b60901170黃彥翔 第十一組 EEBird

整體規劃：

我們在手機的google play 與appstore上找遊戲時，發現了flappy bird這款遊戲，因為好玩，且後續可以擴充的功能很多，因此我們經過討論後，決定寫一款類似於他的遊戲。

工作分配：

我(b06901170)所負責的是:

美工、影片、main.cpp中的背景、障礙物以及排行榜與得分的製作。(barrier.cpp , score.cpp , highscore.cpp)。

蔡易霖(b06901178):

負責鳥的動作以及其他相關判斷。(bird.cpp)

林育萱(b06901005):

負責主程式建構、遊戲開始與結束畫面的相關操作、main.cpp的整合，以及button的製作。(botton.cpp , LTexture.cpp , start.cpp)

個人程式碼:

1. barrier.cpp:

大致上包含五個函式。

1. void move():18~34行

mPosX += mVelX;代表mPoX(障礙物位置)每次經過一個迴圈就會向右移動mVelX個畫素，進而達到連續移動的效果。

而barrier.cpp中第23~29行敘述的是，若障礙物已經完全沒入視窗左方，則將其的位置移到視窗右方，並重新以隨機的方式賦予其新的y座標，已達到障礙物不斷出現，且y方向隨機的效果。（新的mPosX、mPosY的值，是利用四則運算算出來的）

1. void render():36~40行

將障礙物顯示於其位置座標上。

1. void lacationX()、lacationY():41~48 行

將障礙物的中央空隙的座標回報。

1. void reset(int x):50~61行

將物件id設為x，並初始化位置座標與速度。(mPosX、mPosY、mVelX、mVelY)

1. score.cpp:

由三個函式組成：

1. void score():10~24行

若鳥順利通過一個障礙物，則分數+1。

判斷障礙物通過鳥的x座標：

以bl1~bl5記住上一時刻障礙物的x座標，若此刻某個障礙物x座標小於鳥的x座標，且上一刻此障礙物x座標大於鳥的x座標，則代表鳥通過一個障礙物。

1. int scorelength(int x):45~55行

判斷傳入值x的位數並輸出。

1. string scorestring(int x):25~44行

將傳入值轉為string型態，並輸出。

1. highscore.cpp:

由一個函式組成：

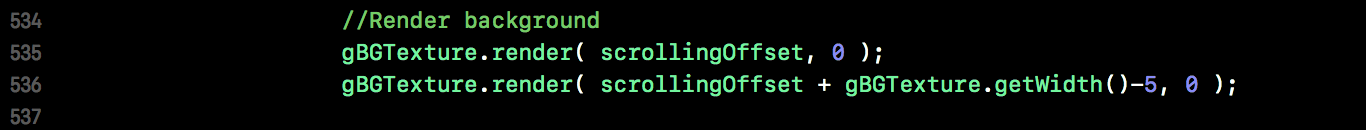
1. void sort():11~23行

比較highscore[i+1]是否大於highscore[i]，若是，則調換他們的位置，進而將highscore[i]由大排到小。

(在main.cpp中，int highscore[10]陣列，的功能是記憶遊戲中出現前十大的score數值，若新出現的score數值大於highscore[9]，則會取代掉highscore[9](main.cpp第599~602行)，輸出前再利用highscore.cpp中的sort()重新排列(main.cpp第571行)，使highscore陣列保持由大至小。)

