

第十二章 對話方塊與功能表控制項



12.1 功能表控制項

12.2 快顯功能表控制項

12.3 工具列控制項

12.4 狀態列控制項

12.5 字型對話方塊控制項

12.6 色彩對話方塊控制項

12.7 檔案對話方塊控制項

12.8 列印文件控制項

12.9 列印格式對話方塊控制項

12.10 預覽列印對話方塊控制項

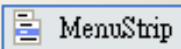
12.11 列印對話方塊控制項

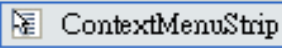
備註：可依進度點選小節

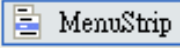


12.1 功能表控制項

■ 工具箱提供兩個和功能表相關的工具，分別是

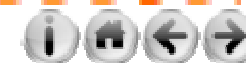
①  功能表

②  快顯功能表

■  控制項允許在設計階段或執行階段中建立功能表。

■ 一般功能表都是在設計階段便已經事先設計好，程式執行時才使用。

■ 將功能表上面建立的「功能選項」稱為「MenuItem」物件。

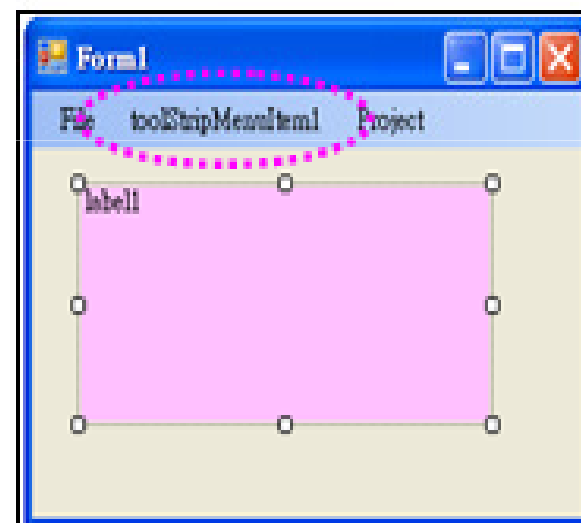
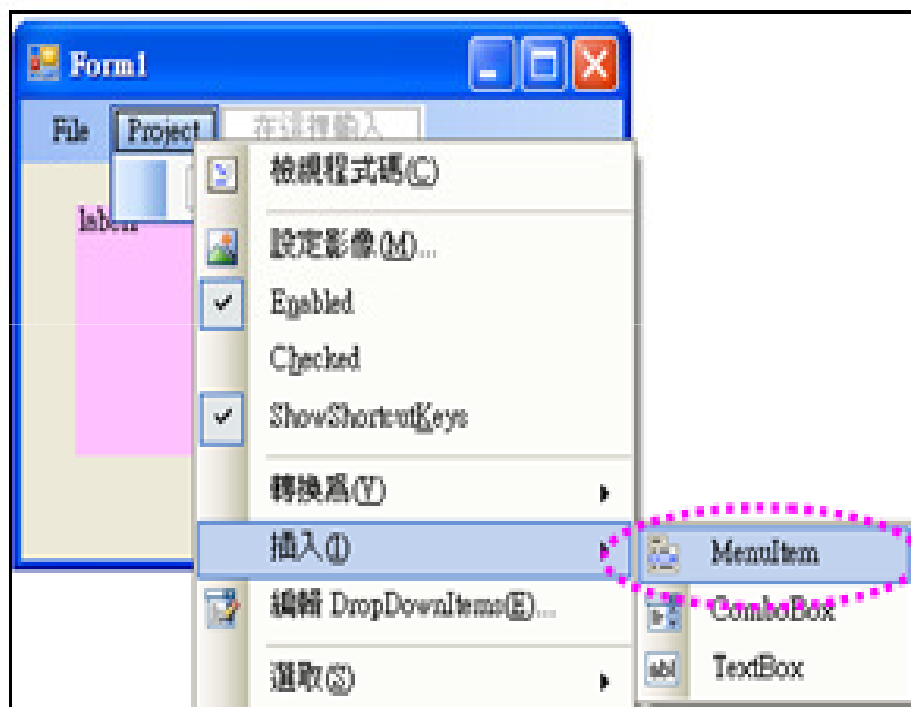


一. 如何建立 MenuStrip 控制項






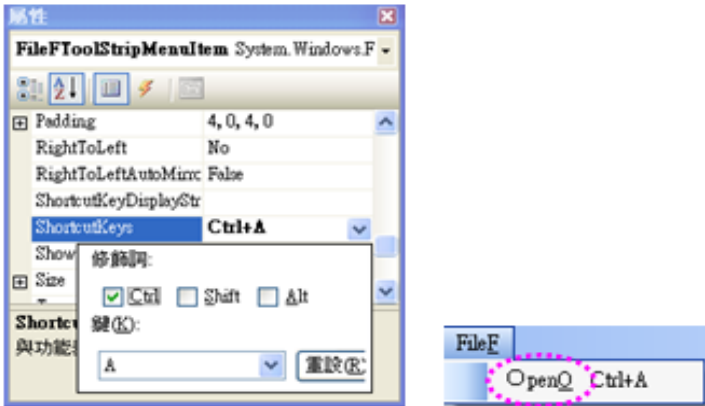


插入功能選項





二. MenuItem 常用屬性

屬性	說明
Text	功能項目上面顯示的文字內容。
Checked	設定功能項目前是否顯示 <input checked="" type="checkbox"/> ，若屬性值為 False 表不顯示；True 時會顯示。例如： 
ShowShortcutKeys	設定是否將功能表項目的快速鍵顯示在該項目上。預設值為 True。
ShortcutKeys	可由清單中選擇功能鍵做為快捷鍵，譬如將 Open 功能選項加上  +  當快捷鍵。 



