogSkillPowerAttackStatus = USING;  }  // 毒氣  else if (dogSkillPoisonGasStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_DOG\_SKILL, false); // Skill按鈕按下音效  WeaponObject.setNormalWeapon(false);  dogSkillPoisonGasStatus = USING;  }  // 治癒  else if (dogSkillOKBandStatus == IS\_HOVER && isNotUsingSKill)  {  CAudio::Instance()->Play(AUDIO\_COM\_SKILL\_HEAL, false); // Skill治癒按鈕按下音效  DogHealthPointBar.AddHP(30);  dogSkillOKBandStatus = USED;  runId = CAT\_PREPARE;  }  // 正常攻擊蓄力  else  {  runId = DOG\_ATTACK\_CHARGE;  Timer = 0;  }  }  }  void CGameStateRun::OnLButtonUp(UINT nFlags, CPoint point) // 處理滑鼠的動作  {  if (runId == DOG\_ATTACK\_CHARGE) {  CAudio::Instance()->Play(AUDIO\_DOG\_ATTACK, false); // 丟武器音效  runId = DOG\_ATTACK\_FIRE;  WeaponObject.DogFire(Timer);  if (dogSkillDoubleAttackStatus == USING)  lastPower = Timer;  }  // 雙人遊戲時貓咪才可以手動蓄力  else if (runId == CAT\_ATTACK\_CHARGE && GameData::is2P) {  CAudio::Instance()->Play(AUDIO\_CAT\_ATTACK, false); // 丟武器音效  runId = CAT\_ATTACK\_FIRE;  WeaponObject.CatFire(Timer);  if (catSkillDoubleAttackStatus == USING)  lastPower = Timer;  }  }  void CGameStateRun::OnMouseMove(UINT nFlags, CPoint point)  {  // Cat skill  // 連擊  if (catSkillDoubleAttackStatus != USED && catSkillDoubleAttackStatus != USING)  {  if (point.x > catSkillDoubleAttackButtonLT.x && point.x < catSkillDoubleAttackButtonRB.x && point.y > catSkillDoubleAttackButtonLT.y && point.y < catSkillDoubleAttackButtonRB.y)  catSkillDoubleAttackStatus = IS\_HOVER;  else  catSkillDoubleAttackStatus = IS\_NOT\_HOVER;  }  // 重擊  if (catSkillPowerAttackStatus != USED && catSkillPowerAttackStatus != USING)  {  if (point.x > catSkillPowerAttackButtonLT.x && point.x < catSkillPowerAttackButtonRB.x && point.y > catSkillPowerAttackButtonLT.y && point.y < catSkillPowerAttackButtonRB.y)  catSkillPowerAttackStatus = IS\_HOVER;  else  catSkillPowerAttackStatus = IS\_NOT\_HOVER;  }  // 毒氣  if (catSkillPoisonGasStatus != USED && catSkillPoisonGasStatus != USING)  {  if (point.x > catSkillPoisonGasButtonLT.x && point.x < catSkillPoisonGasButtonRB.x && point.y > catSkillPoisonGasButtonLT.y && point.y < catSkillPoisonGasButtonRB.y)  catSkillPoisonGasStatus = IS\_HOVER;  else  catSkillPoisonGasStatus = IS\_NOT\_HOVER;  }  // 治癒  if (catSkillOKBandStatus != USED && catSkillOKBandStatus != USING)  {  if (point.x > catSkillOKBandButtonLT.x && point.x < catSkillOKBandButtonRB.x && point.y > catSkillOKBandButtonLT.y && point.y < catSkillOKBandButtonRB.y)  catSkillOKBandStatus = IS\_HOVER;  else  catSkillOKBandStatus = IS\_NOT\_HOVER;  }  // Dog skill  // 連擊  if (dogSkillDoubleAttackStatus != USED && dogSkillDoubleAttackStatus != USING)  {  if (point.x > dogSkillDoubleAttackButtonLT.x && point.x < dogSkillDoubleAttackButtonRB.x && point.y > dogSkillDoubleAttackButtonLT.y && point.y < dogSkillDoubleAttackButtonRB.y)  dogSkillDoubleAttackStatus = IS\_HOVER;  else  dogSkillDoubleAttackStatus = IS\_NOT\_HOVER;  }  // 重擊  if (dogSkillPowerAttackStatus != USED && dogSkillPowerAttackStatus != USING)  {  if (point.x > dogSkillPowerAttackButtonLT.x && point.x < dogSkillPowerAttackButtonRB.x && point.y > dogSkillPowerAttackButtonLT.y && point.y < dogSkillPowerAttackButtonRB.y)  dogSkillPowerAttackStatus = IS\_HOVER;  else  dogSkillPowerAttackStatus = IS\_NOT\_HOVER;  }  // 毒氣  if (dogSkillPoisonGasStatus != USED && dogSkillPoisonGasStatus != USING)  {  if (point.x > dogSkillPoisonGasButtonLT.x && point.x < dogSkillPoisonGasButtonRB.x && point.y > dogSkillPoisonGasButtonLT.y && point.y < dogSkillPoisonGasButtonRB.y)  dogSkillPoisonGasStatus = IS\_HOVER;  else  dogSkillPoisonGasStatus = IS\_NOT\_HOVER;  }  // 治癒  if (dogSkillOKBandStatus != USED && dogSkillOKBandStatus != USING)  {  if (point.x > dogSkillOKBandButtonLT.x && point.x < dogSkillOKBandButtonRB.x && point.y > dogSkillOKBandButtonLT.y && point.y < dogSkillOKBandButtonRB.y)  dogSkillOKBandStatus = IS\_HOVER;  else  dogSkillOKBandStatus = IS\_NOT\_HOVER;  }  }  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。  //  // 顯示背景  Background.ShowBitmap();  // Cat  switch (runId)  {  case CAT\_PREPARE:  CatObject.OnShow(Normal);  break;  case CAT\_ATTACK\_TAKE:  CatObject.OnShow(Attack\_1);  break;  case CAT\_ATTACK\_BEGIN:  if (GameData::is2P)  CatAttackInvertedTriangle.OnShow();  CatObject.OnShow(Attack\_2);  break;  case CAT\_ATTACK\_CHARGE:  CatObject.OnShow(Attack\_3);  break;  case CAT\_ATTACK\_FIRE:  CatObject.OnShow(Attack\_4);  break;  case CAT\_BEGIN\_ATTACKED\_LITTLE:  CatObject.OnShow(BeingAttacked\_Little);  break;  case CAT\_BEGIN\_ATTACKED\_BIG:  CatObject.OnShow(BeingAttacked\_Big);  break;  case CAT\_BEGIN\_ATTACKED\_MISS:  CatObject.OnShow(BeingAttacked\_Miss);  break;  default:  CatObject.OnShow(Normal);  break;  }  // Dog  switch (runId)  {  case DOG\_PREPARE:  DogObject.OnShow(Normal);  break;  case DOG\_ATTACK\_TAKE:  DogObject.OnShow(Attack\_1);  break;  case DOG\_ATTACK\_BEGIN:  DogAttackInvertedTriangle.OnShow();  DogObject.OnShow(Attack\_2);  break;  case DOG\_ATTACK\_CHARGE:  DogObject.OnShow(Attack\_3);  break;  case DOG\_ATTACK\_FIRE:  DogObject.OnShow(Attack\_4);  break;  case DOG\_BEGIN\_ATTACKED\_LITTLE:  DogObject.OnShow(BeingAttacked\_Little);  break;  case DOG\_BEGIN\_ATTACKED\_BIG:  DogObject.OnShow(BeingAttacked\_Big);  break;  case DOG\_BEGIN\_ATTACKED\_MISS:  DogObject.OnShow(BeingAttacked\_Miss);  break;  default:  