第 10 章打地鼠遊戲

簡介：

本單元要介紹如何寫一個打地鼠的遊戲，目標圖案會在預設的幾個位置隨機出現，短時間內消失，玩家必須用滑鼠及時點擊目標取得分數，遊戲還要限時結束且能計分。包含的程式技巧有資源檔案的使用，隨機亂數的操作，計時器的使用，以及用全域變數計時與計分，這些都是遊戲程式中常用的基本技術。

10-1 加入影音資源

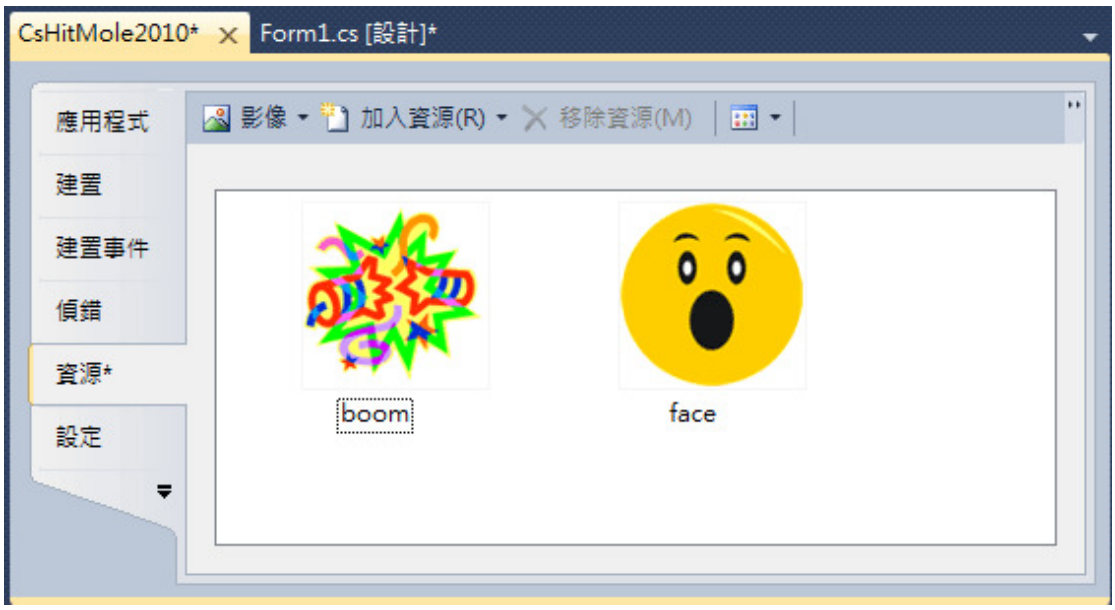
[載入影像的兩種方式]