三. MenuItem 常用事件

Click 事件

當使用者點選功能項目時，會觸動該項目的 **Click** 事件，將執行該功能的程式碼寫在 **Click** 事件處理函式中。

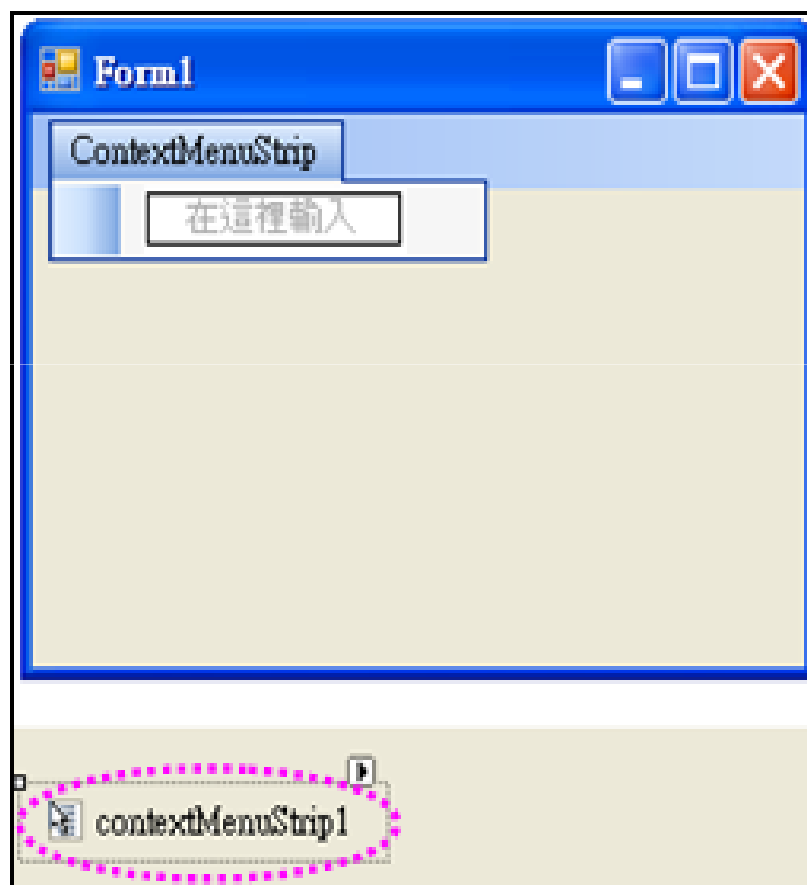


12.2 快顯功能表控制項

- **ContextMenuStrip** 快顯(快捷鍵)功能表控制項，可在程式執行時在特定控制項上面按右鍵，出現快顯功能表。
- 快顯功能表中列出該控制項常用功能，不必點選功能表就能執行功能選項，對使用者而言是非常貼心的設計。
- 建立快顯功能表控制項和功能表控制項方式大致一樣。



一. 如何建立ContextMenuStrip 控制項





二. MenuItem 常用方法

Add方法

ContextMenuStrip 控制項透過 **Items** 集合的 **Add** 方法在執行階段新增選項。

 譬如在 **contextMenuStrip1** 控制項中加入 “內容” 選項，寫法：

```
contextMenuStrip1.Items.Add("內容");
```



三. MenuItem 常用事件

1. Opening 事件

當使用者按右鍵要顯示快顯功能表前，會觸動該控制項所對應 ContextMenuStrip 控制項的 Opening 事件。

此時功能表尚未開啟，若希望在有指定 contextMenuStrip1 控制項的 button1 按鈕上，將選項名稱為 openToolStripMenuItem 功能選項設為失效，寫法：

```
private void contextMenuStrip1_Opening(object sender, CancelEventArgs e)
{
    openToolStripMenuItem.Enabled = false;
}
```

2. Click 事件

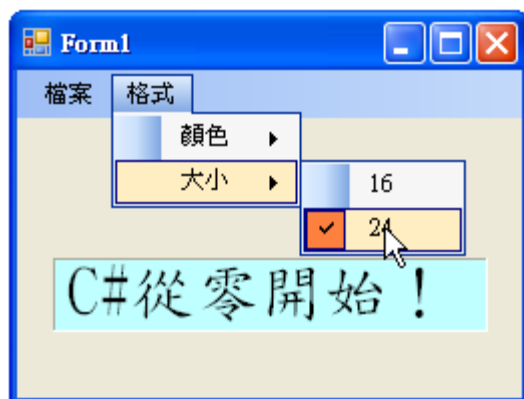
當使用者點選功能項目時，會觸動該項目的 Click 事件，會將執行該功能的程式碼寫在 Click 事件處理函式中。



