

第八章 表單輸出入介面設計



8.1 Form常用的屬性

8.4 TextBox文字方塊控制項

8.2 Form常用的事件

8.5 Button按鈕控制項

8.3 Label和LinkLabel標籤控制項

8.6 MessageBox. Show方法

備註：可依進度點選小節



8.1 Form 常用的屬性

- 表單（Form）是視窗應用程式中最重要的容器（Container）之一。
- 它可容納各種控制項。
- 表單和控制項有些屬性是自己所特有，有些是彼此擁有。
- 每個屬性都有其預設值。
- 不隨意更動預設值以免發生無法預測的情形。
- Visual C# 2008 兩種方式選取屬性：
 - ① 在屬性視窗點選 分類圖示。
 - ② 在屬性視窗點選 按字母順序圖示。





1. Text屬性

- 設定標題欄上面的文字。
- 由於Visual C# 2008 將目前被點選的表單當作用表單，在程式中以 **this** 代替目前作用的表單名稱 **Form1** 。
- 程式設定：
this.Text = "我的表單";



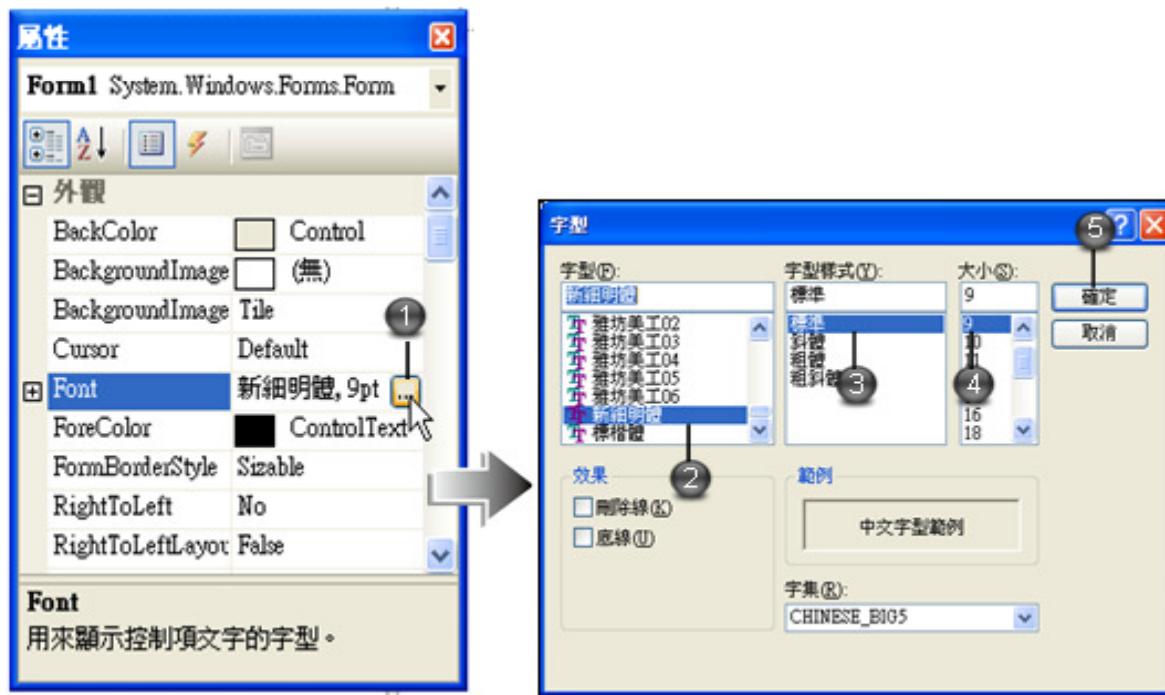
2. ForeColor/BackColor屬性

- ForeColor、BackColor屬性分別用來設定表單的前景色和背景色。
- Visual C# 2008 所建立的控制項具有繼承的特性
- 如ForeColor 屬性值的變更會影響在表單新建立後控制項的字體顏色。
- 程式設定：將目前作用表單的前景色設成淺藍色
`this.ForeColor = Color.Aqua;`



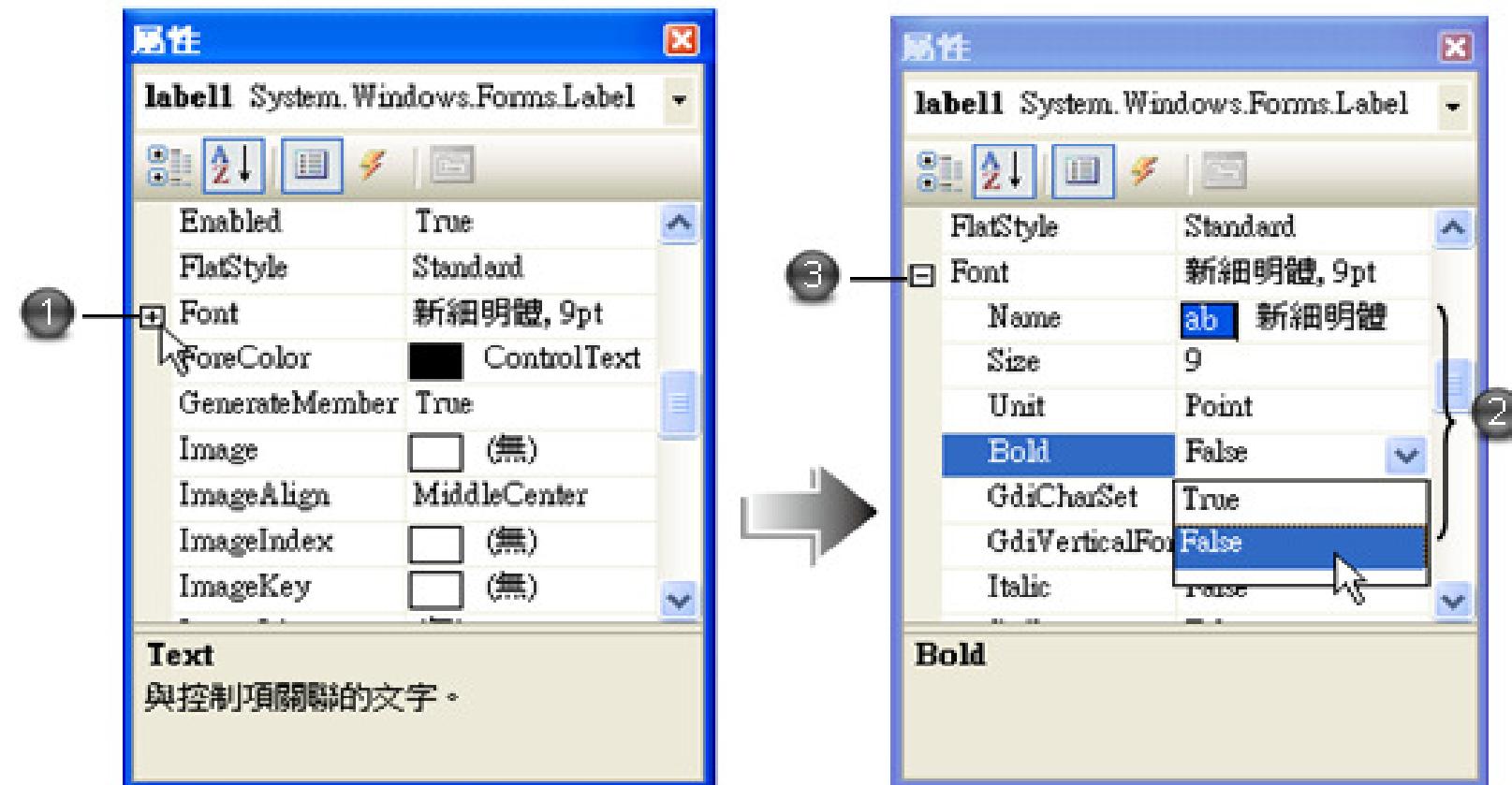
3. Font 屬性

- **Font** 屬性可設定表單的字型，和 **ForeColor** 屬性一樣會影響設定後新建立控制項的字型。
- 當在 **Font** 屬性上按一下，可看到 **Font** 的預設值是新細明體，大小9 pt。





另種方式





■ 程式設定

表單的字型為：“標楷體”、大小為”24”、
樣式為”粗體”，寫法：

```
this.Font = new Font("標楷體", 24, FontStyle.Bold);
```

_____ _____ _____
字體種類 字體大小 字體樣式



4. BackgroundImage/BackgroundImageLayout屬性

- **BackgroundImage** 用來設定表單的背景圖片。
- 預設是空白。
- 可配合 **BackgroundImageLayout** 屬性來配置背景圖。
- 預設值為**Tile**，表背景圖比表單小時，以貼磁磚方式佈滿整個表單。
- 若載入的背景圖比表單大，希望能以目前表單大小顯示整張背景圖，設為 **Stretch**。

