CHAPTER 3

基本輸出入介面設計

- ✧ 學習表單物件常見的屬性
- ✧ 學習表單物件常用的事件
- ✧ 學習 Label 標籤控制項的使用
- ✧ 學習 Button 按鈕控制項的使用
- ✧ 學習 TextBox 文字方塊控制項的使用
- ✧ 學習例外處理技巧
- ✧ 學習使用 Visual Basic 的 InputBox 函式來輸入資料
- ✧ 學習使用 MessageBox.Show()方法輸出提示訊息

3.1 表單物件常見的屬性

Windows 視窗作業系統之所以能夠迅速取代 DOS 系統，就是因為它具有高親和力的圖形化操作介面。Visual C# 亦藉由圖形化操作介面，讓程式設計者可以在表單設計階段，透過工具箱提供的工具，不用寫程式便能快速地建立輸出入介面。程式設計者只要專注在處理流程的程式碼及演算法上，縮短了程式開發的時間。

撰寫 Windows Form 視窗應用程式最基本的輸出入介面就是表單(Form)，表單就是一個視窗，它像一個容器(Container)可以安置透過工具箱的工具所建立的元件，所建立的元件用來做為使用者操作的輸出入介面。我們將工具箱的工具拖曳到表單所建立的元件稱為「控制項」或「物件」。

每個控制項都有其所屬的屬性和方法，每個屬性皆有其預設值，可依程式需求來加以修改，讓同類別的表單或控制項展現不同的外觀和功能。不同種類的控制項和表單可能擁有相同的屬性名稱，但也可能是該控制項所獨有。至於如何修改控制項的屬性值，最快的方式是在表單設計階段透過屬性視窗來設定；另一種方式則是在程式執行階段，依需求在適當時機使用程式碼來設定。

本節先以分類的方式來介紹表單物件常見的屬性，以及如何在表單設計階段透過屬性視窗來設定屬性的方法。若在後面的章節中，這些屬性在其它控制項中可能也被擁有，除非該屬性在該控制項中另有特別的用法，否則不再重複說明。

3.1.1 外觀類型的屬性

屬性名稱	說明
BackColor	設定表單工作區的背景顏色，預設值為 Control。 [例] 將表單的背景色設為黃色，程式碼： BackColor = Color.Yellow ; ↑ 屬性名稱　↑ 屬性值

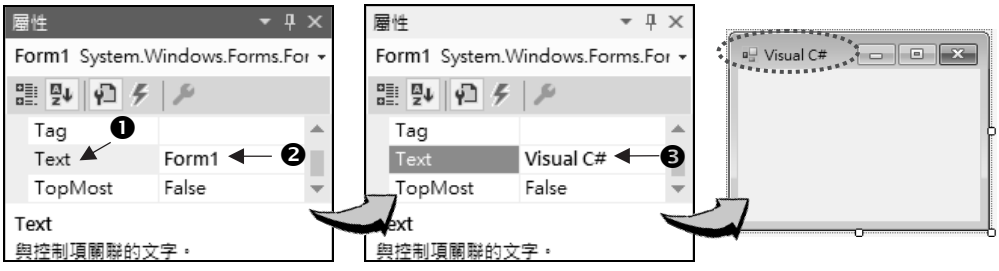
屬性名稱	說明
BackgroundImage	設定表單工作區的背景圖片，預設值為無。 [例] 以 c:\ch03 資料夾的 Image1.jpg 圖片檔當作表單的背景圖， 寫法： BackgroundImage = Image.FromFile("c:\\ch03\\Image1.jpg") ; 或 BackgroundImage = new Bitmap("c:\\ch03\\Image1.jpg") ;
Cursor	設定程式執行時視窗(表單)內的滑鼠游標形狀。預設值 Default。 [例] 將滑鼠游標形狀設為手指，程式寫法： Cursor = Cursors.Hand;
Font	顯示字型對話方塊，在此對話方塊中可設定字型、字型樣式、大小與效果，其預設值為新細明體，9pt。 [例] 將表單內顯示的文字字體為標楷體、大小設為 10、樣式為粗體字，寫法： Font = new Font ("標楷體", 10, FontStyle.Bold); 字型種類 大小 字型樣式
FormBorderStyle	設定表單邊界樣式，設定結果在執行時才看到，共有七種格式： ① None (沒有框線) ② FixedSingle (單線固定) ③ Fixed3D (立體固定) ④ FixedDialog (雙線固定對話方塊) ⑤ Sizable (大小可調整) -預設值 ⑥ FixedToolWindow (單線固定工具視窗) ⑦ SizableToolWindow (可調整工具視窗) [例] 將表單邊界樣式設為立體固定，程式碼： FormBorderStyle = FormBorderStyle.Fixed3D;
Text	表單標題欄上的標題文字(或稱關聯文字)，預設值為 Form1。 [例] 將表單標題文字由預設的 Form1，改設為 "第一個程式"。 寫法： Text = "第一個程式" ;

