|  |  |
| --- | --- |
| **臺北科技大學**  **資訊工程系** | |
| **物件導向程式設計實習-書面報告** | |
| **組別：第 3 組**  **題目：Windows-Mini metro**  **組員：楊信致 105590040**  **鄭宇翔 105590028**  **指導老師：陳偉凱** |

目錄

[壹、 簡介 1](#_Toc484992673)

[一、 製作動機： 1](#_Toc484992674)

[二、 分工： 1](#_Toc484992675)

[貳、 遊戲介紹 2](#_Toc484992676)

[一、 遊戲說明 2](#_Toc484992677)

[二、 遊戲圖形 3](#_Toc484992678)

[三、 遊戲音效](#_Toc484992679) 6

[參、 程式設計](#_Toc484992680) 7

[一、 程式架構](#_Toc484992681) 7

[二、 程式類別 9](#_Toc484992682)

[三、 程式技術](#_Toc484992683) 10

[肆、 結語](#_Toc484992684) 11

[一、 問題及解決方法](#_Toc484992685) 11

[二、 時間表(不含上課時間)](#_Toc484992686) 12

[三、 貢獻比例](#_Toc484992687) 13

[四、 檢核表：](#_Toc484992688) 13

[五、 收獲](#_Toc484992689) 14

[六、 心得](#_Toc484992690) 15

[七、 對於本課程的建議](#_Toc484992691) 17

[伍、 附錄](#_Toc484992692) 18

# **簡介**

## 製作動機：

一開始我們選擇要做Windows的遊戲。我們要做的遊戲是「mini metro」，這是一款為迅速發展的城市設計地鐵路線圖的遊戲，因為我們有實際玩過這款遊戲，覺得自己規劃出一個城市的地鐵路線是一件很有成就感的事，又覺得遊戲中的視覺設計很不錯，所以就想要嘗試自己做出和「mini metro」一樣的遊戲。

## 分工：

楊信致 負責遊戲中主要的功能，像是拉出鐵軌、乘客上下車、乘客移動等等。鄭宇翔 負責遊戲中介面、圖片、音樂素材等等。

# **遊戲介紹**

## 遊戲說明

1. 遊玩方式：
2. 點選車站。
3. 移動滑鼠即可拉出鐵軌。
4. 點選另一個車站製作鐵軌、車廂。
5. 車廂到站時乘客會上車，但是鐵軌路線上要有與乘客同類型的車站乘客才會上車。
6. 乘客到達與自己同類型的車站就會下車。
7. 遊戲規則：

在兩分鐘內運輸越多乘客分數越高。

1. 密技：

I. 按鍵盤上鍵可增加人數10。

II. 按鍵盤下鍵可直接結算。

## 遊戲圖形

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 遊戲讀取 | | | | | | |
|  | | | | Loading | | |
| 遊戲開始背景 | | | | | | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\background.bmp C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\backgroundnight.bmp | 背景(日間模式/夜間模式) | |  | | | 標題(日間模式/夜間模式) |
|  | 鐵路(深藍) | |  | | | 鐵路(藍) |
|  | 鐵路(橘) | | C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\red.bmp | | | 鐵路(紅) |
| 遊戲按鈕 | | | | | | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\startInvert.bmp | 開始按鈕/開始按鈕(滑鼠滑過) | |  | | | 夜間模式/夜間模式按鈕(滑鼠滑過) |
|  | 日間模式/日間模式按鈕(滑鼠滑過) | |  | | | 離開按鈕/離開按鈕(滑鼠滑過) |
|  | 繼續按鈕/繼續按鈕(滑鼠滑過) | | C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\restartInvert.bmp | | | 重新開始按鈕/重新開始按鈕(滑鼠滑過) |
|  | 遊戲說明/遊戲說明 (滑鼠滑過) | |  | | |  |
| 遊戲中 | | | | | | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\map.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\mapNight.bmp | | 背景(日間模式/夜間模式) | | C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\person.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\personnight.bmp | 人(日間模式/夜間模式) | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Circle.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Circle_night.bmp | | 圓形車站(日間模式/夜間模式) | | C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Diamond.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Diamond_night.bmp | 菱形車站(日間模式/夜間模式) | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Hexagon.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Hexagon_night.bmp | | 六角形車站(日間模式/夜間模式) | | C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Pentagon.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Pentagon_night.bmp | 五角形車站(日間模式/夜間模式) | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Square.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Square_night.bmp | | 四方形車站(日間模式/夜間模式) | | C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Triangle.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Station_Triangle_night.bmp | 三角形車站(日間模式/夜間模式) | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\red.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\yellow.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\blue.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\purple.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\choosecolor.bmp | | | | | 車站顏色+顯取顏色外框(日間模式/夜間模式) | |
|  | | | | | 數字(日間模式/夜間模式) | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Passenger_Circle.bmp | | 圓形乘客(日間模式/夜間模式) | |  | 菱形乘客(日間模式/夜間模式) | |
|  | | 六角形乘客(日間模式/夜間模式) | |  | 五角形乘客(日間模式/夜間模式) | |
|  | | 日方形乘客(日間模式/夜間模式) | | C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Passenger_Triangle.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Passenger_Triangle_night.bmp | 三角形乘客(日間模式/夜間模式) | |
|  | | | | | 時鐘 | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_37.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_38.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_39.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_40.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_41.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_42.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_43.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_44.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_45.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_46.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_47.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_48.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_49.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_50.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_51.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_52.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_53.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_54.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_55.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_56.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_57.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_58.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_59.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_60.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_61.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_62.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_63.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_64.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_65.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_66.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_67.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_68.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_69.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_70.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_71.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_72.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_73.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_74.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_75.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_76.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_77.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_78.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_79.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_80.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_81.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_82.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_83.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_84.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_85.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_86.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_87.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_88.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_89.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_90.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_91.bmp | | | | | 時鐘 | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_97.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_98.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_99.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_0.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_1.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_2.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_3.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_4.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_5.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_6.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_7.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_8.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_9.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_10.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_11.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_12.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_13.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_14.bmpC:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Clock_15.bmp | | | | | 時鐘 | |
| C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\gameovernight.bmp  C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\good.bmp  C:\Users\uxiang\AppData\Local\Microsoft\Windows\INetCache\Content.Word\bad.bmp | | | | | 遊戲結算畫面 | |

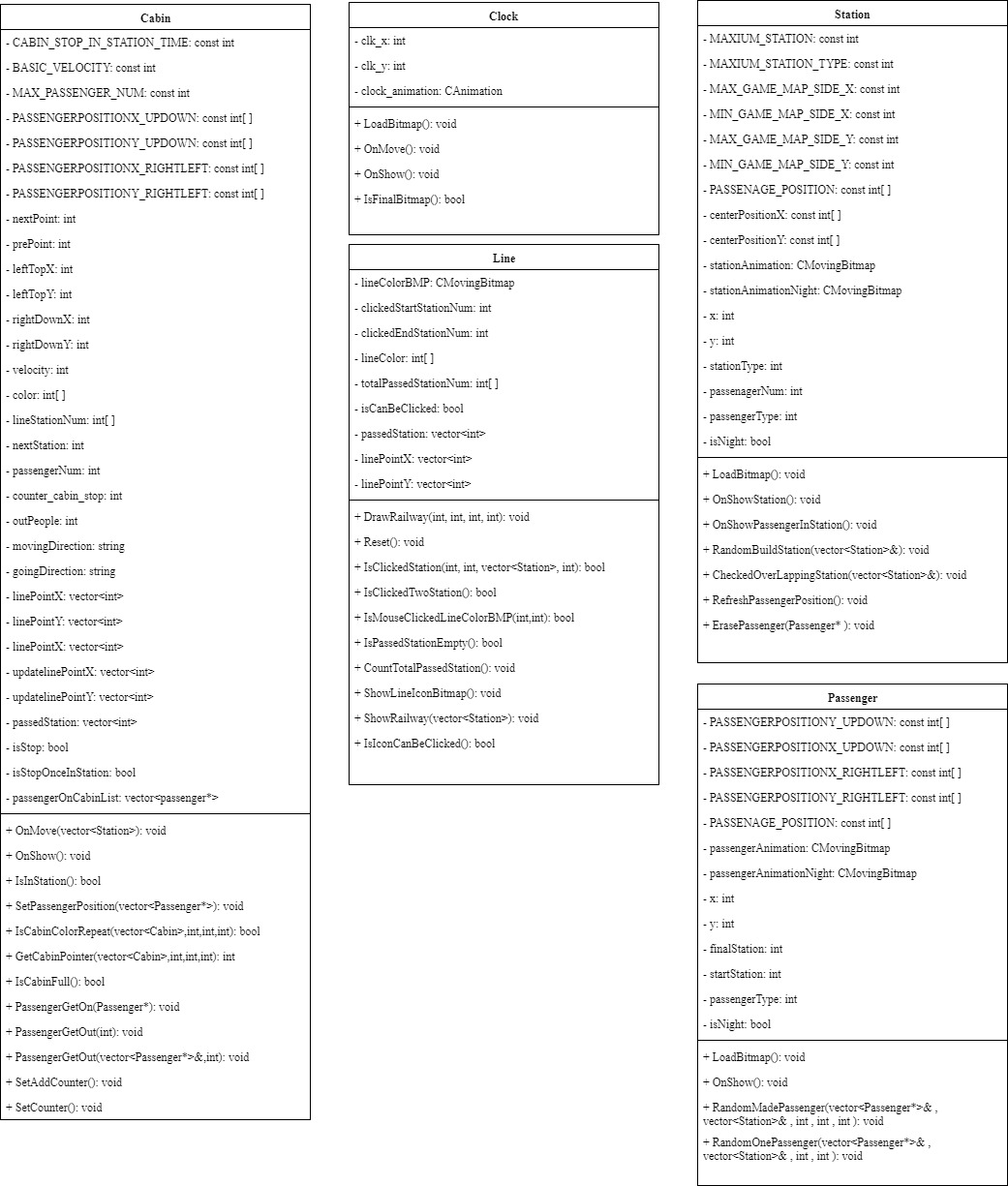
## 遊戲音效

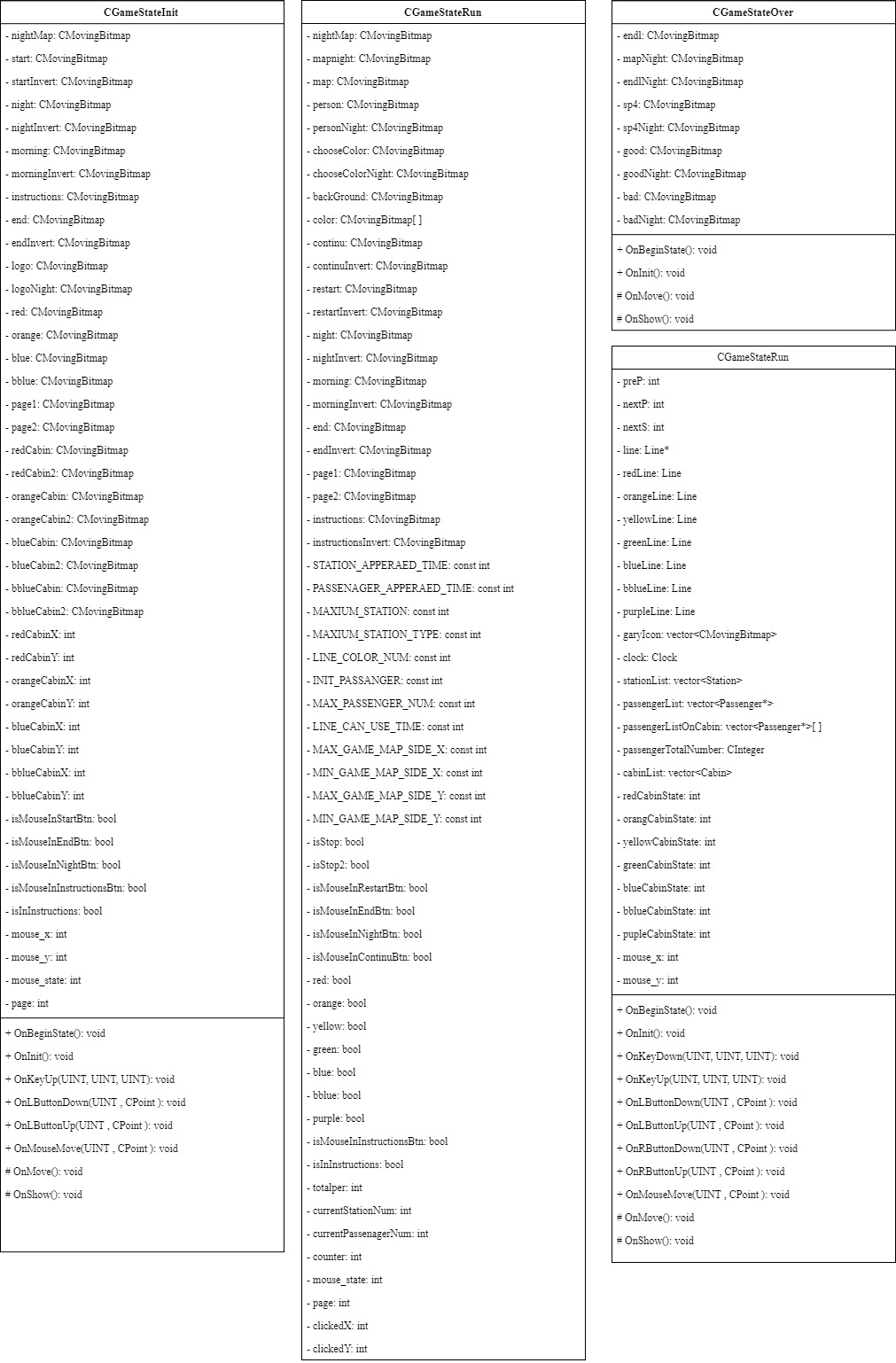
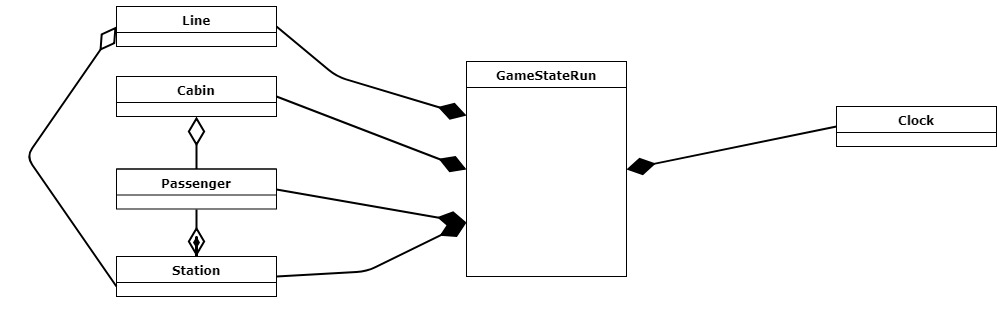
|  |  |  |
| --- | --- | --- |
| 時機 | 用途 | 音效檔名 |
| 整體遊戲 | 背景音樂 | music.mp3 |
| 按鈕 | 滑鼠划上按鈕 | slip.mp3 |
| 滑鼠點下按鈕 | click.mp3 |
| 遊戲中 | 車站出現 | appear.mp3 |
| 選擇顏色 | odd.wav |
| 點選時鐘暫停 | stop1.mp3 |
| stop2.mp3 |
| 點選車站 | select1.mp3 |
| select2.mp3 |

# **程式設計**

## 程式架構

Class Diagram:



****

## 程式類別

|  |  |  |  |
| --- | --- | --- | --- |
| 類別名稱 | .h 檔行數 | .cpp檔行數 | 說明 |
| mygame | 278 | 1546 | 遊戲開頭畫面物件、執行物件、結束狀態 |
| Cabin | 82 | 551 | 控制車廂移動、乘客上下車 |
| Clock | 20 | 64 | 時鐘動畫 |
| Line | 55 | 347 | 繪畫鐵軌、紀錄鐵軌路線座標 |
| Passenger | 38 | 167 | 生成乘客 |
| Station | 48 | 213 | 生成車站 |
| 總行數 | 521 | 2888 |  |

