第 16 章刺球遊戲

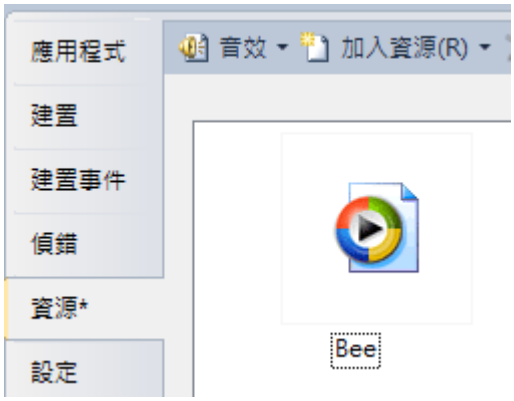
簡介：

本單元設計的遊戲會持續有多種外觀的球自畫面上方落下，玩家可以用鍵盤移動畫面下方的刺針物件，盡量快速地刺破最多氣球。此類遊戲的變形體很多，最常見的可能是開車跑彎道賽車或閃避障礙物之類的遊戲。我們可以在此學到遊戲設計中控制物件持續運動並產生隨機變化的一些技巧，也可學到物件碰撞檢測程式以及 Tag 標籤屬性在遊戲計分上面的應用。

16-1 建立資源檔

[加入圖案與音效資源檔]

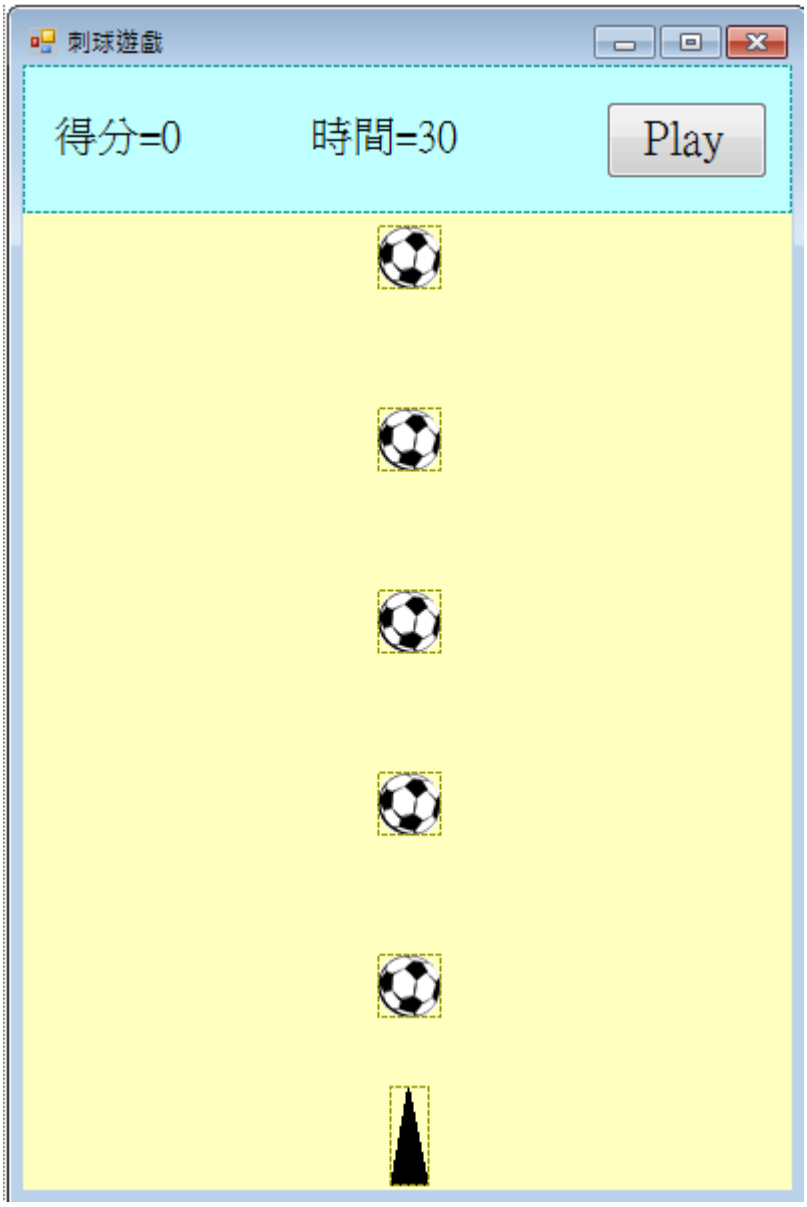
本專案將使用三種不同的球形圖案，還需要一個針狀物的圖案，你可以自行繪製或尋找如上的三個球型圖檔以及一個尖尖的三角形，球的大小為應設定為 32 x 32 像素點，針的圖案約為 25 x 50 點；且分別命名為：B1、B2、B3 與 Pin；再搜尋電腦或網路找一個適合當作刺破氣球時的音效檔案，本範例使用一個名為 Bee.wav 的檔案。請先將以上檔案加入專案資源檔。方法是：打開「方案總管」視窗，按滑鼠右鍵開啟 My Project 項目，打開專案的屬性視窗，點選「資源」頁籤，選擇「加入資源」的「加入現有檔案」，依序選擇上面四個檔案以及音效檔即可。完成畫面影像部分如下，音效檔案則會呈現於音效分類頁：