下面我們以設定表單常用的外觀類型屬性為例，介紹如何在表單設計階段，透過屬性視窗來設定屬性值的方法。

1. 如何設定表單的標題欄文字為「Visual C#」

Step 1 在表單上空白處按一下，將表單設為「作用表單」。此時表單的右側、右下與下方各出現一個小白方框(控制點)，表示表單被選取。

Step 2 到屬性視窗點選 Text 屬性，再按屬性值欄，將標題文字預設值「Form1」修改為「Visual C#」。

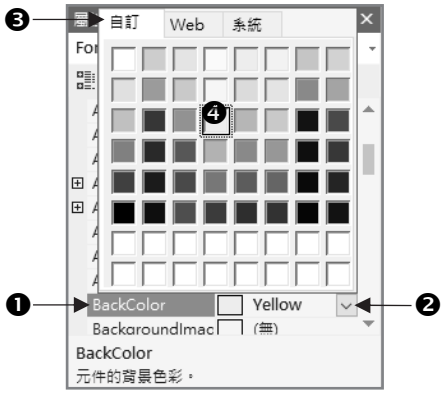


2. 如何設定表單的背景色由預設的「Control」改為「Yellow」

Step 1 先選取表單，再到屬性視窗點選 BackColor 屬性。

Step 2 到屬性值欄右側，點按 ▼ 下拉鈕出現「系統」標籤頁的顏色清單。

Step 3 點選「自訂」標籤頁改為色盤，點選『黃色』色塊，表單內的背景色由預設值「Control」變成「Yellow」。



3. 如何將表單的背景改成圖片顯示，背景圖片存於書附範例『ch03\images\VS.png』

Step 1 選取表單，點選 BackgroundImage 屬性。

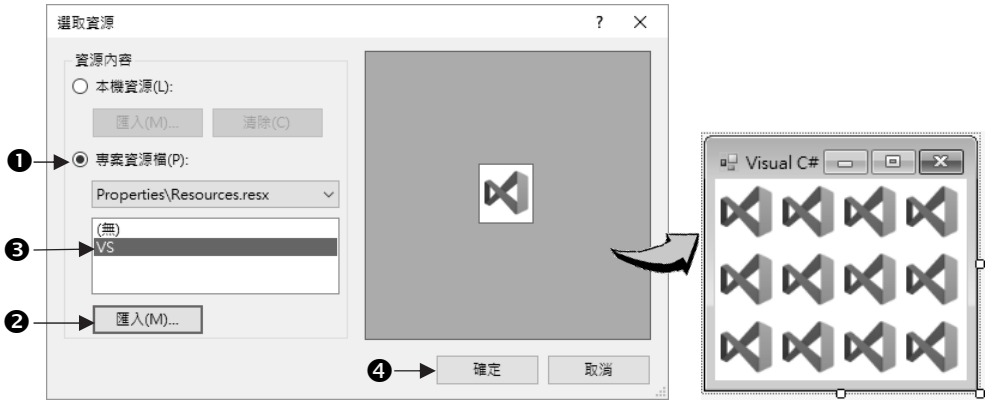
Step 2 按該屬性值欄的 ... 鈕開啟「選取資源」對話方塊。

Step 3 在「選取資源」對話方塊中，到「資源內容」框架內點選「專案資源檔」，再按 匯入(M)... 鈕開啟「開啟」對話方塊。

Step 4 在「開啟」對話方塊中，點選書附範例 [ch03\images\VS.png] 圖片檔，按 開啓(O) 鈕，返回「選取資源」對話方塊。

Step 5 返回「選取資源」對話方塊後，觀察圖片預覽區的內容，按 確定 鈕返回 IDE 整合開發環境。

Step 6 結果表單內的背景圖，以貼磁磚方式呈現所選取的圖片內容。



► 注意

① 本機資源(L): 選取本選項，不會將圖檔加到方案資料夾，複製方案時必須將圖檔和其資料夾路徑一起複製。

② 專案資源檔(P): 選取本選項，自動將圖檔加入到方案資料夾下的 Resource 資料夾內，以方便日後方案進行複製時不用再複製圖檔和其資料夾路徑。

3.1.2 視窗樣式的屬性

屬性名稱	說明
ControlBox	是否顯示標題欄上的控制圖示鈕，如：  、  、  、  ，預設值為 true。
MaximizeBox	是否顯示最大化鈕 ，預設值為 true。
MinimizeBox	是否顯示最小化鈕 ，預設值為 true。
HelpButton	是否顯示說明按鈕 ，預設值為 false。設為 true 時，必須 MaximizeBox 和 MinimizeBox 兩屬性值均為 false，說明按鈕才會顯示。
Icon	設定表單縮小時所用的圖示，預設值為  (圖示)。
ShowInTaskbar	設定當按最小化鈕時，程式圖示是否顯示在螢幕正下方工作列上，預設值為 true 表示顯示；false 表示不顯示。
TopMost	設定表單是否為最上層表單。預設值為 false 表示可被其他視窗遮蓋；若設為 true 表示永遠在最上層。

3.1.3 配置類型的屬性

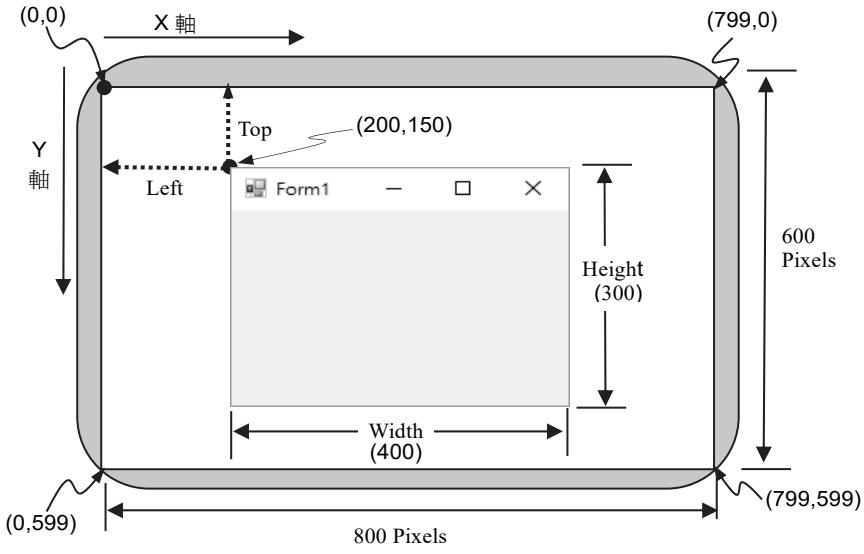
屬性名稱	說明
Location	以螢幕左上角為基準，設定表單左上角位置，向右及向下的座標值，預設值為 0, 0。 [例] 將表單左上角座標設為(200, 200)，程式碼： Location = new Point(200, 200);
Location 的 X 屬性	表單左上角距離螢幕左邊界的水平距離，預設值為 0。 [例] int x = Location.X; ⇦ 取得表單左上角 x 座標值
Location 的 Y 屬性	表單左上角距離螢幕上邊界的垂直距離，預設值為 0。 [例] int y = Location.Y; ⇦ 取得表單左上角 y 座標值
Size	表單的寬度和高度大小，預設值為 300, 300。 [例] Size = new Size(200, 360); ⇦ 將表單設為寬 200、高 360
Size / Width 子屬性	表單的水平寬度，預設值為 300。 [例] int w = Size.Width; ⇦ 取得表單的水平寬度。

屬性名稱	說明
Size / Height 子屬性	表單的垂直高度，預設值為 300。 [例] int h = Size.Height; ⇦ 取得表單的垂直高度。
StartPosition	決定程式初始化視窗在螢幕出現時的位置，共有下列 5 種狀態： ① Manual (手動) ② CenterScreen (螢幕中央) ③ WindowsDefaultLocation (預設位置-預設值) ④ WindowsDefaultBounds (螢幕中央並調整邊界為適當大小) ⑤ CenterParent (父視窗中央)
WindowState	表單執行的狀態，有三種設定方式： ① Normal (一般)：表單為設計階段大小 (預設值)。 ② Minimized (最小化)：表單縮為圖示，置於工作列上。 ③ Maximized (最大化)：表單放大佔滿整個螢幕。 [例] WindowState = FormWindowState.Maximized; ⇦表單最大化

上表中的 StartPosition 屬性值，可設定程式執行時表單視窗在螢幕顯示的起始位置。若要指定表單顯示的座標位置，要先選取表單，再到屬性視窗將 StartPosition 屬性值設為 Manual，然後再設定 Location 屬性值，就可以指定表單的起始座標。若要設定表單以最大化模式充滿整個螢幕要將 WindowState 屬性的屬性值設為 Maximized，而不是設定 MaximizeBox 屬性值為 true。程式的座標和數學的座標不同，容器(表單)左上角座標值為(0, 0)，小括號內第一個參數 X 座標值也就是水平距離；第二個參數為 Y 座標值也就是垂直距離。表單左上角座標水平方向向右為正；垂直方向向下為正。

下圖設定表單左上角座標為(200,150)，表單的寬度和高度為(400,300)，其程式碼寫法：

```
Location = new Point(200,150); ⇦ 設定表單左上角座標
Size = new Size(400,300);      ⇦ 設定表單的寬及高度
```



3.2 表單的常用事件

傳統 DOS 作業系統下，所設計出來的程式通常會按照既定的流程執行，每次執行的結果大致相同。在 Windows 作業系統下，我們將使用者所操作的每一個動作都視為「事件」，事件會被作業系統所攔截，並傳遞給應用程式的處理序來處理，這就是事件驅動(Event-Driven)的觀念。換言之「事件驅動」是指程式執行時，程式會不斷地等待操作者觸發事件，再根據系統所判斷出的事件，執行該事件處理函式內所撰寫的程式碼。由於程式執行時的流程是由操作者決定，因此每次執行流程未必是一樣的。

事件是物件傳送給應用程式的訊息，通知有事情發生需要處理，而傳送訊息的動作稱為「觸動事件」或「引發事件」。所觸動的事件要如何處理，就是以程式碼(指令或敘述)撰寫在事件處理函式裡。例如：家中門鈴響，就是「門鈴」物件的「按一下」事件被觸動，發出訊息 (門鈴聲) 通知有人來。我們可以將處理方式寫在「門鈴_按一下」事件處理函式中。要做處理的流程是：若是熟識的人就開門