## 程式技術

1. RandomMadePassenger(vector<Passenger\*>& passengerList, vector<Station>& stationList, int stationMaxium, int stationTypeNum, int totalPassenagerNum)

這是遊戲開始的時候，用來隨機產生乘客的函式，用到了隨機變數的函式srand和取得時間的time（）函數，讓他下去產生新的亂數。

1. DrawRailway(int startX, int startY, int endX, int endY)

這是用來畫出線路的函式，透過內建函式和傳入的四個點，畫出兩段長方形，而這個畫出長方形的方法跟跟Loding畫面的長條圖是一樣的。

1. SetMovingCabin(string movingDirection, int num)

這是用來決定我車廂的大小用的，因為我們車廂也是用內建的函式畫出來的，所以，需要兩個點去畫出那個長方形，而我利用了他左上角的點去算出右下角的點，這樣就不會因為轉彎或是其他因素，而讓車廂大小跑掉。

# **結語**

## 問題及解決方法

1. 問題：當滑鼠移到按鈕上將按鈕圖片放大會出錯。
2. 解決方式：因為我們在判斷式中用到了圖片的變數，而在最最最一開始的時候，圖片有短暫的時間還沒載入，所以會噴錯誤，要解決就是直接寫點座標讓他判斷。
3. 問題： 時鐘的圖片太多
4. 解決方式： 我們一開始做出時鐘的圖片時，不小心做到了500張，我們覺得有點多，所以後來經過刪減，現在只剩下100張了。
5. 問題：乘客上車之後，會有錯誤
6. 解決方式： 找了很久，後來發現是忘記把在vector裡指標做erase，讓程式操作到空指標而出錯。
7. 問題：所有圖片的前後順序。
8. 解決方式：一開始我們並不曉得圖片的前後關係與onshow的順序有關係，導致在最剛開始時發現我們想要的圖片怎麼都沒有出現，後來才發現有這個問題，調整程式碼之後就解決了。

## 時間表(不含上課時間)

|  |  |  |  |
| --- | --- | --- | --- |
| 週數 | 日期 | 花費時間 | 工作規劃 |
| 0 | 2017-03-14 ~  2017-03-20 | 楊信致：1小時  鄭宇翔：1小時 | Teamwork with Git , Windows Tutorials |
| 1 | 2017-03-16 ~  2017-03-22 | 楊信致：6小時  鄭宇翔：4小時 | 畫面上自動按順序出現10車站 , 計時器 |
| 2 | 2017-03-23~  2017-03-29 | 楊信致：6小時  鄭宇翔：4小時 | 第二個計時器 , 地圖,優化 |
| 3 | 2017-03-30 ~  2017-04-05 | 楊信致：3小時  鄭宇翔： 3小時 | 可用滑鼠拉出路線,加強各物件的關係,優化 |
| 4 | 2017-04-09 ~  2017-04-15 | 楊信致：8小時  鄭宇翔：5小時 | 可拉出自己想要的顏色的線路, 開始遊戲畫面 |
| 5 | 2017-04-13~  2017-04-19 | 楊信致：7小時  鄭宇翔：6小時 | 乘客出現在各車站 , 軌道bug修正 |
| 6 | 2017-04-20 ~  2017-04-26 | 楊信致：2小時  鄭宇翔：1小時 | 完成人物移動、拖移物件解鎖關卡 |
| 7 | 2017-04-27 ~  2017-05-03 | 楊信致：6小時  鄭宇翔：4小時 | 乘客bug,軌道bug修正 |
| 8 | 2017-05-04~  2017-05-10 | 楊信致：8小時  鄭宇翔：10小時 | 車廂,bug |
| 9 | 2017-05-11 ~  2017-05-17 | 楊信致：6小時  鄭宇翔：4小時 | 車廂,bug |
| 10 | 2017-05-18 ~  2017-05-24 | 楊信致：4 小時  鄭宇翔：3小時 | 車廂方向,乘客上車,bug |
| 11 | 2017-05-25 ~  2017-05-31 | 楊信致：4小時  鄭宇翔：3小時 | 車廂BUG,線路bug,乘客可上車 |
| 12 | 2017-06-01 ~  2017-06-07 | 楊信致：6小時  鄭宇翔：4小時 | 乘客可上車 |
| 13 | 2017-06-08 ~  2017-06-14 | 楊信致：5小時  鄭宇翔：4小時 | 遊戲收尾 |
| 14 | 2017-06-09 ~  2017-06-15 | 楊信致：6小時  鄭宇翔：6小時 | 遊戲收尾，報告 |
| 總花費時數 | | 楊信致：78小時  鄭宇翔：62小時 | |

## 貢獻比例

楊信致：60% 程式撰寫、素材搜尋

鄭宇翔：40% 程式撰寫、美工、素材搜尋

## 檢核表：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 項目 | 完成否 | 無法完成的原因 |
| 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 | 修改Help->About | ■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檔 | ■已完成 □未完成 |  |
| 7 | 報告字型、點數、對齊、行距、頁碼等格式正確 | ■已完成 □未完成 |  |
| 8 | 報告封面、側邊格式正確 | ■已完成 □未完成 |  |

## 收獲

楊信致：

我們一開始是因為對另外兩個的語言不熟，所以選擇了C++。在經歷了這一學期的摸索之後，我覺得對於C++的方面有了稍微的小小了解，而且這也是我們第一次自己學習從頭到尾，素材自己弄、程式自己寫，真的獲益良多。

我覺得C++是一個比較困難的程式語言，可是有跟C比起來，還是方便許多。

起初我已經有點遺忘C++ 的程式怎麼寫，以至於剛開始的進度有點緩慢，而且沒辦法順利的找到bug，後來經過我們不斷的詢問後，才知道可以透過中斷點，讓程式停在我們想要停的地方。

鄭宇翔：

經過一學期的摸索之後，有比一開始知道裡面的程式在寫什麼，雖然詳細的內容沒有很了解，對於C++也熟練了點，以前都不會用bool，現在才發現這個很好用，PS也稍微會用了一點，原本還有為了讓時鐘轉得很順去用AE做了很多圖片出來，但因為真的太多了所以就沒放。除錯我只能看剛剛打的程式哪一行出錯，再去看他為什麼錯，常常因為圖片的關係噴出一堆錯。

## 心得

楊信致：

一開始為了挑遊戲，就花了不少時間，後來上YOUTUBE看了之後，就決定了Mini Metro，因為這遊戲我有買，也有玩過。

一開始的時候為了git的問題困擾了我很久，後來發現是老師的伺服器問題，才鬆了一口氣。

原本預計的進度，是在第二次的demo就要完成大部份的遊戲，可是因為有其他的功課，所以我整個遊戲進度就大delay了。幸好，最後進度有趕上，沒有因為delay而開天窗。

C++在我上學習修完課之後我就沒什麼去碰了，所以在剛開始寫程式的時候，我也費了一番功夫去重新熟悉了一下，幸好，當初的記憶都有找回來。可是在途中，也是有寫出了很多bug，可能是因為我對c++還不是很熟吧，所以才會寫出一堆bug，那這個也是我應該要在精進的部份。

整個學期寫程式寫下來，發現跟學期初的自己，又更進步了，因為我在這次的寫遊戲的過程中，學到了很多寫程式的技巧，也要謝謝我的室友，在我寫出bug的時候幫我debug，讓我覺得這個世界還是有溫暖的!

鄭宇翔：

一開始因為不知道做什麼遊戲苦惱很久，後來上網看了很多遊戲就決定是Mini Metro，因為這遊戲的視覺設計讓我很喜歡。

當我看到範例程式的時候基本上是完全看不懂，雖然我有照著老師給的範例去做，但做完之後仍然不知道該從哪裡開始寫，直到信致寫出了一些東西，才可以慢慢看懂，不然我連怎麼讓圖片動都不知道。

Mini Metro原本的設計是可以拉出45度的斜線，但我們試了很多方法都不知道該如何去拉出一條斜線，所以只好以90度轉角來去拉路線，然後遊戲當中很多主要的功能我原本也該負責一部份，但我不知道為什麼都寫不出來，頭腦都打結，可能是我都沒有把流程圖、架構那些東西寫出來，只會想到什麼就寫什麼，所以都寫不出來，信致也很厲害的把我不會的部分也都寫完，我能做的大概就是負責尋找圖片素材、音樂素材，還有一些像是遊戲介面那種小小的功能。

我覺得除了要會寫程式，時間掌控也是一件很重要的事，常常因為別的科目作業很多來當藉口沒時間做，但其實是自己都把作業拖到很晚才開始做，所以才會覺得事情很多事情不夠，真的除了顧課業之外還要好好培養自己的生活習慣。

還有最重要的是團隊合作，團隊合作的話我們兩個都會寫上自己寫的程式，我常常都只能測試結果，都沒有去看信致怎麼寫的，所以到最後很多地方都不太懂，所以要團隊合作的話就要先把自己顧好才行。

其實我對寫遊戲蠻有興趣的，但是實際下去寫才發現很困難，如果將來還有機會選擇寫遊戲我還是會嘗試看看，相信這學期所學到的東西和經驗一定可以幫助我。

## 對於本課程的建議

1. 每一次Demo時，希望能對螢幕廣播。

# **附錄**

|  |
| --- |
| mygame.h |
| #include "Station.h"  #include "Clock.h"  #include "Line.h"  #include "Passenger.h"  #include "Cabin.h"  namespace game\_framework  {  /////////////////////////////////////////////////////////////////////////////  // Constants  /////////////////////////////////////////////////////////////////////////////  enum AUDIO\_ID // 定義各種音效的編號  {  AUDIO\_SLIP, // 0  AUDIO\_CLICK, // 1  AUDIO\_MUSIC, // 2  AUDIO\_SELECT1, // 3  AUDIO\_SELECT2, // 4  AUDIO\_APPEAR, // 5  AUDIO\_STOP1, // 6  AUDIO\_STOP2, // 7  AUDIO\_COLOR // 8  };  /////////////////////////////////////////////////////////////////////////////  // 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 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  protected:  void OnMove(); // 移動遊戲元素  void OnShow(); // 顯示這個狀態的遊戲畫面  private:  CMovingBitmap nightMap;  CMovingBitmap start; //遊戲開始的按鈕圖  CMovingBitmap startInvert; //遊戲開始的顏色反相按鈕圖  CMovingBitmap night; //夜間模式的按鈕圖  CMovingBitmap nightInvert; //夜間模式的顏色反相按鈕圖  CMovingBitmap morning; //日間模式的按鈕圖  CMovingBitmap morningInvert; //日間模式的顏色反相按鈕圖  CMovingBitmap instructions; //遊戲說明的按鈕圖  CMovingBitmap instructionsInvert; //遊戲說明的顏色反相按鈕圖  CMovingBitmap end; //遊戲結束的按鈕圖  CMovingBitmap endInvert; //遊戲結束的顏色反相按鈕圖  CMovingBitmap logo; //遊戲的logo  CMovingBitmap logoNight; //遊戲的顏色反相logo  CMovingBitmap red; //封面的紅色鐵軌  CMovingBitmap orange; //封面的橘色鐵軌  CMovingBitmap blue; //封面的藍色鐵軌  CMovingBitmap bblue; //封面的深藍色鐵軌  CMovingBitmap page1; //遊戲說明第1頁  CMovingBitmap page2; //遊戲說明第2頁  CMovingBitmap redCabin; //封面的紅色車廂  CMovingBitmap redCabin2; //封面的紅色車廂  CMovingBitmap orangeCabin; //封面的橘色車廂  CMovingBitmap orangeCabin2; //封面的橘色車廂  CMovingBitmap blueCabin; //封面的藍色車廂  CMovingBitmap blueCabin2; //封面的藍色車廂  CMovingBitmap bblueCabin; //封面的深藍色車廂  CMovingBitmap bblueCabin2; //封面的深藍色車廂  int redCabinX = -50;  int redCabinY = 270;  int orangeCabinX = 38;  int orangeCabinY = 800;  int blueCabinX = 694;  int blueCabinY = 600;  int bblueCabinX = 675;  int bblueCabinY = 1000;  bool isMouseInStartBtn; //滑鼠在開始按鈕  bool isMouseInEndBtn; //滑鼠在結束按鈕  bool isMouseInNightBtn; //滑鼠在模式按鈕  bool isMouseInInstructionsBtn; //滑鼠在遊戲說明按鈕  bool isInInstructions; //滑鼠在繼續按鈕  int mouse\_x = 0, mouse\_y = 0; //滑鼠位置  int mouse\_state = 1;  int page = 0;  };  /////////////////////////////////////////////////////////////////////////////  // 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:  CMovingBitmap nightMap; //夜間模式背景  CMovingBitmap mapnight; //夜間模式選單背景  CMovingBitmap map; //日間模式背景  CMovingBitmap esc; //選單  CMovingBitmap escNight; //夜間模式選單  CMovingBitmap person; //人的圖案  CMovingBitmap personNight; //夜間模式人的圖案  CMovingBitmap chooseColor; //選取鐵軌顏色外框  CMovingBitmap chooseColorNight; //夜間模式選取鐵軌顏色外框  CMovingBitmap backGround; //日間模式選單背景  CMovingBitmap color[7]; //鐵軌顏色  CMovingBitmap continu; //繼續的按鈕圖  CMovingBitmap continuInvert; //繼續的顏色反相按鈕圖  CMovingBitmap restart; //遊戲重新開始的按鈕圖  CMovingBitmap restartInvert; //遊戲重新開始的顏色反相按鈕圖  CMovingBitmap night; //夜間模式的按鈕圖  CMovingBitmap nightInvert; //夜間模式的顏色反相按鈕圖  CMovingBitmap morning; //日間模式的按鈕圖  CMovingBitmap morningInvert; //日間模式的顏色反相按鈕圖  CMovingBitmap end; //遊戲結束的按鈕圖  CMovingBitmap endInvert; //遊戲結束的顏色反相按鈕圖  CMovingBitmap page1; //遊戲說明第1頁  CMovingBitmap page2; //遊戲說明第2頁  CMovingBitmap instructions; //遊戲說明的按鈕圖  CMovingBitmap instructionsInvert; //遊戲說明的顏色反相按鈕圖  const int STATION\_APPERAED\_TIME = 5; //車站出現的秒數  const int PASSENAGER\_APPERAED\_TIME = 3; //乘客出現的秒數  const int MAXIUM\_STATION = 6; //會出現在地圖上車站總數  const int MAXIUM\_STATION\_TYPE = 6; //車站樣式總共有6個 00~05  const int LINE\_COLOR\_NUM = 7; //線路樣式總共有7個 00~06  const int INIT\_PASSANGER = MAXIUM\_STATION \* 2; //乘客初始個數  const int MAX\_PASSENGER\_NUM = 6; //車廂最多六個乘客  const int LINE\_CAN\_USE\_TIME = 5; //可以拉出其他顏色的時間  //station.h裡面也有相同的屬性要改  const int MAX\_GAME\_MAP\_SIDE\_X = 770 - 135; //實際上的遊戲邊界X軸只有到770  const int MIN\_GAME\_MAP\_SIDE\_X = 30; //實際上的遊戲邊界X軸從30開始  const int MAX\_GAME\_MAP\_SIDE\_Y = 560 - 25; //實際上的遊戲邊界Y軸只有到560  const int MIN\_GAME\_MAP\_SIDE\_Y = 60; //實際上的遊戲邊界Y軸從60開始  bool isStop; //判斷是否暫停  bool isStop2; //判斷是否進選單  bool isMouseInRestartBtn; //滑鼠在重新開始按鈕  bool isMouseInEndBtn; //滑鼠在結束按鈕  bool isMouseInNightBtn; //滑鼠在模式按鈕  bool isMouseInContinuBtn; //滑鼠在繼續按紐  bool red; //是否選紅色鐵軌  bool orange; //是否選橘色鐵軌  bool yellow; //是否選黃色鐵軌  bool green; //是否選綠色鐵軌  bool blue; //是否選藍色鐵軌  bool bblue; //是否選深藍色鐵軌  bool purple; //是否選紫色鐵軌  bool isMouseInInstructionsBtn; //滑鼠在遊戲說明按鈕  bool isInInstructions; //判斷是否在遊戲說明  int totalper = 0;  int currentStationNum; //目前出現到哪個車站  int currentPassenagerNum; //目前有幾個乘客出現  int counter;  int mouse\_state = 1;  int page = 0;  int clickedX;  int clickedY;  int preP = 0, nextP = 0, nextS = 0;  Line\* line; //負責處理火車線路的指標  Line redLine; //紅色線路  Line orangeLine; //橘色線路  Line yellowLine; //黃色線路  Line greenLine; //綠色線路  Line blueLine; //藍色線路  Line bblueLine; //靛色線路  Line purpleLine; //紫色線路  vector<CMovingBitmap> garyIcon; //負責蓋住其他顏色的圖標  Clock clock; //會動的時鐘  vector<Station> stationList; //一堆的車站  vector<Passenger\*> passengerList; //一堆的乘客  vector<Passenger\*> passengerListOnCabin[7];  CInteger passengerTotalNumber; //乘客人數  vector<Cabin> cabinList;  int redCabinState = 0;  int orangCabinState = 0;  int yellowCabinState = 0;  int greenCabinState = 0;  int blueCabinState = 0;  int bblueCabinState = 0;  int pupleCabinState = 0;  int mouse\_x = 0, mouse\_y = 0; //滑鼠位置  };  /////////////////////////////////////////////////////////////////////////////  // 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();  protected:  void OnMove(); // 移動遊戲元素  void OnShow(); // 顯示這個狀態的遊戲畫面  private:  int counter; // 倒數之計數器  CMovingBitmap endl; //結束畫面  CMovingBitmap mapNight; //夜間模式背景  CMovingBitmap endlNight; //夜間模式結束畫面  CMovingBitmap sp4; //評語  CMovingBitmap sp4Night; //評語  CMovingBitmap good; //評語  CMovingBitmap goodNight; //評語  CMovingBitmap bad; //評語  CMovingBitmap badNight; //評語  };  } |