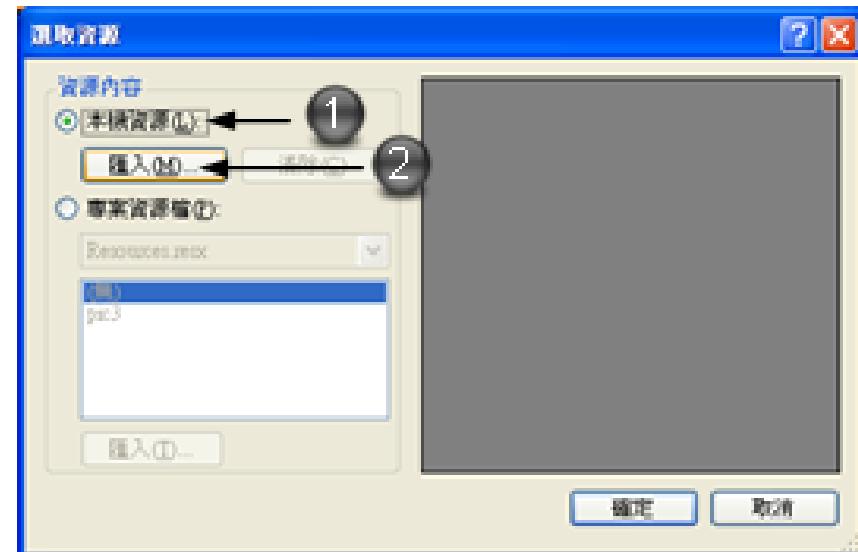
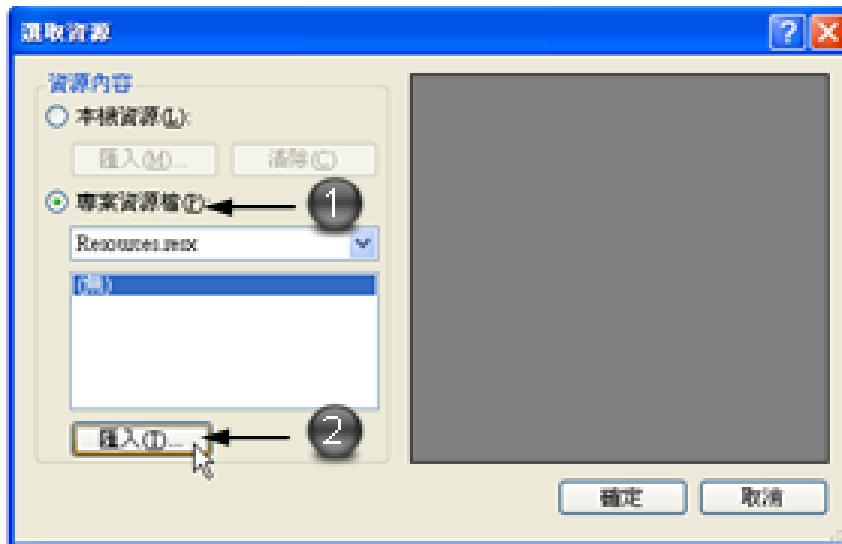
程式設定：

```
this.BackgroundImage = new Bitmap("C:\\CSharp\\pic2.bmp");
this.BackgroundImageLayout = ImageLayout.Tile;
```



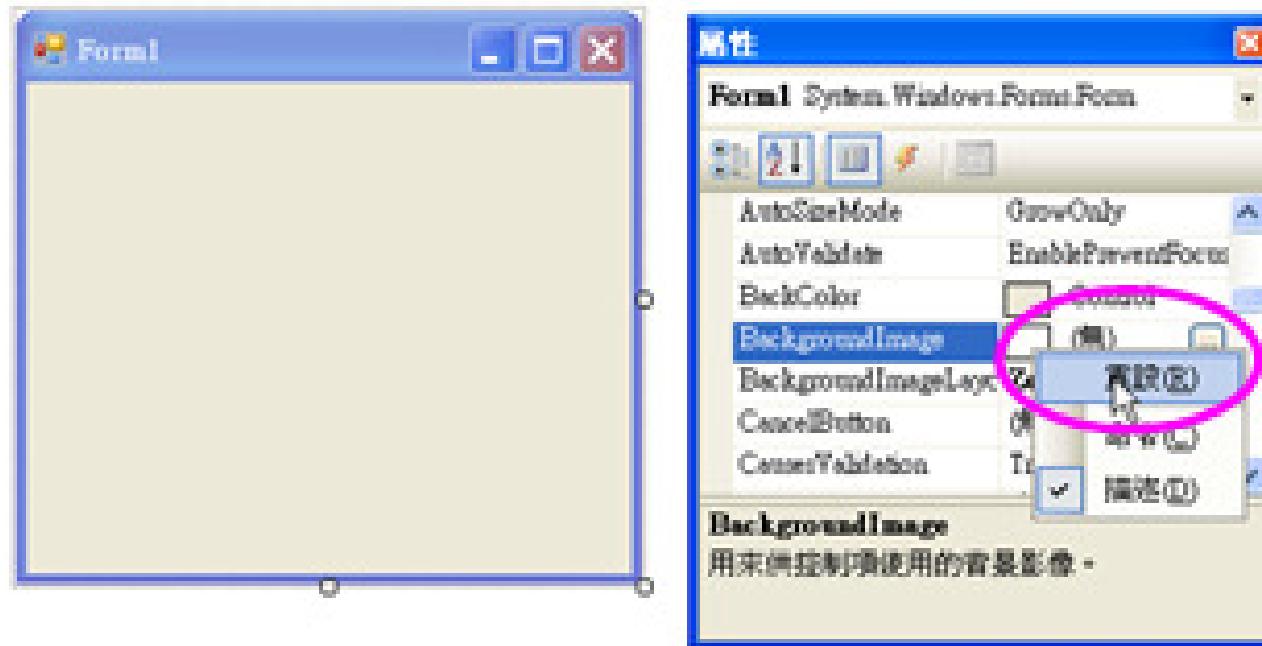
■ 設定圖檔方式：

1. 匯入專案資源檔
2. 指定本機資源





■ 清除圖檔





5. FormBorderStyle屬性 (預設值：Sizable)

- 用來設定表單視窗的邊框樣式，其屬性值有：
 - ① **None**：無邊框、大小固定、無標題欄。
 - ② **FixedSingle**：單線邊框、大小固定、有標題欄。
 - ③ **Fixed3D**：立體邊框、大小固定、有標題欄。
 - ④ **FixedDialog**：單線邊框、無法調整大小、標題欄只有關閉鈕。
 - ⑤ **Sizable**：立體邊框、可調整大小、有標題欄(預設值)。
 - ⑥ **FixedToolWindow**：單線邊框大小固定、標題欄只有結束鈕。
 - ⑦ **SizableToolWindow**：單線邊框、可調大小、標題欄只有結束鈕。



6. Enabled屬性

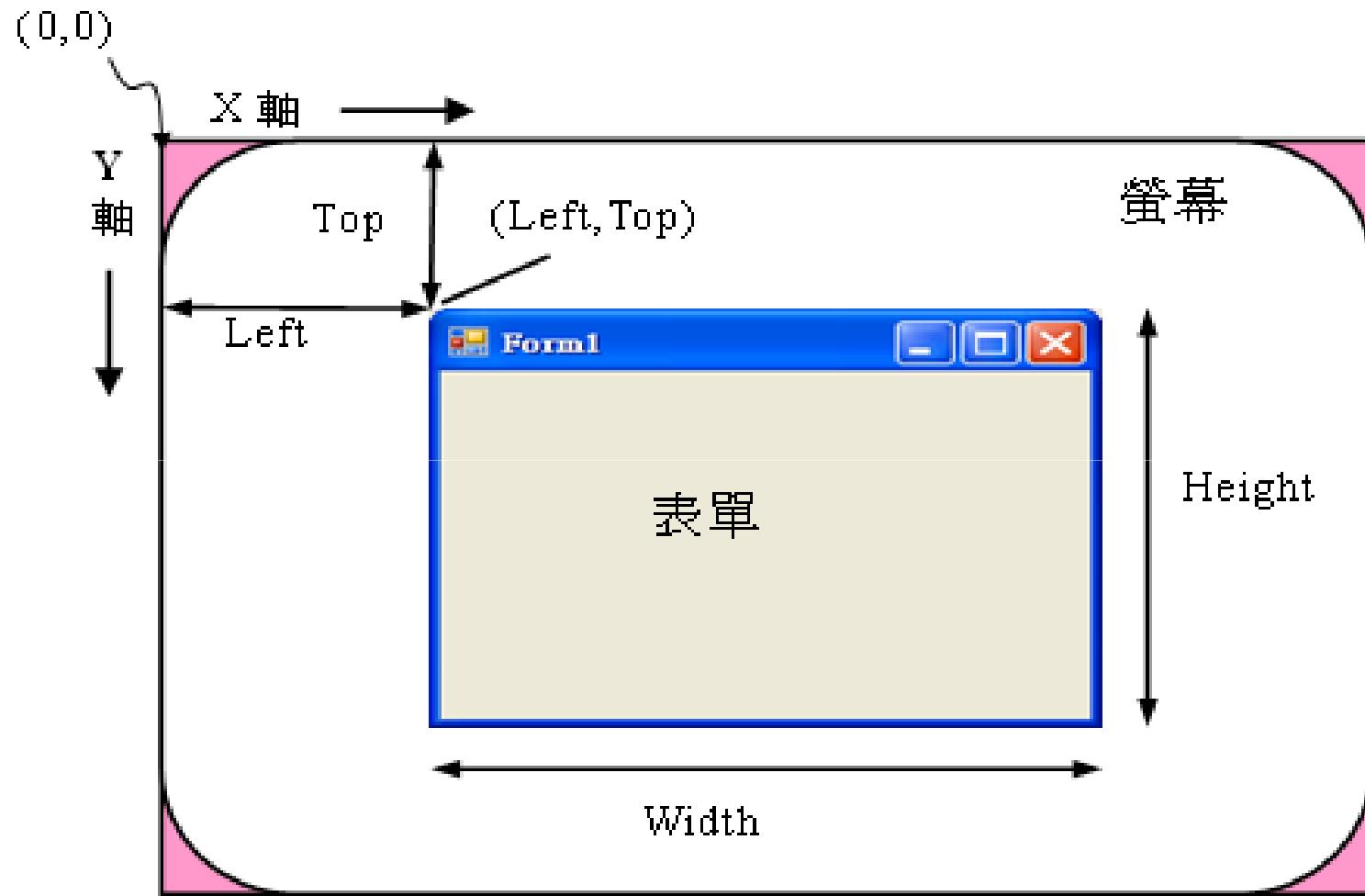
設定表單中的控制項是否有作用。

True：有作用；**False**：無作用。

7. StartPosition 屬性(預設值：**WindowDefaultLocation**)

設定表單開啟時顯示的位置，其值有：

- ① **Manual**（手動）。
- ② **CenterScreen**：置於螢幕中央。
- ③ **WindowDefaultLocation**；預設位置。
- ④ **WindowDefaultBounds**：系統預設位置和大小。
- ⑤ **CenterParent**：父視窗中央。