歡迎；若是推銷員就假裝不在家…。Visual C# 的程式編寫就是以物件的事件為導向，所以了解物件的事件觸動時機就很重要。

Visual C# 對事件處理函式的命名，結合了事件傳送者(表單或控制項)的物件名稱和事件名稱，兩者中間以底線作區隔。例如：表單設計階段，在表單上無控制項處快按兩下，即進入到程式碼編輯區的 Form1_Load 事件處理函式內，等待你由鍵盤輸入程式碼，此時亦會在 Form1.Designer.cs 檔案內自動新增下列敘述，使得當表單被載入時觸發 Form1 表單的 Load 事件時會自動執行 Form1_Load 事件處理函式的程式碼。

```
this.Load += new System.EventHandler(this.Form1_Load);
```

定義將要處理的事件處理函式

上面敘述會自動新增到 Form1.Designer.cs 檔中，其中 ***this*** 用來指定表單的 Load 事件被觸發時，會自動執行 Form1_Load 事件處理函式。至於 Form1_Load 事件處理函式內要處理的程式碼就寫在 Form1.cs 程式檔中的 Load 事件處理函式內，Visual C# 會如下面在程式編輯區自動產生含有 sender 和 e 兩個引數 Form1 表單的 Load 函式。函式大括號內沒有任何敘述，等待你撰寫有關表單載入時要處理工作的程式碼。

物件名稱　事件處理函式名稱

```
private void Form1_Load(object sender, EventArgs e)
{                                    參數 1          參數 2
    預設沒有敘述
}
```

物件包括表單或控制項，它們都有自己本身對應的事件。有些事件可能是某控制項所獨有，也可能其他的控制項也具有。譬如：在表單物件上按一下會觸動該表單的 Click 事件，在按鈕控制項上按一下也會觸動該按鈕的 Click 事件。表單物件有 Load 事件，但是按鈕控制項卻沒有 Load 事件。本節先介紹表單物件常見的五個事件：

表單事件名稱	說明
Load (預設事件)	此事件發生時機是在程式開始執行表單第一次載入時，是表單最早被觸動的事件，也是優先權最高的事件，通常只執行一次。通常在此事件處理函式中設定物件屬性的初值。此事件是表單的預設事件，也就是在表單無控制項的地方快按兩下，即進入表單的 Load 事件程式碼編輯環境。
Activated	此事件的發生時機是每當表單(程式視窗)被點選成為「作用表單」時，即會觸動該表單的 Activated 事件。此事件被執行次數可能不止一次，表單被開啟為「作用視窗」的情況有下列三種： ①表單第一次被載入時，先執行 Load 事件，接著執行 Activated 事件。 ②使用滑鼠游標點選視窗，使它置於其他視窗最上層，此時該視窗就成為「作用視窗」。 ③若程式視窗最小化至工作列，再把它開啟時，該視窗會被放在桌面的最上層，此時該視窗就成為「作用視窗」。
Click	程式執行時，在表單內沒有放置控制項的地方按一下滑鼠左鍵時，就會觸動該表單物件的 Click 事件。
DoubleClick	程式執行時，在表單內沒有放置控制項的地方快按二下滑鼠左鍵時，就會觸動該表單的 DoubleClick 事件。由於執行 DoubleClick 事件前，Click 事件會先被觸動，設計程式時要注意兩事件的先後關係。
Paint	當表單內的控制項被重繪時會觸發此事件。在表單第一次載入時，會依序執行 Load、Activated 和 Paint 事件。若 Form1 遮住 Form2 時，當 Form1 移走時，會執行放在 Form2 的 Paint 事件內的程式碼。另外，只要表單大小有被調整時，也會觸動 Paint 事件。

實作 FileName：EventTest.sln

試依據 Form1 表單物件發生下列事件，在各事件中設定相關屬性的程式碼，以便觀察各事件的觸發時機：

① Load 事件：設表單標題欄的標題文字為 "Load"，並設表單寬度為 500、高度為 300，表單背景色為黃色。

② Activated 事件：每次執行時標題文字增加 ",Act" 字串。

③ Paint 事件：每次執行時標題文字增加 ",Paint" 字串。

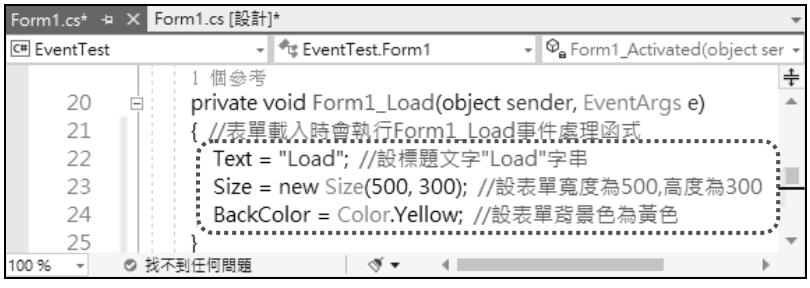
④ Click 事件：每次執行時標題文字增加 ",Click" 字串，表單寬度加寬 10 點。

⑤ DoubleClick 事件：每次執行時標題文字增加 ",Dclick" 字串。

▶ 解題技巧

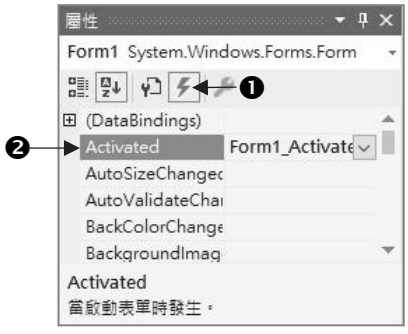
Step 1 建立專案和表單物件的 Form1_Load 事件處理函式

1. 建立專案名稱為「EventTest」的 Windows Forms APP(.NET Framework)應用程式專案。
2. 由於 Load 為表單的預設事件，在表單空白處快按兩下，就會直接進入 Form1_Load 事件處理函式內。
3. 本例要求在 Form1_Load 事件處理函式內做下列屬性初值設定：
   ① 設表單的 Text 屬性值為 "Load"。
   ② 設表單的 Width 屬性值為 500，Height 屬性值為 300。
   ③ 設表單的 BackCloor 屬性值為 Color.Yellow。

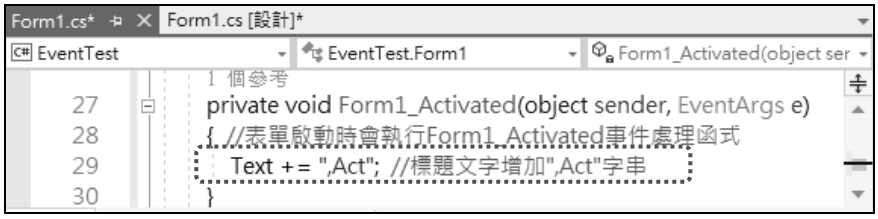


Step 2 建立表單物件的 Form1_Activated 事件處理函式

1. 在「屬性視窗」工具列的 ⚡ 事件鈕上按一下拉出表單的事件清單，移動滑鼠到「Activated」事件名稱上快按兩下，進入表單的 Form1_Activated 事件處理函式。