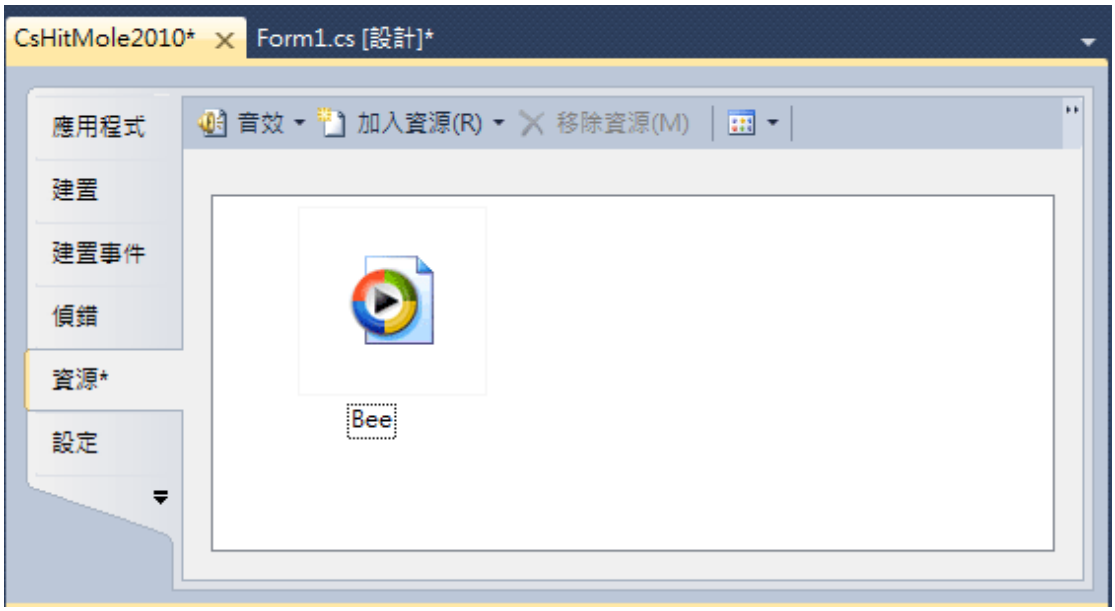
當我們需要在程式專案內使用影像時，基本上有兩種方式：一是在需要使用影像物件的 Image 或 BackgroundImage 屬性欄中選取檔案載入；一是將圖檔載入為專案的「資源」檔。前者的影像隨後就專屬於該物件，後者則變成專案內任何物件都能以程式碼操作隨時引用的「資源」。在本單元中地鼠物件的圖案在執行中需要切換，所以我們選擇採用資源檔的方式載入使用較為方便。

同樣的，音效檔案也可以在設計階段事先載入為資源檔。不論使用哪一種方式，都告訴我們一個概念！我們可以將很多專案需要的圖檔、音效，甚至文字或文件檔案先放在專案內，變成專案內部的東西，就像工具箱物件一樣，使用起來就很方便了！換言之，你的專案需要的很多東西都可以與專案直接整合，不必製作成程式後還得帶著很多檔案同行！

[載入專案資源檔]

載入資源檔的動作是：開啟專案後開啟「方案總管」中的 Properties 項目，進入「專案屬性」頁，選擇「資源」頁籤，選擇「加入資源」→「加入現有檔案」。依序加入兩個圖檔(face.gif & boom.gif)，以及一個音效檔(Bee.wav)，兩圖的長寬都是 100 像素點，這是稍後當作目標(face)以及目標被點中後(boom)的圖案，音效檔(Bee)則作為目標被點中時的聲音。





請注意到載入的資源檔案專案會自動將它們分類，如上面的兩個畫面，不管你是一一載入或一起載入，音效與影像都會分在兩個頁面，不要誤會載入失敗而重複載入。當然這些資源檔不用時也應該將它們刪除，以免你的最終程式負載太多無用的東西，為了寫程式方便，在此也可以將它們的名字變更。