範例演練

設計一個功能表列提供「檔案」和「格式」兩個主功能：

- ① 檔案主功能表下有 結束子選項，
- ② 格式主功能下有 顏色和大小兩個子功能。
- ③ 顏色子功能下有 黑色和 紅色兩個選項。
- ④ 大小子功能下有「16」和「24」兩個選項。
- ⑤ 當功能表選項有異動時，表單上面的標籤控制項內的「C#從零開始」的對應屬性亦跟著改變。
- ⑥ 在標籤控制項上製作一個快顯功能表具有顏色和大小子功能。





1. menuStrip1 各功能項目 Name 物件名稱。



結束 ToolStripMenuItem

lblTitle



黑色 ToolStripMenuItem

紅色 ToolStripMenuItem

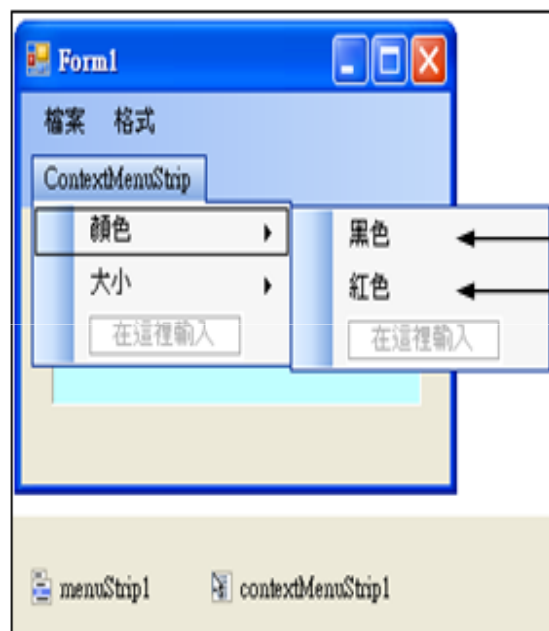


字大小 16 ToolStripMenuItem

字大小 24 ToolStripMenuItem



2. contextMenuStrip1 各功能項目 Name 物件名稱



C 黑色 ToolStripMenuItem

C 紅色 ToolStripMenuItem



C 字大小 16 ToolStripMenuItem

C 字大小 24 ToolStripMenuItem



// FileName : MenuStripDemo.sln

```
01 private void Form1_Load(object sender, EventArgs e)
02 {
03     lblTitle.Font = new Font("標楷體", 16);
04     lblTitle.Text = "C#從零開始！";
05     黑色ToolStripMenuItem.Checked = true;
06     C黑色ToolStripMenuItem.Checked = true;
07     字大小16ToolStripMenuItem.Checked = true;
08     C字大小16ToolStripMenuItem.Checked = true;
09     lblTitle.ContextMenuStrip = contextMenuStrip1;
11     黑色ToolStripMenuItem.Click += new EventHandler(Black_Click);
12     C黑色ToolStripMenuItem.Click += new EventHandler(Black_Click);
14     紅色ToolStripMenuItem.Click += new EventHandler(Red_Click);
15     C紅色ToolStripMenuItem.Click += new EventHandler(Red_Click);
17     字大小16ToolStripMenuItem.Click += new EventHandler(Size16_Click);
18     C字大小16ToolStripMenuItem.Click += new EventHandler(Size16_Click);
20     字大小24ToolStripMenuItem.Click += new EventHandler(Size24_Click);
21     C字大小24ToolStripMenuItem.Click += new EventHandler(Size24_Click);
22 }
```



```
24 private void 結束ToolStripMenuItem_Click(object sender, EventArgs e)
25 {
26     Application.Exit();
27 }
28
29 private void Black_Click(object sender, EventArgs e)
30 {
31     黑色ToolStripMenuItem.Checked = true ;
32     C黑色ToolStripMenuItem.Checked = true;
33     紅色ToolStripMenuItem.Checked = false ;
34     C紅色ToolStripMenuItem.Checked = false ;
35     lblTitle.ForeColor = Color.Black;
36 }
37
38 private void Red_Click(object sender, EventArgs e)
39 {
40     黑色ToolStripMenuItem.Checked = false ;
41     C黑色ToolStripMenuItem.Checked = false ;
42     紅色ToolStripMenuItem.Checked = true ;
43     C紅色ToolStripMenuItem.Checked = true ;
44     lblTitle.ForeColor = Color.Red ;
45 }
```





```
47 private void Size16_Click(object sender, EventArgs e)
48 {
49     字大小16ToolStripMenuItem.Checked = true;
50     C字大小16ToolStripMenuItem.Checked = true;
51     字大小24ToolStripMenuItem.Checked = false ;
52     C字大小24ToolStripMenuItem.Checked = false ;
53     lblTitle.Font = new Font("標楷體", 16);
54 }
55
56 private void Size24_Click(object sender, EventArgs e)
57 {
58     字大小16ToolStripMenuItem.Checked = false ;
59     C字大小16ToolStripMenuItem.Checked = false ;
60     字大小24ToolStripMenuItem.Checked = true ;
61     C字大小24ToolStripMenuItem.Checked = true ;
62     lblTitle.Font = new Font("標楷體", 24);
63 }
```



