


CHAPTER

10

功能表與工具列

- ✧ 學習 MenuStrip 功能表控制項
- ✧ 學習 ContextMenuStrip 快顯功能表控制項
- ✧ 學習 ToolStrip 工具列控制項

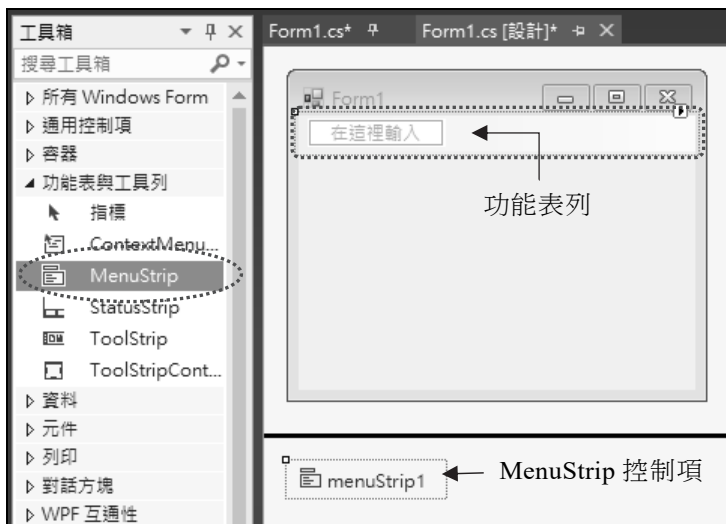
10.1 MenuStrip 功能表控制項

視窗應用程式中，若要將同性質的功能做有系統的分類，功能表是一個非常便利的介面，例如 Word、Excel 等大型的應用程式都提供各式各樣的功能表，這些功能表都是透過工具箱的  MenuStrip 功能表工具在表單上快速建立多層次功能項目的功能表列。至於如何在按下功能項目時能做指定的工作，那就要對被按下功能項目的 Click 事件撰寫該工作的相關程式碼。本章主要議題是介紹如何建立功能表以及如何使功能表的項目可以真正運作。

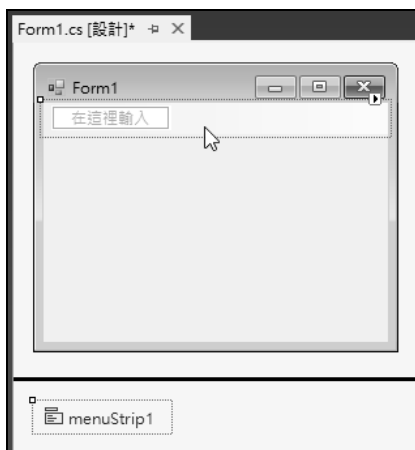
10.1.1 如何建立功能表的項目

一般功能表大都在表單設計階段事先建立，當然也可在程式執行時才建立，但此種機會很少，因此本節著重於如何在表單設計階段建立功能表。

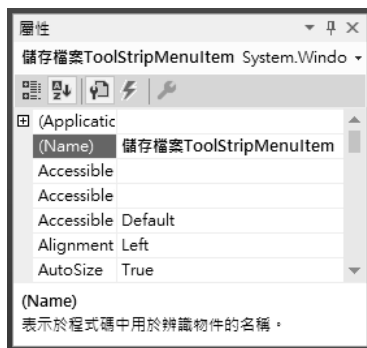
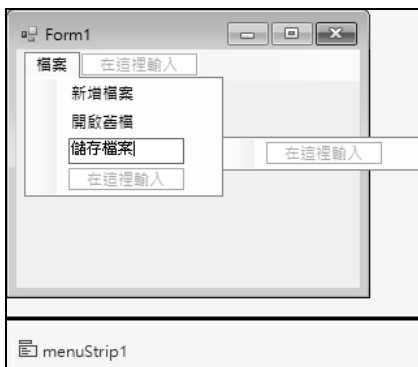
Step 1 從工具箱中將 MenuStrip 功能表工具拖曳到表單中，該控制項如下圖自動置於表單 Form1 的正下方，預設名稱為 menuStrip1，且標題欄正下方會出現功能表列，用來建立相關的功能表項目名稱。



Step 2 當選取表單 Form1 正下方的 menuStrip1 控制項，或左下圖上方的功能表列，都會出現 的輸入方塊。移動滑鼠到此輸入方塊上按一下，出現插入點游標，便可開始鍵入功能表的各項目名稱。譬如右下圖輸入「檔案」主功能項目，此時在項目的下方及右邊也會出現 輸入框，如果在右邊輸入，那就是同層次的項目；若是在下方輸入，那將是該項目的下一層子選項。



Step 3 在第一層輸入「檔案」主功能項目後，接著如左下圖往下(即第二層功能表)輸入『新增檔案』、『開啟舊檔』、『儲存檔案』等三個子功能項目。每層建立的功能表項目都是 ToolStripMenuItem 控制項。



完成上圖設定後，menuStrip1 功能表控制項在主功能項目與各子功能項目處，皆會形成獨立的物件，而這些物件的 Name 屬性值是自動產生，依序為：「檔案 ToolStripMenuItem」、「新增檔案 ToolStripMenuItem」、「開啟舊檔 ToolStripMenuItem」和「儲存檔案 ToolStripMenuItem」。

Step 4 接著在下圖「檔案」主功能項目的右側再建立『編輯』主功能項目功能(即第一層功能項目)，以及正下方建立『複製』、『貼上』、『剪下』三個第二層「編輯」子功能項目，並在「貼上」項目往右邊輸入第三層子功能項目：『貼成物件』、『貼入選取區』、『貼成新影像』。輸入完畢在「貼上」子項目的右邊會自動出現向右箭頭 ▶，表示該項目尚有子功能表。