1. main.cpp背景與分數：

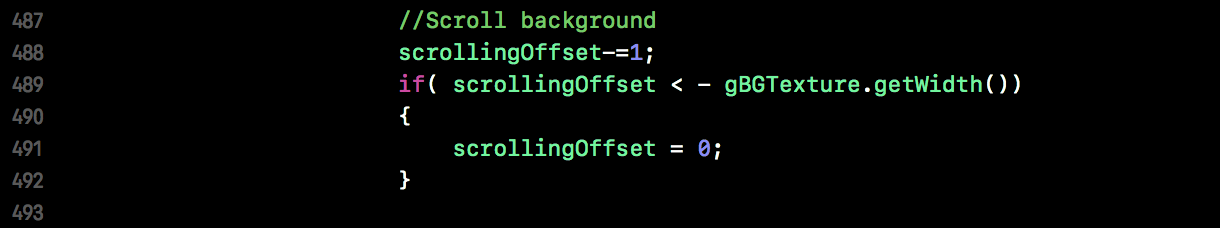
背景：



main.cpp第534~537行

由於要不斷重複一樣的背景，因此一次render兩個背景，且拼接在一起，以達到背景不間斷。

main.cpp第487~494行、第506~512行



同時scrollingOffset不斷減少，讓每一次迴圈時render的位置不同，以達到背景滾動的效果。又當scrollingOffset < -gBGTexture.getWidth()時(也就是背景完全沒入視窗左方時)，將scroingOffset重置，已達到背景不斷重複的錯覺。

分數：

1. main.cpp中，第546行，呼叫Score()以計算當前分數。
2. main.cpp中，第549~566行，目的是在遊戲中顯示當前得分。先判斷是否超過最高得分highscore[0]，若有，則前綴為”NEW HIGH:”；若無，則為”SCORE:” 而後將此字串顯示於指定位置。
3. main.cpp中，第575~579行，目的在於顯示最高得分前十名。將highscore陣列以scorestring()函式轉為string型態，在一一顯示於指定位置（由上而下）

物件導向：

1. 介紹類別(class):
2. 我們一共用了4個class:

Button.h中的Button、Barrier.hpp中的Barrier、Bird.h中的Bird、LTexture.h中的LTexture。

LTexture是在SDL2的教學網站上，每個程式碼基本上都會有的class，它的功能在於可以載入某個外部的.jpg , .png 檔案。

Button 的功能是控制每一個按鍵的出現位置、被選取時的反應以及選取完後續的動作。

Bird 控制的是，鳥的位置、加速度、玩家按“上”時的反應，以及是否發生碰撞。

Barrier 的功能是控制每一個障礙物的生成、障礙物隨機的y座標、回報障礙物中間空隙的位置，以及障礙物不停的向左滾動。

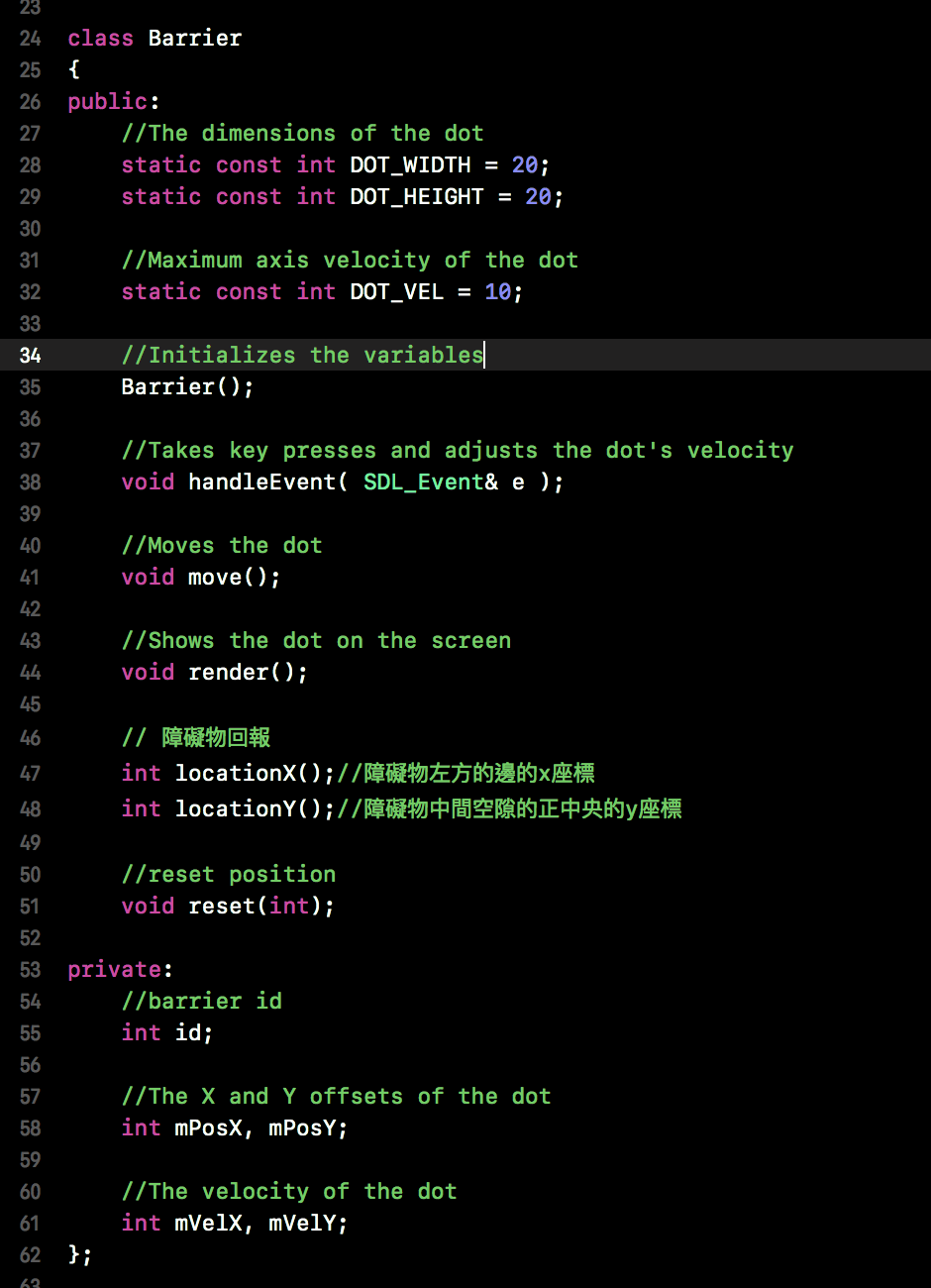
1. 物件化的優點：

Button、Barrier、LTexture都有共通的特性，就是重複性。遊戲中有button不斷出現，Barrier共有五個，LTexture也必須不斷出現以載入許多圖案，因此我們選擇將此三種物件化，避免不斷重複相同的步驟。

Bird雖然只有出現一次，但是由於Bird的許多操作與其他class相似，因此將它物件化較易於我們整理，若之後想要加入多隻鳥已進行多人的pk遊戲的話，也較易於擴充。

1. 封裝(encapsulation):

舉例:class Barrier(in “Barrier.hpp”)

class Barrier 中有兩種存取範圍：

public:

包含全部的function。

我們之所以將function都放在public是因為每一個function都將在此class外被呼叫。

例如：

1. move()在main.cpp中被呼叫，以移動障礙物的座標。
2. render()在main.cpp中被呼叫，以讓螢幕上出現障礙物。
3. locationX()和locationY()被用來呼叫“bird.collision(int x,int y)”(main.cpp 521行)。
4. 而reset(int)則在main.cpp中用來重置障礙物的位置。

private:

包含障礙物id、障礙物位置(mPosX,mPosY)、障礙物速度(mVelX,mVelY)。

此五個參數都只有在class中的function才會被用到，且會影響到整個障礙物的移動模式，因此將它們放在較緊的封裝private中。

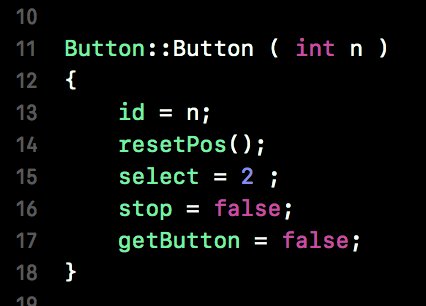
例如：

id只有在class Barrier中的reset()被用到。

1. mPosX、mPosY只有在class Barrier中的reset(int)、move()、render()、lacationX()、lacationY()被用到。
2. mVelX、mVelY只有在class Barrier中的reset(int)、move()被用到。

Barrier.hpp第24~62行

1. 建構子(constructor):