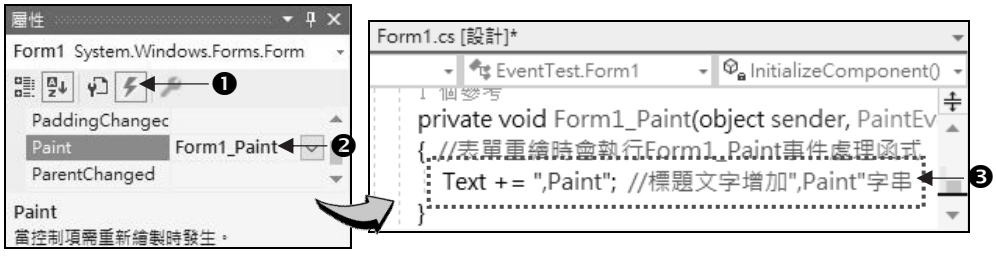


2. 本例要求在 Form1_Activated 事件處理函式內，設表單的 Text 屬性值增加 "，Act" 字串。

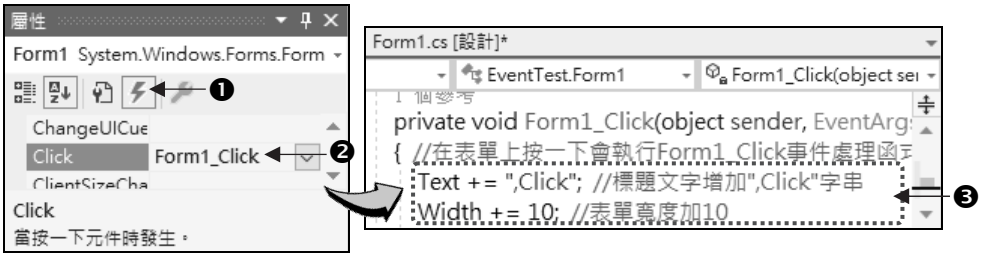


Step 3 建立表單物件的 Form1_Paint 事件處理函式

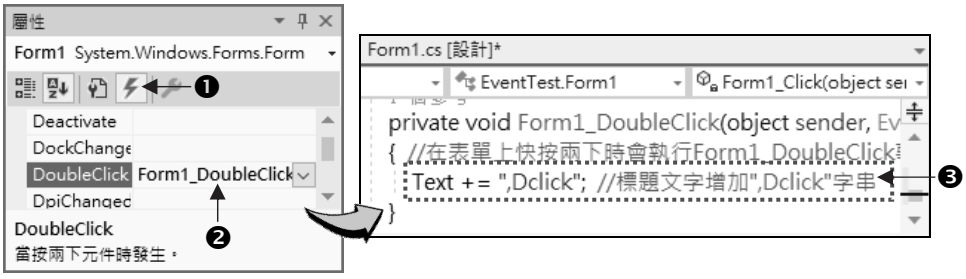
1. 依照 Step2 方法在「屬性視窗」中建立 Form1_Paint 事件處理函式。
2. 在 Form1_Paint 事件處理函式內，撰寫相關程式碼將 ",Paint" 字串插入表單的 Text 屬性值的後面。



Step 4 建立並撰寫表單物件的 Form1_Click 事件處理函式，將 ",Click" 字串插入到表單的 Text 屬性值後面，以及 Width 屬性值增加 10。

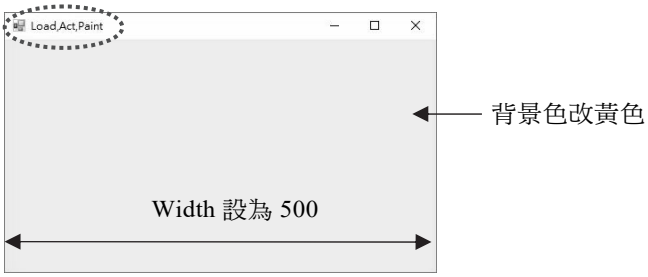


Step 5 建立並撰寫表單物件的 Form1_DoubleClick 事件處理函式，將 ",Dclick" 字串插入到表單的 Text 屬性值後面。

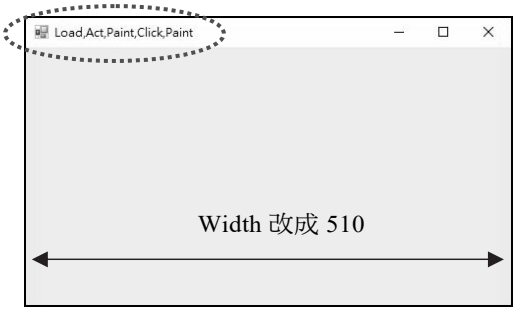


Step 6 按照下列指示操作觀察各事件變化情形

1. 按 F5 功能鍵執行程式，觀察表單的標題文字出現事件發生的先後次序，會發現程式會先執行 Form1_Load 事件處理函式，接著執行 Form1_Activated 事件處理函式，最後才是 Form1_Paint 事件處理函式，所以標題欄顯示 "Load,Act,Paint" 字串。

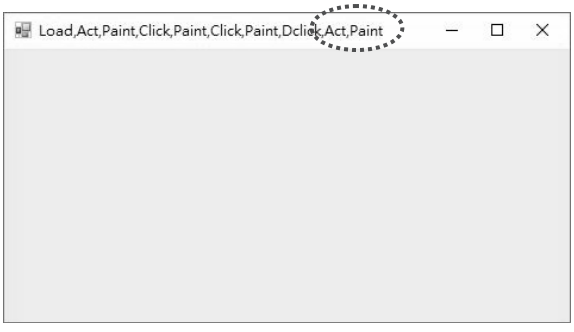


2. 在表單上按一下，會依序觸動 Click 事件和 Paint 事件。Click 事件將表單寬度加 10 變成 510，並將目前標題欄上顯示的 "Load, Act,Paint" 字串後，與本事件產生的 ",Click" 字串合併。由於表單寬度有改變會自動觸動 Paint 事件重繪表單，因此 ",Paint" 字串會合併到目前標題欄文字的後面。



3. 在表單上快按兩下，本程式會依序觸動表單的 Click 事件、Paint 事件、DoubleClick 事件。因為表單的 Click 事件會改變表單的大小，所以該事件後會觸動 Paint 事件來重繪表單。Click 事件使表單的 Width 寬度加 10 改成 520；標題欄會增加 ",Click,Paint" 字串。接著觸動 DoubleClick 事件使表單的標題欄會增加 ", Dclick" 字串。

4. 將表單最小化到視窗最下方工作列，然後點選表單圖示將表單復原，本程式會依序觸動表單的 Activated 和 Paint 事件處理函式，結果如下圖將 ",Act,Paint" 字串合併到目前標題欄顯示文字的後面。



5. 點選程式視窗右上方的 關閉鈕，關閉表單。

Step 7 完整程式碼

FileName: EventTest.sln	
01 using System;	// 1~9行 程式碼為宣告程式所引用的命名空間
02 using System.Collections.Generic;	// 為避免佔用篇幅，除非必要將不再列出
03 using System.ComponentModel;	
04 using System.Data;	

```
05 using System.Drawing;
06 using System.Linq;
07 using System.Text;
08 using System.Threading.Tasks;
09 using System.Windows.Forms;
10
11 namespace EventTest
12 {
13     public partial class Form1 : Form
14     {
15         public Form1()
16         {
17             InitializeComponent();     // 執行初始化動作，例如載入資源...等
18         }
19
20         private void Form1_Load(object sender, EventArgs e)
21         {     // 表單載入時會執行Form1_Load事件處理函式
22             Text = "Load";   //設標題文字為 "Load"字串
23             Size = new Size(500, 300);          // 設表單寬度為500，高度為300
24             BackColor = Color.Yellow;          // 設表單背景色為黃色
25         }
26
27         private void Form1_Activated(object sender, EventArgs e)
28         {     // 表單啟動時會執行Form1_Activated事件處理函式
29             Text += ",Act";       // 標題文字增加",Act"字串
30         }
31
32         private void Form1_Paint(object sender, PaintEventArgs e)
33         {     // 表單重繪時會執行Form1_Paint事件處理函式
34             Text += ",Paint";   // 標題文字增加 ",Paint" 字串
35         }
36
37         private void Form1_Click(object sender, EventArgs e)
38         {     // 在表單上按一下會執行Form1_Click事件處理函式
39             Text += ",Click";   //標題文字增加",Click"字串
40             Width += 10;        //表單寬度加10
```