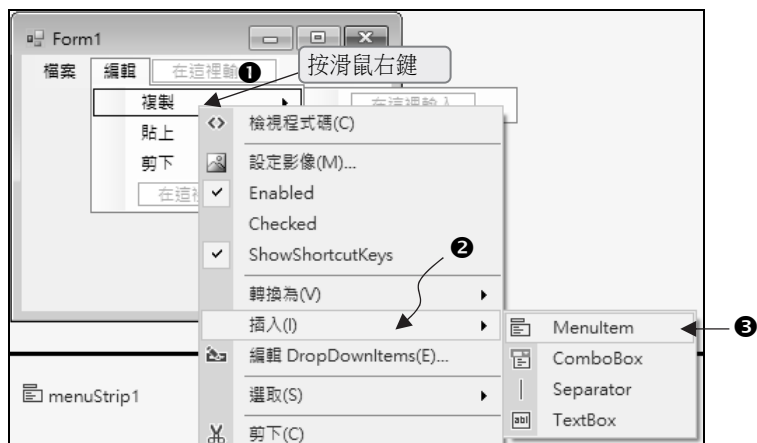


10.1.2 如何新增、刪除、移動功能項目

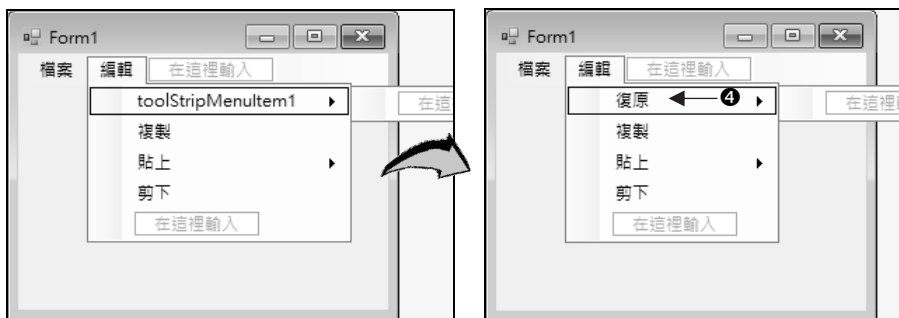
本節將介紹如何在表單設計階段，對已建立的功能表進行功能項目增刪和調整工作：

Step 1 新增功能項目

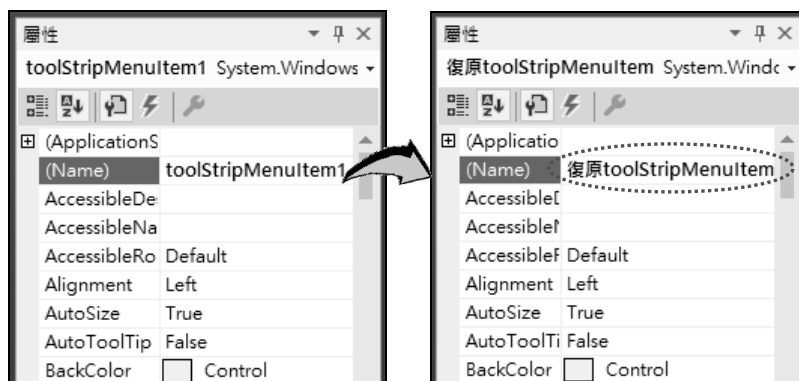
在表單設計階段對已建立好的功能表進行新增一個功能項目。譬如：欲在下圖的「複製」功能項目上方插入「復原」功能項目，其做法是先在「複製」功能項目上按滑鼠右鍵由快顯功能表中選用【插入/MemuItem】指令。



此時，會在「複製」功能項目上方會增加一個項目，預設名稱為「toolStrip MenuItem1」，再將這個項目名稱更改為『復原』即可。

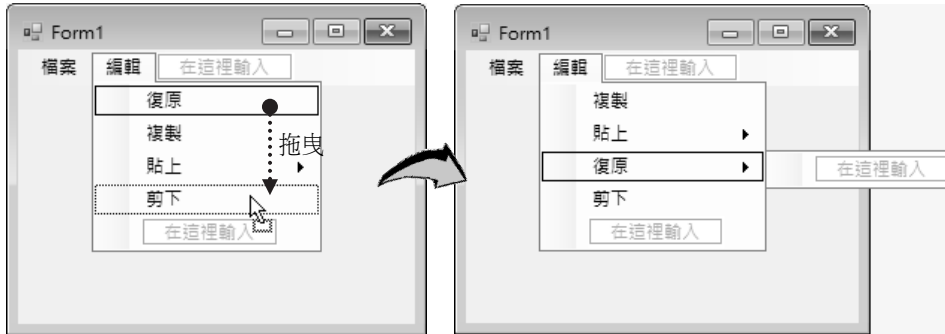


至於 Name 屬性還是要在屬性視窗中設定，為求統一，我們將新增功能項目的 Name 屬性設定為『復原 ToolStripMenuItem』。



Step 2 移動功能項目順序

如果要在功能表上變更功能項目的順序，只要按住項目不放拖曳到適當的位置即可。如下圖中將「復原」項目向下移到「剪下」項目處。結果，「復原」項目會被置放在「剪下」項目的上方。



Step 3 刪除項目

若要刪除某個功能項目，只要選定該功能項目後，直接按鍵盤 **Del** 鍵或按滑鼠右鍵由快顯功能表選擇【刪除(D)】指令即可刪除該功能項目。

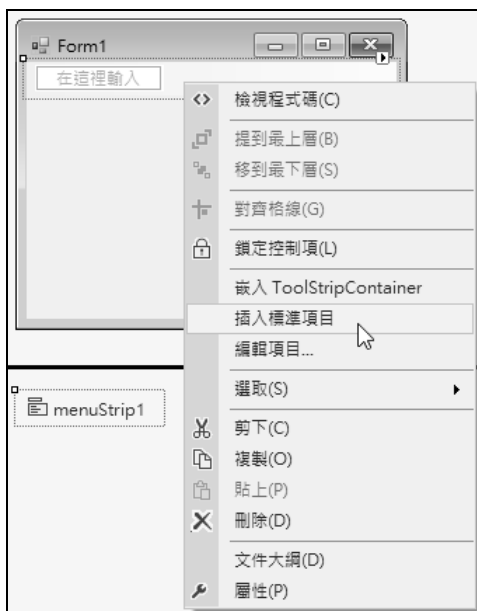
10.1.3 快速建立標準功能表項目

上節中介紹如何在表單的標題欄正下方建立功能表項目，本節將介紹建立功能表的另一種方式，其方法是先快速自動產生一個預設的功能表項目，接著再對不合需求的功能表項目進行增刪和調整的工作。