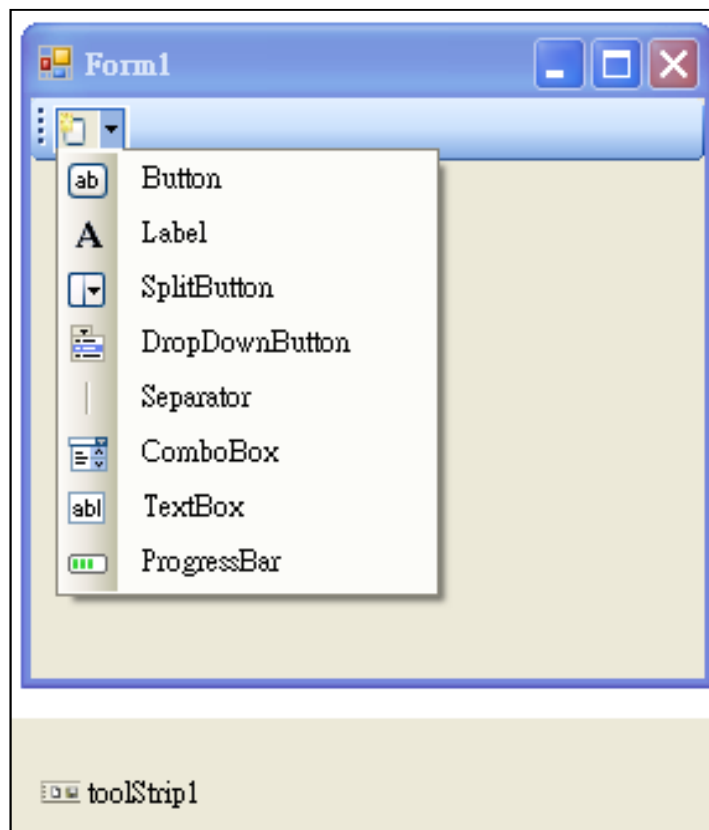

12.3 工具列控制項



- 一般較大型視窗應用程式除提供功能表外，還提供工具列以供快速操作。
- 在工具箱提供視覺化的  工具列控制項及可顯示目前系統狀態的  狀態列控制項。
- ToolStrip 工具列有別於 MenuStrip 功能表。
- 它是以視覺化的圖示功能表來取代文字功能表，感覺比較親切和直覺。
- 透過 ToolStrip 可輕鬆自訂且經常使用的工具列，亦支援進階使用者介面和配置功能，如：停駐、浮動定位、具有文字和影像的按鈕、下拉式控制項、下拉式按鈕及 ToolStrip 項目的執行階段重新排序。

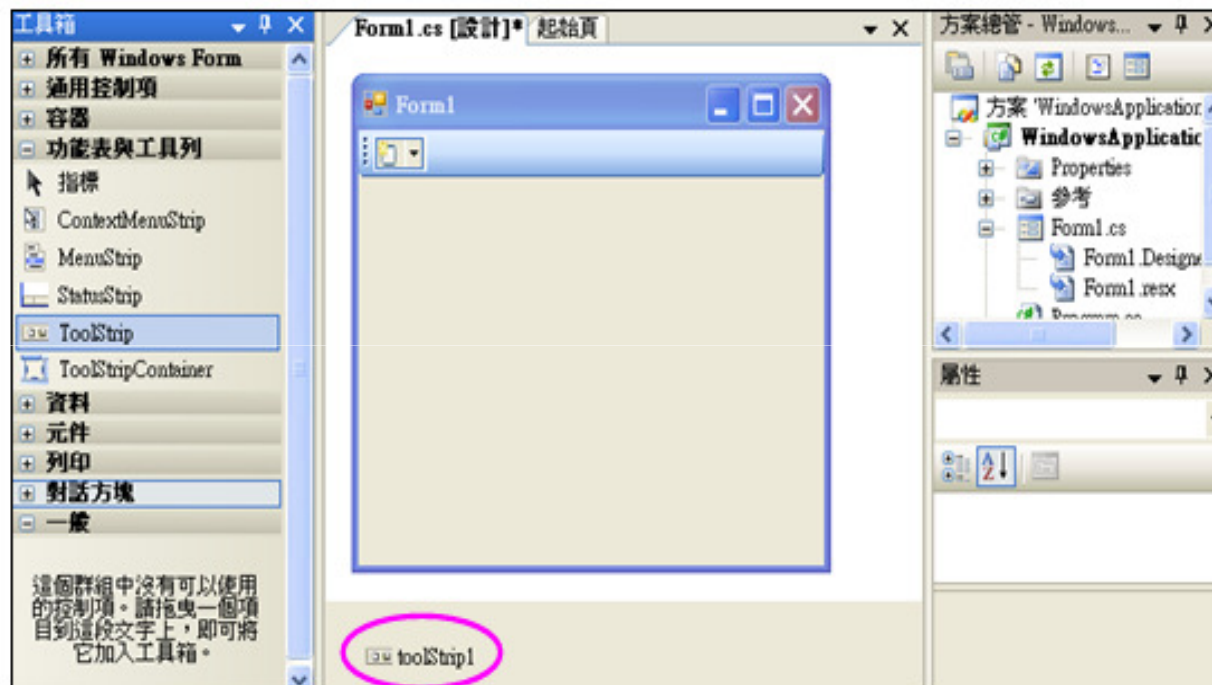


- 由於 ToolStrip 控制項是屬於容器控制項，可容納 Button、Label、SplitButton、DropDownButton、Seperator、ComboBox、TextBox、ProgressBar 等 ToolStrip 控制項提供的物件。