|  |
| --- |
| mygame.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <mmsystem.h>  #include <ddraw.h>  #include "audio.h"  #include "gamelib.h"  #include <cstdlib>  #include <ctime>  #include <vector>  #include "mygame.h"  namespace game\_framework  {  int personss = 0; //運輸人數  bool isNight = false; //判斷是否夜間模式  /////////////////////////////////////////////////////////////////////////////  // 這個class為遊戲的遊戲開頭畫面物件  /////////////////////////////////////////////////////////////////////////////  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  //  // 載入音效  CAudio::Instance()->Load(AUDIO\_SLIP, "sounds\\slip.mp3");  CAudio::Instance()->Load(AUDIO\_CLICK, "sounds\\click.mp3");  CAudio::Instance()->Load(AUDIO\_MUSIC, "sounds\\music.mp3");  // 載入圖片  logo.LoadBitmap(".\\RES\\title.bmp");  logoNight.LoadBitmap(".\\RES\\titlenight.bmp", RGB(0, 0, 0));  nightMap.LoadBitmap(".\\RES\\backgroundnight.bmp");  page1.LoadBitmap(".\\RES\\Instruction.bmp");  page2.LoadBitmap(".\\RES\\Instruction2.bmp");  start.LoadBitmap(".\\RES\\button\\start.bmp");  startInvert.LoadBitmap(".\\RES\\button\\startInvert.bmp");  night.LoadBitmap(".\\RES\\button\\night.bmp");  nightInvert.LoadBitmap(".\\RES\\button\\nightInvert.bmp");  morning.LoadBitmap(".\\RES\\button\\morning.bmp");  morningInvert.LoadBitmap(".\\RES\\button\\morningInvert.bmp");  instructions.LoadBitmap(".\\RES\\button\\Instructions.bmp");  instructionsInvert.LoadBitmap(".\\RES\\button\\InstructionsInvert.bmp");  end.LoadBitmap(".\\RES\\button\\end.bmp");  endInvert.LoadBitmap(".\\RES\\button\\endInvert.bmp");  red.LoadBitmap(".\\RES\\Cover\\red.bmp", RGB(255, 255, 255));  orange.LoadBitmap(".\\RES\\Cover\\orange.bmp", RGB(255, 255, 255));  blue.LoadBitmap(".\\RES\\Cover\\blue.bmp", RGB(255, 255, 255));  bblue.LoadBitmap(".\\RES\\Cover\\bblue.bmp", RGB(255, 255, 255));  redCabin.LoadBitmap(".\\RES\\Cabin\\Cabin\_rad2.bmp", RGB(255, 255, 255));  redCabin2.LoadBitmap(".\\RES\\Cabin\\Cabin\_rad3.bmp", RGB(255, 255, 255));  orangeCabin.LoadBitmap(".\\RES\\Cabin\\Cabin\_orange2.bmp", RGB(255, 255, 255));  orangeCabin2.LoadBitmap(".\\RES\\Cabin\\Cabin\_orange3.bmp", RGB(255, 255, 255));  blueCabin.LoadBitmap(".\\RES\\Cabin\\Cabin\_blue2.bmp", RGB(255, 255, 255));  blueCabin2.LoadBitmap(".\\RES\\Cabin\\Cabin\_blue3.bmp", RGB(255, 255, 255));  bblueCabin.LoadBitmap(".\\RES\\Cabin\\Cabin\_bblue2.bmp", RGB(255, 255, 255));  bblueCabin2.LoadBitmap(".\\RES\\Cabin\\Cabin\_bblue3.bmp", RGB(255, 255, 255));  //設定參數  CAudio::Instance()->Play(AUDIO\_MUSIC);  isMouseInStartBtn = false;  isMouseInNightBtn = false;  isMouseInInstructionsBtn = false;  isMouseInEndBtn = false;  isInInstructions = false;  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%  //  }  void CGameStateInit::OnBeginState()  {  isInInstructions = false;  page = 0;  }  void CGameStateInit::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)  {  const char KEY\_ESC = 27;  const char KEY\_SPACE = ' ';  const char KEY\_RIGHT = 0x27; // keyboard右箭頭  if (nChar == KEY\_RIGHT)  {  if (isInInstructions)  {  if (page == 2)  {  page = 0;  isInInstructions = !isInInstructions;  }  else  {  page++;  }  }  }  /\*if (nChar == KEY\_SPACE)  GotoGameState(GAME\_STATE\_RUN); // 切換至GAME\_STATE\_RUN  else if (nChar == KEY\_ESC) // Demo 關閉遊戲的方法  PostMessage(AfxGetMainWnd()->m\_hWnd, WM\_CLOSE, 0, 0);// 關閉遊戲\*/  }  void CGameStateInit::OnLButtonDown(UINT nFlags, CPoint point)  {  if (start.IsBitmapLoaded() && end.IsBitmapLoaded())  {  if (mouse\_x >= start.Left() && mouse\_x <= start.Left() + start.Width() && mouse\_y >= start.Top() && mouse\_y <= start.Top() + start.Height())  {  CAudio::Instance()->Play(AUDIO\_CLICK);  }  else if (mouse\_x >= end.Left() && mouse\_x <= end.Left() + end.Width() && mouse\_y >= end.Top() && mouse\_y <= end.Top() + end.Height())  {  CAudio::Instance()->Play(AUDIO\_CLICK);  }  }  }  void CGameStateInit::OnLButtonUp(UINT nFlags, CPoint point)  {  if (isInInstructions)  {  }  else  {  if (start.IsBitmapLoaded() && end.IsBitmapLoaded())  {  if (isMouseInStartBtn)  {  GotoGameState(GAME\_STATE\_RUN); // 切換至GAME\_STATE\_RUN  }  else if (isMouseInEndBtn)  {  PostMessage(AfxGetMainWnd()->m\_hWnd, WM\_CLOSE, 0, 0);// 關閉遊戲  }  else if (isMouseInNightBtn)  {  isNight = !isNight;  }  else if (isMouseInInstructionsBtn)  {  page = 1;  isInInstructions = !isInInstructions;  }  }  }  }  void CGameStateInit::OnMouseMove(UINT nFlags, CPoint point)  {  mouse\_x = point.x;  mouse\_y = point.y;  if (!isInInstructions)  {  if (start.IsBitmapLoaded() && end.IsBitmapLoaded())  {  if (mouse\_x > start.Left() && mouse\_x < start.Left() + start.Width() &&  mouse\_y > start.Top() && mouse\_y < start.Top() + start.Height())  {  if (mouse\_state == 1)  {  isMouseInStartBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else if (mouse\_x > end.Left() && mouse\_x < end.Left() + end.Width() &&  mouse\_y > end.Top() && mouse\_y < end.Top() + end.Height())  {  if (mouse\_state == 1)  {  isMouseInEndBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else if (mouse\_x > night.Left() && mouse\_x < night.Left() + night.Width() &&  mouse\_y > night.Top() && mouse\_y < night.Top() + night.Height())  {  if (mouse\_state == 1)  {  isMouseInNightBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else if (mouse\_x > instructions.Left() && mouse\_x < instructions.Left() + instructions.Width() &&  mouse\_y > instructions.Top() && mouse\_y < instructions.Top() + instructions.Height())  {  if (mouse\_state == 1)  {  isMouseInInstructionsBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else  {  mouse\_state = 1;  isMouseInNightBtn = false;  isMouseInStartBtn = false;  isMouseInInstructionsBtn = false;  isMouseInEndBtn = false;  }  }  }  }  void CGameStateInit::OnMove()  {  if (redCabinY > 800)  {  redCabinX = -50;  redCabinY = 270;  }  else if (redCabinY < 375)  {  redCabinY += 2;  redCabinX += 2;  }  else  {  redCabinY += 2;  }  if (orangeCabinY > 333)  {  orangeCabinY -= 2;  }  else if (orangeCabinY < 334 && orangeCabinX > -100)  {  orangeCabinX -= 2;  orangeCabinY -= 2;  }  else if (orangeCabinX <= -100)  {  orangeCabinX = 38;  orangeCabinY = 800;  }  if (bblueCabinY > 425)  {  bblueCabinY -= 2;  }  else if (bblueCabinY <= 424 && bblueCabinX < 850)  {  bblueCabinX += 2;  bblueCabinY -= 2;  }  else if (bblueCabinX >= 850)  {  bblueCabinX = 675;  bblueCabinY = 1000;  }  if (blueCabinY > 70)  {  blueCabinY -= 2;  }  else if (blueCabinY <= 70 && blueCabinX < 1500)  {  blueCabinX += 2;  blueCabinY -= 2;  }  else if (blueCabinX >= 1500)  {  blueCabinX = 694;  blueCabinY = 600;  }  }  void CGameStateInit::OnShow()  {  //設定Logo 位置 和顯示  logo.SetTopLeft((SIZE\_X / 2) - (logo.Width() / 2), SIZE\_Y / 8 \* 1);  logoNight.SetTopLeft((SIZE\_X / 2) - (logo.Width() / 2), SIZE\_Y / 8 \* 1);  //開始按鈕 結束按鈕的位置  start.SetTopLeft((SIZE\_X / 2) - (start.Width() / 2), SIZE\_Y / 8 \* 3);  night.SetTopLeft((SIZE\_X / 2) - (start.Width() / 2), SIZE\_Y / 8 \* 4);  morning.SetTopLeft((SIZE\_X / 2) - (start.Width() / 2), SIZE\_Y / 8 \* 4);  instructions.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 5);  end.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 6);  //反相 開始按鈕 結束按鈕的位置  startInvert.SetTopLeft((SIZE\_X / 2) - (start.Width() / 2), SIZE\_Y / 8 \* 3);  nightInvert.SetTopLeft((SIZE\_X / 2) - (start.Width() / 2), SIZE\_Y / 8 \* 4);  morningInvert.SetTopLeft((SIZE\_X / 2) - (start.Width() / 2), SIZE\_Y / 8 \* 4);  instructionsInvert.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 5);  endInvert.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 6);  //顯示 開始按鈕 結束按鈕  if (!isNight)  {  logo.ShowBitmap();  night.ShowBitmap();  }  else  {  nightMap.ShowBitmap();  morning.ShowBitmap();  logoNight.ShowBitmap();  }  start.ShowBitmap();  instructions.ShowBitmap();  end.ShowBitmap();  //設定鐵軌位置 和顯示  orange.SetTopLeft(0, 280);  orange.ShowBitmap();  if (orangeCabinY > 333)  {  orangeCabin.SetTopLeft(orangeCabinX, orangeCabinY);  orangeCabin.ShowBitmap();  }  else  {  orangeCabin2.SetTopLeft(orangeCabinX, orangeCabinY);  orangeCabin2.ShowBitmap();  }  red.SetTopLeft(0, 307);  red.ShowBitmap();  if (redCabinY < 375)  {  redCabin2.SetTopLeft(redCabinX, redCabinY);  redCabin2.ShowBitmap();  }  else  {  redCabin.SetTopLeft(redCabinX, redCabinY);  redCabin.ShowBitmap();  }  blue.SetTopLeft(699, 0);  blue.ShowBitmap();  if (blueCabinY < 70)  {  blueCabin2.SetTopLeft(blueCabinX, blueCabinY);  blueCabin2.ShowBitmap();  }  else  {  blueCabin.SetTopLeft(694, blueCabinY);  blueCabin.ShowBitmap();  }  bblue.SetTopLeft(691, 337);  bblue.ShowBitmap();  if (bblueCabinY < 425)  {  bblueCabin2.SetTopLeft(bblueCabinX, bblueCabinY);  bblueCabin2.ShowBitmap();  }  else  {  bblueCabin.SetTopLeft(bblueCabinX, bblueCabinY);  bblueCabin.ShowBitmap();  }  //  // 滑鼠移到上面 讓按鈕變色  //  if (isMouseInStartBtn)  {  startInvert.ShowBitmap();  }  else if (isMouseInEndBtn)  {  endInvert.ShowBitmap();  }  else if (isMouseInNightBtn && !isNight)  {  nightInvert.ShowBitmap();  }  else if (isMouseInNightBtn && isNight)  {  morningInvert.ShowBitmap();  }  else if (isMouseInInstructionsBtn)  {  instructionsInvert.ShowBitmap();  }  if (isInInstructions)  {  if (page == 1)  {  page1.SetTopLeft(0, 0);  page1.ShowBitmap();  }  else if (page == 2)  {  page2.SetTopLeft(0, 0);  page2.ShowBitmap();  }  }  //  // Demo螢幕字型的使用，不過開發時請盡量避免直接使用字型，改用CMovingBitmap比較好  //  /\*  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));  //pDC->TextOut(150, 275, "Please click mouse or press SPACE to begin.");  pDC->TextOut(5, 493, "Press Ctrl-F to switch in between window mode and full screen mode.");  char t[30];  sprintf(t, "(%d,%d)", mouse\_x, mouse\_y);  pDC->TextOut(10, 10, t);  if (ENABLE\_GAME\_PAUSE)  pDC->TextOut(6, 531, "Press Ctrl-Q to pause the Game.");  pDC->TextOut(6, 569, "Press Alt-F4 or ESC to Quit.");  pDC->SelectObject(fp); // 放掉 font f (千萬不要漏了放掉)  CDDraw::ReleaseBackCDC(); // 放掉 Back Plain 的 CDC  \*/  }  /////////////////////////////////////////////////////////////////////////////  // 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)  GotoGameState(GAME\_STATE\_INIT);  }  void CGameStateOver::OnBeginState()  {  counter = 30 \* 5; // 5 seconds  }  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(66); // 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  endl.LoadBitmap(".\\RES\\Gameover\\gameover.bmp");  endlNight.LoadBitmap(".\\RES\\Gameover\\gameovernight.bmp", RGB(0, 0, 0));  mapNight.LoadBitmapA(".\\RES\\backgroundnight.bmp");  sp4.LoadBitmap(".\\RES\\Gameover\\sp4.bmp", RGB(255, 255, 255));  sp4Night.LoadBitmap(".\\RES\\Gameover\\sp4night.bmp");  good.LoadBitmap(".\\RES\\Gameover\\good.bmp", RGB(255, 255, 255));  goodNight.LoadBitmap(".\\RES\\Gameover\\goodnight.bmp");  bad.LoadBitmap(".\\RES\\Gameover\\bad.bmp", RGB(255, 255, 255));  badNight.LoadBitmap(".\\RES\\Gameover\\badnight.bmp");  //  // 最終進度為100%  //  ShowInitProgress(100);  }  void CGameStateOver::OnShow()  {  endl.SetTopLeft((SIZE\_X / 2) - (endl.Width() / 2), SIZE\_Y / 8 \* 1);  endlNight.SetTopLeft((SIZE\_X / 2) - (endl.Width() / 2), SIZE\_Y / 8 \* 1);  mapNight.SetTopLeft(0, 0);  sp4.SetTopLeft(0, 0);  sp4Night.SetTopLeft(0, 0);  good.SetTopLeft(0, 0);  goodNight.SetTopLeft(0, 0);  bad.SetTopLeft(0, 0);  badNight.SetTopLeft(0, 0);  if (personss <= 30)  {  if (isNight)  {  sp4Night.ShowBitmap();  endlNight.ShowBitmap();  }  else  {  endl.ShowBitmap();  sp4.ShowBitmap();  }  }  else if (personss > 30 && personss <= 50)  {  if (isNight)  {  goodNight.ShowBitmap();  endlNight.ShowBitmap();  }  else  {  endl.ShowBitmap();  good.ShowBitmap();  }  }  else if (personss > 50)  {  if (isNight)  {  badNight.ShowBitmap();  endlNight.ShowBitmap();  }  else  {  endl.ShowBitmap();  bad.ShowBitmap();  }  }  /\*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\*/  }  /////////////////////////////////////////////////////////////////////////////  // 這個class為遊戲的遊戲執行物件，主要的遊戲程式都在這裡  /////////////////////////////////////////////////////////////////////////////  CGameStateRun::CGameStateRun(CGame\* g)  : CGameState(g)  {  }  CGameStateRun::~CGameStateRun()  {  for (unsigned i = 0; i < passengerList.size(); i++)  {  if (passengerList[i] != nullptr)  {  delete passengerList[i];  }  }  }  void CGameStateRun::OnBeginState()  {  isInInstructions = false;  isMouseInInstructionsBtn = false;  page = 0;  totalper = 0;  passengerTotalNumber.SetIsNight(isNight);  red = true;  orange = false;  yellow = false;  green = false;  blue = false;  bblue = false;  purple = false;  const int HITS\_LEFT = 0;  const int HITS\_LEFT\_X = 680;  const int HITS\_LEFT\_Y = 20;  passengerTotalNumber.SetInteger(HITS\_LEFT); // 指定剩下的撞擊數  passengerTotalNumber.SetTopLeft(HITS\_LEFT\_X, HITS\_LEFT\_Y);  redLine.Reset();  orangeLine.Reset();  yellowLine.Reset();  greenLine.Reset();  blueLine.Reset();  bblueLine.Reset();  purpleLine.Reset();  redLine.SetLineColor(255, 0, 0);  redLine.SetIsCanbeClicked(true);  orangeLine.SetLineColor(255, 144, 0);  yellowLine.SetLineColor(255, 255, 0);  greenLine.SetLineColor(0, 255, 0);  blueLine.SetLineColor(0, 138, 255);  bblueLine.SetLineColor(0, 6, 255);  purpleLine.SetLineColor(144, 0, 255);  for (int i = 0; i < 7; i++)  {  passengerListOnCabin[i].clear();  }  vector <Cabin> a;  cabinList.assign(a.begin(), a.end());  Station\* s = new Station; //用來建立隨機車站列表 及 檢查車站 列表 是否重疊  Passenger\* p = new Passenger; //用來建立隨機乘客個數 及 隨機列表  line = &redLine;  counter = 0;  //建立隨機車站列表  s->RandomBuildStation(stationList);  //檢查車站列表是否有重疊的車站  s->CheckedOverLappingStation(stationList);  s->SetIsNight(isNight);  //建立乘客列表  p->RandomMadePassenger(passengerList, stationList, MAXIUM\_STATION, MAXIUM\_STATION\_TYPE, INIT\_PASSANGER);  p->SetIsNight(isNight);  garyIcon[0].SetTopLeft(285, 570);  garyIcon[1].SetTopLeft(330, 570);  garyIcon[2].SetTopLeft(375, 570);  garyIcon[3].SetTopLeft(420, 570);  garyIcon[4].SetTopLeft(465, 570);  garyIcon[5].SetTopLeft(510, 570);  chooseColor.SetTopLeft(240, 570);  chooseColorNight.SetTopLeft(240, 570);  currentStationNum = 3; //現有車站為三個 遊戲開始 有三個車站  currentPassenagerNum = 0; //一開始出現的乘客數為0  clickedX = clickedY = -1;  delete s;  delete p;  }  void CGameStateRun::OnMove() // 移動遊戲元素  {  if (!isStop)  {  counter++;  clock.OnMove(); //播放clock時鐘動畫  vector<int> pointX;  vector<int> pointY;  vector<int> passedStation;  /\*每 STATION\_APPERAED\_TIME 秒就出一個車站  if (counter % (30 \* STATION\_APPERAED\_TIME) == 0 && currentStationNum < MAXIUM\_STATION)  {  currentStationNum++;  //CAudio::Instance()->Play(AUDIO\_APPEAR);  }  \*/  //每 PASSENAGER\_APPERAED\_TIME 秒就出一個乘客  if (counter % (30 \* PASSENAGER\_APPERAED\_TIME) == 0)  {  Passenger\* p = new Passenger;  p->RandomOnePassenger(passengerList, stationList, MAXIUM\_STATION, MAXIUM\_STATION\_TYPE);  delete p;  }  if (counter >= LINE\_CAN\_USE\_TIME \* 2 \* 30)  orangeLine.SetIsCanbeClicked(true);  if (counter >= LINE\_CAN\_USE\_TIME \* 3 \* 30)  yellowLine.SetIsCanbeClicked(true);  if (counter >= LINE\_CAN\_USE\_TIME \* 4 \* 30)  greenLine.SetIsCanbeClicked(true);  if (counter >= LINE\_CAN\_USE\_TIME \* 5 \* 30)  blueLine.SetIsCanbeClicked(true);  if (counter >= LINE\_CAN\_USE\_TIME \* 6 \* 30)  bblueLine.SetIsCanbeClicked(true);  if (counter >= LINE\_CAN\_USE\_TIME \* 7 \* 30)  purpleLine.SetIsCanbeClicked(true);  if (line->IsClickedTwoStation())  {  int R, G, B;  int countPassedStation[6] = { 0 };  Cabin c ;  line->SetPassedStation(line->GetClickedStartStationNum(), line->GetClickedEndStationNum());  line->SetLinePointXY(stationList);  line->SetClickedStartStationNum(-1);  line->SetClickedEndStationNum(-1);  line->GetLinePointXY(pointX, pointY);  line->GetPassedStationNum(passedStation);  line->GetLineColorRGB(R, G, B);  line->CountTotalPassedStation();  line->GetTotalPassedStation(countPassedStation);  if (!c.IsCabinColorRepeat(cabinList, R, G, B))  {  Cabin buf(pointX[0], pointY[0], R, G, B);  buf.SetLinePoint(pointX, pointY);  buf.SetPassedStation(passedStation);  buf.SetLineStationNum(countPassedStation);  cabinList.push\_back(buf);  }  else  {  int pos = c.GetCabinPointer(cabinList, R, G, B);  cabinList[pos].SetLinePoint(pointX, pointY);  cabinList[pos].SetPassedStation(passedStation);  cabinList[pos].SetLineStationNum(countPassedStation);  }  }  if (!cabinList.empty())  {  for (unsigned i = 0; i < cabinList.size(); i++) //每個車廂  {  cabinList[i].OnMove(stationList);  if (cabinList[i].IsStop()) //火車進站囉  {  //乘客上下車  cabinList[i].SetAddCounter(1);  int nowCabinInStationNum = cabinList[i].GetNextStation(); //目前車廂在哪個車站  //先下車後上車  //========下車========  if ((int)passengerListOnCabin[i].size() > 0 && cabinList[i].GetCounter() == 10)  {  cabinList[i].PassengerGetOut(passengerListOnCabin[i], nowCabinInStationNum);  //cabinList[i].PassengerGetOut(nowCabinInStationNum);  totalper += cabinList[i].GetOutPeople();  passengerTotalNumber.Add(cabinList[i].GetOutPeople());  }  //========上車========  if (cabinList[i].GetCounter() == 20)  {  for (unsigned j = passengerList.size() - 1; j >= 0; j--) //每個乘客  {  if (j > passengerList.size())  {  break;  }  else if ((int)passengerListOnCabin[i].size() < MAX\_PASSENGER\_NUM) //車廂乘客小於 六個  {  int cabinPassedStation[6] = { 0 }; //車廂會經過的站  cabinList[i].GetLineStationNum(cabinPassedStation);  if (cabinPassedStation[passengerList[j]->GetFinalStation()] == 1 && passengerList[j]->GetStartStation() == nowCabinInStationNum)  //確定乘客要去的站有在這條線路上  {  //cabinList[i].PassengerGetOn(passengerList[j]);  passengerListOnCabin[i].push\_back(passengerList[j]);  cabinList[i].SetPassengerPosition(passengerListOnCabin[i]);  stationList[nowCabinInStationNum].ErasePassenger(passengerList[j]);  passengerList.erase(passengerList.begin() + j);  }  }  }  }  if (cabinList[i].GetCounter() == 30)  {  cabinList[i].SetCounter(0);  cabinList[i].SetIsStop(false);  }  }  }  for (unsigned i = 0; i < stationList.size(); i++)  {  stationList[i].RefreshPassengerPosition();  }  }  if (counter == 120 \* 30)  {  personss = totalper;  GotoGameState(GAME\_STATE\_OVER);  }  }  }  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  //  Station\* s = new Station; //用來建立隨機車站列表 及 檢查車站 列 表 是否重疊  instructions.LoadBitmap(".\\RES\\button\\Instructions.bmp");  instructionsInvert.LoadBitmap(".\\RES\\button\\InstructionsInvert.bmp");  page1.LoadBitmap(".\\RES\\Instruction.bmp");  page2.LoadBitmap(".\\RES\\Instruction2.bmp");  clock.LoadBitmap();  esc.LoadBitmap(".\\RES\\esc.bmp", RGB(255, 255, 255));  esc.SetTopLeft(700, 565);  escNight.LoadBitmap(".\\RES\\escnight.bmp", RGB(0, 0, 0));  escNight.SetTopLeft(700, 565);  passengerTotalNumber.LoadBitmap();  mapnight.LoadBitmap(".\\RES\\backgroundnight.bmp");  map.LoadBitmap(".\\RES\\map.bmp");  nightMap.LoadBitmap(".\\RES\\mapNight.bmp");  person.LoadBitmap(".\\RES\\person.bmp", RGB(255, 255, 255));  personNight.LoadBitmap(".\\RES\\personnight.bmp", RGB(0, 0, 0));  backGround.LoadBitmap(".\\RES\\background.bmp");  continu.LoadBitmap(".\\RES\\button\\continu.bmp");  continuInvert.LoadBitmap(".\\RES\\button\\continuInvert.bmp");  restart.LoadBitmap(".\\RES\\button\\restart.bmp");  restartInvert.LoadBitmap(".\\RES\\button\\restartInvert.bmp");  night.LoadBitmap(".\\RES\\button\\night.bmp");  nightInvert.LoadBitmap(".\\RES\\button\\nightInvert.bmp");  morning.LoadBitmap(".\\RES\\button\\morning.bmp");  morningInvert.LoadBitmap(".\\RES\\button\\morningInvert.bmp");  night.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 4);  morning.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 4);  nightInvert.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 4);  morningInvert.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 4);  restart.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 3);  restartInvert.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 3);  continu.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 2);  continuInvert.SetTopLeft((SIZE\_X / 2) - (restart.Width() / 2), SIZE\_Y / 8 \* 2);  end.LoadBitmap(".\\RES\\button\\end.bmp");  endInvert.LoadBitmap(".\\RES\\button\\endInvert.bmp");  instructions.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 5);  instructionsInvert.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 5);  end.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 6);  endInvert.SetTopLeft((SIZE\_X / 2) - (end.Width() / 2), SIZE\_Y / 8 \* 6);  isStop = false;  isStop2 = false;  isMouseInRestartBtn = false;  isMouseInEndBtn = false;  CAudio::Instance()->Load(AUDIO\_STOP1, "sounds\\stop1.mp3");  CAudio::Instance()->Load(AUDIO\_STOP2, "sounds\\stop2.mp3");  CAudio::Instance()->Load(AUDIO\_COLOR, "sounds\\odd.wav");  /\*red(255.0.0),orang(255.144.0),yellow(255.255.0),green(0.255.0),blue(0.138.255),bblue(0.6.255),puple(144.0.255)\*/  redLine.SetLineColor(255, 0, 0);  redLine.SetIsCanbeClicked(true);  orangeLine.SetIsCanbeClicked(true);  yellowLine.SetIsCanbeClicked(true);  greenLine.SetIsCanbeClicked(true);  blueLine.SetIsCanbeClicked(true);  bblueLine.SetIsCanbeClicked(true);  purpleLine.SetIsCanbeClicked(true);  orangeLine.SetLineColor(255, 144, 0);  yellowLine.SetLineColor(255, 255, 0);  greenLine.SetLineColor(0, 255, 0);  blueLine.SetLineColor(0, 138, 255);  bblueLine.SetLineColor(0, 6, 255);  purpleLine.SetLineColor(144, 0, 255);  ShowInitProgress(40);  redLine.LoadBitmap();  orangeLine.LoadBitmap();  yellowLine.LoadBitmap();  greenLine.LoadBitmap();  blueLine.LoadBitmap();  bblueLine.LoadBitmap();  purpleLine.LoadBitmap();  chooseColor.LoadBitmap(".\\RES\\color\\choosecolor.bmp", RGB(255, 255, 255));  chooseColorNight.LoadBitmap(".\\RES\\color\\choosecolornight.bmp", RGB(0, 0, 0));  for (unsigned i = 0; i < 6; i++)  {  CMovingBitmap a;  a.LoadBitmap(".\\RES\\color\\gray.bmp", RGB(255, 255, 255));  garyIcon.push\_back(a);  }  //建立隨機車站列表  s->RandomBuildStation(stationList);  //檢查車站列表是否有重疊的車站  s->CheckedOverLappingStation(stationList);  //載入各車站圖片  ShowInitProgress(50);  CAudio::Instance()->Load(AUDIO\_SELECT1, "sounds\\select1.mp3");  CAudio::Instance()->Load(AUDIO\_SELECT2, "sounds\\select2.mp3");  CAudio::Instance()->Load(AUDIO\_APPEAR, "sounds\\appear.mp3");  delete s;  //Sleep(300); // 放慢，以便看清楚進度，實際遊戲請刪除此Sleep  //  // 繼續載入其他資料  //  //  // 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)  {  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  }  void CGameStateRun::OnKeyUp(UINT nChar, UINT nRepCnt, UINT nFlags)  {  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  const char KEY\_ESC = 27;  if (nChar == KEY\_ESC)  {  if (!isStop2 && !isStop)  {  isStop2 = !isStop2;  isStop = !isStop;  }  }  if (nChar == KEY\_UP) //人數加十  {  if (!isStop && !isStop2)  {  if (passengerTotalNumber.GetInteger() < 999)  {  if (passengerTotalNumber.GetInteger() == 990)  {  totalper += 9;  passengerTotalNumber.Add(9);  }  else  {  totalper += 10;  passengerTotalNumber.Add(10);  }  }  }  }  if (isInInstructions)  {  if (nChar == KEY\_RIGHT)  {  if (isInInstructions)  {  if (page == 2)  {  page = 0;  isInInstructions = !isInInstructions;  }  else  {  page++;  }  }  }  }  if (nChar == KEY\_DOWN) //快速結算  {  if (!isStop && !isStop2)  {  counter = 120 \* 30 - 1;  }  }  }  void CGameStateRun::OnLButtonDown(UINT nFlags, CPoint point)  {  // 這邊負責處理按下滑鼠左鍵的動作  int start = line->GetClickedStartStationNum(); //紀錄起始車站  int end = line->GetClickedEndStationNum(); //紀錄終點車站  if (!isStop)  {  if (start == -1) //沒有任何車站被選取  {  if (line->IsClickedStation(point.x, point.y, stationList, MAXIUM\_STATION)) //檢查是否有點到車站  {  CAudio::Instance()->Play(AUDIO\_SELECT1);  start = line->GetMouseClickedStationNum(point.x, point.y, stationList, MAXIUM\_STATION);  line->SetClickedStartStationNum(start);  if (!(line->IsPassedStationEmpty() || line->GetClickedFirstStation() == line->GetClickedStartStationNum() || line->GetClickedLastStation() == line->GetClickedStartStationNum()))  {  line->SetClickedStartStationNum(-1);  line->SetClickedEndStationNum(-1);  }  }  else  {  line->SetClickedStartStationNum(-1);  line->SetClickedEndStationNum(-1);  }  }  else //如果起點車站已經被選取  {  if (line->IsClickedStation(point.x, point.y, stationList, MAXIUM\_STATION)) //檢查是否有點到車站  {  CAudio::Instance()->Play(AUDIO\_SELECT2);  end = line->GetMouseClickedStationNum(point.x, point.y, stationList, MAXIUM\_STATION);  line->SetClickedEndStationNum(end);  }  else  {  line->SetClickedStartStationNum(-1);  line->SetClickedEndStationNum(-1);  }  }  }  if (!isStop2)  {  if (point.x >= 750 && point.y >= 10 && point.x <= 790 && point.y <= 50)  {  CAudio::Instance()->Play(AUDIO\_SLIP);  if (isStop)  {  CAudio::Instance()->Play(AUDIO\_STOP1);  }  else  {  CAudio::Instance()->Play(AUDIO\_STOP2);  }  isStop = !isStop;  }  }  clickedX = point.x; //用來debug  clickedY = point.y; //用來debug  }  void CGameStateRun::OnLButtonUp(UINT nFlags, CPoint point) // 處理滑鼠左鍵的動作 放開  {  //換鐵軌的顏色  if (redLine.IsMouseClickedLineColorBMP(point.x, point.y))  {  line = &redLine;  CAudio::Instance()->Play(AUDIO\_COLOR);  red = true;  orange = false;  yellow = false;  green = false;  blue = false;  bblue = false;  purple = false;  }  else if (orangeLine.IsMouseClickedLineColorBMP(point.x, point.y))  {  line = &orangeLine;  CAudio::Instance()->Play(AUDIO\_COLOR);  red = false;  orange = true;  yellow = false;  green = false;  blue = false;  bblue = false;  purple = false;  }  else if (yellowLine.IsMouseClickedLineColorBMP(point.x, point.y))  {  line = &yellowLine;  CAudio::Instance()->Play(AUDIO\_COLOR);  red = false;  orange = false;  yellow = true;  green = false;  blue = false;  bblue = false;  purple = false;  }  else if (greenLine.IsMouseClickedLineColorBMP(point.x, point.y))  {  line = &greenLine;  CAudio::Instance()->Play(AUDIO\_COLOR);  red = false;  orange = false;  yellow = false;  green = true;  blue = false;  bblue = false;  purple = false;  }  else if (blueLine.IsMouseClickedLineColorBMP(point.x, point.y))  {  line = &blueLine;  CAudio::Instance()->Play(AUDIO\_COLOR);  red = false;  orange = false;  yellow = false;  green = false;  blue = true;  bblue = false;  purple = false;  }  else if (bblueLine.IsMouseClickedLineColorBMP(point.x, point.y))  {  line = &bblueLine;  CAudio::Instance()->Play(AUDIO\_COLOR);  red = false;  orange = false;  yellow = false;  green = false;  blue = false;  bblue = true;  purple = false;  }  else if (purpleLine.IsMouseClickedLineColorBMP(point.x, point.y))  {  line = &purpleLine;  CAudio::Instance()->Play(AUDIO\_COLOR);  red = false;  orange = false;  yellow = false;  green = false;  blue = false;  bblue = false;  purple = true;  }  if (isStop2)  {  if (!isInInstructions)  {  if (restart.IsBitmapLoaded() && restartInvert.IsBitmapLoaded())  {  if (isMouseInRestartBtn)  {  isStop = !isStop;  isStop2 = !isStop2;  GotoGameState(GAME\_STATE\_INIT);  }  else if (isMouseInEndBtn)  {  PostMessage(AfxGetMainWnd()->m\_hWnd, WM\_CLOSE, 0, 0); // 關閉遊戲  }  else if (isMouseInNightBtn)  {  isNight = !isNight;  }  else if (isMouseInContinuBtn)  {  isStop = !isStop;  isStop2 = !isStop2;  }  else if (isMouseInInstructionsBtn)  {  isInInstructions = !isInInstructions;  page = 1;  }  }  }  }  }  void CGameStateRun::OnMouseMove(UINT nFlags, CPoint point) // 處理滑鼠左鍵的動作 移動  {  // 沒事。如果需要處理滑鼠移動的話，寫code在這裡  if (point.x > MIN\_GAME\_MAP\_SIDE\_X && point.x < MAX\_GAME\_MAP\_SIDE\_X + 135) mouse\_x = point.x;  if (point.y > MIN\_GAME\_MAP\_SIDE\_Y && point.y < MAX\_GAME\_MAP\_SIDE\_Y) mouse\_y = point.y;  if (!isInInstructions && isStop2)  {  if (restart.IsBitmapLoaded() && restartInvert.IsBitmapLoaded())  {  if (mouse\_x > restart.Left() && mouse\_x < restart.Left() + restart.Width() &&  mouse\_y > restart.Top() && mouse\_y < restart.Top() + restart.Height())  {  if (mouse\_state == 1)  {  isMouseInRestartBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else if (mouse\_x > end.Left() && mouse\_x < end.Left() + end.Width() &&  mouse\_y > end.Top() && mouse\_y < end.Top() + end.Height())  {  if (mouse\_state == 1)  {  isMouseInEndBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else if (mouse\_x > night.Left() && mouse\_x < night.Left() + night.Width() &&  mouse\_y > night.Top() && mouse\_y < night.Top() + night.Height())  {  if (mouse\_state == 1)  {  isMouseInNightBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else if (mouse\_x > continu.Left() && mouse\_x < continu.Left() + continu.Width() &&  mouse\_y > continu.Top() && mouse\_y < continu.Top() + continu.Height())  {  if (mouse\_state == 1)  {  isMouseInContinuBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else if (mouse\_x > instructions.Left() && mouse\_x < instructions.Left() + instructions.Width() &&  mouse\_y > instructions.Top() && mouse\_y < instructions.Top() + instructions.Height())  {  if (mouse\_state == 1)  {  isMouseInInstructionsBtn = true;  CAudio::Instance()->Play(AUDIO\_SLIP);  mouse\_state = 0;  }  }  else  {  mouse\_state = 1;  isMouseInContinuBtn = false;  isMouseInNightBtn = false;  isMouseInEndBtn = false;  isMouseInInstructionsBtn = false;  isMouseInRestartBtn = false;  }  }  }  }  void CGameStateRun::OnRButtonDown(UINT nFlags, CPoint point) // 處理滑鼠的動作  {  }  void CGameStateRun::OnRButtonUp(UINT nFlags, CPoint point) // 處理滑鼠的動作  {  }  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。  //  map.SetTopLeft(0, 0); //設置地圖位置  if (isNight)  {  nightMap.ShowBitmap();  }  else  {  map.ShowBitmap(); //把地圖顯示出來  }  clock.OnShow(); //顯示時鐘  //===============顯示各顏色的路線的圖標==============  purpleLine.ShowLineIconBitmap(); //紫色  bblueLine.ShowLineIconBitmap(); //靛色  blueLine.ShowLineIconBitmap(); //藍色  greenLine.ShowLineIconBitmap(); //綠色  yellowLine.ShowLineIconBitmap(); //黃色  orangeLine.ShowLineIconBitmap(); //橘色  redLine.ShowLineIconBitmap(); //紅色  if (!orangeLine.IsIconCanBeClicked())  garyIcon[0].ShowBitmap();  if (!yellowLine.IsIconCanBeClicked())  garyIcon[1].ShowBitmap();  if (!greenLine.IsIconCanBeClicked())  garyIcon[2].ShowBitmap();  if (!blueLine.IsIconCanBeClicked())  garyIcon[3].ShowBitmap();  if (!bblueLine.IsIconCanBeClicked())  garyIcon[4].ShowBitmap();  if (!purpleLine.IsIconCanBeClicked())  garyIcon[5].ShowBitmap();  if (red)  {  chooseColor.SetTopLeft(240, 570);  chooseColorNight.SetTopLeft(240, 570);  }  if (orange)  {  chooseColor.SetTopLeft(285, 570);  chooseColorNight.SetTopLeft(285, 570);  }  if (yellow)  {  chooseColor.SetTopLeft(330, 570);  chooseColorNight.SetTopLeft(330, 570);  }  if (green)  {  chooseColor.SetTopLeft(375, 570);  chooseColorNight.SetTopLeft(375, 570);  }  if (blue)  {  chooseColor.SetTopLeft(420, 570);  chooseColorNight.SetTopLeft(420, 570);  }  if (bblue)  {  chooseColor.SetTopLeft(465, 570);  chooseColorNight.SetTopLeft(465, 570);  }  if (purple)  {  chooseColor.SetTopLeft(510, 570);  chooseColorNight.SetTopLeft(510, 570);  }  passengerTotalNumber.SetIsNight(isNight);  if (isNight)  {  personNight.SetTopLeft(640, 10);  personNight.ShowBitmap();  passengerTotalNumber.ShowBitmap();  chooseColorNight.ShowBitmap();  escNight.ShowBitmap();  }  else  {  chooseColor.ShowBitmap();  person.SetTopLeft(640, 10);  person.ShowBitmap();  passengerTotalNumber.ShowBitmap();  esc.ShowBitmap();  }  //===============================================  // 如果有一個車站被選取 就要畫出來 //換句話說 就是線路要跟著滑鼠動  if (line->GetClickedStartStationNum() != -1 &&  (line->IsPassedStationEmpty() ||  line->GetClickedLastStation() == line->GetClickedStartStationNum()))  line->DrawRailway(stationList[line->GetClickedStartStationNum()].GetX() + 5,  stationList[line->GetClickedStartStationNum()].GetY() + 5, mouse\_x, mouse\_y);  //顯示各顏色線路  purpleLine.ShowRailway(stationList);  bblueLine.ShowRailway(stationList);  blueLine.ShowRailway(stationList);  greenLine.ShowRailway(stationList);  yellowLine.ShowRailway(stationList);  orangeLine.ShowRailway(stationList);  redLine.ShowRailway(stationList);  if (!cabinList.empty())  for (unsigned i = 0; i < cabinList.size(); i++)  {  cabinList[i].OnShow();  cabinList[i].SetPassengerPosition(passengerListOnCabin[i]);  }  for (int i = 0; i < MAXIUM\_STATION; i++)  {  stationList[i].SetIsNight(isNight);  stationList[i].OnShowStation();  }  for (int i = 0; i < (int)passengerList.size(); i++)  {  passengerList[i]->SetIsNight(isNight);  passengerList[i]->OnShow();  }  for (int i = 0; i < 7; i++)  {  for (unsigned j = 0; j < passengerListOnCabin[i].size(); j++)  {  passengerListOnCabin[i][j]->OnShow();  }  }  if (isStop2)  {  if (isNight)  {  mapnight.ShowBitmap();  morning.ShowBitmap();  }  else  {  backGround.ShowBitmap();  night.ShowBitmap();  }  restart.ShowBitmap();  end.ShowBitmap();  continu.ShowBitmap();  instructions.ShowBitmap();  if (isMouseInRestartBtn)  {  restartInvert.ShowBitmap();  }  else if (isMouseInEndBtn)  {  endInvert.ShowBitmap();  }  else if (isMouseInContinuBtn)  {  continuInvert.ShowBitmap();  }  else if (isMouseInNightBtn && isNight)  {  morningInvert.ShowBitmap();  }  else if (isMouseInNightBtn && !isNight)  {  nightInvert.ShowBitmap();  }  else if (isMouseInInstructionsBtn)  {  instructionsInvert.ShowBitmap();  }  if (isInInstructions)  {  if (page == 1)  {  page1.SetTopLeft(0, 0);  page1.ShowBitmap();  }  else if (page == 2)  {  page2.SetTopLeft(0, 0);  page2.ShowBitmap();  }  }  }  }  } |