8. Location、Top、Left屬性

① StartPosition屬性

設為**Manual**（手動）時，才能透過 **Location** 屬性變更表單顯示的位置。

② Location屬性

有兩個子屬性 **X** 和 **Y**，分別代表 **X-座標** 和 **Y-座標**，設計階段可輸入(100,50)直接更改表單位置，也可按 **Location** 前面的展開鈕，直接更改 **X** 和 **Y** 值。



③ Top、Left 屬性

可在設計或程式執行時設定表單位置。

- **Top**屬性：是指表單上緣到螢幕上邊界的距離，單位為像素**pixel**，即表單左上角的Y座標。
- **Left**屬性：是指表單左邊到螢幕左邊界的距離，即表單左上角的X座標。
- 兩者合起來 (**Left,Top**)代表表單左上角座標。
- 程式中設定表單位置有兩種方式：

this.Location = new Point(x,y);

或

this.Left = x; this.Top = y;



9. Size、Width、Height 屬性

① **Size**屬性：設定表單的大小
包含寬度（**Width**）和高度（**Height**）子屬性。

② 程式設定表單大小：

this.Size = new Size(width,height);

或

this.Width = width ; this.Height = height;



10. WindowState屬性 (預設值 : Normal)

用來設定表單視窗開啟時的顯示狀態，其值有：

- ① **Normal**：一般 (預設值)
- ② **Minimized**：視窗最小化
- ③ **Maximized**：視窗最大化

程式設定：

```
this.WindowState = FormWindowState.Maximized ; // 最大化  
this.WindowState = FormWindowState.Minimized ; // 最小化  
this.WindowState = FormWindowState.Normal;      // 正常
```



11. ControlBox、MaximizeBox、MinimizeBox屬性

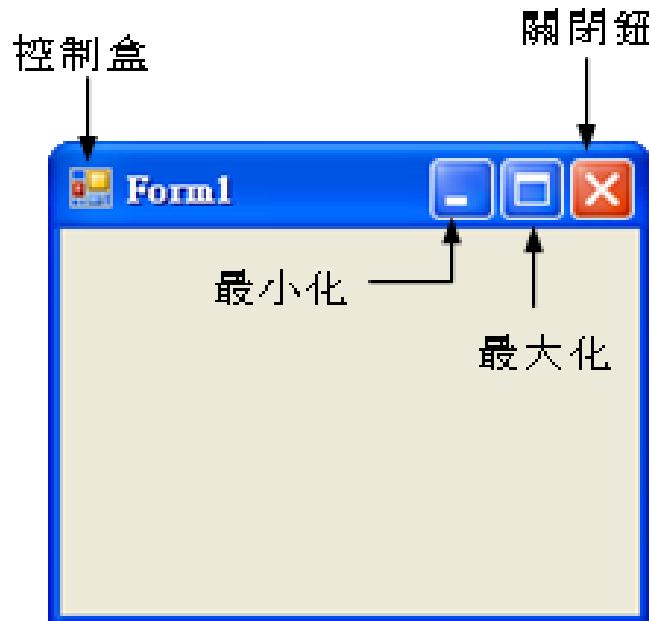
控制盒最小化最大化關閉鈕

- ① 分別設定表單的標題欄是否顯示：
- 控制盒（**ControlBox**屬性）
 - 最大化鈕（**MaximizeBox**屬性）
 - 最小化鈕（**MinimizeBox**屬性）。

設為 **True** 表顯示，
設為 **False** 表不顯示。

- ② 程式設定表單標題欄左上角
不顯示控制盒：

this.ControlBox =false;

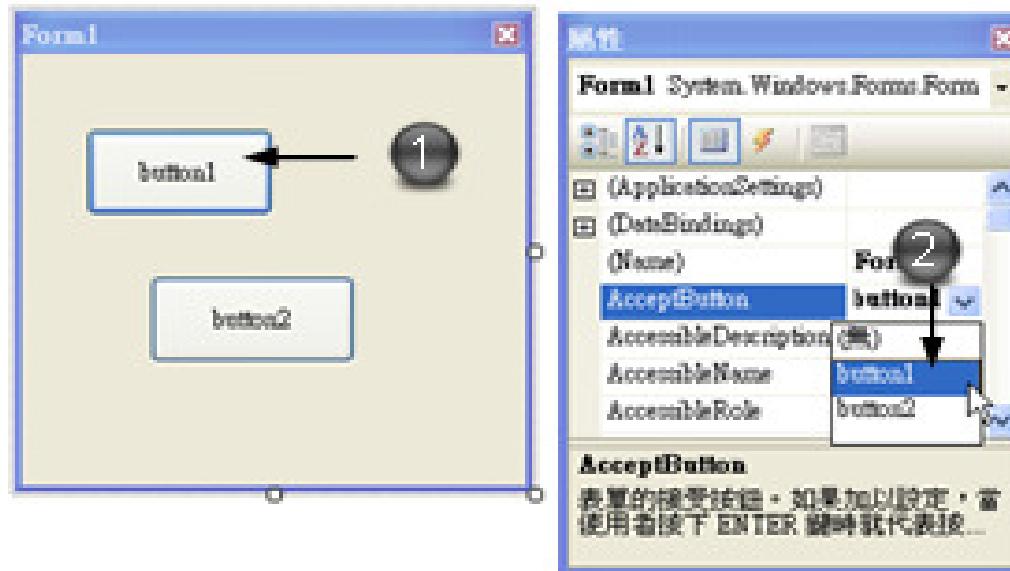




12. AcceptButton 屬性

當表單中有按鈕控制項時，可設定按 鍵相當於按下哪個按鈕控制項。

- 表單有 button1 和 button2 按鈕控制項，將 AcceptButton 屬性設成 button1，執行時按 鍵相當於按 button1 按鈕。
- 程式設定：**this.AcceptButton = button1;**

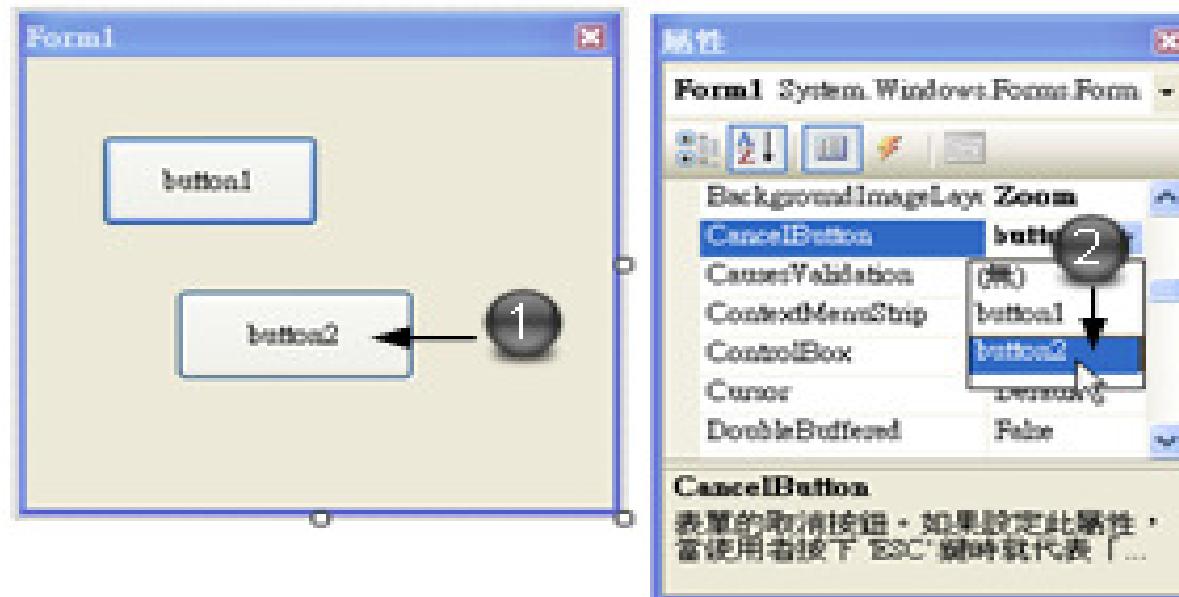




13. CancelButton屬性

當表單中有按鈕控制項時，設定按 **Esc** 鍵相當於按下哪個按鈕控制項。譬如右圖表單欲將 **CancelButton** 屬性設成 **button2** 時，執行時按 **Esc** 鍵相當於按 **button2** 按鈕。