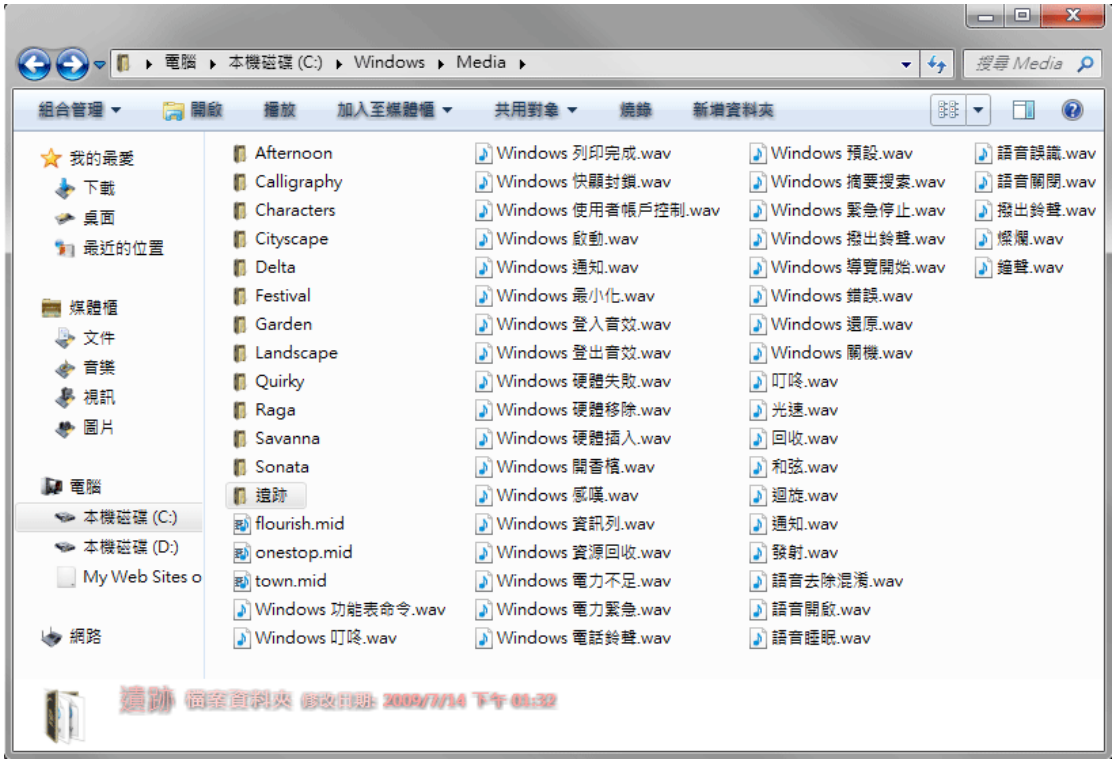
[圖檔的取得與選擇]

手邊沒有適合的圖檔嗎？最簡單的方式是開啟檔案總管，在自己的電腦內搜尋"*.gif"應該就可以找到上百張圖片；不然也可以到 Office 軟體中找到插入「美工圖案」之類的功能，可以用搜尋主題的方式找圖，找到之後可以用複製的方式貼到小畫家等等軟體再存成 gif 格式的檔案。或許有人想問為什要用 gif？jpg 格式圖檔就不行嗎？當然可以！但是這類圖案設計類的圖片用 gif 的壓縮格式會相對清晰且檔案小，jpg 在此壓縮效果與品質較不理想，詳細原因就是較深入的影

像處理議題了。

[取得現成的音效檔]

那麼音效檔案哪裡有呢？一樣的在自己的電腦搜尋*.wav 檔案就可以找到很多，還可以試聽一下找一個喜歡的！通常 Windows 系統內建的音效會放在 C:\Windows\Media 目錄內(如下圖)，在這裡就多到找不玩了！要誇張一點的還可以到你安裝的遊戲軟體目錄內找一找，一定不會失望的！當然這只是學習程式設計時的做法，真的要將成果對外展式或銷售時就必須注意影像及音效的版權問題了！



10-2 表單物件佈置

[表單畫面設計]

將表單標題修改為『打地鼠遊戲』，置入一個容器(panel1) 物件，Dock(停駐)屬性設定為在表單的下方(Bottom)。在 panel1 中置入一個計分用的標籤，寫『分數=0』(label1)；置入一個計時標籤，寫『時間=30』(label2)；加入一個啟動按鍵(button1)文字寫 Play。在此介面簡單，可以將字體放大一點(範例為 18 點)！

在視窗內容區(ClientSize)加入六個有臉形圖案的標籤，做法是先叫用一個標籤，將它的 AutoSize 屬性設為 False；刪除 Text 屬性中的預設文字；選擇 Image 屬性的影像為資源檔中的臉形圖案，長寬調整為符合影像的 100x100 點，再將此標籤複製為六個，並將六個標籤分別改名為 M1~M6(Name 屬性)以簡化後續程式碼的寫作，在此範例中標籤的位置次序是沒關係的，完成畫面如下：