16-2 表單佈置

[計時計分與啟動鈕面板]

我們需要的表單介面如下圖。首先將表單作成長方形，在頂部加入一個 Panel 物件，Dock 屬性設為 Top(停駐在頂部)；其次在面板內加入得分與時間的看板標籤(得分=0，時間=30)，以及一個啟動遊戲的 Play 按鍵。



[計時計分與啟動鈕面板]

在面板下方視窗主要區域內請加入六個 PictureBox，SizeMode 屬性設為 AutoSize，圖案分別使用資源檔案中的 Pin 與 B3，Pin 圖案放在視窗底部，作為刺針物件，並將此 PictureBox 的名稱(name)屬性改為「A」，其餘五個 PictureBox 的名稱改為 P1~P5，並使

得球的垂直間隔大致相等。

16-3 讓球持續落下的程式

[啟動循環落下的目標]

接下來我們希望按下 Play 按鍵時五顆球會等速落下，每一顆球掉到視窗下面之後會再度拉回到最上方，看起來球會源源不絕的一直往下掉。要作到這種效果首先要加入一個 timer1 物件， Interval 屬性維持預設的 100，相當 0.1 秒。接著我們想用迴圈處理五個球的落下動作，程式如下：

```
private void button1_Click(object sender, EventArgs e)
{
    timer1.Start();
}

private void timer1_Tick(object sender, EventArgs e)
{
    foreach (var p in this.Controls)
    {
        if (p is PictureBox)
        {
            PictureBox P = (PictureBox)p;
            if (P.Name.Substring(0, 1) == "P")
            {
                P.Top += 5;
                if (P.Top > this.ClientSize.Height)
                {
                    P.Top = panel1.Bottom - P.Height;
                }
            }
        }
    }
}
```

在 button1 按鍵中只是啟動 timer1；timer1 事件中的程式碼就精彩了！先用 foreach 迴圈一一取出表單上的所有控制項(this.Controls)，取出的物件先看是不是 PictureBox？如果是的話，進一步看看它名稱的第一個字(P.Name.Substring(0, 1))是不是"P"？如果是！那就是球物件了！應該讓它的 Top 屬性加 5，就是高度往下掉 5 點，再檢查如果已經掉到畫面外(> this.ClientSize.Height)就要拉回到畫面上端，從操作面板的下緣重新出現(panel1.Bottom - P.Height)。

[調整物件層次]

測試一下按 Play 鍵之後的效果，此時常常會發現拉回的球壓在 panel1 的上面，因為球在 panel1 之後設計，所以堆疊層次高於 panel1。這時應該將 panel1 拉到最上層，操作畫面如下圖：