|  |
| --- |
| Cabin.h |
| namespace game\_framework  {  class Passenger;  class Station;  class Cabin  {  public:  Cabin();  Cabin(int, int, int, int, int);  ~Cabin();  void SetXY(int, int);  void SetVelocity(int);  void SetMovingCabin(string, int);  void SetRGB(int, int, int);  void SetGoingDirection(string);  void SetMovingDirection(string);  void SetLinePoint(vector<int>, vector<int>);  void SetPassedStation(vector<int>);  int GetX();  int GetY();  int GetVelocity();  int GetNextPoint();  void GetRGB(int& R, int& G, int& B);  void OnMove(vector<Station> totalStationList);  void OnShow();  bool IsInStation(int endX, int endY, vector<Station> totalStationList);  void GetNextPrePoint(int&, int&);  int GetNextStation();  void SetLineStationNum(int num[6]);  void GetLineStationNum(int (&num)[6]);  void SetPassengerPosition(vector<Passenger\*> passengerOnCabin);  int GetPassengerNum();  void SetPassengerNum(int);  void AddPassengerNum(int);  bool IsCabinColorRepeat(vector<Cabin> cabinList, int R, int G, int B);  int GetCabinPointer(vector<Cabin> cabinList, int R, int G, int B);  bool IsCabinFull();  void PassengerGetOn(Passenger\* p);  int PassengerGetOut(int);  int PassengerGetOut(vector<Passenger\*>& passengerOnCabin, int nowStation);  bool IsStop();  void SetIsStop(bool);  void SetAddCounter(int n);  void SetCounter(int n);  int GetCounter();  int GetOutPeople();  private:  //車廂大小 42\*28 每個乘客的邊緣各2px  const int CABIN\_STOP\_IN\_STATION\_TIME = 30 / 2; //車廂停留在車站的時間  const int BASIC\_VELOCITY = 2;  int nextPoint;  int prePoint;  int leftTopX, leftTopY; //車廂的左上角位置  int rightDownX, rightDownY; //車廂的右下角位置  int velocity; //速度  int color[3]; //車廂顏色  int lineStationNum[6] = { 0 };  int nextStation; //下一個車站在totalStation裡的編號  int passengerNum = 0;  int counter\_cabin\_stop;  int outPeople;  string movingDirection = "";  string goingDirection = "";  vector<int> linePointX;  vector<int> linePointY;  vector<int> updatelinePointX;  vector<int> updatelinePointY;  vector<int> passedStation;  const int PASSENGERPOSITIONX\_UPDOWN[6] = {2, 16, 2, 16, 2, 16};  const int PASSENGERPOSITIONY\_UPDOWN[6] = {2, 2, 16, 16, 30, 30};  const int PASSENGERPOSITIONX\_RIGHTLEFT[6] = { 2, 2, 16, 16, 30, 30 };  const int PASSENGERPOSITIONY\_RIGHTLEFT[6] = { 2, 16, 2, 16, 2, 16 };  const int MAX\_PASSENGER\_NUM = 6; //最多六個乘客  bool isStop;  bool isStopOnceInStation;  vector<Passenger\*> passengerOnCabinList; //在車廂的乘客  };  } |