DogObject.OnShow(Normal);  break;  }  // 血量條  CatHealthPointBar.OnShow();  DogHealthPointBar.OnShow();  // Weapon  WeaponObject.OnShow(runId);  // 技能hover動畫撥放  // 二連擊  if (catSkillDoubleAttackStatus != USED && catSkillDoubleAttackStatus != USING)  catSkillDoubleAttackButton.ShowBitmap();  // 重擊  if (catSkillPowerAttackStatus != USED && catSkillPowerAttackStatus != USING)  catSkillPowerAttackHoverAnimation.OnShow();  // 毒氣  if (catSkillPoisonGasStatus != USED && catSkillPoisonGasStatus != USING)  catSkillPoisonGasHoverAnimation.OnShow();  // 治癒  if (catSkillOKBandStatus != USED && catSkillOKBandStatus != USING)  catSkillOKBandHoverAnimation.OnShow();  // Dog skill  // 二連擊  if (dogSkillDoubleAttackStatus != USED && dogSkillDoubleAttackStatus != USING)  dogSkillDoubleAttackButton.ShowBitmap();  // 重擊  if (dogSkillPowerAttackStatus != USED && dogSkillPowerAttackStatus != USING)  dogSkillPowerAttackHoverAnimation.OnShow();  // 毒氣  if (dogSkillPoisonGasStatus != USED && dogSkillPoisonGasStatus != USING)  dogSkillPoisonGasHoverAnimation.OnShow();  // 治癒  if (dogSkillOKBandStatus != USED && dogSkillOKBandStatus != USING)  dogSkillOKBandHoverAnimation.OnShow();  }  } |
| mygame.h |
| #include "GAMEID.h"  #include "Cat.h"  #include "Dog.h"  #include "Weapon.h"  #include "GameData.h"  #include "CatHP.h"  #include "DogHP.h"  namespace game\_framework {  /////////////////////////////////////////////////////////////////////////////  // Constants  /////////////////////////////////////////////////////////////////////////////  /////////////////////////////////////////////////////////////////////////////  // 這個class為遊戲的遊戲開頭畫面物件  // 每個Member function的Implementation都要弄懂  /////////////////////////////////////////////////////////////////////////////  class CGameStateInit : public CGameState {  public:  CGameStateInit(CGame\* g);  void OnInit(); // 遊戲的初值及圖形設定  void OnBeginState(); // 設定每次重玩所需的變數  void OnKeyUp(UINT, UINT, UINT); // 處理鍵盤Up的動作  void OnMouseMove(UINT nFlags, CPoint point); // 處理滑鼠的動作  void OnLButtonUp(UINT nFlags, CPoint point); // 處理滑鼠的動作  protected:  void OnMove(); // 判斷要顯示的畫面  void OnShow(); // 顯示這個狀態的遊戲畫面  private:  // 遊戲開始畫面音樂  bool gameStartMusicDidNotPlay;  // About畫面  CMovingBitmap AboutPicture;  // 遊戲開始畫面背景動畫  CAnimation StartButtonBackground;  // 遊戲開始畫面  CMovingBitmap StartButton\_noHover;  CMovingBitmap StartButton\_hover;  // 遊戲大綱畫面  CMovingBitmap FleabagVsMutt\_noHover;  CMovingBitmap FleabagVsMutt\_hoverHowToPlay;  CMovingBitmap FleabagVsMutt\_hoverLetsPlay;  CAnimation FleabagVsMutt\_titleCat;  // 遊戲說明畫面  CMovingBitmap HowToPlay\_noHover;  CMovingBitmap HowToPlay\_hover;  // 玩家人數選擇畫面  CMovingBitmap ChooseGame\_noHover;  CMovingBitmap ChooseGame\_hover1P;  