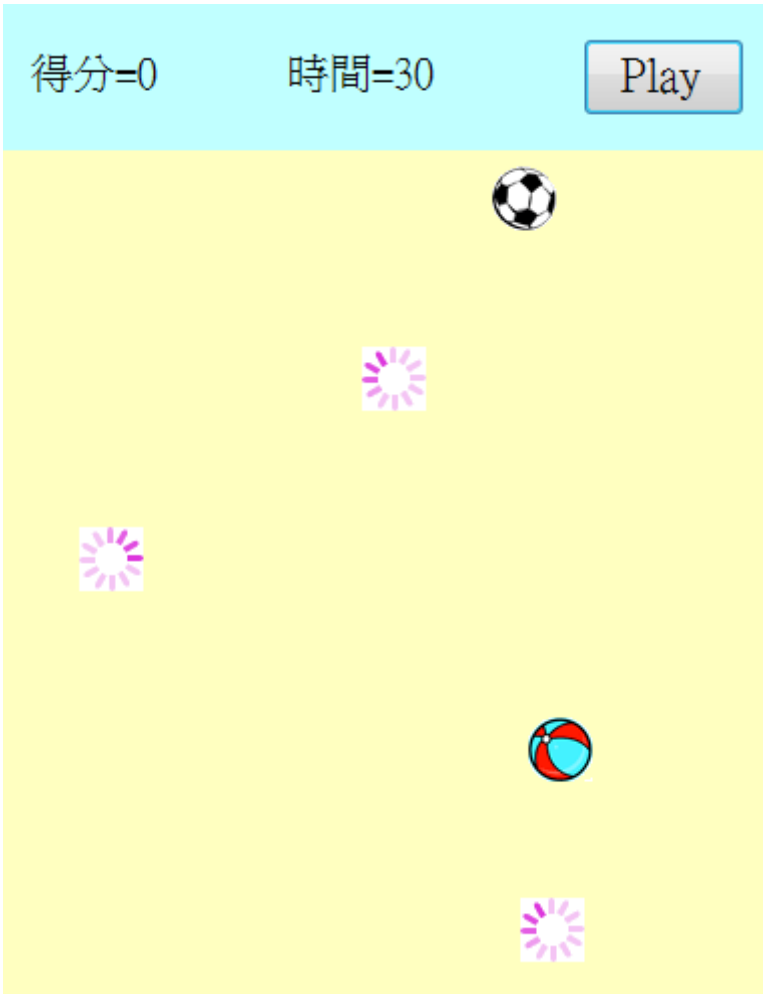
[隨機變化落下的水平位置]

為了讓球的位置以及圖案有隨機的改變，我們想在球拉回到最上面準備再次落下之前用亂數打亂它們的水平位置(Left)；同時間，我們也隨機選擇三種圖案之一(B1~B3)，讓球的外觀也有隨機的變化。為此，首先我們必須宣告建立一個亂數產生器(Random R = new Random;)，接著修改 timer1 事件副程式如下：

```
Random R = new Random();
    private void timer1_Tick(object sender, EventArgs e)
    {
        foreach (var p in this.Controls)
        {
            if (p is PictureBox)
            {
                PictureBox P = (PictureBox)p;
                if (P.Name.Substring(0, 1) == "P")
                {
                    P.Top += 5;
                    if (P.Top > this.ClientSize.Height)
                    {
                        int x = R.Next(this.ClientSize.Width  - P.Width);
                        P.Left = x;
                        P.Top = panel1.Bottom - P.Height;
                        int k = R.Next(3);
                        switch (k)
                        {
                            case 0:
                                P.Image = Properties.Resources.B1;
                                break;
                            case 1:
                                P.Image = Properties.Resources.B2;
                                break;
                            case 2:
                                P.Image = Properties.Resources.B3;
                                break;
                        }
                    }
                }
            }
        }
    }
```

[R.Next(N)的意義]

上面程式中 R.Next(N)的意義就是產生一個大於等於 0，但是小於 N 的整數，所以將物件可以左右移動又不會跑出畫面的最大寬度 this.ClientSize.Width - P.Width 作為參數就可以讓球在此範圍內取得一個任意的新水平位置，就會達到忽左忽右的效果了(請參考螢幕保護程式單元)！同樣的 R.Next(3)會產生 0，1 與 2 三種整數的可能性，我們將此三種狀況對應到三種圖案，新落下的球就會有隨機不同的圖案了！



16-4 控制刺針移動的程式

[使用鍵盤控制物件移動]

在此我們要使用鍵盤的 Z 與 X 來控制刺針左右移動，首先必須設定表單的 KeyPreview 屬性為 True，讓程式掃描鍵盤的事件，否則即使寫好鍵盤程式也不會運作，此屬性預設值是 False。接著寫 Form1 的 KeyDown 事件程式如下，應該就可以用鍵盤左右移動刺針了！

```
private void Form1_KeyDown(object  sender, KeyEventArgs e)
{
    switch (e.KeyCode) {
        case Keys.Z:
            A.Left -= 3;
            break;
        case Keys.X:
            A.Left += 3;
            break;
    }
}
```