|  |
| --- |
| Cabin.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <mmsystem.h>  #include <ddraw.h>  #include <cstdlib>  #include <ctime>  #include <vector>  #include "audio.h"  #include "gamelib.h"  #include "Cabin.h"  #include "Passenger.h"  #include "Station.h"  namespace game\_framework  {  Cabin::Cabin()  { }  Cabin::Cabin(int setX, int setY, int R, int G, int B)  {  nextPoint = 1;  prePoint = 0;  leftTopX = setX;  leftTopY = setY;  nextStation = -1;  counter\_cabin\_stop = 0;  isStop = false;  isStopOnceInStation = false;  rightDownX = setX + 42;  rightDownY = setY + 28;  movingDirection = "up";  goingDirection = "head";  velocity = BASIC\_VELOCITY;  color[0] = R;  color[1] = G;  color[2] = B;  }  Cabin::~Cabin()  {  for (unsigned i = 0; i < passengerOnCabinList.size(); i++)  {  if (passengerOnCabinList[i] != nullptr)  {  delete passengerOnCabinList[i];  }  }  }  void Cabin::SetXY(int setX, int setY)  {  leftTopX = setX;  leftTopY = setY;  rightDownX = setX + 42;  rightDownY = setY + 28;  }  void Cabin::SetVelocity(int v)  {  velocity = v;  }  void Cabin::SetMovingCabin(string movingDirection, int num)  {  if (movingDirection == "up")  {  leftTopY -= num;  rightDownX = leftTopX + 28;  rightDownY = leftTopY + 42;  }  else if (movingDirection == "down")  {  leftTopY += num;  rightDownX = leftTopX + 28;  rightDownY = leftTopY + 42;  }  else if (movingDirection == "right")  {  leftTopX += num;  rightDownX = leftTopX + 42;  rightDownY = leftTopY + 28;  }  else if (movingDirection == "left")  {  leftTopX -= num;  rightDownX = leftTopX + 42;  rightDownY = leftTopY + 28;  }  }  void Cabin::SetRGB(int R, int G, int B)  {  color[0] = R;  color[1] = G;  color[2] = B;  }  void Cabin::SetGoingDirection(string s)  {  goingDirection = s;  }  void Cabin::SetMovingDirection(string s)  {  movingDirection = s;  }  void Cabin::SetLinePoint(vector<int> x, vector<int> y)  {  updatelinePointX.assign(x.begin(), x.end());  updatelinePointY.assign(y.begin(), y.end());  if (linePointX.empty())linePointX.assign(x.begin(), x.end());  if (linePointY.empty())linePointY.assign(y.begin(), y.end());  }  void Cabin::SetPassedStation(vector<int> station)  {  passedStation.assign(station.begin(), station.end());  }  int Cabin::GetX()  {  return leftTopX;  }  int Cabin::GetY()  {  return leftTopY;  }  int Cabin::GetVelocity()  {  return velocity;  }  int Cabin::GetNextPoint()  {  return nextPoint;  }  void Cabin::GetRGB(int& R, int& G, int& B)  {  R = color[0];  G = color[1];  B = color[2];  }  void Cabin::OnMove(vector<Station> totalStationList)  {  if (!isStop)  {  int sizeVecX = linePointX.size();  int sizeVecY = linePointY.size();  int startX = linePointX[prePoint];  int startY = linePointY[prePoint];  int endX = linePointX[nextPoint];  int endY = linePointY[nextPoint];  int nowPointX = leftTopX;  int nowPointY = leftTopY;  // X座標相同 代表向上或向下移動  // Y座標相同 代表向左或向右移動  //================================  //============設定移動方向==========  //================================  if (startX == endX)  {  if (startY > endY) //向上移動  {  movingDirection = "up";  }  else if (startY < endY) //向下移動  {  movingDirection = "down";  }  }  else if (startY == endY)  {  if (startX > endX)// 向左移動  {  movingDirection = "left";  }  else if (startX < endX) //向右移動  {  movingDirection = "right";  }  }  //================================  //==============移動===============  //================================  if (movingDirection == "up")  {  if (nowPointY > endY)  SetMovingCabin(movingDirection, velocity);  else if (nowPointY <= endY)  {  if (IsInStation(endX, endY, totalStationList))  {  isStop = true;  }  //================================  //============判斷起訖站===========  //================================  if (nextPoint == sizeVecX - 1)  {  goingDirection = "back";  }  else if (nextPoint == 0)  {  linePointX.assign(updatelinePointX.begin(), updatelinePointX.end());  linePointY.assign(updatelinePointY.begin(), updatelinePointY.end());  goingDirection = "head";  }  //================================  //============移動到下一站==========  //================================  if (goingDirection == "head")  {  prePoint = nextPoint;  nextPoint++;  }  else if (goingDirection == "back")  {  prePoint = nextPoint;  nextPoint--;  }  }  }  else if (movingDirection == "down")  {  if (nowPointY < endY)  SetMovingCabin(movingDirection, velocity);  else if (nowPointY >= endY)  {  if (IsInStation(endX, endY, totalStationList))  {  isStop = true;  }  //================================  //============判斷起訖站===========  //================================  if (nextPoint == sizeVecX - 1)  {  goingDirection = "back";  }  else if (nextPoint == 0)  {  goingDirection = "head";  linePointX.assign(updatelinePointX.begin(), updatelinePointX.end());  linePointY.assign(updatelinePointY.begin(), updatelinePointY.end());  }  //================================  //============移動到下一站==========  //================================  if (goingDirection == "head")  {  prePoint = nextPoint;  nextPoint++;  }  else if (goingDirection == "back")  {  prePoint = nextPoint;  nextPoint--;  }  }  }  else if (movingDirection == "left")  {  if (nowPointX > endX)  SetMovingCabin(movingDirection, velocity);  else if (nowPointX <= endX)  {  if (IsInStation(endX, endY, totalStationList))  {  isStop = true;  }  //================================  //============判斷起訖站===========  //================================  if (nextPoint == sizeVecX - 1)  {  goingDirection = "back";  }  else if (nextPoint == 0)  {  linePointX.assign(updatelinePointX.begin(), updatelinePointX.end());  linePointY.assign(updatelinePointY.begin(), updatelinePointY.end());  goingDirection = "head";  }  //================================  //============移動到下一站==========  //================================  if (goingDirection == "head")  {  prePoint = nextPoint;  nextPoint++;  }  else if (goingDirection == "back")  {  prePoint = nextPoint;  nextPoint--;  }  }  }  else if (movingDirection == "right")  {  if (nowPointX < endX)  SetMovingCabin(movingDirection, velocity);  else if (nowPointX >= endX)  {  if (IsInStation(endX, endY, totalStationList))  {  isStop = true;  }  //================================  //============判斷起訖站===========  //================================  if (nextPoint == sizeVecX - 1)  {  goingDirection = "back";  }  else if (nextPoint == 0)  {  goingDirection = "head";  linePointX.assign(updatelinePointX.begin(), updatelinePointX.end());  linePointY.assign(updatelinePointY.begin(), updatelinePointY.end());  }  //================================  //============移動到下一站==========  //================================  if (goingDirection == "head")  {  prePoint = nextPoint;  nextPoint++;  }  else if (goingDirection == "back")  {  prePoint = nextPoint;  nextPoint--;  }  }  }  }  }  void Cabin::OnShow()  {  CDC\* pDC = CDDraw::GetBackCDC();  CPen\* pp, p(PS\_NULL, 0, RGB(0, 0, 0));  pp = pDC->SelectObject(&p);  CBrush b(RGB(color[0], color[1], color[2]));  pDC->SelectObject(&b);  pDC->Rectangle(leftTopX, leftTopY, rightDownX, rightDownY);  pDC->SelectObject(pp);  //pDC->SelectObject(pb);  CDDraw::ReleaseBackCDC();  //SetPassengerPosition();  for (unsigned i = 0; i < passengerOnCabinList.size(); i++)  {  passengerOnCabinList[i]->OnShow();  }  }  bool Cabin::IsInStation(int endX, int endY, vector<Station> totalStationList)  {  int sizeVec = totalStationList.size();  for (int i = 0; i < sizeVec; i++)  {  if (totalStationList[i].GetX() == endX && totalStationList[i].GetY() == endY)  {  nextStation = i;  return true;  }  }  return false;  }  void Cabin::GetNextPrePoint(int& pre, int& next)  {  pre = prePoint;  next = nextPoint;  }  int Cabin::GetNextStation()  {  return nextStation;  }  void Cabin::SetLineStationNum(int num[6])  {  for (int i = 0; i < 6; i++)  {  lineStationNum[i] = num[i];  }  }  void Cabin::GetLineStationNum(int(&num)[6])  {  for (int i = 0; i < 6; i++)  {  num[i] = lineStationNum[i];  }  }  void Cabin::SetPassengerPosition(vector<Passenger\*> passengerOnCabin)  {  if (movingDirection == "left" || movingDirection == "right")  {  for (unsigned i = 0; i < passengerOnCabin.size(); i++)  {  passengerOnCabin[i]->SetXY(leftTopX + PASSENGERPOSITIONX\_RIGHTLEFT[i], leftTopY + PASSENGERPOSITIONY\_RIGHTLEFT[i]);  }  }  else if (movingDirection == "down" || movingDirection == "up")  {  for (unsigned i = 0; i < passengerOnCabin.size(); i++)  {  passengerOnCabin[i]->SetXY(leftTopX + PASSENGERPOSITIONX\_UPDOWN[i], leftTopY + PASSENGERPOSITIONY\_UPDOWN[i]);  }  }  }  int Cabin::GetPassengerNum()  {  return passengerNum;  }  void Cabin::SetPassengerNum(int n)  {  passengerNum = n;  }  void Cabin::AddPassengerNum(int n)  {  passengerNum += n;  }  bool Cabin::IsCabinColorRepeat(vector<Cabin> cabinList, int R, int G, int B)  {  for (auto it = begin(cabinList); it != end(cabinList); it++)  {  int RR, GG, BB;  it->GetRGB(RR, GG, BB);  if (RR = R && GG == G && BB == B)  return true;  }  return false;  }  int Cabin::GetCabinPointer(vector<Cabin> cabinList, int R, int G, int B)  {  for (unsigned i = 0; i < cabinList.size(); i++)  {  int RR, GG, BB;  cabinList[i].GetRGB(RR, GG, BB);  if (RR = R && GG == G && BB == B)  return i;  }  return -1;  }  bool Cabin::IsCabinFull()  {  return (passengerNum >= 6); //看乘客有沒有超過6個  }  void Cabin::PassengerGetOn(Passenger\* p)  {  passengerNum++;  passengerOnCabinList.push\_back(p);  //passengerNum = passengerOnCabinList.size();  //SetPassengerPosition();  }  int Cabin::PassengerGetOut(int nowStation)  {  outPeople = 0;  for (unsigned i = passengerOnCabinList.size() - 1; i >= 0; i--)  {  if (i > 6)  {  break;  }  else if (nowStation == passengerOnCabinList[i]->GetFinalStation())  {  delete passengerOnCabinList[i];  passengerOnCabinList.erase(passengerOnCabinList.begin() + i);  outPeople++;  }  passengerNum = (int)passengerOnCabinList.size();  }  return 1;  }  int Cabin::PassengerGetOut(vector<Passenger\*>& passengerOnCabin, int nowStation)  {  outPeople = 0;  for (unsigned i = passengerOnCabin.size() - 1; i >= 0; i--)  {  if (i > 6)  {  break;  }  else if (nowStation == passengerOnCabin[i]->GetFinalStation())  {  delete passengerOnCabin[i];  passengerOnCabin.erase(passengerOnCabin.begin() + i);  outPeople++;  }  passengerNum = (int)passengerOnCabin.size();  }  return 0;  }  bool Cabin::IsStop()  {  return isStop;  }  void Cabin::SetIsStop(bool b)  {  isStop = b;  }  void Cabin::SetAddCounter(int n)  {  counter\_cabin\_stop += n;  }  void Cabin::SetCounter(int n)  {  counter\_cabin\_stop = n;  }  int Cabin::GetCounter()  {  return counter\_cabin\_stop;  }  int Cabin::GetOutPeople()  {  return outPeople;  }  } |