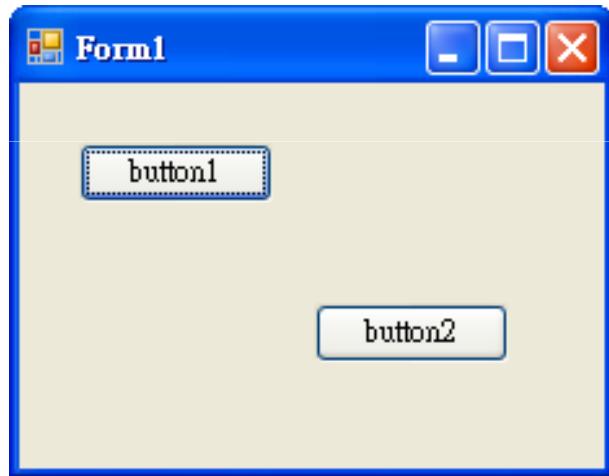
程式設定：**this.CancelButton = button2;**



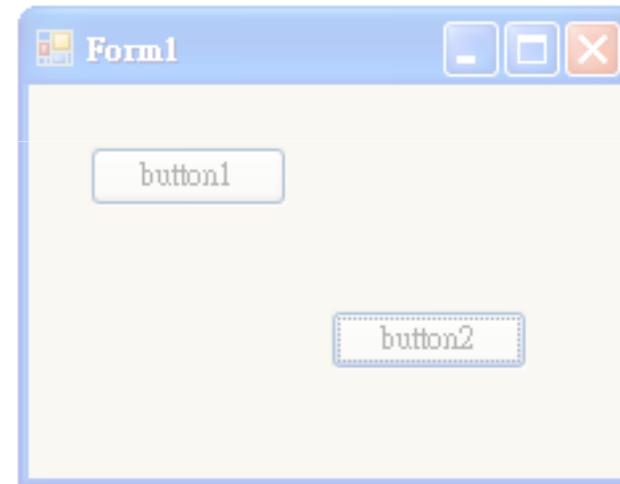


14. Opacity屬性（預設值：100%）

設定表單的透明度，其值由
0%（完全透明）~ 100%（完全不透明）。



透明度=100%



透明度=30%



15. ShowInTaskbar屬性（預設值：True）

- 用來設定表單是否顯示在工作列中
- **True** 表當表單最小化時會置於工作列的上面
- **False** 表最小化時不出現在工作列上面隱藏起來
- 可按 **Alt** + **Tab** 鍵切換重新顯示出來。



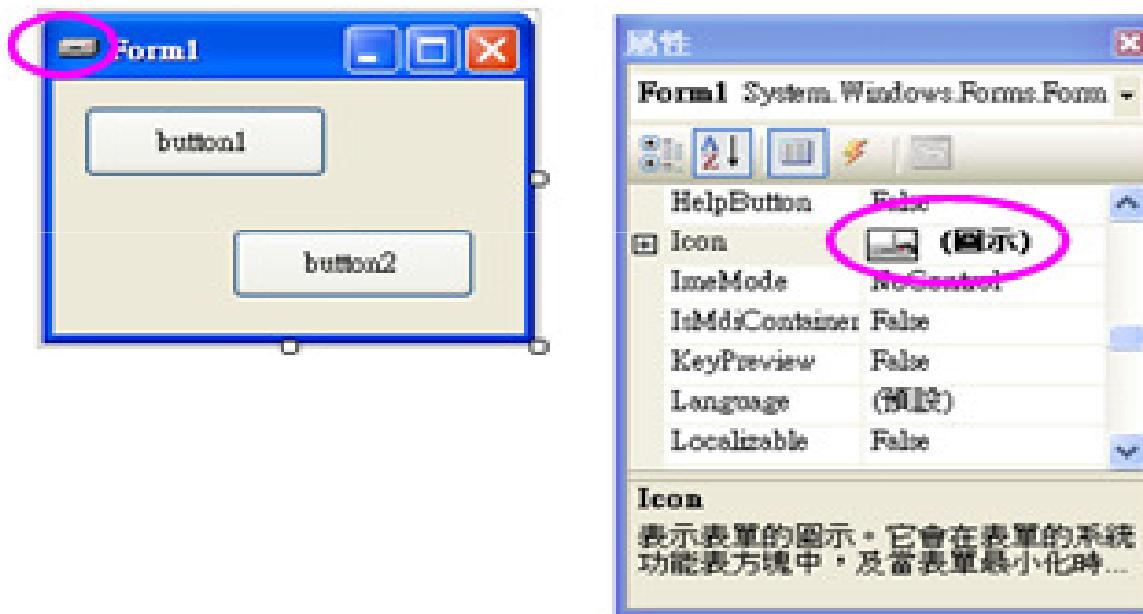
16. TopMost屬性（預設值：**False**）

- 用來設定表單是否永遠出現在所有視窗最上層。
- 若設為 **False** 表不必置於所有視窗的最上層。
- 若設為 **True** 表永遠保持置於所有視窗的最上層。



17. Icon屬性

用來更改標題欄左邊的小圖示。





8.2 Form 常用的事件

- 事件(Event)是可用程式碼來回應或處理的動作。
- 在 Windows 下，表單啟動、按下或放開滑鼠、拖曳滑鼠、按下或放開鍵盤的按鍵、打開或關閉視窗等動作的處理狀況都屬於「事件」。
- 將為回應事件時所要執行的程式碼稱為事件處理(Event-handle) 函式。
- VC# 每個表單和控制項都提供一組預先定義好的事件處理函式，事件處理函式內預設空的程式碼，需要時再依需求撰寫相關程式碼。
- 程式執行中，若觸發到該事件，會將此事件處理函式內的程式碼執行一次。



- VC# 2008 中的事件處理函式是 void 型別。但系統本身有自己呼叫事件方式，和一般 void 方法呼叫方式不相同。
- VC# 宣告事件處理函式是使用 += 運算子及 new EventHandler 來指定控制項的事件觸發時要執行哪個函式。
- 當 button1 的 Click 事件被觸發時會執行 button1_Click 事件處理函式：

```
this.button1.Click += new System.EventHandler(this.button1_Click);
```



- 在 button1 按鈕控制項快按滑鼠左鍵兩下，下面敘述自動增加到 Form1.Designer.cs 檔，不用理會它：
`this.button1.Click += new System.EventHandler(this.button1_Click);`
- 接著進入 Form1.cs 檔中的 button1_Click 事件處理函式，此時可在 button1_Click 事件處理函式內撰寫 button1 按鈕 Click 事件所要處理的程式碼：

```
private void button1_Click(object sender, EventArgs e)
```

```
{
```

```
.....
```

```
}
```



■ Load事件

- 表單第一次載入時觸動此事件
- 表單載入後此事件一直到程式結束都不會再執行。
- 可用來設定變數初值或更改控制項屬性值。

■ Activated事件

- 當表單第一次啟動時，此事件緊接在 Load 事件後被觸動執行的事件。
- 當程式執行時只要表單被選取變成作用表單時，也會觸動此事件，不像 Load 事件只在啟動時執行一次。



■ Click事件

當使用者用滑鼠在表單沒有控制項的地方按一下滑鼠左鍵，就會觸動表單的**Click**事件。

■ DoubleClick事件

當使用者用滑鼠在表單空白處快按兩下，就會觸動**DoubleClick**事件。但要特別注意的是，觸動**DoubleClick**事件的同時也會觸動**Click**事件，而且**Click**事件會先觸動。