在此使用滑鼠拖曳來控制刺針當然也無不可，用鍵盤其實靈活度較低，會增加一些玩遊戲的難度，但是同學們應該盡可能學會各種程式技巧，以備需要時參酌情況使用。

16-5 刺破氣球的程式

[刺到氣球了嗎？]

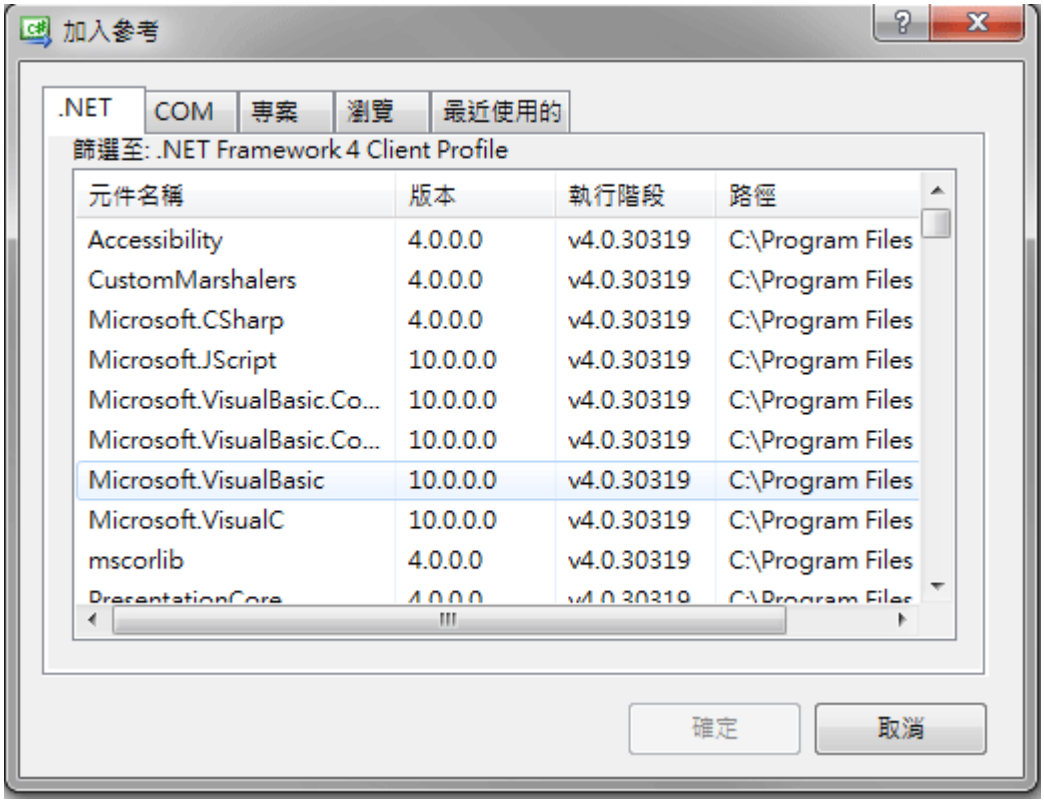
氣球被刺破應該立即消失(Visible=False)，且加計分數！不過在氣球重新下落之前不管氣球是否已被刺破一定要變回可見的狀態！在此的困難點在於如何判斷針已經刺到了氣球？這又是一個「碰撞」檢測程式的問題。我們的邏輯是在球移動到新位置時檢查刺針的頂端(針尖)是否在球的長寬範圍之內？如果答案為真！就算刺破了！相關程式需加在操控氣球下落的 timer1 事件中氣球移動之後(P.Top += 5;)加入以下這一段：

```
if (P.Visible)
{
    if (P.Bottom > A.Top)
    {
        int Ac = A.Left + A.Width / 2;
        if (Ac > P.Left && Ac < P.Right)
        {
            P.Visible = false;
        }
    }
}
```

上面程式以 if (P.Visible)為始，先看球是否可見(Visible=True)？如果已經被刺破的球 Visible 屬性會變成 False，不可見就無需檢測是否會被刺破了！接著是以程式 if (P.Bottom > A.Top)檢查球的高度是否可能被刺破？就是球的底部(P.Bottom)座標(Y)必須大於針尖(A.Top)。再來是檢查「針尖」(A 物件的中央線)的 X 座標(Ac)是不是位於球的左(Left)右(Right)邊緣之間？所有條件都符合時，就是球被刺破了，會導致 P.Visible = False 的結果。