|  |
| --- |
| Clock.h |
| namespace game\_framework  {  /////////////////////////////////////////////////////////////////////////////  // 這個class提供會一直彈跳的球  // 看懂就可以改寫成自己的程式了  /////////////////////////////////////////////////////////////////////////////  class Clock  {  public:  Clock();  void LoadBitmap(); // 載入圖形  void OnMove(); // 移動  void OnShow(); // 將圖形貼到畫面  bool IsFinalBitmap(); //是不是播到最後一張bitmap了  private:  int clk\_x, clk\_y; // 圖形座標  CAnimation clock\_animation; // 利用動畫作圖形  };  } |

|  |
| --- |
| Clock.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <mmsystem.h>  #include <ddraw.h>  #include "audio.h"  #include "gamelib.h"  #include "Clock.h"  namespace game\_framework  {  Clock::Clock()  {  clk\_x = 750;  clk\_y = 10;  }  void Clock::LoadBitmap()  {  char\* clock\_filename[100] = {  ".\\RES\\ClockImage\\Clock\_0.bmp" ,".\\RES\\ClockImage\\Clock\_1.bmp" ,".\\RES\\ClockImage\\Clock\_2.bmp" ,".\\RES\\ClockImage\\Clock\_3.bmp" ,".\\RES\\ClockImage\\Clock\_4.bmp" ,  ".\\RES\\ClockImage\\Clock\_5.bmp" ,".\\RES\\ClockImage\\Clock\_6.bmp" ,".\\RES\\ClockImage\\Clock\_7.bmp" ,".\\RES\\ClockImage\\Clock\_8.bmp" ,".\\RES\\ClockImage\\Clock\_9.bmp" ,  ".\\RES\\ClockImage\\Clock\_10.bmp",".\\RES\\ClockImage\\Clock\_11.bmp",".\\RES\\ClockImage\\Clock\_12.bmp",".\\RES\\ClockImage\\Clock\_13.bmp",".\\RES\\ClockImage\\Clock\_14.bmp",  ".\\RES\\ClockImage\\Clock\_15.bmp",".\\RES\\ClockImage\\Clock\_16.bmp",".\\RES\\ClockImage\\Clock\_17.bmp",".\\RES\\ClockImage\\Clock\_18.bmp",".\\RES\\ClockImage\\Clock\_19.bmp",  ".\\RES\\ClockImage\\Clock\_20.bmp",".\\RES\\ClockImage\\Clock\_21.bmp",".\\RES\\ClockImage\\Clock\_22.bmp",".\\RES\\ClockImage\\Clock\_23.bmp",".\\RES\\ClockImage\\Clock\_24.bmp",  ".\\RES\\ClockImage\\Clock\_25.bmp",".\\RES\\ClockImage\\Clock\_26.bmp",".\\RES\\ClockImage\\Clock\_27.bmp",".\\RES\\ClockImage\\Clock\_28.bmp",".\\RES\\ClockImage\\Clock\_29.bmp",  ".\\RES\\ClockImage\\Clock\_30.bmp",".\\RES\\ClockImage\\Clock\_31.bmp",".\\RES\\ClockImage\\Clock\_32.bmp",".\\RES\\ClockImage\\Clock\_33.bmp",".\\RES\\ClockImage\\Clock\_34.bmp",  ".\\RES\\ClockImage\\Clock\_35.bmp",".\\RES\\ClockImage\\Clock\_36.bmp",".\\RES\\ClockImage\\Clock\_37.bmp",".\\RES\\ClockImage\\Clock\_38.bmp",".\\RES\\ClockImage\\Clock\_39.bmp",  ".\\RES\\ClockImage\\Clock\_40.bmp",".\\RES\\ClockImage\\Clock\_41.bmp",".\\RES\\ClockImage\\Clock\_42.bmp",".\\RES\\ClockImage\\Clock\_43.bmp",".\\RES\\ClockImage\\Clock\_44.bmp",  ".\\RES\\ClockImage\\Clock\_45.bmp",".\\RES\\ClockImage\\Clock\_46.bmp",".\\RES\\ClockImage\\Clock\_47.bmp",".\\RES\\ClockImage\\Clock\_48.bmp",".\\RES\\ClockImage\\Clock\_49.bmp",  ".\\RES\\ClockImage\\Clock\_50.bmp",".\\RES\\ClockImage\\Clock\_51.bmp",".\\RES\\ClockImage\\Clock\_52.bmp",".\\RES\\ClockImage\\Clock\_53.bmp",".\\RES\\ClockImage\\Clock\_54.bmp",  ".\\RES\\ClockImage\\Clock\_55.bmp",".\\RES\\ClockImage\\Clock\_56.bmp",".\\RES\\ClockImage\\Clock\_57.bmp",".\\RES\\ClockImage\\Clock\_58.bmp",".\\RES\\ClockImage\\Clock\_59.bmp",  ".\\RES\\ClockImage\\Clock\_60.bmp",".\\RES\\ClockImage\\Clock\_61.bmp",".\\RES\\ClockImage\\Clock\_62.bmp",".\\RES\\ClockImage\\Clock\_63.bmp",".\\RES\\ClockImage\\Clock\_64.bmp",  ".\\RES\\ClockImage\\Clock\_65.bmp",".\\RES\\ClockImage\\Clock\_66.bmp",".\\RES\\ClockImage\\Clock\_67.bmp",".\\RES\\ClockImage\\Clock\_68.bmp",".\\RES\\ClockImage\\Clock\_69.bmp",  ".\\RES\\ClockImage\\Clock\_70.bmp",".\\RES\\ClockImage\\Clock\_71.bmp",".\\RES\\ClockImage\\Clock\_72.bmp",".\\RES\\ClockImage\\Clock\_73.bmp",".\\RES\\ClockImage\\Clock\_74.bmp",  ".\\RES\\ClockImage\\Clock\_75.bmp",".\\RES\\ClockImage\\Clock\_76.bmp",".\\RES\\ClockImage\\Clock\_77.bmp",".\\RES\\ClockImage\\Clock\_78.bmp",".\\RES\\ClockImage\\Clock\_79.bmp",  ".\\RES\\ClockImage\\Clock\_80.bmp",".\\RES\\ClockImage\\Clock\_81.bmp",".\\RES\\ClockImage\\Clock\_82.bmp",".\\RES\\ClockImage\\Clock\_83.bmp",".\\RES\\ClockImage\\Clock\_84.bmp",  ".\\RES\\ClockImage\\Clock\_85.bmp",".\\RES\\ClockImage\\Clock\_86.bmp",".\\RES\\ClockImage\\Clock\_87.bmp",".\\RES\\ClockImage\\Clock\_88.bmp",".\\RES\\ClockImage\\Clock\_89.bmp",  ".\\RES\\ClockImage\\Clock\_90.bmp",".\\RES\\ClockImage\\Clock\_91.bmp",".\\RES\\ClockImage\\Clock\_92.bmp",".\\RES\\ClockImage\\Clock\_93.bmp",".\\RES\\ClockImage\\Clock\_94.bmp",  ".\\RES\\ClockImage\\Clock\_95.bmp",".\\RES\\ClockImage\\Clock\_96.bmp",".\\RES\\ClockImage\\Clock\_97.bmp",".\\RES\\ClockImage\\Clock\_98.bmp",".\\RES\\ClockImage\\Clock\_99.bmp"  };  for (int i = 0; i < 100; i++) // 載入動畫(由100張圖形構成)  clock\_animation.AddBitmap(clock\_filename[i], RGB(0, 0, 0));  clock\_animation.SetDelayCount(5); // 延遲播放速度  }  void Clock::OnMove()  {  clock\_animation.OnMove(); // 執行一次animation.OnMove()，animation才會換圖  }  void Clock::OnShow()  {  clock\_animation.SetTopLeft(clk\_x, clk\_y);  clock\_animation.OnShow();  }  bool Clock::IsFinalBitmap()  {  return clock\_animation.IsFinalBitmap();  }  } |