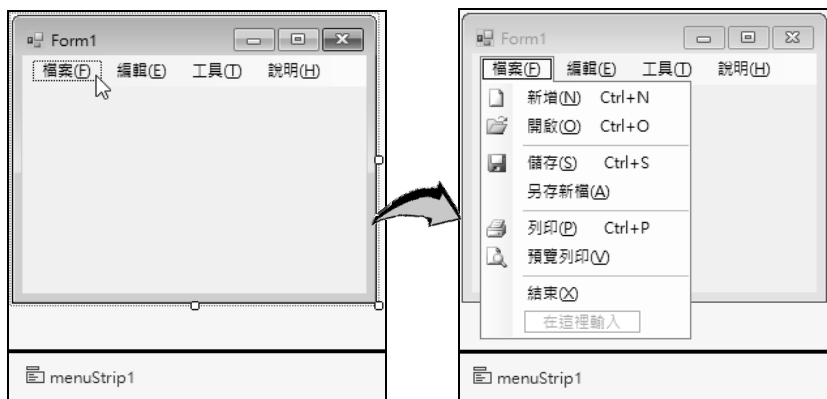
Step 1 先刪除「檔案」與「編輯」主功能項目，成為空白功能表列

先選定該主功能項目後，直接按鍵盤 **Del** 鍵，即可將該功能項目連同子項目一併刪除。

Step 2 在空白的功能表列上，按下滑鼠右鍵，從快顯功能表中選擇【插入標準項目】指令。

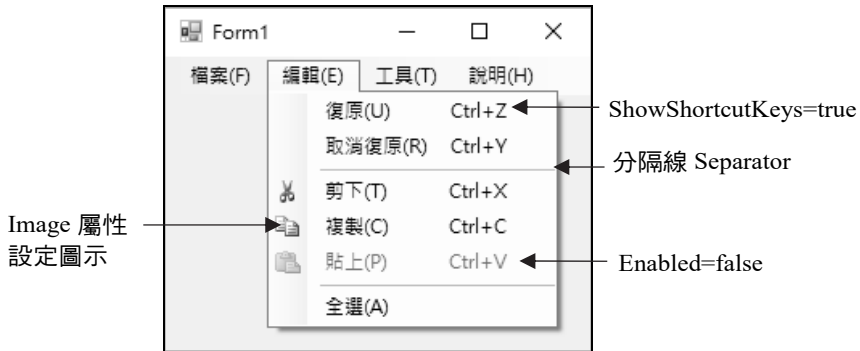


Step 3 下圖即是自動產生的標準功能表項目，接下來就是對不需要的功能項目進行增刪或調整的工作。



10.1.4 如何設定功能表項目的屬性

建立功能表項目時，還可以設定一些具有輔助功能的屬性，使得選取功能表項目提高方便性。主要常用的屬性有：Enabled、Checked、ShortcutKeys、ShowShortcutKeys 及 Image。

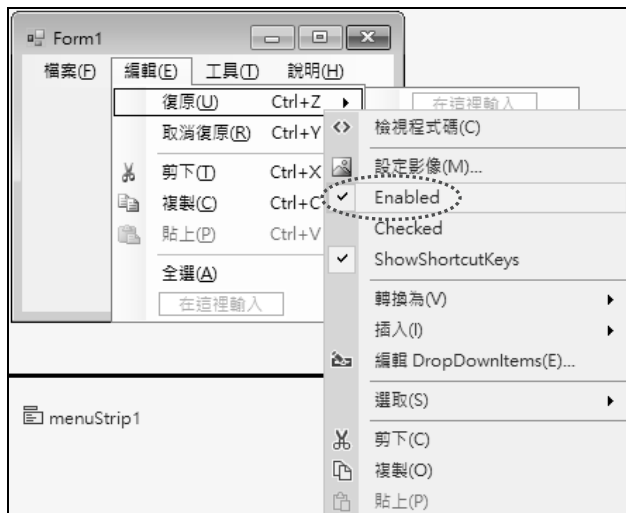


1. Enabled 屬性 (預設值：true)

用來設定該功能項目是否允許被點選。

- ① 若設為 true，表示該功能項目允許被點選(有效)。
- ② 若設為 false，該功能項目呈現淡灰色，
表示該功能項目在程式執行時不允許被點選(無效)。

如何在表單設計時設定此屬性值，可透過屬性視窗或在該功能項目上按滑鼠右鍵，由快顯功能表中點選「Enabled」項目來切換 Enabled 屬性值。
若 Enabled 屬性前出現 ☒ 勾號表示設為 true。



- 例** 在程式執行中將名稱為「復原 UToolStripMenuItem」的功能項目設為無效，寫法如下：

```
復原 UToolStripMenuItem.Enabled = false;
```

2. Checked 屬性 (預設值：false)

用來標示該功能項目是處於使用中或關閉狀態，當該功能項目前面出現 ☒ 勾號表示該屬性值為 true，代表該功能項目正在使用中。在表單設計階段時，可透過屬性視窗或在該功能項目上按滑鼠右鍵，由快顯功能表中點選【Checked】項目來切換 Checked 屬性值。當 Checked 屬性為 true 時，該項目前有打勾符號，代表該功能項目正在使用中。



- 例** 在程式執行中將「自訂 CToolStripMenuItem」功能項目前面加上核取記號，其寫法如下：

```
自訂 CToolStripMenuItem.Checked = false;
```

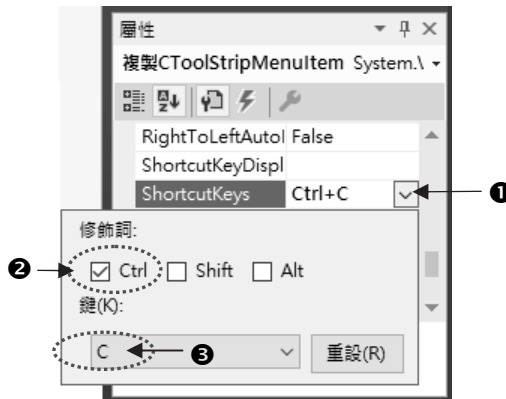
3. ShortcutKeys、ShowShortcutKeys (快速鍵、顯示快速鍵)屬性

ShortCutKeys 屬性允許在功能項目名稱後面加單一字母當快速鍵，譬如：「複製(C)」(鍵入方式為「複製(&C)」)，也可設定組合鍵(如 **Ctrl** + **C**) 快速鍵，讓操作者藉由按鍵方式取代滑鼠點選，以提升操作速度。在表單設計階段設定功能項目的快速鍵操作方式如下：