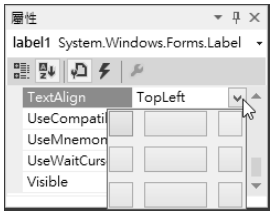
```
41          }
42
43          private void Form1_DoubleClick(object sender, EventArgs e)
44          {    // 在表單上快按兩下時會執行Form1_DoubleClick事件處理函式
45              Text += ",Dclick";   // 標題文字增加 ",Dclick" 字串
46          }
47      }
48 }
```

3.3 標籤控制項

使用 Label 標籤控制項可以在「表單」上提供文字的提示訊息，用來顯示程式執行過程或最後結果。但是標籤控制項在執行時只能顯示文字和數字資料，但無法透過鍵盤來輸入文字。點選工具箱中 Label 標籤工具，然後在表單上按一下或拖曳，就會在表單建立一個名稱為 label1 標籤控制項，label1 是標籤控制項的預設名稱。

3.3.1 標籤控制項的常用屬性

屬性名稱	說明
Name	為標籤控制項的物件名稱以供程式中參用，預設值為 label1。
AutoSize	控制項的寬度是否隨文字的寬度自動調整，預設值為 true。 [例] 將 label1 標籤控制項寬度固定不自動調整。程式寫法： label1.AutoSize = false;
BorderStyle	設定標籤的框線樣式： ① None(沒有框線) -預設值 ② FixedSingle(單線固定) ③ Fixed3D(立體固定) [例] 將 label1 標籤控制項邊框設成立體固定。程式寫法： label1.BorderStyle = BorderStyle.Fixed3D;
Font / Name	可用來設定顯示字體的字型名稱，不同字型名稱會顯示不一樣效果的字體，預設值為新細明體。
Font / Size	用來設定字體大小，預設值為 9。

屬性名稱	說明
Font / Unit	設定字體大小的單位，有下列 6 種： ① World(全局座標系統) ② Pixel(像素) ③ Point(點數-印表機用的單位，一點為 1/72 英吋) -預設值 ④ Inch(英吋) ⑤ Document(文件單位-一單位為 1/300 英吋) ⑥ Millimeter(公厘)。
Font / Bold	true (以粗體字顯示)、false (非粗體字) -預設值
Font / Italic	true (以斜體字顯示)、false (非斜體字) -預設值
Font / Strikeout	true (字體顯示時加刪除線)、false (不加刪除線) -預設值
Font / Underline	true (字體顯示時加底線)、false (不加底線) -預設值
ForeColor	設定物件或控制項的前景色，在標籤控制項中 ForeColor 屬性就是用來設定文字的顏色，預設值為 ControlText。
Image	顯示圖形，使用方式與表單的 BackgroundImage 相同，預設值為無。
ImageAlign	當 Image 屬性有存入圖片時，用來安排圖片在控制項上面的位置。屬性值和 TextAlign 相同。 [例] 將 ImageAlign 設為 MiddleLeft(左中)，TextAlign 設為 MiddleRight (右中)，結果為： 
Text	標籤控制項上面顯示的文字，可當輸入的提示訊息或顯示輸出結果。預設值為 label1。
TextAlign	控制項上面 Text 屬性值對齊方式： TopLeft(左上)-預設值、MiddleLeft(左中)、BottomLeft(左下)、TopMiddle(中上)、MiddleCenter(置中)、BottomMiddle(中下)、TopRight(右上)、MiddleRight(右中)、BottomRight(右下) [例] 將 label1 標籤控制項內的文字設成右上角顯示，程式寫法： label1.TextAlign = ContentAlignment.TopRight;  ⬆由清單中直接點選對齊的位置

每個物件都有 Name 物件名稱屬性，以方便在程式中呼叫使用，剛建立的物件系統會給予預設名稱，如表單物件預設名稱為「Form1」、「Form2」…；標籤控制項預設名稱為「label1」、「label2」…。物件在程式中除了可延用預設名稱

外，也可更改易辨識的名稱。為物件命名時最好在名稱前加上前置字串。使用前置字串程式中較易辨識是哪類的物件，後面接著的名稱則代表其功能。如：標籤控制項上面顯示價格「100 元」，其控制項名稱可命名為『LblPrice』，開頭的『Lbl』代表是標籤控制項，而『Price』代表價格。要注意的是，『100 元』是標籤的 Text 屬性值，而『LblPrice』才是 Name 屬性值，在程式中以此名稱來表示。

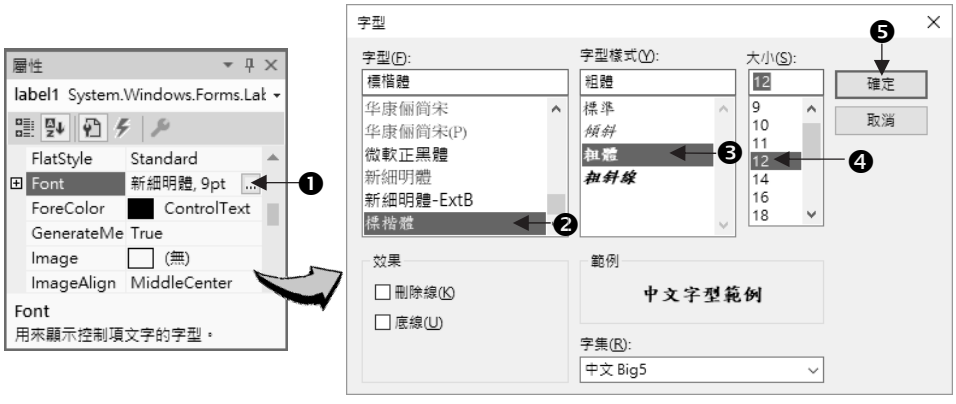
```
LblPrice.Text = "100 元" ;
```

3.3.2 Font 屬性的設定

標籤控制項主要是用來顯示文字，當作輸入的提示訊息或顯示輸出結果。下面操作步驟可將 label1 標籤控制項的字型大小設成 12，字型種類設為標楷體，樣式設為粗體：

Step 1 點選標籤控制項，使其出現控制點(小方框) label1 。

Step 2 到屬性視窗點選 Font 屬性，再按屬性值的 ... 按鈕，開啟下圖的「字型」對話方塊。然後在對話方塊中設定字型的各種屬性值。



只要物件含有 Text 屬性，都可在程式中使用 Font 類別來設定字型的種類、大小、樣式。譬如：下面敘述將 label1 標籤控制項內字體設為『標楷體』、大小『12』，並以『粗體』顯示，其程式寫法如下：

```
label1.Font = new Font("標楷體", 12, FontStyle.Bold) ;
```

字體　大小　字型樣式

字型樣式(FontStyle)的列舉參數有五種樣式：FontSytle.Bold(粗體)、FontStyle.Italic(斜體)、FontStyle.Regular(標準)、FontStyle.Strikeout(刪除線)和FontStyle.Underline(底線)。若文字大小設為 10、字體為標楷體、字型樣式為(粗體+斜體)一起顯示，程式寫法：

```
label1.Font = new Font ("標楷體", 10, FontStyle.Bold | FontStyle.Italic );
```

3.4 按鈕控制項

在表單上設計輸出入畫面時，常常會使用到按鈕(Button)來設計 確定、取消、開啓(O) … 等按鈕。當你在表單上面其中一個按鈕上按一下，會執行該按鈕對應的 Click 事件處理函式(預設事件)，在函式中是按鈕所要達成特定功能的程式碼。按鈕控制項常用的屬性如下：

屬性名稱	說明
Enabled	設定按鈕按下去是否有效。 ① true：按鈕按下去有效(預設值) button1； ② false：按鈕無效 button1。 [例] 將 button1 按鈕設為無效，按鈕上面的文字會呈灰色。 即設按鈕的 Click 事件不會被觸動。寫法： button1.Enabled = false;
TabIndex	設定控制項駐停的順序，屬性值會按照控制項建立的順序，由 0 開始依序編號，必要時可以自行修改。程式執行時當按 Tab 鍵，表單上的控制項會依該順序輪流成為作用控制項。
TabStop	設定控制項是否可駐停(焦點 Focus)，若可駐停按 Tab 鍵時，該控制項才有機會被停駐，輪流成為作用控制項，預設值為 true (表示可駐停)。
Visible	決定按鈕是否顯現。true 按鈕可見(預設值)；false 按鈕被隱藏。 [例] button1.Visible = false; //將 button1 按鈕設為隱藏看不到