■ KeyPress事件

當使用者按鍵盤時，就會觸動**KeyPress**事件。



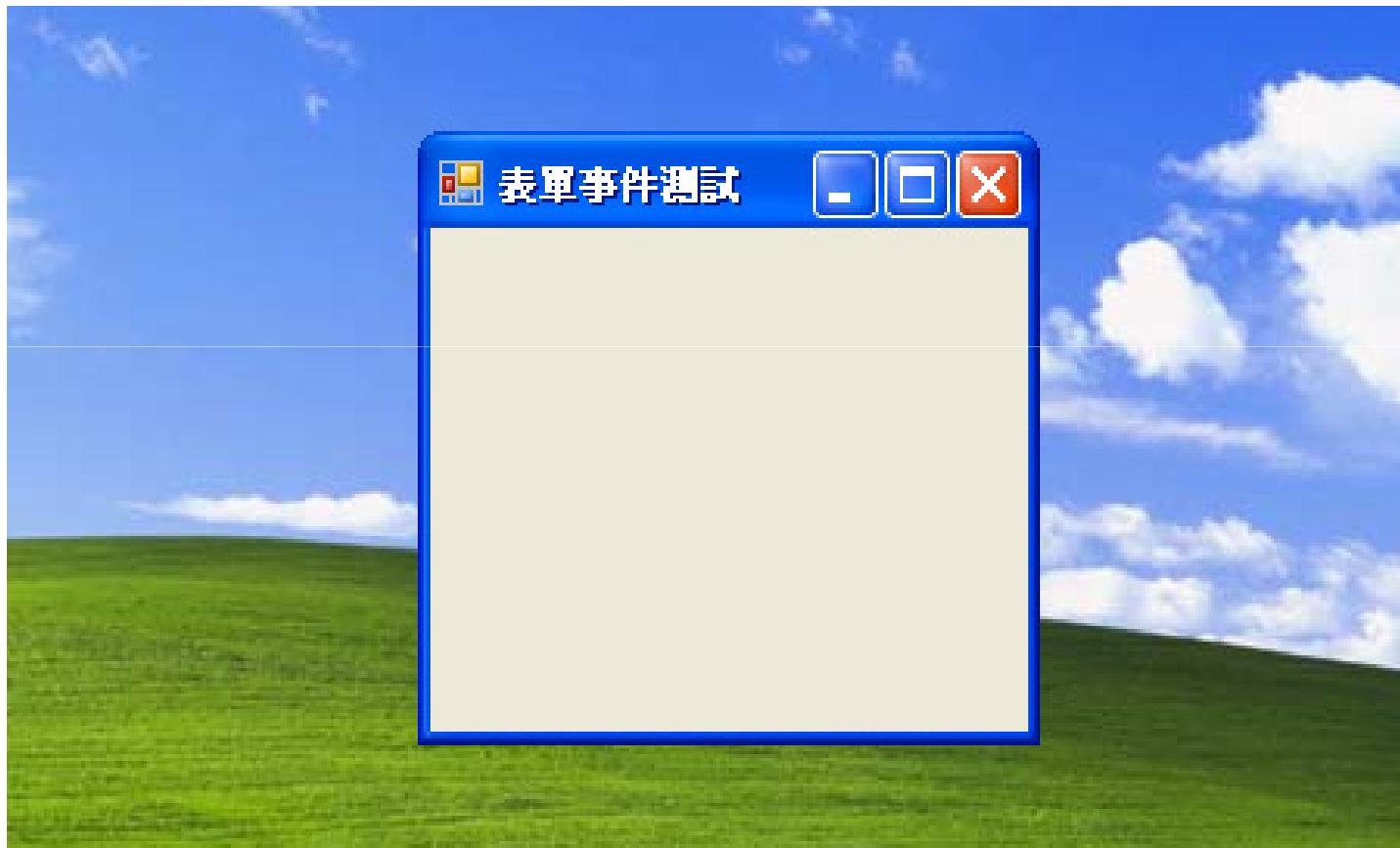
範例演練

■ 試寫一個測試本節介紹的表單五個常用事件。

- ① 當表單載入時，將表單置於螢幕的正中央，
並在 標題欄顯示”表單事件測試”。
- ② 在表單上按一下，表單左上角座標固定不動
表單寬度增長**10 Pixels**，高度減**10 Pixels**。
- ③ 在表單上快按兩下使整個表單往右移 **20 Pixels**。
- ④ 在桌面按一下，表單變成無作用，再按表單一下，
表單又變成作用表單，重新置於螢幕正中央且表單
大小恢復成開始執行時的大小。
- ⑤ 當在鍵盤上按任一鍵結束程式執行。



執行情形





```
// FileName : form1.sln
01 using System;
02 using System.Collections.Generic;
03 using System.ComponentModel;
04 using System.Data;
05 using System.Drawing;此部份程式區段為引用相關的命名空間是編寫程式時自動產生的，為節省篇幅，各章節範例程式碼中除非有必要，此部份將不列出。
06 using System.Text;
07 using System.Windows.Forms;
08
09 namespace form1
10 {
11     public partial class Form1 : Form
12     {
13         public Form1()
14         {
15             InitializeComponent();
16         }
17     }
18 }
```



```
17  
18     int form_left, form_top, form_width, form_height;  
19     private void Form1_Load(object sender, EventArgs e)  
20     {  
21         form_left = this.Left;  
22         form_top = this.Top;  
23         form_width = this.Width;  
24         form_height = this.Height;  
25         this.Text = "表單事件測試";  
26     }  
27     private void Form1_Click(object sender, EventArgs e)  
28     {  
29         this.Width += 10;  
30         this.Height -= 11;  
31     }
```



```
32  private void Form1_DoubleClick(object sender, EventArgs e)
33  {
34      this.Left += 20;
35  }
36  private void Form1_Activated(object sender, EventArgs e)
37  {
38      this.Location = new Point(form_left, form_top);
39      this.Size = new Size(form_width, form_height);
40  }
41  private void Form1_KeyPress(object sender, KeyPressEventArgs e)
42  {
43      Application.Exit();
44  }
45 }
46 }
```



8.3 Label和 LinkLabel標籤控制項

8.3.1 標籤控制項

- 標籤控制項 **A Label** 是非常重要的輸出介面
- 可用來在表單上面顯示文字、圖片以及小圖示。



屬性	說明
AutoSize	設定控制項的寬度是否隨文字長度自動縮放。若設為True，控制項的寬度會隨文字長度自動調整，此時控制項左上角出現小白框如無法調整；若設為False則控制項允許手動調整，控制項四周出現八個小白框如所示。預設值為True。
BackColor	用來設定標籤控制項的背景色。假設表單有背景圖，若將label1標籤控制項的背景色設為透明，可避免標籤控制項的背景色遮住破壞表單背景圖。其程式寫法如下： <code>label1.BackColor=Color.Transparent;</code>
BorderStyle	設定標籤的邊框樣式。其值如下： ①None：不加邊框(預設值) ②FixedSingle：加邊框 ③Fixed3D：立體凹陷



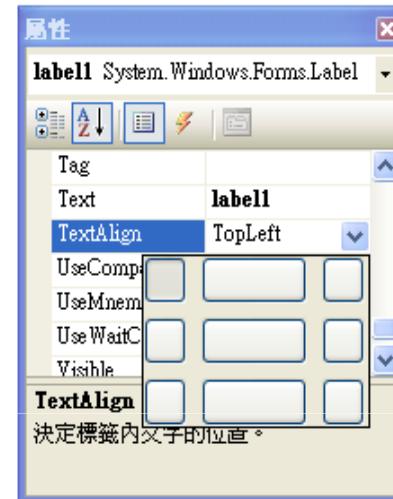
Enabled	設定該控制項是否有作用。若設為True，該控制項有作用即可點選；若為False，該控制項文字會以淺灰色顯示。 預設值為True。
Font	設定控制項上面顯示文字的字型，上一章已介紹過設計階段的設定方法，至於程式中的寫法如下： label1.Font = new Font(字型, 大小, 樣式); 例如： label1.Font = new Font("標楷體", 16, FontStyle.Bold);
ForeColor	設定控制項上面顯示文字的顏色。
Text	用來設定控制項上面顯示的訊息，只能顯示資料，無法輸入資料。表單的Text屬性則設定標題欄名稱。其程式寫法如下： label1.Text="Hello!";



TextAlign

設定文字在標籤控制項顯示的位置，其值有：

- ① **TopLeft** 左上(預設值)
- ② **TopCenter** (中上)
- ③ **TopRight** (右上)
- ④ **MiddleLef** (左中)
- ⑤ **MiddleCenter** (中央)
- ⑥ **MiddleRight** (右中)
- ⑦ **BottomLeft** (左下)
- ⑧ **BottomCenter**(中下)
- ⑨ **BottomRight** (右下)



UseMnemonic

若設為 **True**，則 **Text** 屬性值中「&」後的第一字為提示字元。例如：**Text** 屬性值為「&Help」，「H」就為提示字元，程式執行時會顯示為「**Help**」。

Visible

設定該控制項是否被隱藏。若設為 **True**，表示該控制項在表單看得到；若設為 **False**，將該控制項隱藏。要注意在設計階段雖設為隱藏，還是看得到該控制項，執行時才會自動隱藏。預設值為 **True**。



8.3.2 連結標籤控制項

A LinkLabel

■ 連結標籤控制項具備

- ① 標籤控制項的功能外。
- ② 還增加一些超連結的功能。



一. LinkLabel常用屬性

屬性	說明
LinkColor	設定控制項上面超連結文字的起始顏色，預設藍色
LinkVisited	設定控制項上面超連結文字超連結後顏色是否設為 VisitedLinkColor 屬性值，以和尚未點選過的超連結控制項區分，預設值為True表示會變色。
VisitedLinkColor	設定控制項上面超連結文字已經超連結過的顏色，預設值為紫色。



DisabledLinkColor	設定控制項超連結被停用時超連結文字的顏色，預設值為灰色。
LinkBehavior	設定超連結文字是否要加底線，說明如下： ① SystemDefault ：系統預設(預設值) ② AlwaysUnderline ：加底線 ③ HoverUnderline ：游標在文字上才加底線 ④ NeverUnderline ：不加底線
LinkArea	設定控制項上面文字允許超連結的範圍。屬性值為 (Start,Length)，Start為文字起始位置；Length為長度。 例如：Text = "SuperMan"，LinkArea屬性值為 (5,3)，則表示由第六(5+1)個字起取3個字，所以超連結的文字為 "Man"。



二. LinkLabel 常用事件

當使用者按到有超連結文字時，觸動 **LinkClicked** 事件。

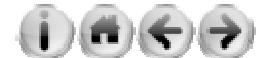
1. 超連到網站

語法：**System.Diagnostics.Process.Start("網址URL");**

[簡例]

超連結到 ” discovery” 網站

System.Diagnostics.Process.Start("http://www.discover.com");



■ 2. 超連結到指定的資料檔或執行檔

語法：**System.Diagnostics.Process.Start("路徑\\檔名");**

[簡例] 超連結到 C槽test資料夾的readme.doc檔案

```
System.Diagnostics.Process.Start("c:\\test\\readme.doc");
```