- ① 單一字母快速鍵：譬如「複製(C)」，設定方式是先點選功能項目，出現插入點閃爍游標再鍵入「複製(&C)」。此種模式在程式執行時可使用 **Alt** + **C** 快速鍵取代滑鼠點選。




- ② 組合鍵快速鍵：譬如 **Ctrl** + **C** 組合按鍵，設定方式是先選定功能項目，接著到屬性視窗中，按 `ShortcutKeys` 屬性欄的下拉式清單，從清單中選取所提供的鍵，快速鍵必須由一個輔助鍵即 **Ctrl**、**Shift** 或 **Alt**，加上一個單一鍵組成。如下圖所示：將「複製」項目設定了一個快速鍵 **Ctrl** + **C**。



當某個功能項目設定快速鍵後，可進入屬性視窗更改 `ShowShortcutKeys` 屬性來設定該功能項目是否顯示快速鍵。若設為 `true` 會顯示快速鍵(預設值為 `true`)。若要設為不顯示，可直接在功能項目上按滑鼠右鍵，由快顯功能表中選取「`ShowShortcutKeys`」選項，取消其前面的勾選，就可以不顯示快速鍵提示。



4. Image 小圖示

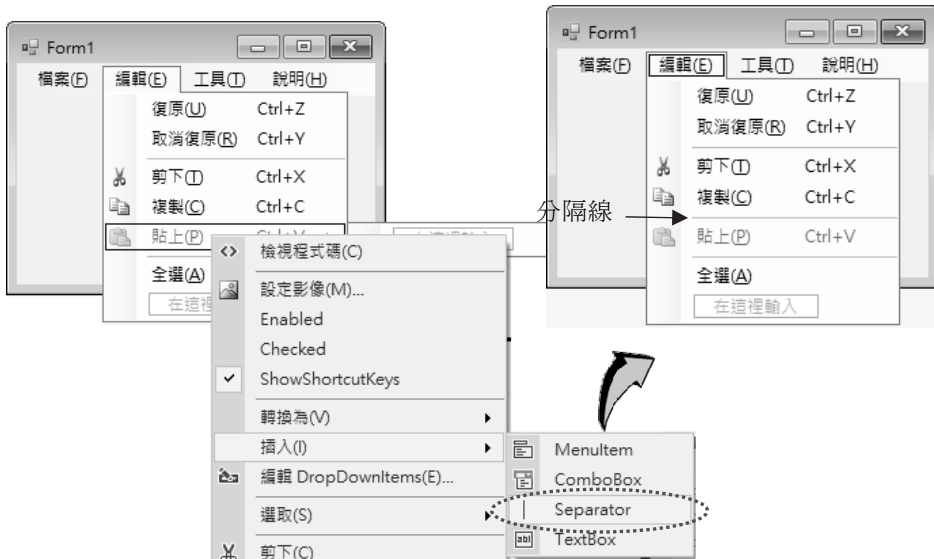
每個功能表項目的前面允許設定一個小圖示，以增加功能表的視覺效果，這個圖示是利用 Image 屬性來設定，其操作方式是選取功能項目後，到屬性視窗中點選 Image 屬性，按欄位右邊的  鈕出現「選取資源」對話方塊，選擇適當圖案。Image 屬性可接受的圖形檔格式有：gif、jpg、bmp、wmf 及 png 幾種，圖形檔的大小並沒有限制，圖片會自動縮小到適當的大小。但是建議還是先將圖片檔縮小，以減少專案檔的大小。

5. 分隔線

分隔線可將性質相近的功能項目放在一個區域內，彼此間以分隔線隔開，讓功能表更容易區別。在設計功能表時，在輸入項目名稱時鍵入「-」字元，該項目位置就會形成一條分隔線。



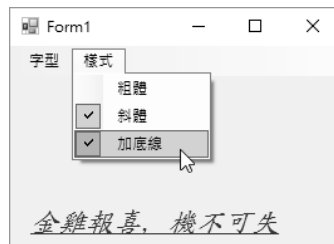
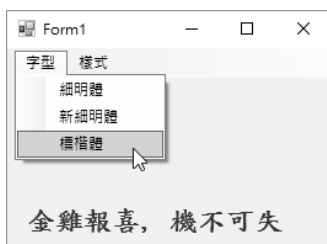
另一種方式是，只要先選取要放置分隔線的下一個功能項目，然後按滑鼠右鍵，由出現的快顯功能表中選擇【插入/Separator】，就會在該功能項目上面增加一條分隔線。



實作 FileName : MenuStrip.sln

設計一個含有「字型」和「樣式」兩個主功能項目的功能表，其中「字型」主功能項目用來更改字型種類：含有「細明體」、「新細明體」、「標楷體」子功能項目；「樣式」主功能項目用來設定字體樣式：含有「粗體」、「斜體」、「加底線」子功能項目，可多選。所有設定將對 label1 標籤控制項上面顯示『金雞報喜，機不可失』做字型 and 樣式設定。

► 輸出要求



► 解題技巧

Step 1 建立輸出入介面

1. 新增專案，新專案名稱為「MenuStrip」。
2. 依輸出要求，在表單建立下列控制項：
 - ① 將 label1 標籤控制項置入表單。
 - ② 將 menuStrip1 功能表控制項置入表單，建立下圖各個功能表選項。



Step 2 分析問題

1. 表單載入時，相關屬性設定值寫在表單的 Form1_Load()事件處理函式：
 - ① 在 label1 標籤控制項上面先顯示「金雞報喜，機不可失」。
 - ② 將 label1 標籤控制項的前景色設為紅色。
 - ③ 將 label1 標籤控制項的字型種類預設為細明體、大小設為 16。
 - ④ 預設樣式的粗體項目為勾選狀態。
2. 分別在細明體、新細明體、標楷體的子功能項目的 Click 事件處理函式內撰寫相對應的功能。例如：按「細明體」功能選項時將 label1 標籤的字型設為細明體，原字型樣式(label1.Font.Style)。按「標楷體」的功能選項時，將 label1 標籤的字型設為標楷體，原字型樣式。其他以此類推。

```
label1.Font = new Font("細明體", 16, label1.Font.Style);
```