[播放刺破氣球的音效]

在此我們還要播放之前匯入的音效，請參考打地鼠單元結的做法。先到功能表的「專案」項目選取「加入參考」，出現選擇視窗後到.NET 頁籤選擇「Microsoft Visual Basic」



再到程式碼前端加上兩個 using 指令：

```
using Microsoft.VisualBasic;
using Microsoft.VisualBasic.Devices;
```

接著在程式碼的公用區域，也就是在 timer1_Click 副程式外面加上一行宣告「Computer」物件 C 的程式碼：

```
Computer C = new Computer();
```

接下來再到刺破氣球的程式碼處加入音效播放程式如下：

```
P.Visible = false;
C.Audio.Play(Properties.Resources.Bee, AudioPlayMode.Background);
```

試試看，現在刺破氣球時就會有聲音了！

16-6 計時的程式

[計時與遊戲結束]

寫計時的程式需要宣告計一個變數 T=30，並加入一個 timer2 物件設定 Interval=1000 (一秒鐘)，每秒鐘 T 要減 1，直到 0 為止，到 T=0 時就顯示一個"Game over!"訊息，並停止計時器，順便也將計分的變數 S=0 一起宣告，程式碼如下：

```
int T=30;
int S=0;
private void timer2_Tick(object sender, EventArgs e)
{
    T -= 1;
    label2.Text = "時間=" + T.ToString();
    if (T == 0)
    {
        timer1.Stop();
        timer2.Stop();
        MessageBox.Show("Game over!", "CsBaloon");
    }
}
```

[啟動遊戲時的變數重設]

當然，相對應的在啟動遊戲之初的 button1 事件中必須也加入啟動計時器 timer2 的程式碼。還有計時計分的變數與看板也都必須復原，程式碼修改如下：

```
private void button1_Click(object sender, EventArgs e)
{
    timer1.Start();
    timer2.Start();
    S = 0;
    label1.Text = "得分=" + S.ToString();
    T = 30;
    label2.Text = "時間=" + T.ToString();
}
```

現在會動也會破，時間也會倒數了！只是球戳破之後就再也不會出現了！因為我們只是將它變不見(Visible=false) 沒有變回可見狀態，應該變回來的時機就是從上面重新落下之前，到 timer1 事件插入下面粗體字程式即可：

```
P.Top = panel1.Bottom - P.Height;
P.Visible = true;
int k = R.Next(3);
```

16-7 計分的程式

[用 Tag 屬性定義分數]

最後還必須加入刺破球時計分的機制，可以想見程式應該寫在球被刺破的時候，而且我們有三種球，可以賦予每一種球不同的分數！但如何定義球的分數呢？簡單的方法是將球的分數與種類結合，在此我們又可以利用到 Tag 屬性來操作了！首先我們定義讓圖案 B1 的球值 5 分，B2=3 分，B3=1 分，在設計階段以及球更換圖案時依據圖案將分數寫在 Tag 屬性之中！接著再將 timer1 中產生不同圖案的程式略做修改(粗體字)如下：

```
int k = R.Next(3);
switch (k)
{
    case 0:
        P.Image = Properties.Resources.B1;
        P.Tag = 5;
        break;
    case 1:
        P.Image = Properties.Resources.B2;
        P.Tag = 3;
        break;
    case 2:
        P.Image = Properties.Resources.B3;
        P.Tag = 1;
        break;
}
```

[分數就是刺破氣球的 Tag 值相加]

接著在檢查是否刺破氣球的部分則加入計分程式，如下粗體字部分：

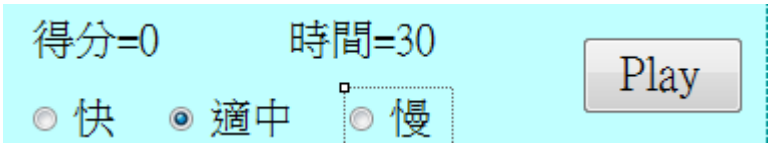
```
if (P.Visible)//如果球尚未被刺破
 {
  if (P.Bottom > A.Top)
   {
    int Ac = A.Left + A.Width / 2;
    if (Ac > P.Left && Ac < P.Right)
     {
      P.Visible = false;
      C.Audio.Play(Properties.Resources.Bee, AudioPlayMode.Background);
      S += int.Parse(P.Tag.ToString());
      label1.Text = "得分=" + S.ToString();
     }
   }
  }
```