一. 如何建立 ToolStrip 控制項



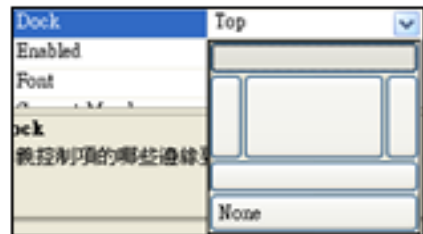


二. ToolStripButton 工具圖示常用屬性

屬性	說明
Text	設定工具列上面的按鈕工具所顯示的文字  。
TextDirection	設定工具圖示上面文字的方向。
TextImageRelation	設定文字和圖片的相關位置。其值為： ① ImageBeforeText：表示圖片在文字之前  ，為預設值。 ② TextAboveImage：文字在圖示上面  。 ③ TextBeforeImage：表示文字在圖示之前  。 ④ ImageAboveText：表示圖示在文字之上  。
Image	設定在工具列圖示上面的圖案。
ToolTipText	設定當滑鼠移到工具列按鈕上顯示的提示文字。 



三. ToolStrip 常用屬性

屬性	說明
Items	設定 ToolStrip 控制項中工具列按鈕的集合。
Size	設定工具列的大小，預設值為 (292,39)。
Dock	設定 ToolStrip 控制項在表單的位置，預設值為 Top (在表單的上方) 



四. ToolStrip 常用事件

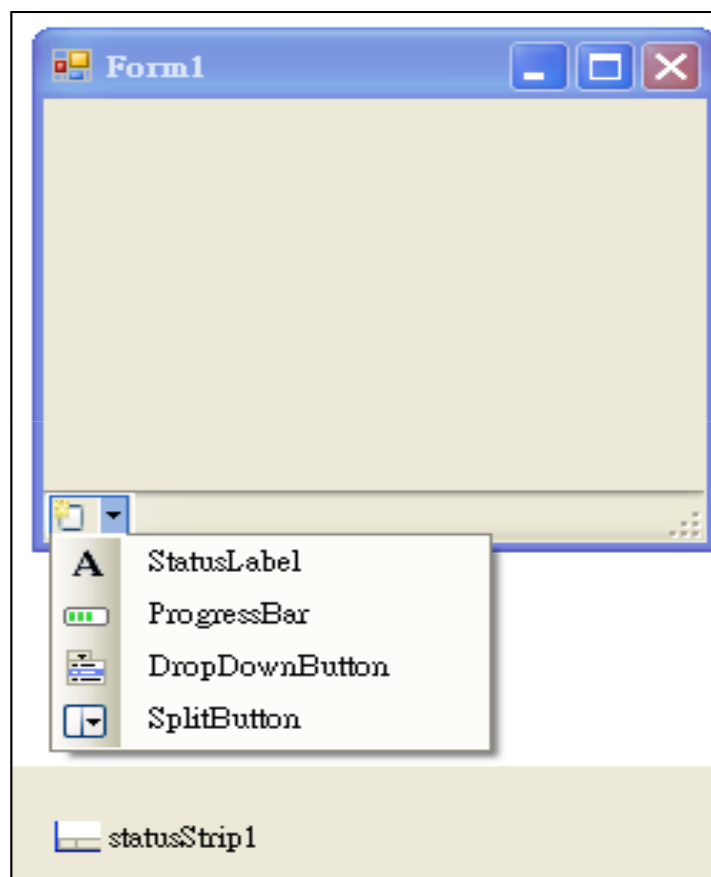
Click 事件

當使用者在 ToolStrip 控制項的工具圖示上按一下，會觸動該圖示名稱的 Click 事件，可將要處理事情的程式碼寫在該事件處理函式內。

 若有些圖示作相同的事情可透過共享事件方式來處理。



12.4 狀態列控制項





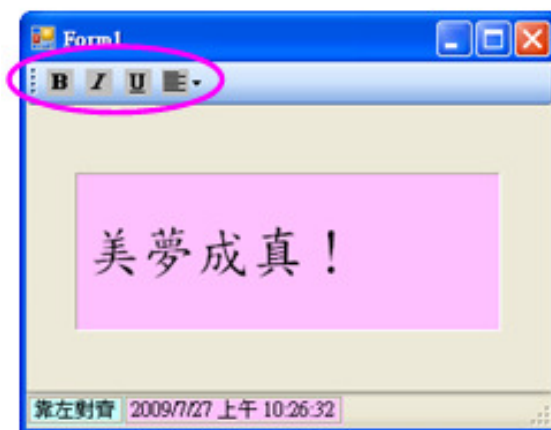
二. StatusStrip 常用屬性

屬性	說明
Items	設定 StatusStrip 控制項中狀態列上面的元件集合。
Visible	設定狀態列中是否顯示狀態列，預設值為 False，屬性值必須設為 True 才能看見狀態列面板。
Dock	設定狀態列在表單的位置，預設值為 Bottom（在表單的下方）。
ShowItemToolTips	是否在狀態列的項目上顯示提示訊息