3. 分別在粗體、斜體、加底線的子功能選項的 Click 事件處理函式內撰寫相對應的功能。例如：按「粗體」的功能項目時，將 label1 原字型樣式和粗體樣式作「互斥(^)」運算，若原為粗體作 ^ 運算後會改為非粗體。另外將 Checked 屬性作「非(!)」運算，若原為勾選，作「!」運算後會改為不勾選。

```
label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Bold);
粗體 ToolStripMenuItem.Checked = !(粗體 ToolStripMenuItem.Checked);
```

Step 3 編寫程式碼

```
FileName : MenuStrip.sln
01 namespace MenuStrip
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             label1.Text = "金雞報喜，機不可失";
13             label1.ForeColor = Color.Red;           // 設前景色為紅色
14             label1.Font = new Font("細明體", 16, FontStyle.Bold);
15             粗體 ToolStripMenuItem.Checked = true; // 預設粗體項目被勾選
16         }
17
18         private void 細明體 ToolStripMenuItem_Click(object sender, EventArgs e)
19         {
20             // 設字型為細明體、大小為 16、原字型樣式
21             label1.Font = new Font("細明體", 16, label1.Font.Style);
22         }
23
24         private void 新細明體 ToolStripMenuItem_Click(object sender, EventArgs e)
25         {
26             label1.Font = new Font("新細明體", 16, label1.Font.Style);
27         }
28     }
29 }
```

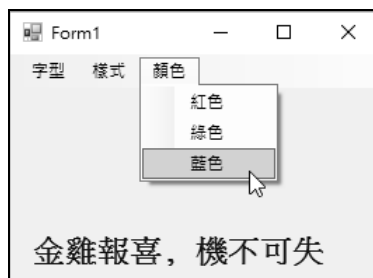
```

28
29     private void 標楷體 ToolStripMenuItem_Click(object sender, EventArgs e)
30     {
31         label1.Font = new Font("標楷體", 16, label1.Font.Style);
32     }
33
34     private void 粗體 ToolStripMenuItem_Click(object sender, EventArgs e)
35     {
36         label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Bold);
37         粗體 ToolStripMenuItem.Checked = !(粗體 ToolStripMenuItem.Checked);
38     }
39
40     private void 斜體 ToolStripMenuItem_Click(object sender, EventArgs e)
41     {
42         label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Italic);
43         斜體 ToolStripMenuItem.Checked = !(斜體 ToolStripMenuItem.Checked);
44     }
45
46     private void 加底線 ToolStripMenuItem_Click(object sender, EventArgs e)
47     {
48         label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Underline);
49         加底線 ToolStripMenuItem.Checked = !(加底線 ToolStripMenuItem.Checked);
50     }
51 }
52 }


```

► 馬上練習

修改上面範例在主功能選項加上「顏色」功能以及在新增的「顏色」新建「紅色」、「綠色」、「藍色」的子選項功能。按藍色的功能項目，則將 label1 標籤的字型顏色設為藍色，其他以此類推。



10.2 ContextMenuStrip 快顯功能表控制項

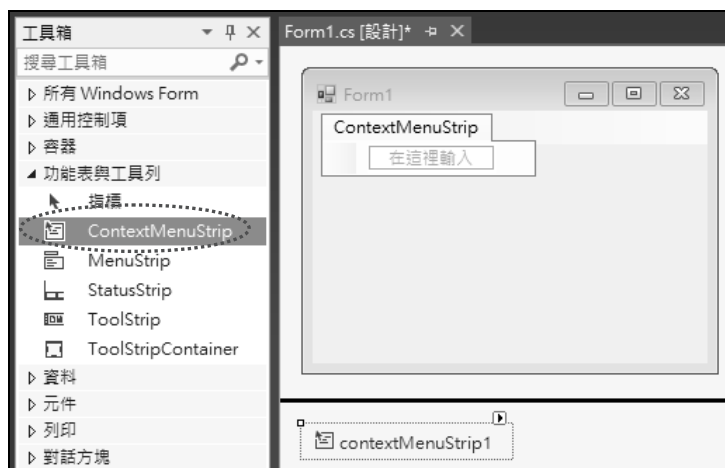
 ContextMenuStrip 快顯功能表控制項的外觀有點像 MenuStrip 的子功能表，在視窗應用程式中時常看到將某物件的常用功能置於快顯功能表中。當滑鼠在該物件上點按滑鼠右鍵，馬上很直覺式地出現快顯功能表，以方便選取功能選項。譬如：下圖，在表單的文字方塊上面，按滑鼠右鍵會出現如下圖的快顯功能表，供您選取功能項目。Visual C# 所提供的快顯功能表控制項，允許將一個快顯功能表控制項同時指定(連結)給表單上多個控制項共用。



10.2.1 如何建立快顯功能表的項目

快顯功能表的項目建立方式與 MenuStrip 控制項相同，先從工具箱中拖曳 ContextMenuStrip 工具到表單內。接下來設定輸入名稱方式都一樣，許多屬性的設定也一樣，相同的部分在本節中將不再贅述。下圖是快顯功能表輸入功能項目的步驟，輸入項目的方式和 MenuStrip 控制項相同。

Step 1 在下圖表單中先建立 contextMenuStrip1 控制項。



Step 2 在 contextMenuStrip1 內輸入快顯功能表的項目
依下圖所示，輸入「剪下」、「複製」、「貼上」三個功能項目。

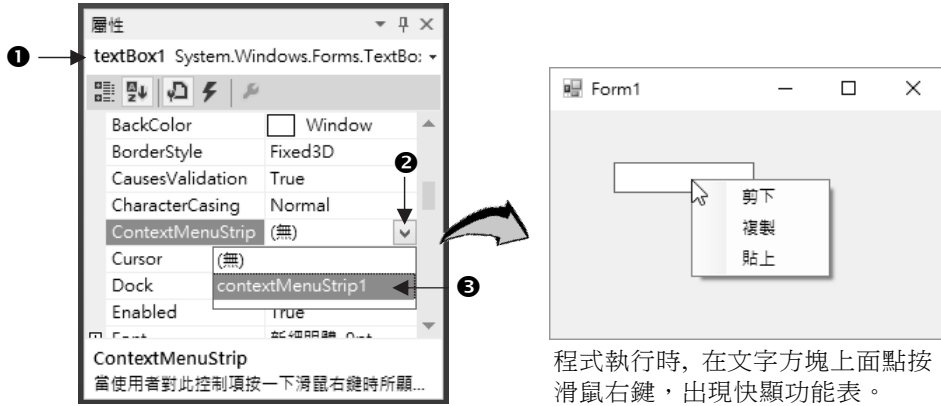


Step 3 點選功能項目(如「剪下」)，就可以到屬性視窗中設定屬性值。

10.2.2 如何將控制項與快顯功能表建立連結

當快顯功能表建立功能項目完畢後會發現表單上沒有任何東西出現，也沒有任何功能表，因為快顯功能表必須藉由表單或表單上面的控制項的 `ContextMenuStrip` 屬性來連結才會產生作用。幾乎每個控制項(包括表單)都含有 `ContextMenuStrip` 屬性，可以先點選需要快顯功能表的控制項，接著在該控制項的 `ContextMenuStrip` 屬性下拉鈕清單中選取使用哪個快顯功能表控制項。下