|  |
| --- |
| Line.h |
| namespace game\_framework  {  class Line  {  public:  Line();  ~Line();  void DrawRailway(int, int, int, int); //畫 出軌 道  void SetLineColor(int,int,int); //設定線路顏色  void SetClickedStartStationNum(int); //設定ClickedStartStationNum  void SetClickedEndStationNum(int); //設定ClickedEndStationNum  void SetPassedStation(int,int); //設定有經過的車站  void SetLinePointXY(vector<Station>); //設定轉折點的XY座標  int GetClickedStartStationNum(); //取得ClickedStartStationNum  int GetClickedEndStationNum(); //取得ClickedEndStationNum  int GetMouseClickedStationNum(int,int,vector<Station>,int); //取得滑鼠點到的那個車站的編號  void GetLineColorRGB(int& R, int& G, int& B); //取得線路顏色RGB  int GetLineColor(); //取得線路顏色編號 red=0 orange=1 ....  int GetClickedFirstStation(); //取得線路車站起點  int GetClickedLastStation(); //取得線路車站終點  void GetLinePointXY(vector<int>&,vector<int>&);  void GetPassedStationNum(vector<int>&);  void GetTotalPassedStation(int(&num)[6]);  void Reset();  bool IsClickedStation(int, int, vector<Station>, int); //檢查是否點到車站 有就回傳ture 如果沒有則回傳false  bool IsClickedTwoStation(); //檢查是否已經點了兩個車站了  bool IsMouseClickedLineColorBMP(int,int);//檢查是否滑鼠有點到下方線路顏色的圖片 //換線路顏色要用的啦  bool IsPassedStationEmpty(); //檢查是否有在地圖上拉出線路了  void CountTotalPassedStation(); //計算線路所經過的車站類型  void LoadBitmap();  void ShowLineIconBitmap(); //顯示畫面底部的的圖片icon  void ShowRailway(vector<Station>); //劃出線路  bool IsIconCanBeClicked();  void SetIsCanbeClicked(bool b);  private:  CMovingBitmap lineColorBMP;  int clickedStartStationNum;  int clickedEndStationNum;  int lineColor[3];  bool isCanBeClicked;  int totalPassedStationNum[6] = {0}; //線路所經過的車站類型  vector<int> passedStation; //經過的車站的編號  vector<int> linePointX; //劃出線路點的X座標 與lineYPoint連動  vector<int> linePointY; //劃出線路點的Y座標 與lineXPoint連動  //red(255.0.0),orang(255.144.0),yellow(255.255.0),green(0.255.0),blue(0.138.255),bblue(0.6.255),puple(144.0.255)  };  } |

|  |
| --- |
| Line.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <mmsystem.h>  #include <ddraw.h>  #include <vector>  #include "audio.h"  #include "gamelib.h"  #include "Station.h"  #include "Line.h"  #include "Cabin.h"  namespace game\_framework  {  Line::Line()  : clickedStartStationNum(-1), clickedEndStationNum(-1), lineColor{0, 0, 0}, isCanBeClicked(false)  {}  Line::~Line()  {}  void Line::DrawRailway(int startX, int startY, int endX, int endY)  {  const int BORDERSIZE = 14;  //const int rectHeight = 20;  CDC\* pDC = CDDraw::GetBackCDC();  CPen\* pp, p(PS\_NULL, 0, RGB(0, 0, 0));  pp = pDC->SelectObject(&p);  //比對start 跟 end 去判斷怎麼畫出線來  CBrush b1(RGB(lineColor[0], lineColor[1], lineColor[2]));  pDC->SelectObject(&b1);  if (endX >= startX && endY >= startY) //結束點在右下  {  pDC->Rectangle(startX, startY, startX + BORDERSIZE, endY + BORDERSIZE);  pDC->Rectangle(startX, endY, endX + BORDERSIZE, endY + BORDERSIZE);  }  else if (endX < startX && endY > startY) //結束點在左下  {  /\*pDC->Rectangle(startX, startY, startX + BORDERSIZE, endY + BORDERSIZE);  pDC->Rectangle(startX + BORDERSIZE, endY, endX, endY + BORDERSIZE);\*/  pDC->Rectangle(startX, startY, endX, startY + BORDERSIZE);  pDC->Rectangle(endX, endY, endX + BORDERSIZE, startY);  }  else if (endX <= startX && endY <= startY) //結束點在左上  {  /\*pDC->Rectangle(startX, startY, endX, startY + BORDERSIZE);  pDC->Rectangle(endX, endY, endX + BORDERSIZE, startY + BORDERSIZE);\*/  pDC->Rectangle(startX, startY + BORDERSIZE, startX + BORDERSIZE, endY);  pDC->Rectangle(startX + BORDERSIZE, endY, endX + BORDERSIZE, endY + BORDERSIZE);  }  else if (endX > startX && endY < startY) //結束點在右上  {  pDC->Rectangle(startX, startY, endX + BORDERSIZE, startY + BORDERSIZE);  pDC->Rectangle(endX, endY, endX + BORDERSIZE, startY + BORDERSIZE);  }  //釋放 pen brush、Back、Plain的CDC  pDC->SelectObject(pp);  //pDC->SelectObject(pb);  CDDraw::ReleaseBackCDC();  }  void Line::SetLineColor(int r, int g, int b)  {  lineColor[0] = r;  lineColor[1] = g;  lineColor[2] = b;  }  void Line::SetClickedStartStationNum(int num)  {  clickedStartStationNum = num;  }  void Line::SetClickedEndStationNum(int num)  {  clickedEndStationNum = num;  }  void Line::SetPassedStation(int start, int end)  {  if (passedStation.empty()) //從來沒有拉過線路  {  passedStation.push\_back(start);  passedStation.push\_back(end);  }  else  {  if (passedStation.back() == start) //從尾巴拉線路出去  {  passedStation.push\_back(end);  }  }  }  void Line::SetLinePointXY(vector<Station> stationList)  {  int vecSize = passedStation.size();  vector<int> bufferLinePointX;  vector<int> bufferLinePointY;  for (int i = 0; i < vecSize - 1; i++)  {  int stationStartNum = passedStation[i];  int stationEndNum = passedStation[i + 1];  int startX = stationList[stationStartNum].GetX();  int startY = stationList[stationStartNum].GetY();  int endX = stationList[stationEndNum].GetX();  int endY = stationList[stationEndNum].GetY();  if (bufferLinePointX.empty()) bufferLinePointX.push\_back(startX);  if (bufferLinePointY.empty()) bufferLinePointY.push\_back(startY);  if (endX >= startX && endY >= startY) //結束點在右下  {  bufferLinePointX.push\_back(startX);  bufferLinePointY.push\_back(endY);  }  else if (endX < startX && endY > startY) //結束點在左下  {  bufferLinePointX.push\_back(endX);  bufferLinePointY.push\_back(startY);  }  else if (endX <= startX && endY <= startY) //結束點在左上  {  bufferLinePointX.push\_back(startX);  bufferLinePointY.push\_back(endY);  }  else if (endX > startX && endY < startY) //結束點在右上  {  bufferLinePointX.push\_back(endX);  bufferLinePointY.push\_back(startY);  }  bufferLinePointX.push\_back(endX);  bufferLinePointY.push\_back(endY);  }  linePointX.assign(bufferLinePointX.begin(), bufferLinePointX.end());  linePointY.assign(bufferLinePointY.begin(), bufferLinePointY.end());  }  int Line::GetClickedStartStationNum()  {  return clickedStartStationNum;  }  int Line::GetClickedEndStationNum()  {  return clickedEndStationNum;  }  int Line::GetMouseClickedStationNum(int x, int y, vector<Station> stationList, int currentStation)  {  for (int i = 0; i < currentStation; i++)  if (x >= stationList[i].GetX() && x <= stationList[i].GetX() + 25 && y >= stationList[i].GetY() && y <= stationList[i].GetY() + 25)  return i;  return -1;  }  void Line::GetLineColorRGB(int& R, int& G, int& B)  {  R = lineColor[0];  G = lineColor[1];  B = lineColor[2];  }  int Line::GetLineColor()  {  if (lineColor[0] == 255 && lineColor[1] == 0 && lineColor[2] == 0)  return 0;  else if (lineColor[0] == 255 && lineColor[1] == 144 && lineColor[2] == 0)  return 1;  else if (lineColor[0] == 255 && lineColor[1] == 255 && lineColor[2] == 0)  return 2;  else if (lineColor[0] == 0 && lineColor[1] == 255 && lineColor[2] == 0)  return 3;  else if (lineColor[0] == 0 && lineColor[1] == 138 && lineColor[2] == 255)  return 4;  else if (lineColor[0] == 0 && lineColor[1] == 6 && lineColor[2] == 255)  return 5;  else //if (lineColor[0] == 144 && lineColor[1] == 0 && lineColor[2] == 255)  return 6;  //red(255.0.0),orang(255.144.0),yellow(255.255.0),green(0.255.0),blue(0.138.255),bblue(0.6.255),puple(144.0.255)  }  int Line::GetClickedFirstStation()  {  return passedStation.empty() ? -1 : passedStation.front();  }  int Line::GetClickedLastStation()  {  return passedStation.empty() ? -1 : passedStation.back();  }  void Line::GetLinePointXY(vector<int>& pointX, vector<int>& pointY)  {  pointX.assign(linePointX.begin(), linePointX.end());  pointY.assign(linePointY.begin(), linePointY.end());  }  void Line::GetPassedStationNum(vector<int>& station)  {  station.assign(passedStation.begin(), passedStation.end());  }  void Line::GetTotalPassedStation(int (&num)[6])  {  for (int i = 0; i < 6; i++)  {  num[i] = totalPassedStationNum[i];  }  }  void Line::Reset()  {  vector<int> a;  passedStation.assign(a.begin(), a.end());  linePointX.assign(a.begin(), a.end());  linePointY.assign(a.begin(), a.end());  for (int i = 0; i < 6; i++)  {  totalPassedStationNum[i] = 0;  }  for (int i = 0; i < 3; i++)  {  lineColor[i] = 0;  }  clickedStartStationNum = -1;  clickedEndStationNum = -1;  }  bool Line::IsClickedStation(int x, int y, vector<Station> stationList, int currentStation)  {  for (int i = 0; i < currentStation; i++)  if (x >= stationList[i].GetX() && x <= stationList[i].GetX() + 25 && y >= stationList[i].GetY() && y <= stationList[i].GetY() + 25)  return true;  return false;  }  bool Line::IsClickedTwoStation()  {  if (clickedStartStationNum != -1 && clickedEndStationNum != -1 && clickedStartStationNum != clickedEndStationNum)  return true;  return false;  }  bool Line::IsMouseClickedLineColorBMP(int mouseX, int mouseY)  {  if (mouseX > lineColorBMP.Left() && mouseX < lineColorBMP.Left() + lineColorBMP.Width() &&  mouseY > lineColorBMP.Top() && mouseY < lineColorBMP.Top() + lineColorBMP.Height() && isCanBeClicked)  return true;  return false;  }  bool Line::IsPassedStationEmpty()  {  return passedStation.empty();  }  void Line::CountTotalPassedStation()  {  int vecSize = passedStation.size();  for (int i = 0; i < vecSize; i++)  {  for (int j = 0; j < 7; j++)  {  if (passedStation[i] == j)  {  totalPassedStationNum[j] = 1;  }  }  }  }  void Line::LoadBitmap()  {  //red(255.0.0),orang(255.144.0),yellow(255.255.0),green(0.255.0),blue(0.138.255),bblue(0.6.255),puple(144.0.255)  if (lineColor[0] == 255 && lineColor[1] == 0 && lineColor[2] == 0)  lineColorBMP.LoadBitmap(".\\RES\\color\\red.bmp", RGB(255, 255, 255));  else if (lineColor[0] == 255 && lineColor[1] == 144 && lineColor[2] == 0)  lineColorBMP.LoadBitmap(".\\RES\\color\\orang.bmp", RGB(255, 255, 255));  else if (lineColor[0] == 255 && lineColor[1] == 255 && lineColor[2] == 0)  lineColorBMP.LoadBitmap(".\\RES\\color\\yellow.bmp", RGB(255, 255, 255));  else if (lineColor[0] == 0 && lineColor[1] == 255 && lineColor[2] == 0 )  lineColorBMP.LoadBitmap(".\\RES\\color\\green.bmp", RGB(255, 255, 255));  else if (lineColor[0] == 0 && lineColor[1] == 138 && lineColor[2] == 255)  lineColorBMP.LoadBitmap(".\\RES\\color\\blue.bmp", RGB(255, 255, 255));  else if (lineColor[0] == 0 && lineColor[1] == 6 && lineColor[2] == 255)  lineColorBMP.LoadBitmap(".\\RES\\color\\bblue.bmp", RGB(255, 255, 255));  else if (lineColor[0] == 144 && lineColor[1] == 0 && lineColor[2] == 255)  lineColorBMP.LoadBitmap(".\\RES\\color\\purple.bmp", RGB(255, 255, 255));  }  void Line::ShowLineIconBitmap()  {  if (lineColor[0] == 255 && lineColor[1] == 0 && lineColor[2] == 0)  lineColorBMP.SetTopLeft(240, 570);  else if (lineColor[0] == 255 && lineColor[1] == 144 && lineColor[2] == 0)  lineColorBMP.SetTopLeft(285, 570);  else if (lineColor[0] == 255 && lineColor[1] == 255 && lineColor[2] == 0)  lineColorBMP.SetTopLeft(330, 570);  else if (lineColor[0] == 0 && lineColor[1] == 255 && lineColor[2] == 0)  lineColorBMP.SetTopLeft(375, 570);  else if (lineColor[0] == 0 && lineColor[1] == 138 && lineColor[2] == 255)  lineColorBMP.SetTopLeft(420, 570);  else if (lineColor[0] == 0 && lineColor[1] == 6 && lineColor[2] == 255 )  lineColorBMP.SetTopLeft(465, 570);  else if (lineColor[0] == 144 && lineColor[1] == 0 && lineColor[2] == 255 )  lineColorBMP.SetTopLeft(510, 570);  lineColorBMP.ShowBitmap();  }  void Line::ShowRailway(vector<Station> stationList)  {  int vecSize = passedStation.size();  for (int i = 0; i < vecSize - 1; i++)  {  int stationA = passedStation[i];  int stationB = passedStation[i + 1];  DrawRailway(stationList[stationA].GetX() + 5, stationList[stationA].GetY() + 5, stationList[stationB].GetX() + 5, stationList[stationB].GetY() + 5);  }  }  bool Line::IsIconCanBeClicked()  {  return isCanBeClicked;  }  void Line::SetIsCanbeClicked(bool b)  {  isCanBeClicked = b;  }  } |