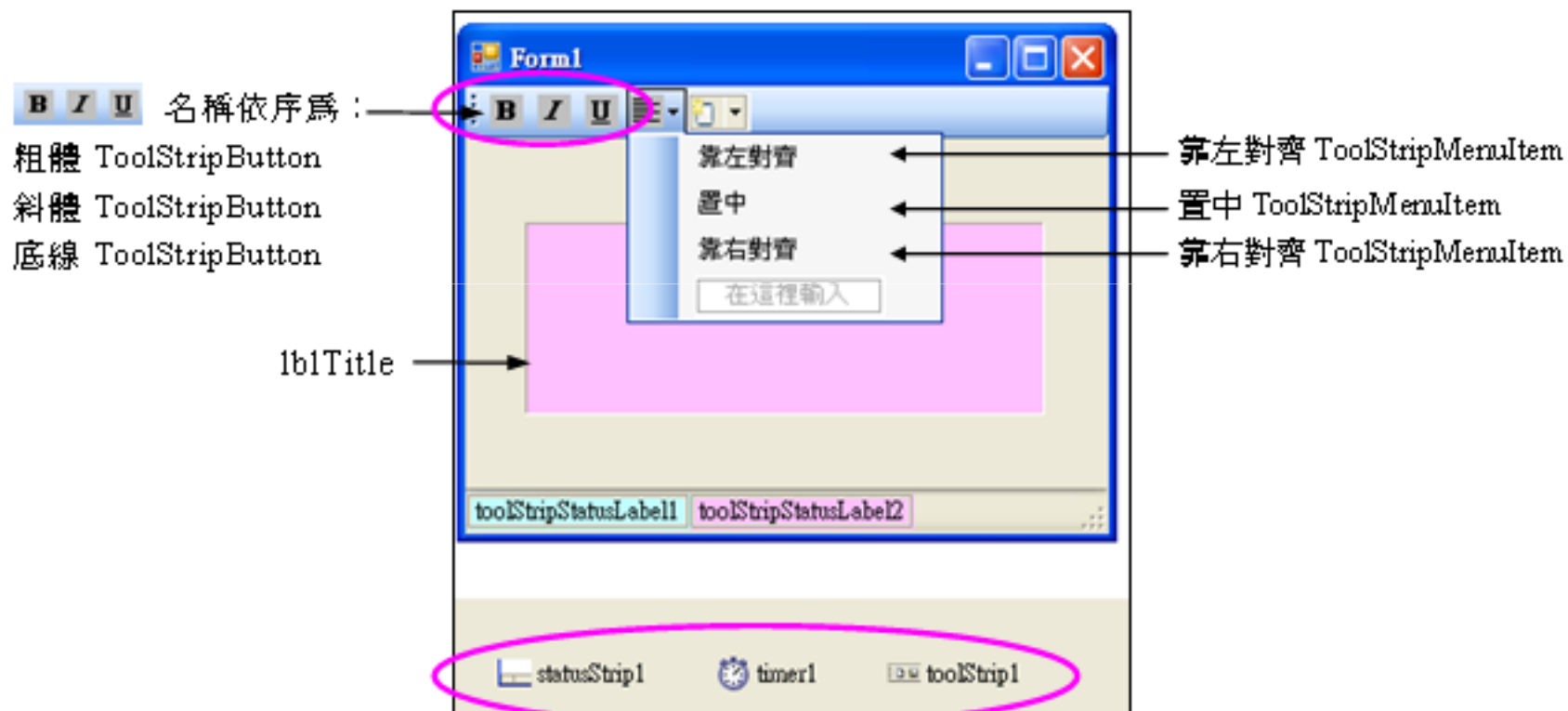
範例演練

在視窗上有 ToolStrip 工具列，工具列有四個按鈕分別是「粗體」、「斜體」、「加底線」、「對齊」功能來調整文字的顯示樣式。另外「對齊」有下拉式功能表，其中有「靠左對齊」、「置中對齊」、「靠右對齊」等三個功能。在視窗底下有 StatusStrip 狀態列，顯示目前文字的樣式和今天日期和目前時間。表單中間的標籤控制項 ” 美夢成真！” 文字會顯示設定效果。





Step1 設計輸出入介面





FileName : ToolStripDemo.sln

01

03 string bold, italic, underline, align;

04

05 private void Form1_Load(object sender, EventArgs e)

06 {

07 lblTitle.Text = "美夢成真！";

08 lblTitle.TextAlign = ContentAlignment.BottomLeft;

09 lblTitle.Font = new Font("標楷體", 24);

10 粗體ToolStripButton.ToolTipText = "粗體";

11 斜體ToolStripButton.ToolTipText = "斜體";

12 底線ToolStripButton.ToolTipText = "底線";

13 toolStripDropDownButton1.ToolTipText = "對齊";

14 粗體ToolStripButton.CheckOnClick = true;

15 斜體ToolStripButton.CheckOnClick = true;