Button.cpp第11~18行

在Button.cpp中有用到建構子。

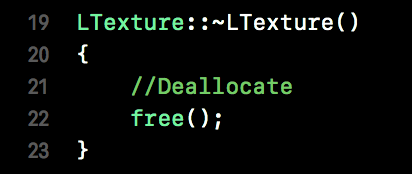
在宣告一個button物件的時候，利用建構子，我們可以：

初始化此button物件的id，以利我們之後用id一個一個控制button物件。

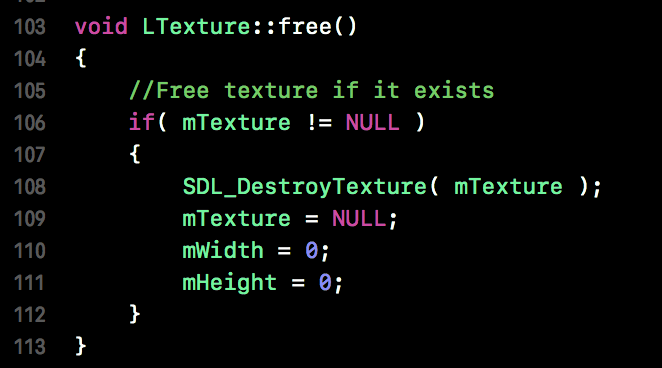
利用函式resetPos()初始化其位置。

初始化select (select為「被選到的button」)

初始化stop、getButton這兩個pruvate的參數。



LTexture.cpp第19~23行



LTexture.cpp第103~113行

解構子(destructor):

在Lexture.cpp中有用到解構子。

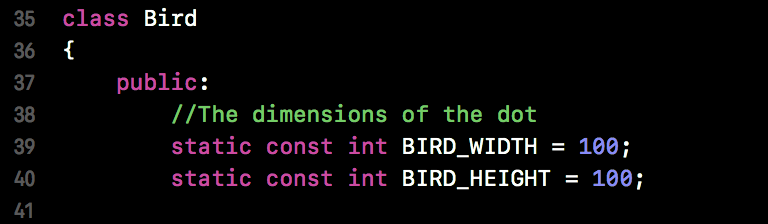
在一個LTexture物件被消除時，會啟動此解構子。

此解構子會呼叫class LTexture中的free()，free()會檢查mTexture是否為NULL，若mTexture不為NULL，便會將其變為NULL，以釋放記憶體。

(mTexture為載入的.png檔案或.jpg檔案)

1. 類別技巧：

static:



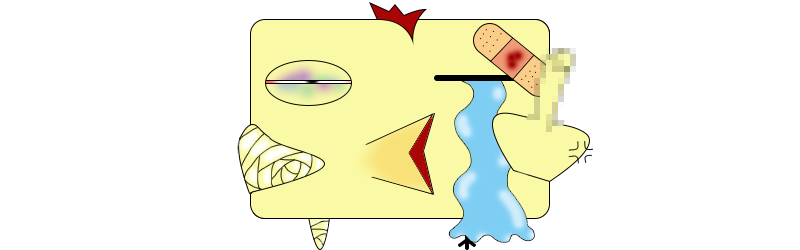
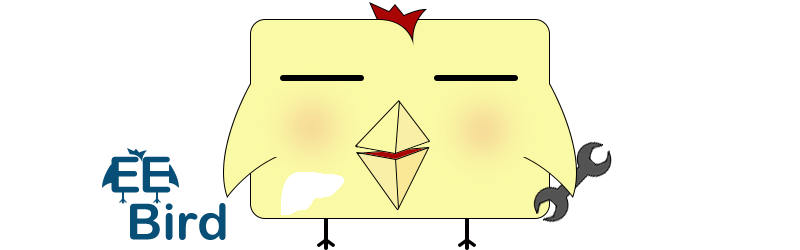
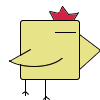
Bird.h第35~40行

在Bird.h中用了static。

因為不論有幾隻鳥，他們照片的長寬皆是一樣的，因此我們使用static，以避免不斷宣告一樣的參數。

(BIRD\_WIDTH以及BIRD\_HEIGHT將在判斷碰撞時用到)

圖片設計：

遊戲背景是台北街頭。剛來到台北，總覺得街頭的地標相當耀眼，相當輝煌。

與其說是遊戲背景，不如說是我對於這個大城市的第一印象。

撞到障礙物受傷後的鳥。

很痛，所以比中指，但是因為這是一個老幼皆適合的普遍級遊戲，所以被打碼了。

封面照。

呆呆鳥，拿著工具，缺著肝，符合外界對於電機系，或是工程師爆肝又時常作為工具人的形象

上升時的鳥，安逸，是封面照的側面

下降時的鳥，驚恐