■ 3. 超連結到電子信箱

語法：**System.Diagnostics.Process.Start("mailto:電子信箱");**

[簡例] 超連結到電子信箱

```
System.Diagnostics.Process.Start("mailto:tom@hinet.net");
```



範例演練

超連結練習。表單載入時出現下圖進入雅虎網站相關提示訊息。當按「雅虎網站簡介」超連結文字時，開啟專案 bin/Debug 資料夾下的 Yahoo.txt 檔。若點選「雅虎網站」中「雅虎」超連結文字時，會連結到雅虎 tw.yahoo.com。

<p>1. 程式開始執行</p> 	<p>2. 連結「雅虎網站介紹」(yahoo.txt)</p> 
--	--



上機

■ Step1 設計輸出入介面





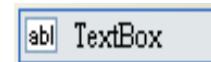
Step3 撰寫程式碼

```
// FileName : link1.sln

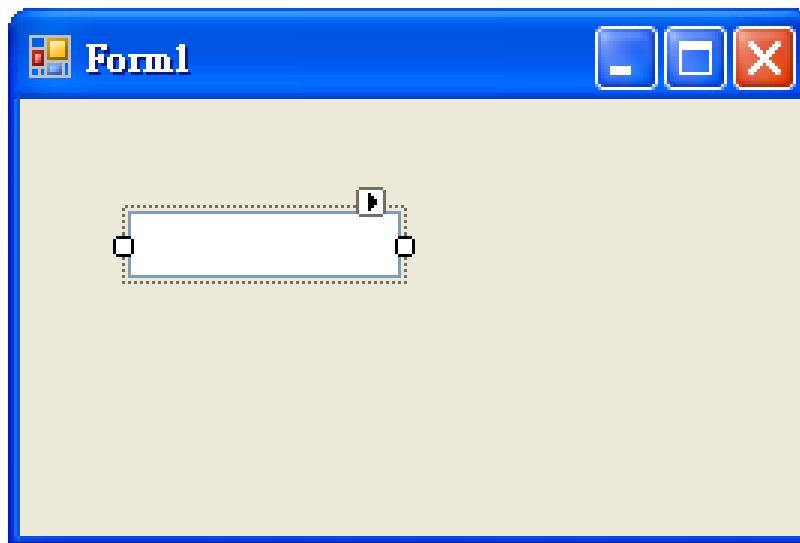
01 private void Form1_Load(object sender, EventArgs e)
02 {
03     lblTitle.Text = "雅虎網站是常用的入口網站\n請按下列超連結至雅虎網站！";
04     lblTitle.Font = new Font("標楷體", 14, FontStyle.Bold);
05 }
06 private void InkReadMe_LinkClicked(object sender, LinkLabelLink
    ClickedEventArgs e)
07 {
08     System.Diagnostics.Process.Start("yahoo.txt");
09 }
10 Private void InkYahoo_LinkClicked(object sender, LinkLabelLink
    ClickedEventArgs e)
11 {
12     System.Diagnostics.Process.Start("http://tw.yahoo.com");
13 }
```



8.4 TextBox文字方塊控制項



- 文字方塊控制項是用來輸入資料
- 也可用來顯示資料
- 建立時出現小白框用來調整控制項的左右寬度。

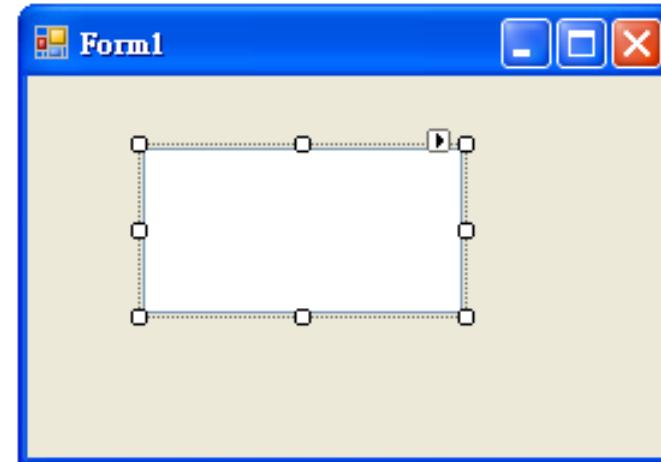
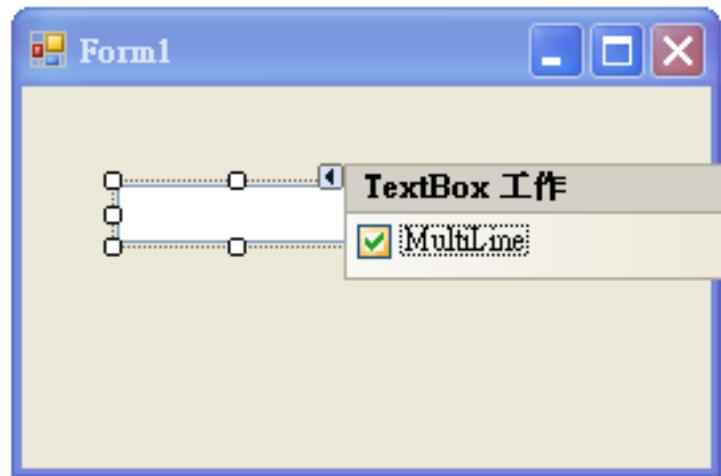




■ 智慧標籤 (Smart Tag)

文字方塊的右上角，顯示文字方塊的常用屬性供你直接選取，而不必再到屬性視窗中點選。

- 勾選 MultiLine 表將 MultiLine 屬性設為 True，允許資料由單行改多行顯示





屬性	說明				
Text	用來接受使用者輸入的資料。輸入的資料視為字串資料型別，若需要轉成數值資料型別，可透過 <code>int.Parse()</code> 方法來轉換，如下寫法： <code>int score = int.Parse(textBox1.Text);</code>				
TextAlign	設定文字在控制項內顯示位置，其值有： ① Left：靠左(預設值) ② Right：靠右 ③ Center：置中				
Lines	Lines 屬性類似 Text 屬性，只是 Lines 的資料型別為字串陣列，用於多行輸入時。例如要取得第 2 行的字串，其寫法如下： <code>//第一行陣列索引值為 0 string input_str = textBox1.Lines[1];</code> 例：右下圖 <code>textBox1</code> 文字方塊控制項內輸入兩行資料： <table border="1"><tr><td><code>label1.Text=textBox1.Lines[0]; label1 結果取得："VC#"</code></td><td></td></tr><tr><td><code>label2.Text=textBox1.Lines[1]; label2 結果取得："最好的選擇"</code></td><td></td></tr></table>	<code>label1.Text=textBox1.Lines[0]; label1 結果取得："VC#"</code>		<code>label2.Text=textBox1.Lines[1]; label2 結果取得："最好的選擇"</code>	
<code>label1.Text=textBox1.Lines[0]; label1 結果取得："VC#"</code>					
<code>label2.Text=textBox1.Lines[1]; label2 結果取得："最好的選擇"</code>					



ScrollBars	當多行輸入時，設定是否出現捲軸。其值有： ① None：無(預設值) ② Vertical：顯示垂直捲軸 ③ Horizontal：顯示水平捲軸 ④ Both：顯示垂直和水平捲軸
MaxLength	設定允許接受輸入文字的最大長度(0~32,767)，屬性值若為 0，代表長度不設限。預設值為 32,767。
CharacterCasing	設定輸入英文字母是否轉換成大小寫，屬性值有： ① Normal：不轉換(預設值) ② Upper：轉成大寫 ③ Lower：轉成小寫 ④ Normal：為預設值
AutoSize	設定文字方塊控制項大小是否隨字型大小自動縮放，預設值為 True。
MultiLine	設定是否允許多行輸入。若設為 False 表示限單行輸入；True 為多行輸入。預設值為 False。



MultiLine	設定是否允許多行輸入。若設為 False 表示限單行輸入；True 為多行輸入。預設值為 False。
WordWrap	設定輸入文字長度超過控制項寬度時是否可以自動換行，本屬性只有 MultiLine 屬性值為 True 時有效。預設值為 True。
PasswordChar	設定輸入資料時，以替代字元取代輸入字元，常用於輸入密碼時。
ReadOnly	設定控制項的文字是否為唯讀，預設值為 False；若設為 False 表示該控制項可以輸入和顯示資料。若設為 True，則效果如標籤控制項。
AcceptsReturn、 AcceptsTab	分別設定當多行輸入時，是否可以接受 和 鍵。
Length	取得控制項內文字的長度，本屬性在執行階段才能使用。如下寫法： <code>int str_length = txtInput.Text.Length;</code>



二. TextBox常用事件

- **TextChanged** 是文字方塊控制項的預設事件。
- 當使用者在文字方塊控制項內輸入文字而改變 **Text**屬性值時，會觸動**TextChanged**事件。
- 在**TextChanged**事件中我們可以檢查輸入字元是否正確或將輸入的資料同步反應在其它相關的控制項。