3.5 文字方塊控制項

在表單上使用「標籤」工具只能顯示文字，卻無法接受文字的輸入或修改的工作。假若允許使用者對表單上的文字資料做輸入或修改的動作，此時就必須使用工具箱的 TextBox 文字方塊工具來完成。所以「文字方塊」是可以用來輸入、修改和顯示文字資料的物件。如果想對文字方塊內的文字做更細部的設定，例如多種字型樣式、段落縮排、項目符號...，則可以使用本書第十一章介紹的豐富文字方塊控制項。

3.5.1 文字方塊控制項的常用屬性

屬性名稱	說明
MaxLength	設定文字方塊內可輸入的最多字元數目，預設值為 32,767。 [例] textBox1.MaxLength = 5;　// 限輸入 5 個字元
PasswordChar	輸入字元時，所輸入的字元不直接顯示，改由指定的字元取代。適用於密碼輸入，預設為不使用。 [例] textBox1 文字方塊控制項輸入時，改用*星號字元取代。 textBox1.PasswordChar = '*' ;
Text	將輸入的文字以字串方式存到 Text 屬性中，若程式在設計或執行階段此屬性值有異動，該控制項上面的文字亦跟著異動。
ReadOnly	設定文字方塊內的文字資料是唯讀不允許修改。預設值為 false 表示允許修改；若為 true 表示唯讀和標籤控制項一樣只能顯示文字。 [例] textBox1 文字方塊控制項設成唯讀狀態 textBox1.ReadOnly = true;
Multiline	當顯示文字資料超過控制項所設定寬度時，決定是否採多行或單行顯示資料。預設值為 false，不允許多行顯示；若設為 true 表示允許多行顯示。[註 1] [例] textBox1 文字方塊控制項設成多行顯示。 textBox1.Multiline = true;
WordWrap	當 Multiline 屬性值設為 true 時，可進一步設定文字是否自動換行，預設值為 true 表示自動換行 [註 2]。

屬性名稱	說明
ScrollBars	用來設定在多行顯示的文字方塊控制項內，是否出現垂直或水平捲軸。有下列屬性值： ① None (預設值：無)　② Horizontal (水平捲軸) ③ Vertical (垂直捲軸)　④ Both (水平與垂直捲軸兩者皆有) [例] textBox1 文字方塊控制項設成有水平捲軸 textBox1.ScrollBars =ScrollBars.Horizontal;

[註 1] 由於文字方塊控制項的 Multiline (多行)屬性其預設屬性值為 false，也就是單行顯示。所以當文字資料超過文字方塊控制項寬度時，超出的文字資料無法顯示出來。若文字內容需要多行顯示時，要將 Multiline 屬性設為 true，此時文字方塊控制項就可以拖曳大小來容納多行文字。

[註 2] 若希望超過文字方塊控制項寬度的資料會自動移到下一行，可以將 WordWrap (自動換行)屬性設為 true。當 WordWrap 屬性設為 false 時，則可以設定 ScrollBars (捲軸)屬性，使得控制項能顯示垂直或水平捲軸，供使用者拖曳來瀏覽文字。

3.5.2 文字與數值間資料型別的轉換

文字方塊控制項中最常使用的屬性就是 Text 屬性，不管輸入的資料是文字或是數值，Visual C# 必須將資料轉成字串才能存入 Text 屬性中。在程式中可使用 Convert.To xxx 方法(xxx 即為指定的資料型別名稱)，將變數或資料轉成適當的數值資料才能做正確的運算。其語法如下：

語法

```
變數 = Convert.To xxx (變數或資料) ;
```

例 Convert.ToInt32("168")　[結果] 168　(將字串 "168" 轉成 int 型別)

例 Convert.ToSingle("168.77")　[結果] 168.77 (將字串 "168.77" 轉成 float 型別)

例 Convert.ToDouble("77.1456")　[結果] 77.1456 (將字串 "77.1456" 轉成 double 型別)

例 Convert.ToBoolean(1)　[結果] true　(將數值 1 轉成布林型別)

例 Convert.ToString(168)　[結果] "168"　(將整數 168 轉成字串型別)

3.5.3 數值格式化輸出字串

上一小節介紹 Convert.ToString()方法，可以將數值資料轉成字串型別資料。本小節將再介紹另一種語法，除了可以將數值資料轉成字串型別資料外，還可以指定輸出的格式化樣式，語法如下：

語法

```
數值變數.ToString (格式符號字串)
```

語法中常用的格式符號和使用說明如下表所示，若省略格式符號預設為 G：

格式符號	說明
G、g	G、g 代表以一般數值格式顯示。例如： double num = 123.45; label1.Text = num.ToString("G");　//會顯示「123.45」
Dn、dn	Dn、dn 代表以 n 位數顯示，不足處補 0。例如： int num = 123; label1.Text = num.ToString("D4");　//會顯示「0123」
Cn、cn	Cn、cn 代表以貨幣方式顯示，小數部分四捨五入到 n 位數。例如： double num = 1234.56; label1.Text = num.ToString("C1");　//會顯示「NT$1,234.6」
Fn、fn	Fn、fn 代表小數部分四捨五入到 n 位數。例如： double num = 1234.56; label1.Text = num.ToString("F3");　//會顯示「1234.560」
N、n	N、n 代表以千位號方式顯示，小數部分四捨五入到二位數。例如： double num = 1234.567; label1.Text = num.ToString("N");　//會顯示「1,234.57」

3.5.4 文字方塊控制項的常用方法

所謂「方法」(Method) 就是指 Visual C#所提供物件或控制項的特定功能。在程式執行的階段，可以使用物件或控制項的「方法」來協助快速完成指定的工作。

一、Clear()方法

Clear()方法可將文字方塊的文字內容清成空白。在程式執行時，要將 textBox1 文字方塊內顯示的文字清除，其寫法如下：

```
textBox1.Clear();　或　textBox1.Text = "";　//兩者功能相同
```

二、Focus()方法

設定文字方塊控制項為駐停焦點(Focus)。所謂「駐停焦點」就是使某個控制項成為作用物件，以供使用者操作。必須該控制項含有 TabIndex、TabStop 屬性，才可以設定駐停作用。要使用 Focus 方法的控制項，其 Visible 與 Enabled 屬性值必須都設為「true」才有效。Focus 方法的語法：

語法

```
物件名稱.Focus( );
```

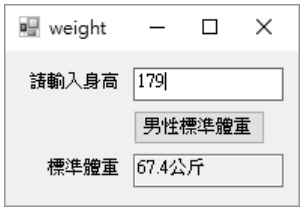
例 將游標移到 textBox1 文字方塊控制項上面。其寫法如下：

```
textBox1.Focus();
```

實作 FileName：weight.sln

設計一個計算成年男性標準體重的程式。使用者輸入身高(最多三位整數)後，點擊 男性標準體重 按鈕，就會計算並顯示標準體重(小數部分四捨五入取 1 位數)。假設男性的標準體重計算公式：(身高-170)× 0.6+ 62。