CMovingBitmap ChooseGame\_hover2P;  // 選擇遊戲難度頁面各難度動畫  CAnimation SelectYourLevel\_hoverBeginnerAnimation;  CAnimation SelectYourLevel\_hoverAverageAnimation;  CAnimation SelectYourLevel\_hoverBringItOnAnimation;  // 選擇遊戲難度頁面  CMovingBitmap SelectYourLevelBackground;  CMovingBitmap SelectYourLevel\_hoverBeginner;  CMovingBitmap SelectYourLevel\_hoverAverage;  CMovingBitmap SelectYourLevel\_hoverBringItOn;  // 現在的畫面狀態  GAME\_PAGE\_ID CurrentPage;  // 所有按鈕的CPoint 左上(LT)與右下(RB)  CPoint StartPage\_StartButtonLT;  CPoint StartPage\_StartButtonRB;  CPoint FleabagVsMuttPage\_HowToPlayButtonLT;  CPoint FleabagVsMuttPage\_HowToPlayButtonRB;  CPoint FleabagVsMuttPage\_LetsPlayButtonLT;  CPoint FleabagVsMuttPage\_LetsPlayButtonRB;  CPoint HowToPlayPage\_LetsPlayButtonLT;  CPoint HowToPlayPage\_LetsPlayButtonRB;  CPoint ChooseGamePage\_1PlayerButtonLT;  CPoint ChooseGamePage\_1PlayerButtonRB;  CPoint ChooseGamePage\_2PlayerButtonLT;  CPoint ChooseGamePage\_2PlayerButtonRB;  CPoint SelectYourLevelPage\_ImBeginnerButtonLT;  CPoint SelectYourLevelPage\_ImBeginnerButtonRB;  CPoint SelectYourLevelPage\_ImAverageButtonLT;  CPoint SelectYourLevelPage\_ImAverageButtonRB;  CPoint SelectYourLevelPage\_BringItOnButtonLT;  CPoint SelectYourLevelPage\_BringItOnButtonRB;  };  /////////////////////////////////////////////////////////////////////////////  // 這個class為遊戲的遊戲執行物件，主要的遊戲程式都在這裡  // 每個Member function的Implementation都要弄懂  /////////////////////////////////////////////////////////////////////////////  class CGameStateRun : public CGameState {  public:  CGameStateRun(CGame\* g);  ~CGameStateRun();  void OnBeginState(); // 設定每次重玩所需的變數  void OnInit(); // 遊戲的初值及圖形設定  void OnKeyDown(UINT, UINT, UINT);  void OnKeyUp(UINT, UINT, UINT);  void OnLButtonDown(UINT nFlags, CPoint point); // 處理滑鼠的動作  void OnLButtonUp(UINT nFlags, CPoint point); // 處理滑鼠的動作  void OnMouseMove(UINT nFlags, CPoint point); // 處理滑鼠的動作  void OnRButtonDown(UINT nFlags, CPoint point); // 處理滑鼠的動作  void OnRButtonUp(UINT nFlags, CPoint point); // 處理滑鼠的動作  protected:  void OnMove(); // 移動遊戲元素  void OnShow(); // 顯示這個狀態的遊戲畫面  private:  // 音效  bool catAttackIsNotPlaying;  bool dogAttackIsNotPlaying;  bool gameRunMusicHadNotPlay;  // 戰鬥中  CMovingBitmap Background;  // 狗勾 貓貓  Dog DogObject;  Cat CatObject;  Weapon WeaponObject;  // 攻擊倒三角  CAnimation CatAttackInvertedTriangle;  CAnimation DogAttackInvertedTriangle;  // 血量條  CatHP CatHealthPointBar;  DogHP DogHealthPointBar;  GAME\_RUN\_ID runId;  int Timer;  int lastPower;  // 技能狀態  SKILL\_STATUS catSkillDoubleAttackStatus;  SKILL\_STATUS catSkillPowerAttackStatus;  SKILL\_STATUS catSkillPoisonGasStatus;  SKILL\_STATUS catSkillOKBandStatus;  SKILL\_STATUS