[為何要用 Label 顯示圖片？]

或許有同學會問：顯示影像的物件不是應該用 PictureBox 嗎？為何在此要用 Label？圖片還不太好設定咧！主要原因是 PictureBox 相對於 Label 是個比較複雜功能較多的物件，這會佔去較多的記憶體，多出的功能如影像縮放旋轉等等，我們在此又用不到！如果可以節省下這些電腦資源，你的程式會運轉更快，也有更大的設計空間，譬如放置更多地鼠。

[記憶體能省則省]

這其實是一個概念的學習，雖然現在的電腦硬體都很好，記憶體也很大，但是同樣的環境下，如果你可以讓自己的程式使用記憶體更少，跑得更快始終是會有優勢的！尤其現在流行在記憶體較小的手機(或平板電腦)上設計程式時，節省資源的概念就更加重要了！

10-3 用物件名稱取得物件參考的程式

遊戲程式內常常有很多物件必須隨時用程式控制它們的：出現、隱藏、移動或改變圖案等等，這時如果這些物件是陣列就很方便(寫成 label[?].Visible=…)，譬如有一大堆標籤要隱藏時用迴圈只需幾行程式即可完成。但是可惜 C#程式目前版本預設不使用物件陣列，因此簡單的處理方式是讓物件名稱有一定的規律，以便藉此規律以迴圈大量處理這些物件，不必一個物件寫一行程式，像

M1.Visibl3=false; M2.Visible=false; M3.Visible=false; …這樣。

在同版本的 VB 2008 程式可以用 Me.Controls("M"+i.Tostring)這樣的語法取得名稱為"M？"物件控制權，但是很不幸目前我在 C#中還找不到類似的簡潔語法(如果有人知道請告知，感激不盡)。所以我們在此先自力救濟，寫一個可以用名字取得標籤物件參考的副程式：

```
private Label getLabel(String name)
{
    foreach (var c in this.Controls)
    {
        if (c is Label)
        {
            Label i = (Label)c;
            if (i.Name == name)
            { return i; }
        }
    }
    return null;
}
```

[Controls 物件集合的意義]

其中 Controls 翻譯成控制項，其實就是指表單(this)上的**物件們**，foreach 的迴圈一一取出未知資料型態(var)的控制項 c，接著看如果它是一個標籤(Label)則明確宣告一個標籤變數 i 代表此物件，再細看此物件的名稱(name 屬性)是不是與此副程式指定的參數(name)一樣？如果是的話表示找到目標物件了，就回傳(return)此物件。於是呼叫此副程式的地方就可以傳入要找的物件名稱，取得的回傳值就是要找的物件了！有了這樣的副程式我們才有可能用迴圈處理像 M1~M6 這樣的多個物件的動作。

必須注意到物件集合是有階層的，容器類的物件如 panel1 之內可以裝其他的控制項，此時這些物件就不在 this.Controls 集合內，而是在 this.panel1.Controls 裡面。讓人胡塗的是它們通常沒有明確的介面操作可以界定哪個物件是設在哪個容器之內，譬如一個 Label 拖曳到 Panel 之內就自動成為它的成員了！真的想明確處理就必須到 Designer.cs 檔案內寫程式，譬如真正讓 label1 變成 panel1 一員的程式是：this.panel1.Controls.Add(this.label1); 刪除這一行，label1 就不再屬於 panel1 而是回歸到 this.Controls 的一個成員了，即使它的位置還是在 panel1 上面。

10-4 啟動遊戲→使圖案隨機出現

[定時切換地鼠位置的程式]

回到我們的打地鼠程式，要讓地鼠定時的隨機出現一隻，最簡單的方法是利用一個計時器，每次時間到時先將所有地鼠隱藏，再用亂數隨機選一隻出來顯示。同時間「Play」的按鍵就用來啟動這個計時器。請自工具箱的「元件」類加