範例演練

試寫一個美金和台幣兌換程式。當在匯率和兌換美金兩個輸入框中資料有異動時，所兌換的新台幣金額亦會跟著改變。

執行情形

1. 開始

匯率換算

匯率 : 1 美金 = 台幣

兌換金額 : 美金 台幣

2. 輸入匯率

匯率換算

匯率 : 1 美金 = 台幣

兌換金額 : 美金 台幣



■ Step1 設計輸出入介面

匯率換算

匯率 : 1美金 = 新台幣
Name=txtRate

兌換金額 : 美金 新台幣
Name=txtUS Name=lblNT
Enabled=False



Step2 撰寫程式碼

```
// FileName : textbox1.sln
01  private void Form1_Load(object sender, EventArgs e)
02  {
03      this.Text = "美金兌換新台幣";
04      txtRate.Text = "0";
05      txtUS.Text = "0";
06      txtRate.TextChanged += new EventHandler(txt_TextChanged);
07      txtUS.TextChanged += new EventHandler(txt_TextChanged);
08  }
09
10     private void txt_TextChanged(object sender, EventArgs e)
11  {
12      try
13      {
14          double n = double.Parse(txtRate.Text) * double.Parse(txtUS.Text);
15          lbINT.Text = n.ToString();
16      }
17      catch (Exception ex)
18      {
19  }
```



8.5 Button按鈕控制項

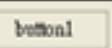


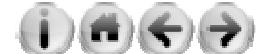
- 按鈕控制項是重要的輸入介面
- 大部分視窗都會用按鈕控制項作為功能的選用。
- 按鈕控制項除可顯示文字外，也可顯示小圖示。



屬性	說明
Text	來設定按鈕控制項上顯示的文字。
Cursor	設定當滑鼠在按鈕控制項上面時，滑鼠游標所顯示的圖示。
Image	設定按鈕控制項上面顯示的圖片。預設值是不顯示圖片，若設定圖片後，可以再利用 ImageAlign 屬性來設定圖片顯示的位置。
ImageAlign	ImageAlign 屬性可設定圖片的顯示位置，其值有： ① TopLeft：左上(預設) ② TopCenter：中上 ③ TopRight：右上 ④ MiddleLeft：左中 ⑤ MiddleCenter：中央 ⑥ MiddleRight：右中 ⑦ BottomLeft：左下 ⑧ BottomCenter：中下 ⑨ BottomRight：右下



FlatStyle	當滑鼠在按鈕控制項上方按鈕的顯示方式，其值有： ① Standard：預設以立體顯示  button1 ② Flat：平面  button ③ PopUp：原為平面滑鼠按下時以  立體顯示 ④ System：系統設定
Enabled	設定按鈕控制項是否有作用？預設值為 True，若為 True 表按鈕有效；若為 False 按鈕無作用以灰色顯示。
DialogResult	設定按下按鈕傳回的值，預設值：None。其值有： ① None (不回傳) ② Ok (確定) ③ Cancel (取消) ④ Abort (放棄) ⑤ Retry (重試) ⑥ Yes (是) ⑦ No (否)。



範例演練

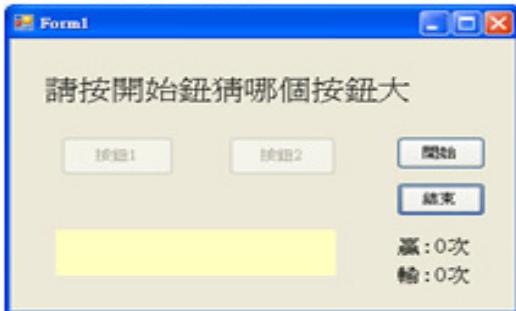
設計一個比大小的遊戲。

- ① 程式開始時，只有 **開始**、**結束** 鈕有效。
- ② 當按 **開始** 鈕，提示訊息改為「請選擇按鈕1或2」，**開始** 鈕無效 **按鈕1**、**按鈕2**、**結束** 鈕有效，產生兩個1~99間不重複的亂數。
- ③ 當按 **按鈕1**、**按鈕2** 鈕之一，將產生的亂數顯示到按鈕上面，若所按的鈕值較大，顯示「你猜對了！」，否則顯示「你猜錯了！」
- ④ 接著將 **按鈕1**、**按鈕2** 設為無效，**開始**、**結束** 鈕設為有效，提示訊息改為「請按開始鈕猜哪個按鈕大」。
- ⑤ 各畫面都顯示累積輸贏的次數。
- ⑥ 按 **結束** 鈕結束程式。



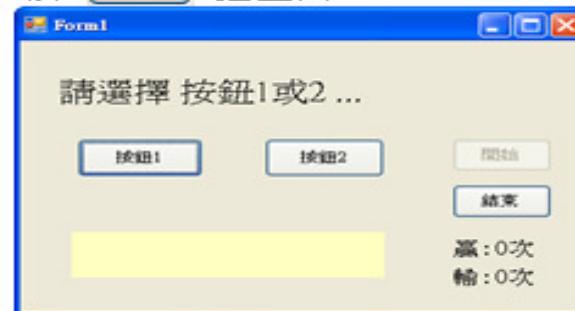
執行情形

① 程式開始畫面



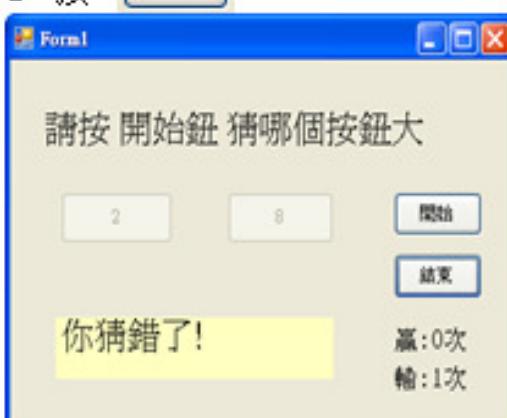
開始鈕有效，按鈕 1 和 2 失效

② 按 開始 鈕畫面



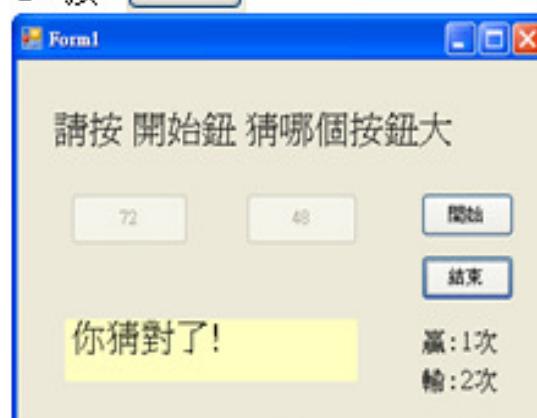
開始鈕失效，按鈕 1 和 2 有效

③ 按 按鈕1



你猜錯了!

④ 按 按鈕2

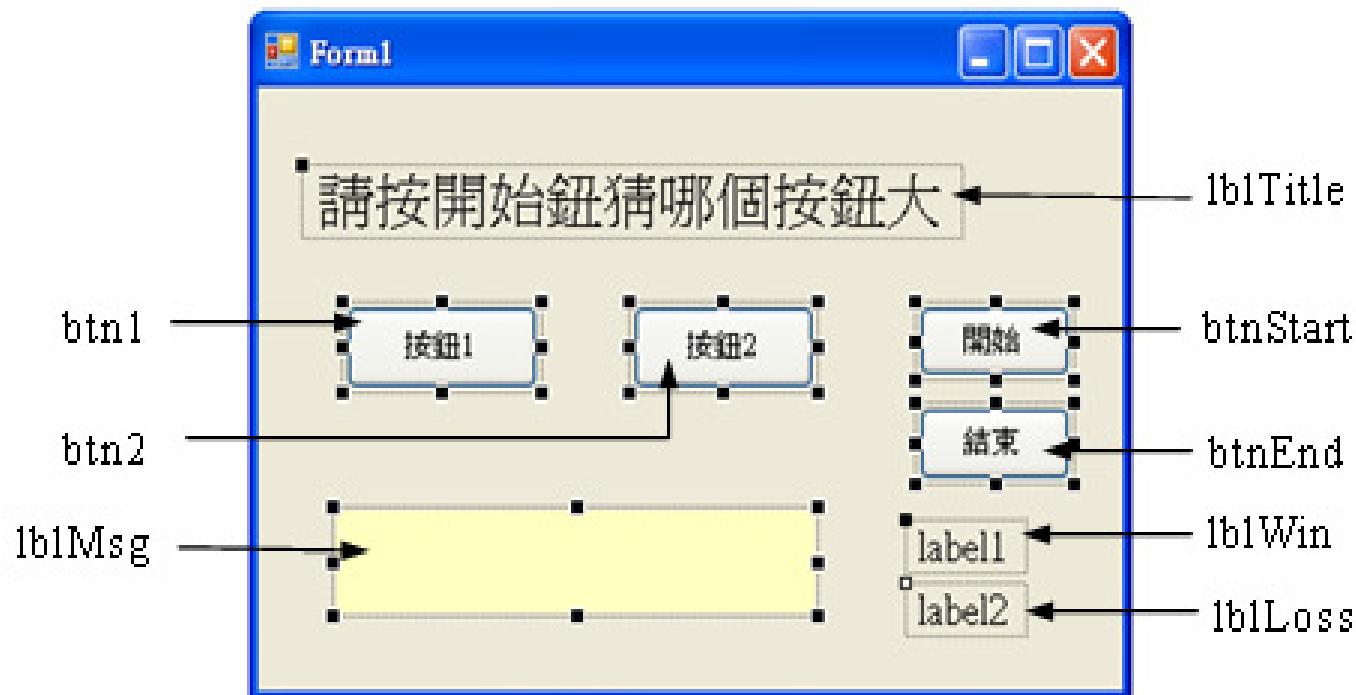


你猜對了!