dogSkillDoubleAttackStatus;  SKILL\_STATUS dogSkillPowerAttackStatus;  SKILL\_STATUS dogSkillPoisonGasStatus;  SKILL\_STATUS dogSkillOKBandStatus;  // 技能按鈕圖片  CMovingBitmap catSkillDoubleAttackButton;  CMovingBitmap dogSkillDoubleAttackButton;  // 技能按鈕hover動畫  CAnimation catSkillPowerAttackHoverAnimation;  CAnimation catSkillPoisonGasHoverAnimation;  CAnimation catSkillOKBandHoverAnimation;  CAnimation dogSkillPowerAttackHoverAnimation;  CAnimation dogSkillPoisonGasHoverAnimation;  CAnimation dogSkillOKBandHoverAnimation;  // 所有按鈕的CPoint 左上(LT)與右下(RB)  CPoint catSkillDoubleAttackButtonLT;  CPoint catSkillDoubleAttackButtonRB;  CPoint catSkillPowerAttackButtonLT;  CPoint catSkillPowerAttackButtonRB;  CPoint catSkillPoisonGasButtonLT;  CPoint catSkillPoisonGasButtonRB;  CPoint catSkillOKBandButtonLT;  CPoint catSkillOKBandButtonRB;  CPoint dogSkillDoubleAttackButtonLT;  CPoint dogSkillDoubleAttackButtonRB;  CPoint dogSkillPowerAttackButtonLT;  CPoint dogSkillPowerAttackButtonRB;  CPoint dogSkillPoisonGasButtonLT;  CPoint dogSkillPoisonGasButtonRB;  CPoint dogSkillOKBandButtonLT;  CPoint dogSkillOKBandButtonRB;  };  /////////////////////////////////////////////////////////////////////////////  // 這個class為遊戲的結束狀態(Game Over)  // 每個Member function的Implementation都要弄懂  /////////////////////////////////////////////////////////////////////////////  class CGameStateOver : public CGameState {  public:  CGameStateOver(CGame\* g);  void OnBeginState(); // 設定每次重玩所需的變數  void OnInit();  void OnMouseMove(UINT nFlags, CPoint point); // 處理滑鼠的動作  void OnLButtonUp(UINT nFlags, CPoint point); // 處理滑鼠的動作  protected:  void OnMove(); // 移動遊戲元素  void OnShow(); // 顯示這個狀態的遊戲畫面  private:  bool isHoverReplay;  // 遊戲開始畫面音樂  bool gameOverMusicHadNotPlay;  // 貓貓贏了  CMovingBitmap CatWin\_background;  CMovingBitmap CatWin\_replayHover;  CAnimation CatWin\_titleAnimation;  CAnimation CatWin\_catAnimation;  CAnimation CatWin\_dogAnimation;  // 狗溝贏了  CMovingBitmap DogWin\_background;  CMovingBitmap DogWin\_replayHover;  CAnimation DogWin\_catAnimation;  CAnimation DogWin\_dogAnimation;  // replay按鈕的CPoint 左上(LT)與右下(RB)  CPoint CatWin\_ReplayButtonLT;  CPoint CatWin\_ReplayButtonRB;  CPoint DogWin\_ReplayButtonLT;  CPoint DogWin\_ReplayButtonRB;  };  } |
| Weapon.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <ddraw.h>  #include "gamelib.h"  #include "GAMEID.h"  #include "Weapon.h"  namespace game\_framework {  /////////////////////////////////////////////////////////////////////////////  // Weapon: Weapon class  /////////////////////////////////////////////////////////////////////////////  Weapon::Weapon()  {  weaponPositionX = 0;  weaponPositionY = 559;  weaponSizeFactor = 1;  isNormalWeapon = true;  }  int Weapon::GetX\_LT(GAME\_RUN\_ID runId)  {  if (runId == CAT\_ATTACK\_FIRE)  return weaponPositionX + 14 \* (int)weaponSizeFactor;  else if (runId == DOG\_ATTACK\_FIRE)  return weaponPositionX + 7 \* (int)weaponSizeFactor;  else  return -1;  }  int Weapon::GetY\_LT(GAME\_RUN\_ID runId)  {  if (runId == CAT\_ATTACK\_FIRE)  return weaponPositionY + 3 \* (int)weaponSizeFactor;  else if (runId == DOG\_ATTACK\_FIRE)  return weaponPositionY + 21 \* (int)weaponSizeFactor;  else  return -1;  }  int Weapon::GetX\_RB(GAME\_RUN\_ID runId)  {  if (runId == CAT\_ATTACK\_FIRE)  return weaponPositionX + 78 \* (int)weaponSizeFactor;  else if (runId == DOG\_ATTACK\_FIRE)  return weaponPositionX + 80 \* (int)weaponSizeFactor;  else  return -1;  }  int Weapon::GetY\_RB(GAME\_RUN\_ID runId)  {  if (runId == CAT\_ATTACK\_FIRE)  return weaponPositionY + 79 \* (int)weaponSizeFactor;  else if (runId == DOG\_ATTACK\_FIRE)  return weaponPositionY + 65 \* (int)weaponSizeFactor;  else  return -1;  }  void Weapon::Initialize()  {  SetPower(0, 0);  SetPosition(0, 0);  Cat\_Weapon.SetTopLeft(0, 0);  Dog\_Weapon.SetTopLeft(0, 0);  Cat\_PoisionWeapon.SetTopLeft(0, 0);  Dog\_PoisionWeapon.SetTopLeft(0, 0);  isNormalWeapon = true;  }  void Weapon::LoadBitmap()  {  Cat\_Weapon.LoadBitmap("GamePicture/GameRun/Cat/Weapon.bmp", RGB(180, 0, 255));  Dog\_Weapon.LoadBitmap("GamePicture/GameRun/Dog/Weapon.bmp", RGB(180, 0, 255));  Cat\_PoisionWeapon.LoadBitmap("GamePicture/GameRun/Cat/PoisionWeapon.bmp", RGB(180, 0, 255));  Dog\_PoisionWeapon.LoadBitmap("GamePicture/GameRun/Dog/PoisionWeapon.bmp", RGB(180, 0, 255));  }  void Weapon::OnMove(GAME\_RUN\_ID runId)  {  if (runId == CAT\_ATTACK\_FIRE || runId == DOG\_ATTACK\_FIRE) {  SetPower(velocityX, velocityY + 10); // 10 是重力  SetPosition(weaponPositionX + velocityX, weaponPositionY + velocityY);  }  if (isNormalWeapon) {  // 根據目前的條件改變狀態  if (runId == CAT\_ATTACK\_FIRE) {  Cat\_Weapon.SetTopLeft(weaponPositionX, weaponPositionY);  }  else if (runId == DOG\_ATTACK\_FIRE) {  Dog\_Weapon.SetTopLeft(weaponPositionX, weaponPositionY);  }  }  else {  if (runId == CAT\_ATTACK\_FIRE) {  Cat\_PoisionWeapon.SetTopLeft(weaponPositionX, weaponPositionY);  }  else if (runId == DOG\_ATTACK\_FIRE) {  Dog\_PoisionWeapon.SetTopLeft(weaponPositionX, weaponPositionY);  }  }  }  void Weapon::OnShow(GAME\_RUN\_ID runId)  {  if (isNormalWeapon) {  // 根據狀態顯示貓或狗的武器  if (runId == CAT\_ATTACK\_FIRE) {  Cat\_Weapon.ShowBitmap(weaponSizeFactor);  }  else if (runId == DOG\_ATTACK\_FIRE) {  Dog\_Weapon.ShowBitmap(weaponSizeFactor);  }  }  else {  if (runId == CAT\_ATTACK\_FIRE) {  Cat\_PoisionWeapon.ShowBitmap(weaponSizeFactor);  }  else if (runId == DOG\_ATTACK\_FIRE) {  Dog\_PoisionWeapon.ShowBitmap(weaponSizeFactor);  }  }  }  void Weapon::SetPower(int powerX, int powerY)  {  velocityX = powerX;  velocityY = powerY;  }  void Weapon::SetPosition(int x, int y)  {  weaponPositionX = x;  weaponPositionY = y;  }  void Weapon::DogFire(int timer)  {  int power = timer \* 5;  SetPosition(1088, 693); // 骨頭起始發射位置  SetPower(-power \* 4 / 5, -power); // 往左上角發射  }  void Weapon::CatFire(int timer)  {  int power = timer \* 5;  SetPosition(182, 588); // 骨頭起始發射位置  SetPower(power \* 4 / 5, -power); // 往左上角發射  }  bool Weapon::isHitGround()  {  // Y值大於814時代表落地  return weaponPositionY > 814 || GetX\_LT(CAT\_ATTACK\_FIRE) > 1333 || GetX\_RB(DOG\_ATTACK\_FIRE) < 0;  }  int Weapon::isHitCat()  {  // -1:too far, 0:miss, 1:lightly attacked, 2:heavly attacked]  // 判斷武器是不是飛到一半  if (GetX\_LT(DOG\_ATTACK\_FIRE) < 1333 / 2 && GetY\_RB(DOG\_ATTACK\_FIRE) > 586) {  // 判斷武器是不是完全在重擊區或擦傷區或是miss  if (GetX\_LT(DOG\_ATTACK\_FIRE) > 74 && GetX\_RB(DOG\_ATTACK\_FIRE) < 200)  return 2;  else if (GetX\_LT(DOG\_ATTACK\_FIRE) > 24 && GetX\_RB(DOG\_ATTACK\_FIRE) < 250)  return 1;  else  return 0;  }  else  return -1;  }  int Weapon::isHitDog()  {  // -1:too far, 0:miss, 1:lightly attacked, 2:heavly attacked]  // 判斷武器是不是飛到一半  if (GetX\_LT(CAT\_ATTACK\_FIRE) > 1333 / 2 && GetY\_RB(CAT\_ATTACK\_FIRE) > 677) {  // 判斷武器是不是完全在重擊區或擦傷區或是miss  if (GetX\_LT(CAT\_ATTACK\_FIRE) > 1092 && GetX\_RB(CAT\_ATTACK\_FIRE) < 1238)  return 2;  else if (GetX\_LT(CAT\_ATTACK\_FIRE) > 1042 && GetX\_RB(CAT\_ATTACK\_FIRE) < 1288)  return 1;  else  return 0;  }  else  return -1;  }  void Weapon::setSize(double factor)  {  weaponSizeFactor = factor;  }  void Weapon::setNormalWeapon(bool normal)  {  isNormalWeapon = normal;  }  } |
| Weapon.h |
| #pragma once  namespace game\_framework {  class Weapon  {  public:  Weapon();  int GetX\_LT(GAME\_RUN\_ID runId); // 投擲物左上角 x 座標  int GetY\_LT(GAME\_RUN\_ID runId); // 投擲物左上角 y 座標  int GetX\_RB(GAME\_RUN\_ID runId); // 投擲物右下角 x 座標  int GetY\_RB(GAME\_RUN\_ID runId); // 投擲物右下角 y 座標  void Initialize(); // 設定  void OnMove(GAME\_RUN\_ID runId); // 根據目前的條件改變狀態  void OnShow(GAME\_RUN\_ID runId); // 根據目前的狀態貼上貓貓圖片  void LoadBitmap();  void SetPower(int powerX, int powerY);  void SetPosition(int x, int y);  void DogFire(int timer);  void CatFire(int timer);  bool isHitGround();  int isHitCat();  int isHitDog();  void setSize(double);  void setNormalWeapon(bool);  protected:  // 武器投擲物座標  int weaponPositionX;  int weaponPositionY;  int velocityX;  int velocityY;  CMovingBitmap Cat\_Weapon;  CMovingBitmap Dog\_Weapon;  CMovingBitmap Cat\_PoisionWeapon;  CMovingBitmap Dog\_PoisionWeapon;  bool isNormalWeapon;  double weaponSizeFactor;  };  } |