入一個 timer1，將其 Interval(時間間隔)屬性設為 1000(等於一秒)。接著雙擊 Play 按鍵以及 timer1 物件寫程式如下：

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

private void timer1_Tick(object sender, EventArgs e)
{
    for (int i = 1; i <= 6; i++)
    {
        Label x = getLabel("M" + i.ToString());
        x.Visible = false;
    }
    int j = R.Next(6) + 1;
    Label y = getLabel("M" + j.ToString());
    y.Visible = true;
}
```

程式首先建置一個隨機數產生器 R，在按鍵 button1 啟動計時器，計時器程式一開始會用迴圈讓 M1~M6 的標籤通通消失，接著使用 R 產生一個(1~6)之間的隨機亂數 j，再取得"Mj"這個物件的參考讓它變成可見的(Visible)，就完成了每一秒鐘隨機出現一隻地鼠的動作。

[隨機選擇一隻地鼠的程式]

程式碼 int j = R.Next(6) + 1 的意義是宣告一個 1 到 6 之間隨機選擇的整數，R.Next(6)會產生一個大於等於 0 小於 6 的整數，也就是 0~5 之間的任意數，加上 1 之後就是 1~6 的隨機數了！接著再用 getLabel 取得 M"j"這個物件讓它顯示出來(Visible=true)就完工了！

10-5 打地鼠囉

[計時與計分的機制]

這個遊戲需要持續的計時計分，所以程式一定需要全域的公用變數來紀錄這兩個參數，請先在程式開始處這樣宣告，當然雙斜線後是方便閱讀程式的註解。

```
int S = 0; //分數
int T = 30; //時間
```

接下來我們要加入打地鼠的互動程式。我們先從 M1_Click 事件寫起：

```
private void M1_Click(object sender, EventArgs e)
{
    S += 1;
    label1.Text = "得分=" + S.ToString();
    M1.Image =Properties.Resources.boom;
}
```

上述程式 S+=1 就是加分，之後隨即在 label1 顯示目前得分為多少？接著 M1 的 Image 屬性要變成爆炸圖案(boom)，因為之前已經將圖案設為資源，所以程式呼叫方法為：(專案的)Properties・Resource・boom。此時測試你的程式，發現只有 M1 對於點即會產生加分及變臉(圖案變成爆炸)的效果！但是其他五個地鼠沒反應，且爆炸圖不會變回人臉圖案。

[共用打地鼠事件副程式]

要讓 M1 之外的其他地鼠的 Click 事件共用程式 M1_Click，必須用程式碼交代清楚，方法與小畫家單元中程式開啟時共用開新檔案功能(6.-2 節)相同，請到 Form_Load 事件中加入以下程式碼，這樣表單一載入時，六個 Click 事件就都指向 M1_Click 了！

```
private void Form1_Load(object sender, EventArgs e)
{
    this.M2.Click += new System.EventHandler(this.M1_Click);
    this.M3.Click += new System.EventHandler(this.M1_Click);
    this.M4.Click += new System.EventHandler(this.M1_Click);
    this.M5.Click += new System.EventHandler(this.M1_Click);
    this.M6.Click += new System.EventHandler(this.M1_Click);
}
```

同時也必須將 M1_Click 中的物件名稱 M1 變成代名詞 sender，M1.Image…的程式修改如下：

```
Label x = (Label)sender;
x.Image = Properties.Resources.boom;
```

[復原圖案的程式]

在此，當六個物件共用一段程式時，sender 就是觸發事件的物件，可能為 M1~M6 中的任何一個，如果每次點擊地鼠，不管點到誰都只改變 M1 的圖案當然不合理！其實 sender 就是為了這種狀況設計的，不然就無法共用事件副程式了！試試看，現在六隻地鼠都可以回應點擊變臉的效果了！但是再次出現時不會變回原來的 face 圖案，這必須在 timer1 中地鼠要出現(Visible=true)之前加入一行程式如下：