▶ 輸出要求



▶ 解題技巧

Step 1 建立輸出入介面

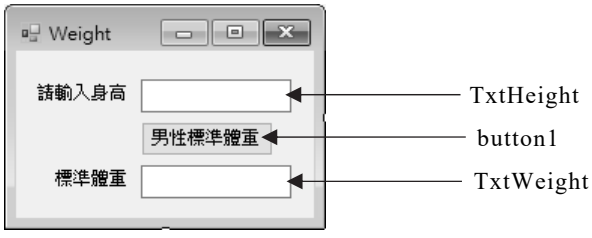
1. 新增專案並以「weight」為新專案名稱。
2. 建立輸出入介面

由執行結果可知，本實作必須在表單上建立下列各控制項名稱：

① TxtHeight 文字方塊控制項用來接受使用者輸入之身高。

② TxtWeight 文字方塊控制項用來顯示計算後之體重。

③ button1 按鈕控制項，用來執行計算該身高之標準體重。



Step 2 問題分析

1. 在表單載入時會觸動該表單 Load 事件，應程式要求在 Form1_Load 事件處理函式內做下列屬性初值設定：

① TxtHeight 文字方塊控制項限制只能鍵入三位數，所以將 MaxLength 屬性值設為 3。

② 由於執行時 TxtWeight 文字方塊控制項，只能顯示文字不能修改，所以將該控制項的 ReadOnly 屬性值設為 true 成為唯讀。

③ 程式開始執行時，希望 TxtWeight 文字方塊設為作用控制項，等待使用者輸入資料，將 TabIndex 屬性值設為 0，成為第一個可駐停控制項。

④ 將 TxtHeight 文字方塊的 Text 屬性預設成 170。

2. 按 button1 男性標準體重 鈕會觸發該按鈕的 Click 事件，在 Click 事件處理函式內要完成下列動作：

① 先將 TxtHeight.Text 透過 Convert.ToInt32()方法將輸入的字串轉成數值，置入 height 變數中，取得使用者輸入的身高資料。

② 利用題目所提示的公式，計算出標準體重。

③ 將換算後之數值用 ToString("f1")方法轉成字串，顯示到小數以下 1 位，再加上「公斤」字串後，指定給 TxtWeight 的 Text 屬性。

④ 用 Focus() 讓 TxtHeight 取得駐停焦點，等候使用者再輸入。

Step 3 完整程式碼

FileName: weight.sln

```
01 namespace weight
02 {
03     public partial class Form1 : Form
04     {
05         public Form1()
06         {
07             InitializeComponent();
08         }
09
10         private void Form1_Load(object sender, EventArgs e)
11         {
12             TxtHeight.MaxLength = 3;      //設最多只能輸入3位數
13             TxtWeight.ReadOnly = true;    //設為唯讀不能輸入
14             TxtHeight.TabIndex = 0;       //設為第一個停駐焦點
15             TxtHeight.Text = "170";       //設定預設值
16         }
17
18         private void button1_Click(object sender, EventArgs e)
19         {
20             int height = Convert.ToInt32(TxtHeight.Text);          //字串轉成整數
21             double weight = ( height - 170) * 0.6 + 62;            //計算標準體重
22             TxtWeight.Text = weight.ToString("f1") + "公斤";   //顯示標準體重
23             TxtHeight.Focus();                    //將停駐焦點移到TxtHeight
24         }
25     }
26 }
```

▶ 馬上練習

將上例再加上一個計算成年女性標準體重的按鈕，使用者可自行選擇要計算男性或女性體重。假設女性的標準體重計算公式：(身高-158)× 0.5+ 52。

3.5.5 例外處理

程式執行時發生問題或有異常狀況發生，導致無法繼續執行時，此時會由系統自動發出一個訊號稱為「例外」(Exception)。譬如，陣列索引超出範圍、數字碰到除以零、資料型別轉換失敗等都會產生例外，此時程式會自動結束。但有時應程式需求，不希望遇到這些例外就自動結束，希望能對這些例外加以處理，再對這些例外撰寫程式碼做相關的處理，我們將此過程稱為「例外處理」(Exception Handle)。例如前一個範例，若使用者在 TxtHeight 文字方塊內輸入 "一百八" 字串資料時，按 男性標準體重 鈕計算時，會因為字串無法轉換成整數資料而發生例外，導致程式中止無法執行。

為了避免執行期間發生錯誤而中斷程式，在程式中可加上 try{…}catch{….} 來做偵測，其寫法如下：

語法

```
try
{
      // 將可能會發生錯誤的程式區段置於此處
}
catch [(Exception ex)]
{
      // 將發生錯誤時要處理的程式區段置於此處
}
```

實作 FileName：TryDemo.sln

延續前一範例，在 button1_Click 事件處理程序中，加入 try…catch 例外處理敘述。當使用者在 TxtHeight 文字方塊輸入錯誤資料時，會顯示 "請輸入整數！" 的例外狀況訊息，再將 TxtHeight 設為作用控制項並清空資料，等待使用者重新輸入。

▶ 輸出要求

▶ 解題技巧

Step 1 新增專案

以「TryDemo」為專案名稱，輸出入介面如上例 weight.sln。

Step 2 問題分析

1. 按 男性標準體重 鈕觸發 button1_Click 事件處理函式，在函式中會從 TxtHeight 中讀取使用者輸入的身高，因為可能輸入錯誤的資料型別，所以必須將會發生例外錯誤的程式碼放入 try 區塊中。
2. 當發生例外錯誤時，所要處理的程式碼則寫在 catch 區塊中。發生例外錯誤時，在 TxtWeight 文字方塊上顯示 「請輸入整數！」的訊息；另外用 Clear()方法清空 TxtHeight 中的錯誤資料。

Step 3 完整程式碼

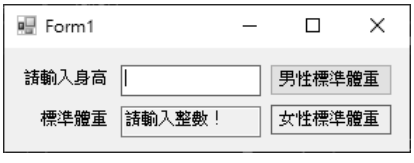
FileName: TryDemo.sln

```
01 namespace TryDemo
02 {
03     public partial class Form1 : Form
04     {
05         public Form1()
06         {
07             InitializeComponent();
08         }
09
10         private void Form1_Load(object sender, EventArgs e)
11         {
12             TxtHeight.MaxLength = 3;
13             TxtWeight.ReadOnly = true;
14             TxtHeight.TabIndex = 0;
15         }
```

```
16
17          private void button1_Click(object sender, EventArgs e)
18          {
19              try
20              {     // 可能發生錯誤的程式碼放在try區塊中
21                  int height = Convert.ToInt32(TxtHeight.Text);
22                  double weight = (height - 170) * 0.6 + 62;
23                  TxtWeight.Text = weight.ToString("f1") + "公斤";
24              }
25              catch
26              {   // 發生錯誤時執行catch區塊
27                  TxtWeight.Text = "請輸入整數！";    // 顯示提示訊息
28                  TxtHeight.Clear();                  // 清空錯誤資料
29              }
30              TxtHeight.Focus ();
31          }
32      }
33 }
```

▶ 馬上練習

將前例的馬上練習，增加例外錯誤處理功能。



3.5.6 文字方塊控制項的常用事件

一、TextChanged 事件(預設事件)

在程式執行時，當文字方塊的 Text 屬性值有異動，就會觸動該文字方塊的 TextChanged 事件。所以可以將和 Text 屬性值有關的敘述，寫在 TextChanged 事件處理函式中。在前面範例中，我們將計算體重的敘述寫在按鈕的 Click 事件中。如果將敘述改寫在 TextChanged 事件中，輸入資料有變動就立即顯示體重的互動效果，可省略按下按鈕的動作。

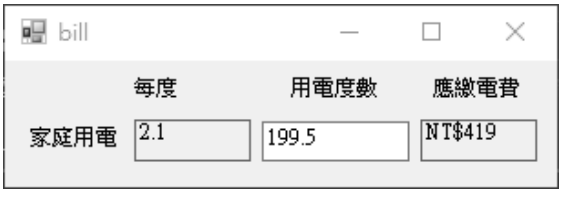
二、Enter 事件

當文字方塊控制項取得駐停焦點(Focus)時，就會觸發 Enter 事件，通常用來在該事件中設定文字方塊的文字預設值。例如希望某文字方塊預設值為「台中市」，就可以在 Enter 事件中設 Text 屬性值為「台中市」，每當用滑鼠或 Tab 鍵移動駐停焦點到該文字方塊時，文字方塊上面就自動改為「台中市」。