至此遊戲執行結果畫面可能如下，測試時請注意不同的球是不是會有不同的分數？

16-8 速度調整的介面

[使用 RadioButton 選速度]

相信很多同學都對球落下的速度快慢有點意見，當然你可以直接到程式中改變「P.Top += 5」中的那個數字"5"就可以了！但是如果有操作介面會更好。請先到面板 panel1 裡面加入三個 RadioButton 物件(單選用)，畫面如下：



請注意先將「適中」項目的 Checked(勾選)屬性設為 True 表示速度會有個預設的值。請注意，同一物件容器(如 Panel)內的所有 RadioButton 只有一個可以是 Checked=True，也就是只有一個可以被選取，任何一個被選或 Checked 設定為 True 時其他的 RadioButton

會自動變成未選取狀態。接著請一一雙擊各個 RadioButton 寫程式如下：

```
int V = 5;
private void radioButton1_CheckedChanged(object  sender, EventArgs e)
{
    V = 7;
}

private void radioButton2_CheckedChanged(object  sender, EventArgs e)
{
    V = 5;
}

private void radioButton3_CheckedChanged(object  sender, EventArgs e)
{
    V = 3;
}
```

在此，必須先將下落速度宣告成一個全域變數 V，接著依 radioButton1~radioButton3 被勾選的狀態決定 V=3，5 或 7。當然落下速度既然已成為一個變數 V 了，原來控制下落的程式碼也應該改成這樣：

```
P.Top += V;
```

16-9 進階挑戰

一、如何設計使用滑鼠移動刺針？

提示：模仿射擊遊戲將刺針物件寫成可拖曳的物件。

二、如何設計到達一定分數可以過關增加玩的時間？

提示：檢查總分 S 大於某個分數時增加 T 值。

三、如何加入背景音樂？

提示：背景音樂也是音效，可以在啟動遊戲時開始播放，遊戲結束停止。

課後閱讀

談談迴圈

至今我們已經用過 for 以及 foreach 兩種迴圈，for 是指定起終點索引值與間距值作運算的(C#作數學單元)，適合用於已知數量並具有連續索引的物件或陣列；foreach 除了本單元我們也在工具箱單元用過，都是用來搜尋所有表單上的物件之用，這個語法不強調取得物件的次序，但是可使用於一堆沒有明確序號的『雜物』！即使是有索引的陣列，如果你的處理目的與次序無關，用 foreach 也是可以的，這樣的好處是你不必寫出迴圈的起始、結束與間距值，方便多多。

除了這兩種迴圈其實還有一種常用迴圈 do…while，語法結構如下：

do
{
程式碼
}while(條件式)

這類迴圈主要是根據區塊內的程式碼每一次執行的結果，決定是不是回到迴圈再作一次？譬如猜數字吧？猜錯就再來一次，猜對就跳出迴圈了！這種語法其實比較多出現於數值分析計算，在圖形化介面程式，尤其是遊戲程式比較少用到，這種迴圈結束時機比較不明確，是最有可能產生無限循環迴圈的語法，一旦發生無窮迴路，程式就會一直卡住了！

被遺忘的 Tag 屬性

[控制項的備註欄]

在打地鼠的單元中我們首次使用到物件的 Tag(標籤)屬性來記錄某個地鼠是不是已經被點擊過，如果被點過就不能重複加分；在本單元中我們再次用到它作為不同種類氣球擊破時可得的分數。事實上 Tag 是一個常常被設計者遺忘的屬性，多數的書籍中也不會刻意提起。它基本上是一個文數字型態未定的變數，設計者給它文字它就可以存文字，給它數值也可以存數值，任何想要附記於此物件的資料都可以寫進去，非常實用。

[如果不使用 Tag…]

我們可以想像一下如果不用 Tag 的解決方案，通常就是必須宣告與多個物件相對應的陣列。譬如本單元有五顆球，要記得每次球的圖片更改時計分的變化就必須類似這樣宣告一個陣列： Q[0]對應到球 P1，Q[1]對應到球 P2，依此類推…。這樣不是不行，但是程式的可閱讀性就差很多了，必須隨時提醒自己 Q 陣列就是 P 物件們的分數，而 Tag 本來就是 P 物件們自己的屬性，用 Tag 寫程式看起來就清楚多了！