Project1_Group16

Main

> Main.cpp

僅呼叫一個 MainWindow 物件名為 mainwindow，並將其顯示。

MainWindow

> mainwindow.h
> mainwindow.cpp
> mainwindow_game_init.h

MainWindow 物件生成時，我們首先讓 Ctor 初始化一些內容：

- 設定顯示視窗 `view` 大小

> 💡 `view` 為 QGraphicsView 的物件，可以視為視窗。我們可以透過將不同的 QGraphicsScene 物件 (場景) 傳入 `view` 以達成切換畫面的效果。

- 將開始遊戲按鈕連接訊號槽到 `MainWindow::on_start_button_clicked()` 來呼叫 `MainWindow::game_init()` 以初始化遊戲畫面
- 利用 QTimer 建立一個計時 10ms 的計時器，再利用 訊號槽與觸發機制 設定每 10ms 呼叫一次 `MainWindow::update_frame()` 來更新畫面內容
- 呼叫成員函數 `MainWindow::start_init()` 來初始化開始畫面

> 💡 我們將 `gamestatus` 分為 3 種狀態：
> 0 ⇒ 開始畫面
> 1 ⇒ 遊戲中
> 2 ⇒ 結算畫面

▼ MainWindow::update_frame()

被呼叫頻率：每 10ms 會被訊號槽呼叫

功能：偵測目前遊戲狀態來更新畫面內容

- 當遊戲狀態為 0 (開始畫面) 時，將 `view` 設定為 `start_scene`
- 當遊戲狀態為 1 (遊戲中) 時，呼叫：
  - `MainWindow::all_move_detection()` 決定該讓馬力歐相對畫面移動或是讓畫面相對馬力歐移動

> 💡 我們讓馬力歐在移動時能夠固定在畫面中央的方法是讓畫面上的所有物件相對馬力歐去做移動，但當馬力歐在最左邊的場景時，因為最左邊的場景物件不能再往右移動 (意即露出場地外的部分)，因此需要讓馬力歐在畫面上的絕對座標去移動。`all_move_detection()` 就是用來決定是誰要相對誰移動的函數

  - 所有需要重力加速度的物件的成員函數 `move()` 以達成重力加速度的效果
  - 呼叫所有子彈物件的成員函數 `fly()` 以達成讓子彈依特定斜率飛行的效果

> 💡 我們讓物件(馬力歐、方塊、蘑菇等)實現重力加速度的方式是為每個物件額外定義 dx 和 dy 兩個變數，而遊戲狀態中每次 update_frame() 被呼叫時，就會接著去呼叫這些需要實現重力加速度的物件的成員函數 move()，讓它們的 x 和 y 座標分別增加 dx 和 dy，而重力加速度的實現只需要在每次 move() 後讓 dy 改變固定值，就如同將速度值加上一個加速度值一般。

  - 偵測若達到 GameOver 的其中之一個條件即呼叫 `Mainwindow::end_init()` 來初始化結算畫面。
- 當遊戲狀態為 2 (結算畫面) 時，將 `view` 設定為 `game_scene`

▼ MainWindow::start_init()

- 將遊戲狀態設為 0 (開始畫面)
- 將要提交給視窗 `view` 的場景 `cur_scene` 設為 `start_scene`
- 載入開始畫面的圖片(QPixmap 物件)，並生成一個開始畫面的物件(QGraphicsPixmapItem 物件)，再將圖片傳入物件，並設定物件的座標後加入目前的場景 `cur_scene` 當中
- 載入開始遊戲按鈕的圖片(QPixmap 物件)，並生成一個開始遊戲按鈕(QPushButton 物件)，再生成一個開始遊戲按鈕的物件(ButtonItem 物件)，再將圖片與按鈕傳入物件，並設定物件的座標後加入目前的場景 `cur_scene` 當中

> 💡 ButtonItem 是我們自定義的類別，功能是將 QPushButton 物件轉成 QGraphicsItem 物件

▼ MainWindow::end_init()

- 將遊戲狀態設為 2 (結算畫面)
- 將結算畫面的三個物件 (白底背景、GameOver圖片、結算文字) 的 x 座標設為 1800 (剛好在畫面外)
- 將白底背景物件 (QGraphicsRectItem 物件)、GameOver背景物件 (QGraphicsPixmapItem 物件)、文字物件 (QGraphicsTextItem 物件) 加入目前的場景 `cur_scene` 當中。

  > 💡 文字物件會去偵測遊戲結束的原因以決定顯示的內容。

- 新增一個 fade_in_timer (QTimer 物件) 來讓結算畫面的三個物件 (白底背景、GameOver圖片、結算文字) 能夠逐漸移動到視窗中央，透過將 fade_in_timer 設定為每 1ms 觸發一次並將訊號槽連接到 `Mainwindow::game_over_fade_in` 來實現。

▼ MainWindow::game_over_fade_in()

被呼叫頻率：每 1ms 被呼叫一次

每次被呼叫：

- 讓結算畫面的三個物件 (白底背景、GameOver圖片、結算文字) 的 x 座標往左移一點點。
- 偵測是否已經移到底(左側)：
  - 刪除 fade_in_timer 計時器
  - 載入重新遊戲按鈕的圖片(QPixmap 物件)，並生成一個重新遊戲按鈕(QPushButton 物件)，再生成一個重新遊戲按鈕的物件(ButtonItem 物件)，再將圖片與按鈕傳入物件，並設定物件的座標後加入目前的場景 `cur_scene` 當中。
  - 將重新遊戲按鈕連接訊號槽到 `MainWindow::on_restart_button_clicked()` 來呼叫 `MainWindow::game_restart()` 進行重新遊戲的參數更新。

▼ MainWindow::on_restart_button_clicked()

▼ MainWindow::keyPressEvent(QKeyEvent *event)

▼ MainWindow::keyReleaseEvent(QKeyEvent *event)
▼ MainWindow::mousePressEvent(QMouseEvent *event)
▼ MainWindow::all_move_detection()

▼ MainWindow::all_horizontal_move(int moving_unit)

Mario

About Size Changing

Hp

Score

Bullet

Blocks

Floor brick

Normal brick

Box brick

Broken brick

Invisible brick

Stone brick

Water pipe

Items

Coin

Fire flower

Flag & Flag pole

Mashrooms

Super Mashroom

Toxic Mashroom