```
Label y = getLabel("M" + j.ToString());
y.Image = Properties.Resources.face;
y.Visible = true;
```

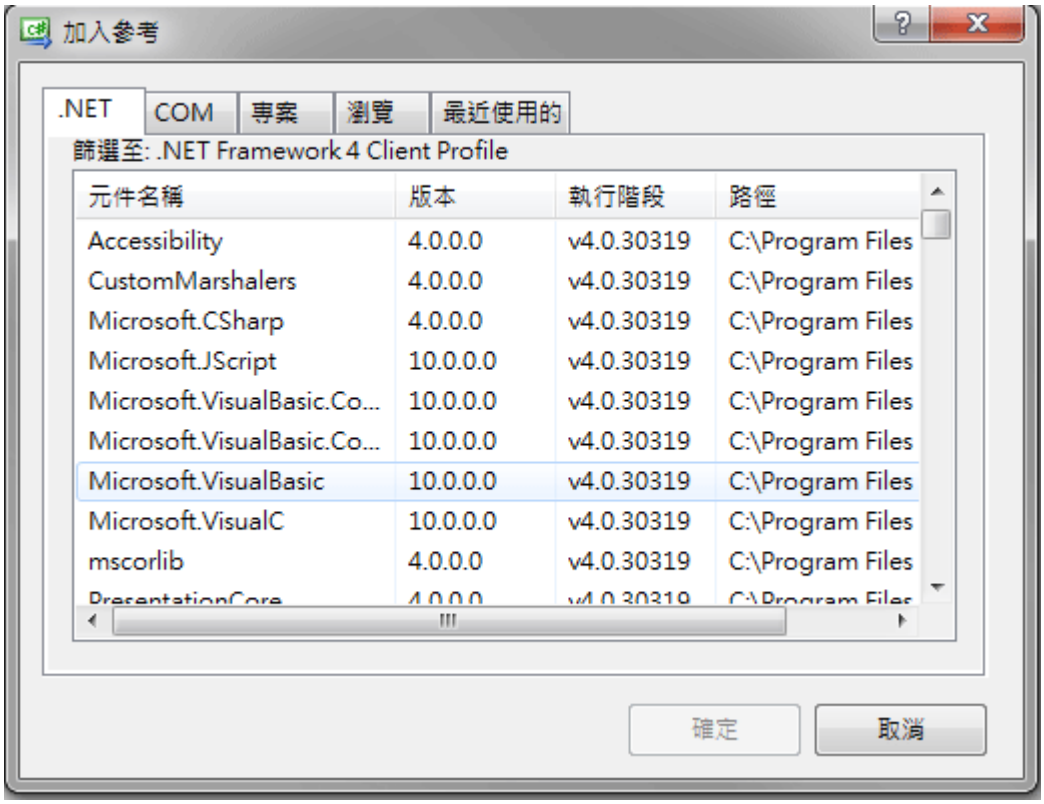
10-6 加入音效

[借用 VB 程式播放音效]

有點不好意思的事情是 C#沒有自己預設的音效播放程式，所以必須向 VB

借用！好像 VB 才是寫遊戲的正確語言？要用音效首先必須先到功能表的「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」

[加入參考]



[宣告匯入]

這樣還不行， C#是非常嚴謹的！還要到程式碼前端加上兩個 using 指令：

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

這樣才算是準備好可以宣告建立播放音效的物件了！請在程式碼的公用區域，就是在副程式外面加上一行宣告「Computer」物件 C 的程式碼：

[宣告與使用播放音效物件]

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

接下來再到 M1_Click 程式中加入音效播放程式如下：