圖中的操作步驟是將文字方塊(textBox1)與 context MenuStrip1 控制項產生連結，也就是說當在 textBox1 文字方塊控制項上按滑鼠右鍵時會如右下圖出現 contextMenuStrip1 的快顯功能表。



實作 FileName: ContextMenuStrip.sln

延續上例新增一個快顯功能表。在 label1 標籤上按右鍵，可由「字型」快顯功能表選項下的子選項，設定 label1 標籤的字型種類。可由「樣式」快顯功能表選項下的子選項，設定 label1 標籤的字型樣式。

► 輸出要求



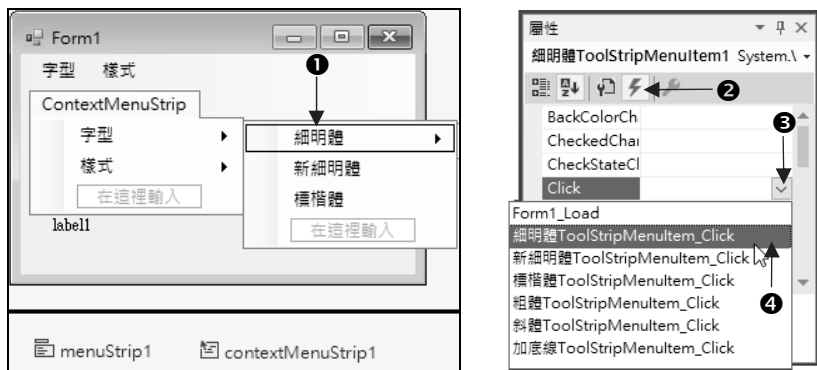
► 解題技巧

Step 1 建立輸出入介面

1. 承上例新增專案，但以 ContextMenuStrip 為新專案名稱。
2. 建立下面控制項以及設定相關屬性：
 - ① 將 contextMenuStrip1 功能表控制項放入表單。
 - ② 如下圖所示，建立各個快顯功能表項目。



- ③ 使用上節連結快顯功能表方法，將 label1 的 ContextMenuStrip 屬性設為 contextMenuStrip1，使得標籤控制項與此快顯功能表產生連結。
3. 設定 contextMenuStrip1 各個功能項目的事件
- 由於 menuStrip1 與 contextMenuStrip1 的各個功能項目所做的事情都一樣，可以用共享事件。以「細明體」功能目項為例，如下圖所示順序操作，讓「細明體 ToolStripMenuItem1」的 Click 事件共用「細明體 ToolStripMenuItem_Click」事件處理函式。



Step 2 分析問題

在粗體、斜體、加底線的子功能項目的 Click 事件處理函式內，增加快顯功能表各項目的勾選狀態程式。以「粗體」功能目項為例，如下：

```
label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Bold);
// 設定功能表列中粗體功能項目的勾選狀態
粗體 ToolStripMenuItem.Checked = !(粗體 ToolStripMenuItem.Checked);
// 設定快顯功能表列中粗體功能項目的勾選狀態
粗體 ToolStripMenuItem1.Checked = !(粗體 ToolStripMenuItem1.Checked);
```

Step 3 編寫程式碼（粗體字部分的程式是與上例不同處）

FileName : ContextMenuStrip.sln

```
01 namespace ContextMenuStrip
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             label1.Text = "金雞報喜，機不可失";
13             label1.ForeColor = Color.Red;           // 設前景色為紅色
14             label1.Font = new Font("細明體", 16, FontStyle.Bold);
15             粗體 ToolStripMenuItem.Checked = true;    // 預設粗體項目被勾選
16             粗體 ToolStripMenuItem1.Checked = true;  // 預設粗體項目被勾選
17         }
18
19         private void 細明體 ToolStripMenuItem_Click(object sender, EventArgs e)
20         {
21             // 設字型為細明體、大小為 16、原字型樣式
22             label1.Font = new Font("細明體", 16, label1.Font.Style);
23         }
24
25         private void 新細明體 ToolStripMenuItem_Click(object sender, EventArgs e)
26         {
27             label1.Font = new Font("新細明體", 16, label1.Font.Style);
28         }
```

```

29
30     private void 楷楷體 ToolStripMenuItem_Click(object sender, EventArgs e)
31     {
32         label1.Font = new Font("標楷體", 16, label1.Font.Style);
33     }
34
35     private void 粗體 ToolStripMenuItem_Click(object sender, EventArgs e)
36     {
37         label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Bold);
38         // 設定功能表列中粗體功能項目的勾選狀態
39         粗體 ToolStripMenuItem.Checked = !(粗體 ToolStripMenuItem.Checked);
40         // 設定快顯功能表列中粗體功能項目的勾選狀態
41         粗體 ToolStripMenuItem1.Checked = !(粗體 ToolStripMenuItem1.Checked);
42     }
43
44     private void 斜體 ToolStripMenuItem_Click(object sender, EventArgs e)
45     {
46         label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Italic);
47         斜體 ToolStripMenuItem.Checked = !(斜體 ToolStripMenuItem.Checked);
48         斜體 ToolStripMenuItem1.Checked = !(斜體 ToolStripMenuItem1.Checked);
49     }
50
51     private void 加底線 ToolStripMenuItem_Click(object sender, EventArgs e)
52     {
53         label1.Font = new Font(label1.Font, label1.Font.Style ^ FontStyle.Underline);
54         加底線 ToolStripMenuItem.Checked = !(加底線 ToolStripMenuItem.Checked);
55         加底線 ToolStripMenuItem1.Checked = !(加底線 ToolStripMenuItem1.Checked);
56     }
57 }
58 }