實作 FileName：bill.sln

試設計一個計算家庭電費的程式。假設家庭用電每 1 度電電費 2.1 元，程式執行時只要使用者輸入用電度數，會立即計算出應繳電費。當移動插入點到用電度數文字方塊時，會將文字內容設為 0 等待輸入。

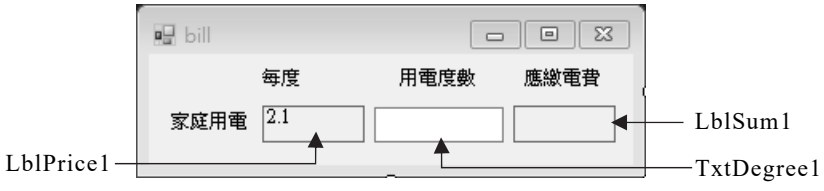
▶ 輸出要求



▶ 解題技巧

Step 1 建立輸出入介面

1. 新增專案並以「bill」為新專案名稱。
2. 由輸出結果可知，本實作只允許輸入用電度數，所以在表單建立 TxtDegree1 文字方塊。另外建立 LblPrice1 和 LblSum1 標籤控制項，分別顯示每度單價和金額計算的結果。



Step 2 問題分析

1. 在 Load 事件中使用 Focus()方法，使 TxtDegree1 文字方塊取得駐停焦點。
2. 本實作要求當插入點移入文字方塊時，會將文字內容設定為 0 等待輸入，所以必須將文字方塊設定的工作寫在 Enter 事件處理函式內。寫法為：TxtDegree1.Text = "0"。
3. 由於本實作要求用電度數一有改變，金額會立即計算，所以計算金額的程式碼要寫在 TextChanged 事件處理函式內。演算法如下：
   ① 輸入的字串使用 Convert.ToDouble ()方法轉換成數值。
   ② 計算結果以 ToString("C0")方法轉成字串，再指定給 LblSum1 的 Text 屬性。
   ③ ①～② 必須放置在 try 區塊中，以便處理例外錯誤的發生。
   ④ 在 catch 程式區段內將金額改設為 0，來避免顯示錯誤的資料。

Step 3 完整程式碼

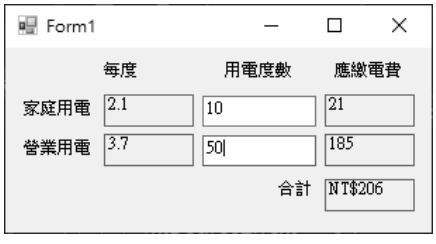
FileName: bill.sln

```
01 namespace bill
02 {
03     public partial class Form1 : Form
04     {
05         public Form1()
06         {
07             InitializeComponent();
08         }
09
10         private void Form1_Load(object sender, EventArgs e)
11         {
12             TxtDegree1.Text = "0";      // 預設文字方塊內容為0
13             TxtDegree1.Focus();         // 預設文字方塊取得駐停焦點
14         }
15
16         private void TxtDegree1_Enter(object sender, EventArgs e)
17         {   // 當文字方塊取得駐停焦點時執行
18             TxtDegree1.Text = "0";
19         }
20
21         private void TxtDegree1_TextChanged(object sender, EventArgs e)
```

```
22          {   // 文字方塊的Text屬性改變時執行
23              try
24              {
25                  double sum1;
26                  sum1 = Convert.ToDouble(LblPrice1.Text) *
                            Convert.ToDouble(TxtDegree1.Text);    // 計算電費金額
27                  LblSum1.Text = sum1.ToString("C0");     // 顯示電費金額
28              }
29              catch
30              {
31                  LblSum1.Text = "0";
32              }
33          }
34      }
35 }
```

▶ 馬上練習

延續上面的電費計算程式，增加營業用電的計算，度數預設為 0。程式執行時只要輸入家庭用電和營業用電的度數，會立即計算出電費金額，以及合計的總金額。

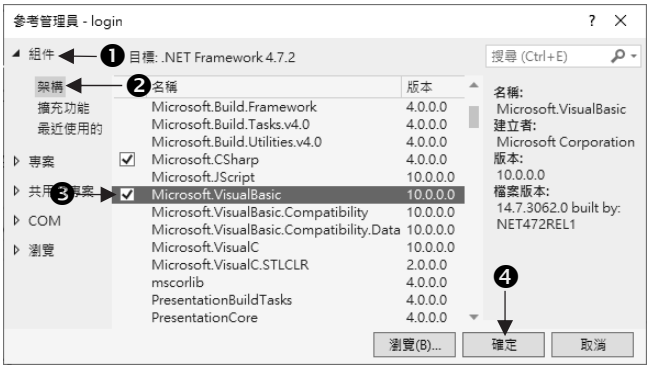


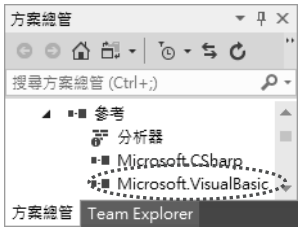
3.6 InputBox 函式

在 Visual Basic 中提供好用的 InputBox 函式，可免去在表單上建立控制項，就可以直接顯示輸入對話方塊。使用者可在所提供的文字框內輸入資料，再按 確定 鈕，就可達到輸入資料的目的。那在 Visual C# 程式中是否也可以使用 Visual Basic 的 InputBox 函式嗎？答案是可以的。這是因為 .NET 統一了不同模型的程式庫，讓不同的程式語言如：Visual Basic、Visual C#、Visual C++ 都可共同使用 .NET Framework 類別程式庫。在 Visual C# 中若要使用 Visual Basic 的 InputBox 函式，必須在專案中加入 Microsoft.VisualBasic 元件，操作步驟如下：

Step 1 在專案加入 Microsoft.VisualBasic 元件

執行功能表的【專案(P)/加入參考(R)...】開啟「參考管理員」視窗，接著依下圖操作在專案中加入 Microsoft.VisualBasic 元件。完成後方案總管視窗中的「參考」資料夾內，就會加入 Microsoft.VisualBasic 元件。

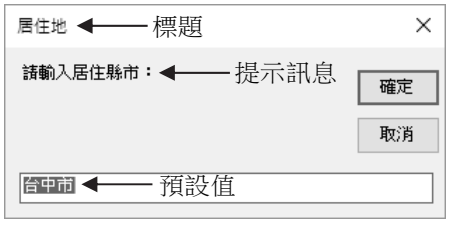




Step 2 加入 Microsoft.VisualBasic 元件後，接著即可使用 Microsoft.Visual Basic.Interaction.InputBox() 函式開啟輸入對話方塊，在此對話方塊鍵入的資料會以字串型別放入等號左邊指定的字串變數，語法如下：

```
string 字串變數 = Microsoft.VisualBasic.Interaction.InputBox
                  ( 提示訊息 [, [標題] [, [預設值] [, Xpos, Ypos] ] ] );
```

右圖使用 InputBox 輸入對話方塊函式，將輸入值存放到所宣告的 city 字串變數。寫法如下：




```
string city = Microsoft.VisualBasic.Interaction.InputBox ("請輸入居住縣市：",
              "居住地", "台中市" );
```

呼叫 InputBox 函式時，會自動出現一個對話方塊給使用者，用來等待使用者由鍵盤輸入文字，並將輸入的文字傳給等號左邊的字串變數。如果使用者在 TextBox 中輸入資料後，再按 確定 鈕，則資料會傳給 city 字串變數；假如按 取消 鈕，則 city 值會是空字串("")。