```
private void M1_Click(object sender, EventArgs e)
{
    S += 1;
    label1.Text = "得分=" + S.ToString();
    Label x = (Label)sender;
    x.Image = Properties.Resources.boom;
    C.Audio.Play(Properties.Resources.Bee, AudioPlayMode.Background);
}
```

播放音效功能是由 Computer 物件的 Audio.Play 函數執行，第一個參數是使用資源檔中的 Bee 音效，第二個參數 AudioPlayMode.Background 是指「背景播放」，就是播音效時其它的視窗動作都不要暫停的意思。試試看，此時地鼠被點中時就會有聲音了！

10-7 防止快手搶分

至此還有個小漏洞，如果手腳快的玩家可以趁同一隻地鼠未消失之前連續點擊得分，這好像有點不合理？記得嗎？我們之前提過人的反射動作可以快到近 1/10 秒！所以此地地鼠出現一秒鐘之內被連點兩三次是很可能的！

[使用 Tag 屬性]

我們必須將地鼠是否已經被點擊的狀態記住，被點過就不能再回應點擊動作(加分)了，但下次出現之前又要恢復為未點擊狀態，傳統技術上好像必須建立一個陣列來記住這個資訊。但事實上多數物件都有個 Tag(標籤)屬性可以擔任這個暫時註記的工作，我們就用它來記錄吧！被點過的 Tag = 1，未被點就是 0 了！加入此一機制之後程式碼如下：

```
private void timer1_Tick(object sender, EventArgs e)
{
    for (int i = 1; i <= 6; i++)
    {
        Label x = getLabel("M" + i.ToString());
        x.Image = Properties.Resources.face;
        x.Visible = false;
    }
    int j = R.Next(6) + 1;
    Label y = getLabel("M" + j.ToString());
    y.Tag = 0; //未被點狀態
    y.Visible = true;
}

private void M1_Click(object sender, EventArgs e)
{
    Label x = (Label)sender;
    if ((int)x.Tag == 0)
    {
        S += 1;
        label1.Text = "得分=" + S.ToString();
        x.Image = Properties.Resources.boom;
        C.Audio.Play(Properties.Resources.Bee, AudioPlayMode.Background);
        x.Tag = 1;
    }
}
```

簡單說就是在地鼠出現時定義它的 Tag 是 0，被點擊時先檢查它的 Tag，如果是 0 才可以執行加分等動作，最後讓 Tag=1。請注意(int)x.Tag == 0 的語法，(int)是明確指定 x.Tag 的資料型態為整數，這樣才能與整數 0 做比較運算，看起來有些多餘，但是 C#是嚴格要求這樣寫的！

或許有人會問：何必這麼麻煩？點擊到地鼠時直接讓它消失(Visible = false)不就點不到第二次，也不會連續計分了嗎？但是這樣設計就不能看到被點中時的爆炸圖案了！設計方式很多，算是各有利弊吧？

10-8 計時機制

[計時與遊戲結束的機制]

接下來要加入一個倒數計時的機制讓遊戲可以按時結束。方法是使用一個新的計時物件 timer2，Interval 屬性也是設定為 1000(一秒)，每次減一秒並顯示於計時標籤(label2)，時間到了就會關掉 timer1 和它自己(timer2)，遊戲也就停止了，此時還應該出現一個 Game Over 的訊息！當然，timer2 本身也必須在遊戲開始時開啟，而且分數與時間都必須在此重設！

[ProgressBar 物件的使用]

在此為了增加視覺效果，可以增加一個工具箱中通用控制項的 ProgressBar 物件，將它停駐(Dock)在控制面板的底部(Bottom)，Maximium 屬性設為最大時間的 30，Value 也設為 30，表示從此開始倒數，之後程式碼只要將時間變數 T

與 progressBar1 的 Value 屬性同步，我們就會看到如沙漏一般的倒數計時圖形化介面了。最後 Play 按鍵與 timer2 計時器的程式如下：

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

private void timer2_Tick(object sender, EventArgs e)
{
    T -= 1;
    label2.Text = "時間=" + T.ToString();
    progressBar1.Value = T;
    if (T == 0)
    {
        timer1.Stop();
        timer2.Stop();
        MessageBox.Show("Game over!");
    }
}
```