```

► 馬上練習

修改上面實作，增加顏色功能項目，以及紅色、綠色、藍色的快選功能項目。當按下綠色的功能項目，則將 label1 標籤的字型顏色設為綠色，其他以此類推。



10.3 ToolStrip 工具列控制項

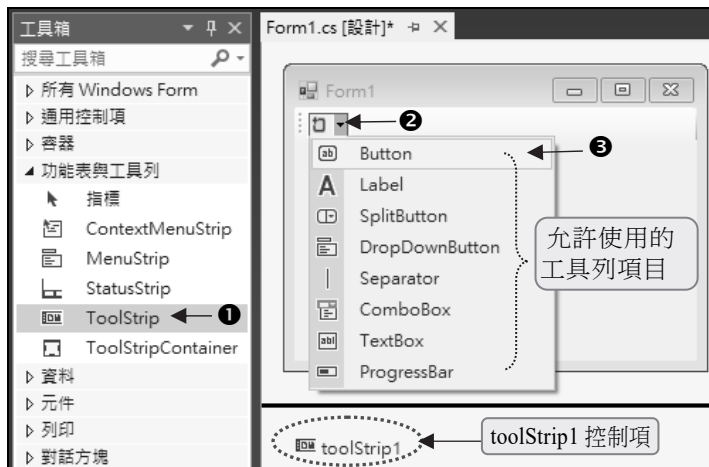
在 Windows 環境下的應用程式，常將一些常用的功能項目以按鈕圖示組合成一個工具列，掛在功能表的正下方以方便快速選取常用的功能項目。譬如您目前正在使用 Visual C# 的整合開發環境的標準工具列即是如此。

10.3.1 如何建立自訂工具列的項目

透過在工具箱中的 **ToolStrip** 工具，可以輕易地依需求自訂一個工具列。ToolStrip 控制項預設放在表單的正上方。ToolStrip 和表單一樣亦是一個控制項容器，它是由一些按鈕、標籤、下拉式按鈕、文字方塊所組合而成的集合。至於建立 ToolStrip 工具列控制項的操作方式如下：

Step 1 建立 ToolStrip 工具列控制項

在工具箱的 **ToolStrip** 工具上快按兩下，會在表單標題欄的正下方產生一個空白的工具列。



Step 2 建立工具列的項目

ToolStrip 允許使用的工具項目如上圖所示。在工具列的 **ToolStrip** 下拉鈕按一下，由清單中選取需要的項目。上圖即選取 **Button** 按鈕工具項目，表示使用按鈕當自訂工具列的第一個項目。

Step 3 設定項目屬性

先點選要設定的工具項目，屬性視窗就會顯示該項目的屬性。例如 Image 屬性可設定工具項目中顯示的圖示、Text 屬性可以設定顯示的文字。

10.3.2 ToolStrip 工具列控制項常用的屬性

1. Items 屬性

是 ToolStrip 控制項中工具列項目的集合。


2. Dock 屬性

設定工具列在表單的位置，預設值為 Top(在表單的上方)。

例 設定 toolStrip1 工具列控制項安置在表單下方，程式碼寫法如下：

```
toolStrip1.Dock = DockStyle.Bottom;
```

10.3.3 工具列項目常用的屬性

ToolStrip 是一個控制項容器，其中可以安置按鈕、標籤、下拉式按鈕、文字方塊等不同種類的工具項目。不同的工具項目的屬性不同，本書只介紹 ToolStrip 控制項  Button 按鈕的常用屬性：

1. Text 屬性

是 ToolStrip 控制項中工具列項目的集合。

2. Image 屬性

設定工具項目上顯示的圖示。

3. DisplayStyle 屬性

設定文字和圖示的顯示狀態，其值有 None、Text、Image(預設值)和 ImageAndText。

4. TextImageRelation 屬性

設定文字和圖示的相對位置，預設值為 ImageBeforeText，表示圖示在文字前面。

5. ToolTipText 屬性

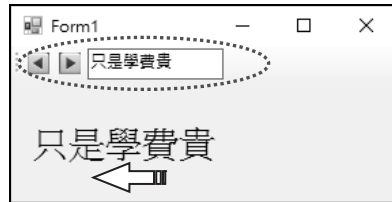
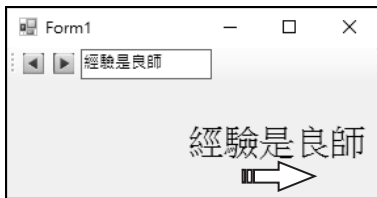
設定當使用者滑鼠移到工具項目上，所顯示的提示文字。



實作 FileName : RunWord.sln

製作一個跑馬燈程式。開始執行時跑馬燈會以工具列文字方塊項目預設的「經驗是良師」跑馬燈文字置於標籤控制項上面，每隔 0.1 秒由左向右移動 10 點。工具列含有三個項目，工具列的文字方塊項目允許改變跑馬燈的文字，當在工具列按 ► 鈕文字會由左向右移動、按 ◀ 鈕改為由右向左移動。

► 輸出要求



► 解題技巧

Step 1 建立輸出入介面

1. 新增專案並以「RunWord」為新專案名稱。
2. 建立輸出入介面及設定相關屬性：
 - ① 建立 toolStrip1 控制項，在上面新增兩個按鈕(圖檔在書附光碟 ch10/images 資料夾)和一個文字方塊項目。
 - ② 建立 timer1 計時器控制項每隔 0.1 秒來移動標籤控制項上面設定的跑馬燈文字。



Step 2 分析問題

1. 宣告 `move_d` 為表單 `Form1` 的成員(全域)變數，以便讓所有事件處理函式共用。`move_d` 變數用來記錄跑馬燈文字移動方向，其資料型別為 `Boolean`，初值設為 `true` 表示目前跑馬燈方向是由左向右移動。若設為 `false` 表示跑馬燈方向是由右向左移動。
2. 在表單載入的 `Form1_Load` 事件處理函式中必須先啟動 `timer1` 計時器，將 `Enabled` 屬性設為 `true` 啟動計時器。
3. 由於每隔 0.1 秒(`timer1.Interval = 100`)必須依 `move_d` 布林變數判斷跑馬燈文字的移動方向，將在 `timer1_Tick` 事件處理函式中根據 `move_d` 值來設定 `lblMsg` 的 `Left` 屬性值。每次移動 10 Pixels，若跑出表單邊界，由表單另一端邊界跑入。程式碼寫法如下：

```
if (move_d == true)    // true 由左向右移
{
    lblMsg.Left += 10;
    if (lblMsg.Left >= this.Width) lblMsg.Left = -lblMsg.Width;
}
else                  // false 由左向右移
{
    lblMsg.Left -= 10;
    if (lblMsg.Left <= -lblMsg.Width) lblMsg.Left = this.Width;
}
```