上機

■ Step1 設計輸出入介面





■ Step3 撰寫程式碼

```
// FileName :guess.sln
01 int no1, no2, win, loss;
02
03 private void Form1_Load(object sender, EventArgs e)
04 {
05     win = loss = 0;
06     lblWin.Text = "贏 :" + win.ToString() + " 次";
07     lblLoss.Text = "輸 :" + loss.ToString() + " 次";
08     btn1.Enabled = false;
09     btn2.Enabled = false;
10 }
```



```
12 private void btnStart_Click(object sender, EventArgs e)
13 {
14     lblTitle.Text = "請選擇 按鈕1或2 ...";
15     lblMsg.Text = "";
16     btn1.Enabled = true;
17     btn2.Enabled = true;
18     btnStart.Enabled = false;
19     btn1.Text = "按鈕1";
20     btn2.Text = "按鈕2";
21     Random ranobj = new Random();
22     no1 = ranobj.Next(1, 100);
23     do
24         no2 = ranobj.Next(1, 100);
25     while (no1 == no2);
26 }
```



```
28 private void btn1_Click(object sender, EventArgs e)
29 {
30     btn1.Text = Convert.ToString(no1);
31     btn2.Text = Convert.ToString(no2);
32     if (no1 > no2)
33     {
34         lblMsg.Text = "你猜對了!";
35         win++;
36     }
37     else
38     {
39         lblMsg.Text = "你猜錯了!";
40         loss++;
41     }
```



```
42     lblWin.Text = "贏 :" + win.ToString() + " 次";
43     lblLoss.Text = "輸 :" + loss.ToString() + " 次";
44     lblTitle.Text = "請按 開始鈕 猜哪個按鈕大";
45     btnStart.Enabled = true;
46     btn1.Enabled = false;
47     btn2.Enabled = false;
48 }
49
50 private void btn2_Click(object sender, EventArgs e)
51 {
52     btn1.Text = Convert.ToString(no1);
53     btn2.Text = Convert.ToString(no2);
54     if (no2 > no1)
55     {
56         lblMsg.Text = "你猜對了!";
57         win++;
58     }
59     else
60     {
61         lblMsg.Text = "你猜錯了!";
62         loss++;
63     }
```



```
64     lblWin.Text = "贏 :" + win.ToString() + " 次";
65     lblLoss.Text = "輸 :" + loss.ToString() + " 次";
66     lblTitle.Text = "請按 開始鈕 猜哪個按鈕大";
67     btnStart.Enabled = true;
68     btn1.Enabled = false;
69     btn2.Enabled = false;
70 }
71
72 private void btnEnd_Click(object sender, EventArgs e)
73 {
74     Application.Exit();
75 }
```



8.6 MessageBox.Show方法

■ **MessageBox.Show()** 方法可產生允許顯示

- ① 提示訊息
- ② 標題欄名稱
- ③ 提示圖示
- ④ 相關按鈕

■ 語法：

DialogResult 傳回值 = MessageBox.Show(訊息, 標題, 按鈕常數, 圖示常數);



```
DialogResult result =  
    MessageBox.Show("確定結束程式？", "井字遊戲",  
    MessageBoxButtons.OKCancel,  
    MessageBoxIcon.Exclamation);
```



DialogResult常數	說明
OK	使用者按 鈕的傳回值
Cancel	使用者按 鈕的傳回值
Abort	使用者按 鈕的傳回值
Retry	使用者按 鈕的傳回值
Ignore	使用者按 鈕的傳回值
Yes	使用者按 鈕的傳回值
No	使用者按 鈕的傳回值
None	使用者未按鈕的傳回值



按鈕常數

用來設定視窗中顯示的按鈕。常數如下：

MessageBoxButtons 常數	顯示的按鈕
OK	
OkCancel	
AbortRetryIgnore	
YesNoCancel	
YesNo	
RetryCancel	



圖示常數

用來設定視窗中顯示的圖示：

MessageBoxIcon 常數	顯示的圖示
Asterisk	
Error	
Exclamation	
Question	
None	不顯示圖示



範例演練

試寫一個帳號及密碼檢查程式。程式一開始要求輸入

- ① 帳號(帳號為 " yahoo")
- ② 密碼 (密碼為 " 1688")

當帳號及密碼正確連結到奇摩網站 <http://www.kimo.com.tw>

帳號及密碼不正確可再重新輸入，連續錯誤三次結束程式。



執行情形

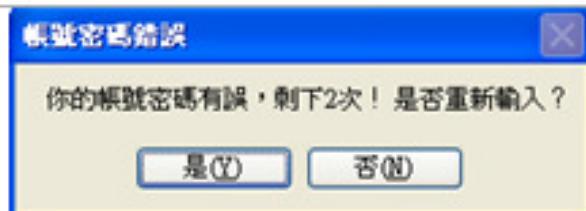
① 密碼輸入畫面：



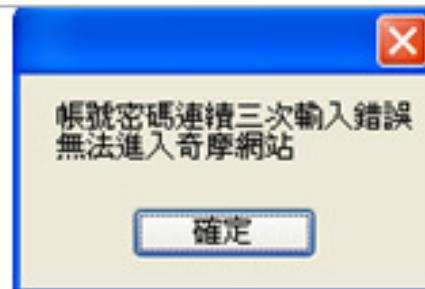
② 密碼輸入正確畫面：



③ 密碼錯誤未超過 3 次時

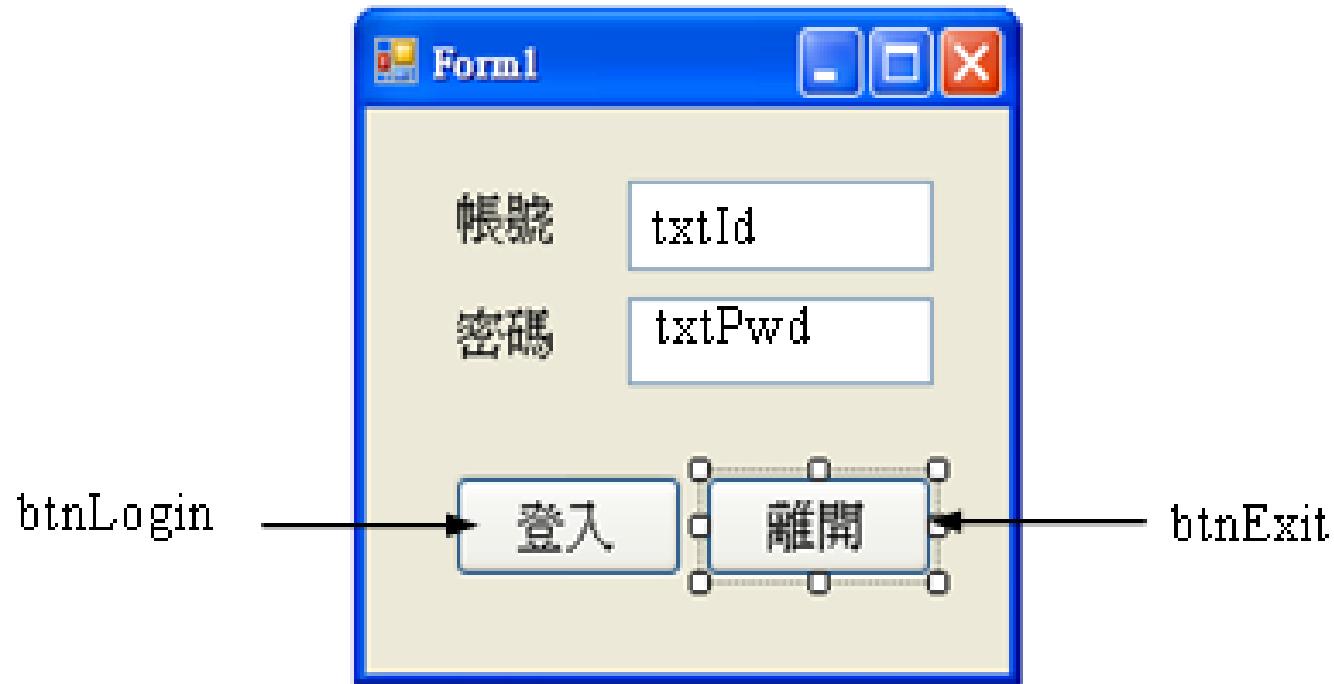


④ 密碼錯誤超過 3 次時





■ 設計輸出入介面





■ 撰寫程式碼

```
// FileName : msgbox1.sln

01 int num;
02
03 private void btnLogin_Click(object sender, EventArgs e)
04 {
05     DialogResult result;
06     num += 1;
07     if (txtId.Text == "yahoo" && txtPwd.Text == "1688")
08     {
09         MessageBox.Show("歡迎光臨, 奇摩網站");
10         System.Diagnostics.Process.Start("http://www.kimo.com.tw");
11     }
```



```
12     else
13     {
14         if (num == 3)
15         {
16             MessageBox.Show("帳號密碼連續三次輸入錯誤\n無法進入奇摩網站");
17             Application.Exit();
18         }
19     else
20     {
21         result = MessageBox.Show("你的帳號密碼有誤，剩下" + (3 - num) +
22         "次！是否重新輸入？", "帳號密碼錯誤", MessageBoxButtons.YesNo);
23         if (result == DialogResult.No) Application.Exit();
24     }
25 }
26
27 private void btnExit_Click(object sender, EventArgs e)
28 {
29     Application.Exit();
30 }
```