執行中畫面如下：

10-9 多個地鼠同時出現

[使用迴圈顯示多個地鼠]

多數這一類遊戲到了後半段都會變快且同時出現不只一隻地鼠。要讓我們的程式也可以同時出現好幾隻地鼠相當簡單，只要加上一個迴圈就可以了！還記得挑選「一」隻地鼠的程式段落嗎？用迴圈將它包起來就 OK 了！

```
for (int k = 1; k <= 2;k++ )
{
    int j = R.Next(6) + 1;
    Label y = getLabel("M" + j.ToString());
    y.Image = Properties.Resources.face;
    y.Tag = 0;
    y.Visible = true;
}
```

請注意到，迴圈重複執行亂數選取時每次都是獨立事件，因此可能會有「相同」或「不同」的結果，上面的迴圈 for (int k = 1; k <= 2;k++)會無條件執行兩次選擇地鼠的動作，它其實有可能會重複，所以有時候出現兩隻，有時一隻地鼠都是正常的！

10-10 進階挑戰

一、可以讓地鼠不要限制在固定位置嗎？

提示：可以如螢幕保護程式範例，每次出現時隨機改變地鼠位置。

二、如何設計不同難度的關卡？

提示：可用 Interval 調整速度，用迴圈控制地鼠出現的個數。

課後閱讀

談談資料型別轉換

程式設計時會用到很多種資料型別，從簡單的文字、數字到影像、控制項乃至自己建立的資料類別等等。在 C#語言中這是非常嚴肅的議題，譬如數字之下還分成 int, float, double 等等多達十餘種數值型別，所以並不是數字就一定可以定義給另一個數字變數，譬如下列程式是會當掉的！原因是 double 是一個可以表達較精細複雜數字的型別，而 int 整數無法保證完全承接 double 的正確內容，在某些語言可能就自動簡化數字算了(如 VB？)，C#則絕對不允許這種轉換！

```
double d = 10.333;
int k = d;
```

相反的，將簡單資料自動轉成複雜資料型別則是可以的，譬如將整數 int 定義給 double 變數。那麼要如何處理「繁到簡」的資料轉換呢？一般有兩種寫法：

```
double d=10.333;
int k= (int)d;
int m = System.Convert.ToInt32(d);
```

(資料型別)的簡單寫法在此處理能力有限，是無條件去小數的；使用 Convert 函式則可以有四捨五入的功能。此外在本單元也用到這種型別轉換：

```
Label x = (Label)sender;
```

可見工具箱的控制項(如 Label)也是資料型別，上面程式就是明確宣告將事件觸發者轉換為 Label 型別並定義給 Label 變數 x。但是不必天真到相信上述語法可以將一個 TextBox 轉成 Label！如果 sender 不是 Label 物件，上面的程式還是會當掉的！

所以在此看來 C#可能太官僚(不夠友善親民)了一點，一方面規定如上的型別轉換(Label)非寫不可，但是卻『只能轉換**原本就是 Label** 的物件』！哈哈，那還用轉換嗎？下面這段程式也是一樣，第二行必須加(int)才可以通過程式編譯，但是因為第一行將 x.Tag 定義為文字"aa"，所以到了程式執行階段依舊會因為文字"aa"轉不成 int 而當掉！

```
x.Tag = "aa";
if ((int)x.Tag == 0)
  {...}
```

資料型別認真講可以介紹好一個月，但是我認為實際用到的種類不多！還是用到甚麼再介紹甚麼就好，目前基本資料型別認得 int(整數), double(倍準數)以及 string(字串)就差不多了！這裡只是講講 C#的壞話，提醒大家知道他古板僵化的一面，希望下一版 C#這些蠢蠢的轉換機制可以變得更聰明，也親民一點！不要再用『語法嚴謹』這種話來唬弄人了！實在沒必要的。