4. 程式執行中當在 `toolStripTextBox1` 工具列上的文字方塊上更改文字會觸動該項目的 `TextChanged` 事件，在該事件中將工具列上更改的跑馬燈文字指定給 `lblMsg` 標籤控制項。

```
lblMsg.Text = toolStripTextBox1.Text;
```

Step 3 編寫程式碼**FileName : RunWord.sln**

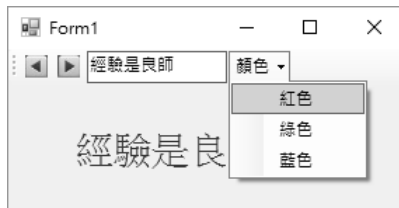
```
01 namespace RunWord
02 {
03     public partial class Form1 : Form
04     {
```

```
05     public Form1()
06     {
07         InitializeComponent();
08     }
09
10     bool move_d = true;    // 記錄跑馬燈文字移動方向
11
12     private void Form1_Load(object sender, EventArgs e)
13     {
14         timer1.Enabled = true;
15         timer1.Interval = 100;
16         toolStripTextBox1.Text = "經驗是良師";
17     }
18
19     private void timer1_Tick(object sender, EventArgs e)
20     {
21         if (move_d == true)    // true 由左向右移
22         {
23             LblMsg.Left += 10;
24             if (LblMsg.Left >= this.Width) LblMsg.Left = -LblMsg.Width;
25         }
26         else    //false 由左向右移
27         {
28             LblMsg.Left -= 10;
29             if (LblMsg.Left <= -LblMsg.Width) LblMsg.Left = this.Width;
30         }
31     }
32
33     private void toolStripButton1_Click(object sender, EventArgs e)
34     {
35         move_d = false;
36     }
37
38     private void toolStripButton2_Click(object sender, EventArgs e)
39     {
40         move_d = true;
41     }
42
43     private void toolStripTextBox1_TextChanged(object sender, EventArgs e)
44     {
45         LblMsg.Text = toolStripTextBox1.Text;
46     }
```

```
47     }
48 }
```

► 馬上練習

修改上面實作，增加一個 DropDownButton 項目，其中有紅色、綠色、藍色三個項目，可以設定跑馬燈文字顏色。



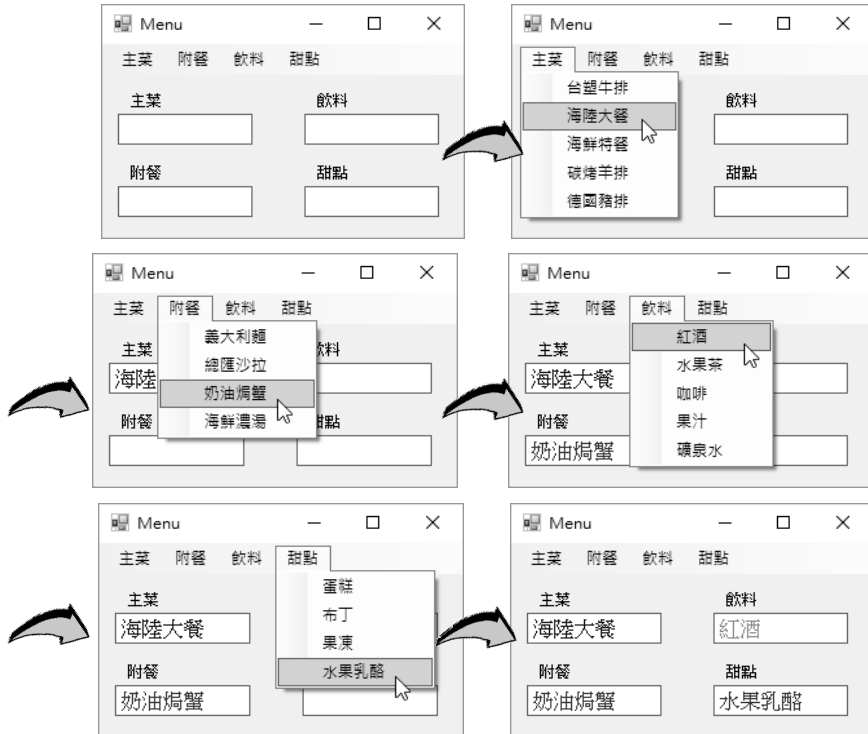
► 10.4 習題

一、選擇題

- 控制項要與快顯功能表建立關聯必須使用下列那個屬性？
(A) Text (B) DataSource (C) Color (D) ContextMenuStrip
- 欲建立工具列，必須使用下列哪個控制項來建立？
(A) ToolBar (B) ToolStrip (C) ToolMenu (D) MenuStrip
- 工具列控制項的項目欲同時顯示文字和圖示，必須將 DisplayStyle 屬性設為？
(A) None (B) Text (C) Image (D) ImageAndText
- 功能表項目是什麼物件？
(A) MenuBox (B) MenuItem (C) MenuButton (D) Button
- 欲在功能表項目中加入一個分隔線，可在功能表按下右鍵，並執行快顯功能表？
(A) [新增/分隔線] (B) [插入/分隔線]
(C) [新增/Separator] (D) [插入/Separator]
- 工具列控制項的項目只要顯示文字，必須將 DisplayStyle 屬性設為？
(A) None (B) Text (C) Image (D) ImageAndText

二、程式設計

- 設計一個餐廳點餐系統。系統中有一個功能表，有四個主功能項目，其下包含子功能項目。安排四個標籤，當選擇每一個主功能下的子功能項目，要在對應的標籤控制項中，顯示所選功能項目名稱。請參考下圖：



- 製作如下圖的圖片瀏覽器程式，在圖片方塊控制項按右鍵出現快顯功能表，您可以透過快顯功能表的「第一張」、「下一張」、「上一張」、「最上一張」功能選項來切換圖檔。