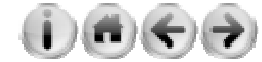
16 底線ToolStripButton.CheckOnClick = true;



```
17  toolStripStatusLabel1.BorderSides = ToolStripStatusLabelBorderSides.All;
18  toolStripStatusLabel2.BorderSides = ToolStripStatusLabelBorderSides.All;
19  timer1.Interval = 1000;
20  timer1.Enabled = true;
21  bold = "";
22  italic = "";
23  underline = "";
24  align = "靠左對齊";
25  toolStripStatusLabel1.Text = align;
26  toolStripStatusLabel2.Text = DateTime.Now.ToString();
27  lblTitle.TextAlign = ContentAlignment.MiddleLeft;
28 }
29
```



```
30 private void timer1_Tick(object sender, EventArgs e)
31 {
32     toolStripStatusLabel2.Text = DateTime.Now.ToString();
33 }
34
35 private void 粗體ToolStripButton_Click(object sender, EventArgs e)
36 {
37     lblTitle.Font = new Font("標楷體", 24, (FontStyle.Bold ^ lblTitle.Font.Style));
38     italic = 粗體ToolStripButton.CheckState == CheckState.Checked ? "粗體" : "";
39     toolStripStatusLabel1.Text = bold + " " + italic + " " + underline + " " + align;
40 }
41
42 private void 斜體ToolStripButton_Click(object sender, EventArgs e)
43 {
44     lblTitle.Font = new Font("標楷體", 24, (FontStyle.Italic ^ lblTitle.Font.Style));
45     bold = 斜體ToolStripButton.CheckState == CheckState.Checked ? "斜體" : "";
46     toolStripStatusLabel1.Text = bold + " " + italic + " " + underline + " " + align;
47 }
```



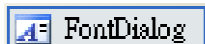
```
49 private void 底線ToolStripButton_Click(object sender, EventArgs e)
50 {
51     lblTitle.Font = new Font("標楷體", 24, (FontStyle.Underline ^ lblTitle.Font.Style));
52     underline = 底線ToolStripButton.CheckState == CheckState.Checked ? "加底線" : "";
53     toolStripStatusLabel1.Text = bold + " " + italic + " " + underline + " " + align;
54 }
55
56 private void 靠左對齊ToolStripMenuItem_Click(object sender, EventArgs e)
57 {
58     lblTitle.TextAlign = ContentAlignment.MiddleLeft;
59     align = "靠左對齊";
60     toolStripStatusLabel1.Text = bold + " " + italic + " " + underline + " " + align;
61 }
62
```



```
63 private void 置中ToolStripMenuItem_Click(object sender, EventArgs e)
64 {
65     lblTitle.TextAlign = ContentAlignment.MiddleCenter;
66     align = "置中對齊";
67     toolStripStatusLabel1.Text = bold + " " + italic + " " + underline + " " + align;
68 }
69
70 private void 靠右對齊ToolStripMenuItem_Click(object sender, EventArgs e)
71 {
72     lblTitle.TextAlign = ContentAlignment.MiddleRight;
73     align = "靠右對齊";
74     toolStripStatusLabel1.Text = bold + " " + italic + " " + underline + " " + align;
75 }
```



12.5 字型對話方塊控制項



- 字型對話方塊控制項圖提供設定字型的種類、字型的樣式、字型的大小、字型的效果等功能。
- 屬於幕後執行的控制項，置於表單的正下方。
- 程式執行中，欲開啟字型對話方塊，使用 ShowDialog 方法來開啟。





一. FontDialog 常用屬性

屬性	說明
Font	設定和取得在字型對話方塊所做的字型各項設定。譬如：欲將 fontDialog1 字型對話方塊所設定字型種類、樣式、大小指定給 label1 標籤控制項，寫法如下： <code>label1.Font = fontDialog1.Font;</code>
Color	設定和取得字型對話方塊中所設定的顏色。譬如：欲將 fontDialog1 字型對話方塊中指定的顏色當做 label1 標籤控制項的字型顏色，寫法如下： <code>label1.ForeColor = fontDialog1.Color;</code>
MaxSize / MinSize	設定字型對話方塊中，字型大小可以選取的最大和最小點數。預設值為 0 代表停用。
ShowColor	設定字型對話方塊中，是否加入色彩清單，預設值為 False 表示未加入色彩清單。



二. FontDialog 常用方法

1. ShowDialog 方法

用來在執行中開啟 字型對話方塊，平時是不顯現。

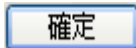
開啟方式：

語法：`fontDialog1.ShowDialog();`

若程式中有使用到 `fontDialog1` 控制項，可透過對話方塊中的 、 這兩個回應鈕來判斷是否要更新？

① 如按  鈕，傳回值為 `DialogResult.OK` 列舉常數；

② 如按  鈕，傳回值為 `DialogResult.Cancel` 列舉常數。

程式執行時希望按  鈕時，才將 `fontDialog1` 字型對話方塊控制項內字型的相關設定指定給 `label1`，寫法：

```
if (fontDialog1.ShowDialog() == DialogResult.OK)
{
    label1.Font = fontDialog1.Font;
}
```