|  |
| --- |
| Passenger.h |
| namespace game\_framework  {  class Station;  class Passenger  {  public:  Passenger();  Passenger(int, int, int, int, int);  void LoadBitmap(); //載入圖形  void SetType(int);  void SetXY(int, int);  void SetX(int n);  void SetY(int n);  void SetIsNight(bool);  int GetStartStation();  int GetFinalStation();  int GetPassengerType();  void RandomMadePassenger(vector<Passenger\*>& passengerList, vector<Station>& stationList, int stationMaxium, int stationTypeNum, int passenagerNum);  void OnShow(); //將圖形貼到畫面  void RandomOnePassenger(vector<Passenger\*>& passengerList, vector<Station>& stationList, int stationMaxium, int stationTypeNum);  ~Passenger();  private:  CMovingBitmap passengerAnimation; //乘客圖形  CMovingBitmap passengerAnimationNight; //乘客圖形  int finalStation; //終點  int startStation; //起點  int x, y; //圖形座標  int passengerType; //乘客類型  bool isNight = false;  //乘客在車廂的位置  const int PASSENGERPOSITIONX\_UPDOWN[6] = { 2, 16, 2, 16, 2, 16 };  const int PASSENGERPOSITIONY\_UPDOWN[6] = { 2, 2, 16, 16, 30, 30 };  const int PASSENGERPOSITIONX\_RIGHTLEFT[6] = { 2, 2, 16, 16, 30, 30 };  const int PASSENGERPOSITIONY\_RIGHTLEFT[6] = { 2, 16, 2, 16, 2, 16 };  //乘客在車站的位置  const int PASSENAGE\_POSITION[10] = { 25, 36, 47, 58, 69, 80, 91, 102, 113, 124 };  };  } |

|  |
| --- |
| Passenger.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <mmsystem.h>  #include <ddraw.h>  #include <cstdlib>  #include <ctime>  #include "audio.h"  #include "gamelib.h"  #include "Station.h"  #include "Passenger.h"  namespace game\_framework  {  Passenger::Passenger()  : passengerType(0), x(0), y(0)  { }  Passenger::Passenger(int type, int start, int end, int x, int y)  : passengerType(type), startStation(start), finalStation(end), x(x), y(y)  { }  void Passenger::LoadBitmap()  {  if (passengerType == 0)  {  passengerAnimation.LoadBitmap(".\\RES\\passenger\\Passenger\_Circle.bmp", RGB(255, 255, 255));  passengerAnimationNight.LoadBitmap(".\\RES\\passenger\\Passenger\_Circle\_night.bmp", RGB(255, 255, 255));  }  else if (passengerType == 1)  {  passengerAnimation.LoadBitmap(".\\RES\\passenger\\Passenger\_Triangle.bmp", RGB(255, 255, 255));  passengerAnimationNight.LoadBitmap(".\\RES\\passenger\\Passenger\_Triangle\_night.bmp", RGB(255, 255, 255));  }  else if (passengerType == 2)  {  passengerAnimation.LoadBitmap(".\\RES\\passenger\\Passenger\_Square.bmp", RGB(255, 255, 255));  passengerAnimationNight.LoadBitmap(".\\RES\\passenger\\Passenger\_Square\_night.bmp", RGB(255, 255, 255));  }  else if (passengerType == 3)  {  passengerAnimation.LoadBitmap(".\\RES\\passenger\\Passenger\_Diamond.bmp", RGB(255, 255, 255));  passengerAnimationNight.LoadBitmap(".\\RES\\passenger\\Passenger\_Diamond\_night.bmp", RGB(255, 255, 255));  }  else if (passengerType == 4)  {  passengerAnimation.LoadBitmap(".\\RES\\passenger\\Passenger\_Pentagon.bmp", RGB(255, 255, 255));  passengerAnimationNight.LoadBitmap(".\\RES\\passenger\\Passenger\_Pentagon\_night.bmp", RGB(255, 255, 255));  }  else if (passengerType == 5)  {  passengerAnimation.LoadBitmap(".\\RES\\passenger\\Passenger\_Hexagon.bmp", RGB(255, 255, 255));  passengerAnimationNight.LoadBitmap(".\\RES\\passenger\\Passenger\_Hexagon\_night.bmp", RGB(255, 255, 255));  }  }  void Passenger::SetType(int type)  {  passengerType = type;  }  void Passenger::SetXY(int xP, int yP)  {  x = xP;  y = yP;  }  void Passenger::SetX(int n)  {  x = n;  }  void Passenger::SetY(int n)  {  y = n;  }  void Passenger::SetIsNight(bool z)  {  isNight = z;  }  int Passenger::GetStartStation()  {  return startStation;  }  int Passenger::GetFinalStation()  {  return finalStation;  }  int Passenger::GetPassengerType()  {  return passengerType;  }  void Passenger::RandomMadePassenger(vector<Passenger\*>& passengerList, vector<Station>& stationList, int stationMaxium, int stationTypeNum, int totalPassenagerNum)  {  passengerList.clear();  int start = -1;  int end = -1;  int x = -1;  int y = -1;  int type = -1;  srand((unsigned)time(NULL));  for (int i = 0; i < totalPassenagerNum; i++)  {  start = rand() % stationMaxium;  type = rand() % stationTypeNum;  end = type;  while (start == type) start = rand() % stationMaxium;  stationList[start].SetAddPassenagerNum(1);  x = stationList[start].GetX();  y = stationList[start].GetY();  Passenger\* buf = new Passenger(type, start, end, x, y);  buf->LoadBitmap();  passengerList.push\_back(buf);  stationList[start].SetPassengerToStationPosition(buf);  }  }  void Passenger::OnShow()  {  passengerAnimation.SetTopLeft(x, y);  passengerAnimationNight.SetTopLeft(x, y);  if (isNight)  {  passengerAnimationNight.ShowBitmap();  }  else  {  passengerAnimation.ShowBitmap();  }  }  void Passenger::RandomOnePassenger(vector<Passenger\*>& passengerList, vector<Station>& stationList, int stationMaxium, int stationTypeNum)  {  int start = -1;  int end = -1;  int x = -1;  int y = -1;  int type = -1;  srand((unsigned)time(NULL));  start = rand() % stationMaxium;  type = rand() % stationTypeNum;  end = type;  while (start == type) start = rand() % stationMaxium;  stationList[start].SetAddPassenagerNum(1);  x = stationList[start].GetX();  y = stationList[start].GetY();  Passenger\* buf = new Passenger(type, start, end, x, y);  buf->LoadBitmap();  passengerList.push\_back(buf);  stationList[start].SetPassengerToStationPosition(buf);  }  Passenger::~Passenger()  {  }  } |

|  |
| --- |
| Station.h |
| namespace game\_framework  {  class Passenger;  class Station  {  public:  Station();  Station(int, int, int);  void LoadBitmap(); //載入圖形  void SetXY(int, int); //設定座標  void SetPassenagerNum(int); //設定目前乘客數量  void SetAddPassenagerNum(int); //設定增加目前乘客數量  void SetIsNight(bool);  int GetX();  int GetY();  int GetCenterPositionX();  int GetCenterPositionY();  int GetPassenagerNum();  void SetType(int);  void RandomBuildStation(vector<Station>&);  void CheckedOverLappingStation(vector<Station>&);  void SetPassengerToStationPosition(Passenger\* p);  void RefreshPassengerPosition();  void ErasePassenger(Passenger\* p);  void OnShowStation(); //將圖形貼到畫面  void OnShowPassengerInStation();  ~Station();  private:  const int MAXIUM\_STATION = 6; //會出現在地圖上的只有6個車站  const int MAXIUM\_STATION\_TYPE = 6; //車站樣式總共有6個 00~05  //mygame.h 裡面也有相同的屬性要改  const int MAX\_GAME\_MAP\_SIDE\_X = 770 - 135; //實際上的遊戲邊界X軸只有到770  const int MIN\_GAME\_MAP\_SIDE\_X = 30; //實際上的遊戲邊界X軸從30開始  const int MAX\_GAME\_MAP\_SIDE\_Y = 560 - 25; //實際上的遊戲邊界Y軸只有到560  const int MIN\_GAME\_MAP\_SIDE\_Y = 60; //實際上的遊戲邊界Y軸從60開始  const int PASSENAGE\_POSITION[10] = {25, 36, 47, 58, 69, 80, 91, 102, 113, 124};  const int centerPositionX; //中心X座標  const int centerPositionY; //中心Y座標  int x, y; //圖形座標  int stationType; //車站類型  int passenagerNum; //總共有幾個乘客在車站  bool isNight = false;  CMovingBitmap stationAnimation; //利用動畫作圖形  CMovingBitmap stationAnimationNight; //利用動畫作圖形  vector<Passenger\*> passengerInStationList; //在車站的乘客  };  } |

|  |
| --- |
| Station.cpp |
| #include "stdafx.h"  #include "Resource.h"  #include <mmsystem.h>  #include <ddraw.h>  #include <cstdlib>  #include <ctime>  #include "audio.h"  #include "gamelib.h"  #include "Passenger.h"  #include "Station.h"  namespace game\_framework  {  Station::Station()  : stationType(0), x(0), y(0), centerPositionX(0), centerPositionY(0)  { }  Station::Station(int type, int set\_x, int set\_y)  : stationType(type), x(set\_x), y(set\_y), centerPositionX(set\_x + 13), centerPositionY(set\_y + 13), passenagerNum(0)  { }  void Station::LoadBitmap()  {  if (stationType == 0)  {  stationAnimation.LoadBitmap(".\\RES\\Station\\Station\_Circle.bmp", RGB(255, 255, 255));  stationAnimationNight.LoadBitmap(".\\RES\\Station\\Station\_Circle\_night.bmp", RGB(255, 255, 255));  }  else if (stationType == 1)  {  stationAnimation.LoadBitmap(".\\RES\\Station\\Station\_Triangle.bmp", RGB(255, 255, 255));  stationAnimationNight.LoadBitmap(".\\RES\\Station\\Station\_Triangle\_night.bmp", RGB(255, 255, 255));  }  else if (stationType == 2)  {  stationAnimation.LoadBitmap(".\\RES\\Station\\Station\_Square.bmp", RGB(255, 255, 255));  stationAnimationNight.LoadBitmap(".\\RES\\Station\\Station\_Square\_night.bmp", RGB(255, 255, 255));  }  else if (stationType == 3)  {  stationAnimation.LoadBitmap(".\\RES\\Station\\Station\_Diamond.bmp", RGB(255, 255, 255));  stationAnimationNight.LoadBitmap(".\\RES\\Station\\Station\_Diamond\_night.bmp", RGB(255, 255, 255));  }  else if (stationType == 4)  {  stationAnimation.LoadBitmap(".\\RES\\Station\\Station\_Pentagon.bmp", RGB(255, 255, 255));  stationAnimationNight.LoadBitmap(".\\RES\\Station\\Station\_Pentagon\_night.bmp", RGB(255, 255, 255));  }  else if (stationType == 5)  {  stationAnimation.LoadBitmap(".\\RES\\Station\\Station\_Hexagon.bmp", RGB(255, 255, 255));  stationAnimationNight.LoadBitmap(".\\RES\\Station\\Station\_Hexagon\_night.bmp", RGB(255, 255, 255));  }  }  void Station::SetXY(int set\_x, int set\_y)  {  x = set\_x;  y = set\_y;  }  void Station::SetPassenagerNum(int num)  {  passenagerNum = num;  }  void Station::SetAddPassenagerNum(int num)  {  passenagerNum += num;  }  void Station::SetIsNight(bool z)  {  isNight = z;  }  int Station::GetX()  {  return x;  }  int Station::GetY()  {  return y;  }  int Station::GetCenterPositionX()  {  return centerPositionX;  }  int Station::GetCenterPositionY()  {  return centerPositionY;  }  int Station::GetPassenagerNum()  {  return passenagerNum;  }  void Station::SetType(int type)  {  stationType = type;  }  void Station::RandomBuildStation(vector<Station>& stationList)  {  vector<Station> a;  stationList.assign(a.begin(), a.end());  int x = -1;  int y = -1;  int type = -1;  srand((unsigned)time(NULL));  for (int i = 0; i < MAXIUM\_STATION; i++)  {  //type = rand() % MAXIUM\_STATION\_TYPE; //隨機選擇車站的形狀  type = i;  x = MIN\_GAME\_MAP\_SIDE\_X + rand() % (MAX\_GAME\_MAP\_SIDE\_X - MIN\_GAME\_MAP\_SIDE\_X);// 隨機選車站的X位置  y = MIN\_GAME\_MAP\_SIDE\_Y + rand() % (MAX\_GAME\_MAP\_SIDE\_Y - MIN\_GAME\_MAP\_SIDE\_Y);// 隨機選車站的Y位置  Station buf(type, x, y);  stationList.push\_back(buf);  stationList[i].LoadBitmap();  }  return;  }  void Station::CheckedOverLappingStation(vector<Station>& stationList)  {  for (int i = 1; i < MAXIUM\_STATION; i++)  {  for (int j = 0; j < i; j++)  {  while (1)  {  if (stationList[j].GetX() >= stationList[i].GetX() - 50 && stationList[j].GetX() <= stationList[i].GetX() + 50 &&  stationList[j].GetY() >= stationList[i].GetY() - 50 && stationList[j].GetY() <= stationList[i].GetY() + 50)  {  x = MIN\_GAME\_MAP\_SIDE\_X + rand() % (MAX\_GAME\_MAP\_SIDE\_X - MIN\_GAME\_MAP\_SIDE\_X);// 隨機選車站的X位置  y = MIN\_GAME\_MAP\_SIDE\_Y + rand() % (MAX\_GAME\_MAP\_SIDE\_Y - MIN\_GAME\_MAP\_SIDE\_Y);// 隨機選車站的Y位置  stationList[i].SetXY(x, y);  }  else break;  }  }  }  }  void Station::SetPassengerToStationPosition(Passenger\* p)  {  p->SetXY(x + 25 + (passenagerNum - 1) \* 10 + passenagerNum \* 1, y);  passengerInStationList.push\_back(p);  }  void Station::RefreshPassengerPosition()  {  passenagerNum = (int)passengerInStationList.size();  for (int i = 0; i < passenagerNum; i++)  {  passengerInStationList[i]->SetXY(x + 25 + i \* 10 + i \* 1, y);  }  }  void Station::ErasePassenger(Passenger\* p)  {  for (unsigned i = passengerInStationList.size() - 1; i >= 0 ; i--)  {  if (i >= 6)  {  break;  }  else if (passengerInStationList[i]->GetStartStation() == p->GetStartStation() &&  passengerInStationList[i]->GetFinalStation() == p->GetFinalStation())  {  passengerInStationList.erase(passengerInStationList.begin() + i);  return;  }  }  passenagerNum = (int)passengerInStationList.size();  }  void Station::OnShowStation()  {  stationAnimation.SetTopLeft(x, y);  stationAnimationNight.SetTopLeft(x, y);  if (isNight)  {  stationAnimationNight.ShowBitmap();  }  else  {  stationAnimation.ShowBitmap();  }  }  void Station::OnShowPassengerInStation()  {  for (unsigned i = 0; i < passengerInStationList.size(); i++)  {  passengerInStationList[i]->OnShow();  }  }  Station::~Station()  {  }  } |