2. Reset 方法

將fontDialog1字型對話方塊控制項所有屬性，
都還原為預設值。

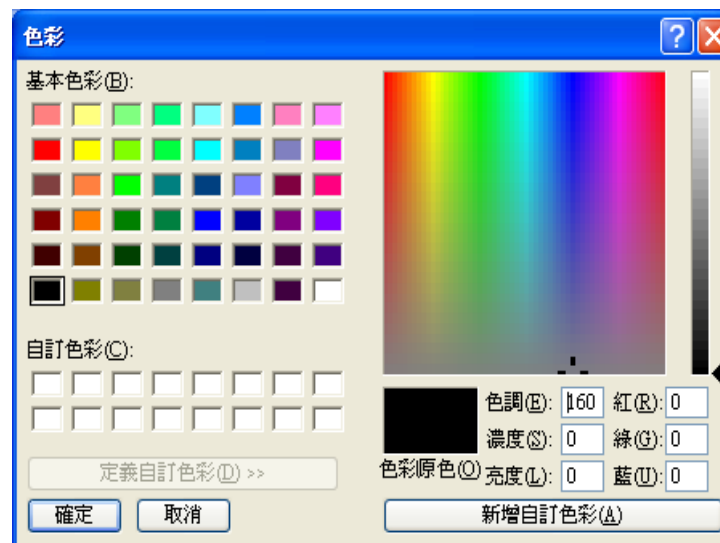
語法：**fontDialog1.Reset();**



12.6 色彩對話方塊控制項



- 色彩對話方塊控制項提供設定顏色
- 屬於幕後執行的控制項，置於表單的正下方。
- 執行中欲開啟色彩對話方塊，使用 **ShowDialog** 方法來開啟





一. ColorDialog 常用屬性

屬性	說明
Color	設定和取得色彩對話方塊中使用者設定的顏色。譬如： 將 colorDialog1 色彩對話方塊內的顏色設定，指定給 label1 標籤控制項當做背景色，其寫法如下： <code>label1.BackColor = colorDialog1.Color;</code>
AllowFullOpen	用來設定自訂色彩色盤按鈕是否有效。預設值為 True，表示  按鈕有效，當按下此鈕如右上圖在右邊開啓自訂色彩色盤供選取顏色值。若設為 False 表示  按鈕無效，無法自訂色彩。
FullOpen	當 AllowFullOpen 為 True 時，此屬性才有效。預設值為 True 表示一開始在右邊自動開啓自訂色彩色盤。若為 False 表示一開始不打開 [自訂色彩(D)] 色盤，必須在左上圖按  按鈕才打開右上圖自訂色彩色盤。



二. ColorDialog 常用方法

1. ShowDialog 方法

- 彩對話方塊 和字型對話方塊一樣屬幕後執行控制項
- 平時不會顯現出來。
- 程式中欲開啟色彩對話方塊時，同樣透過 ShowDialog 方法。寫法：

colorDialog1.ShowDialog();

2. Reset 方法

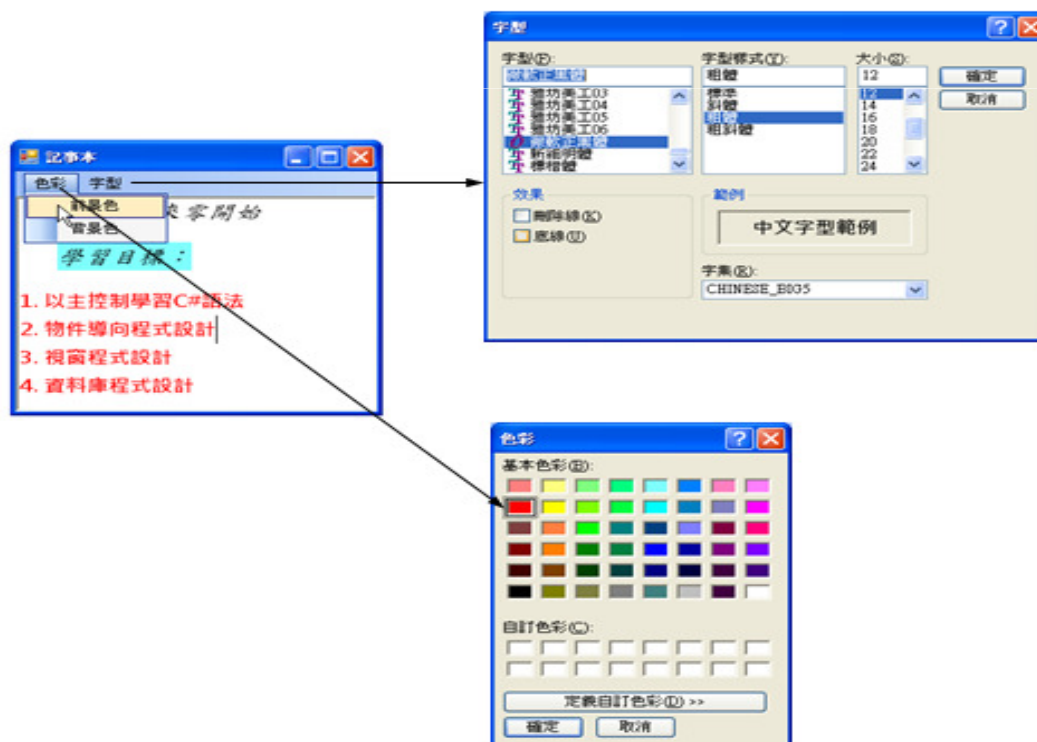
用來將colorDialog1 色彩對話方塊控制項所有屬性，全部還原為預設值。寫法：

colorDialog1.Reset();



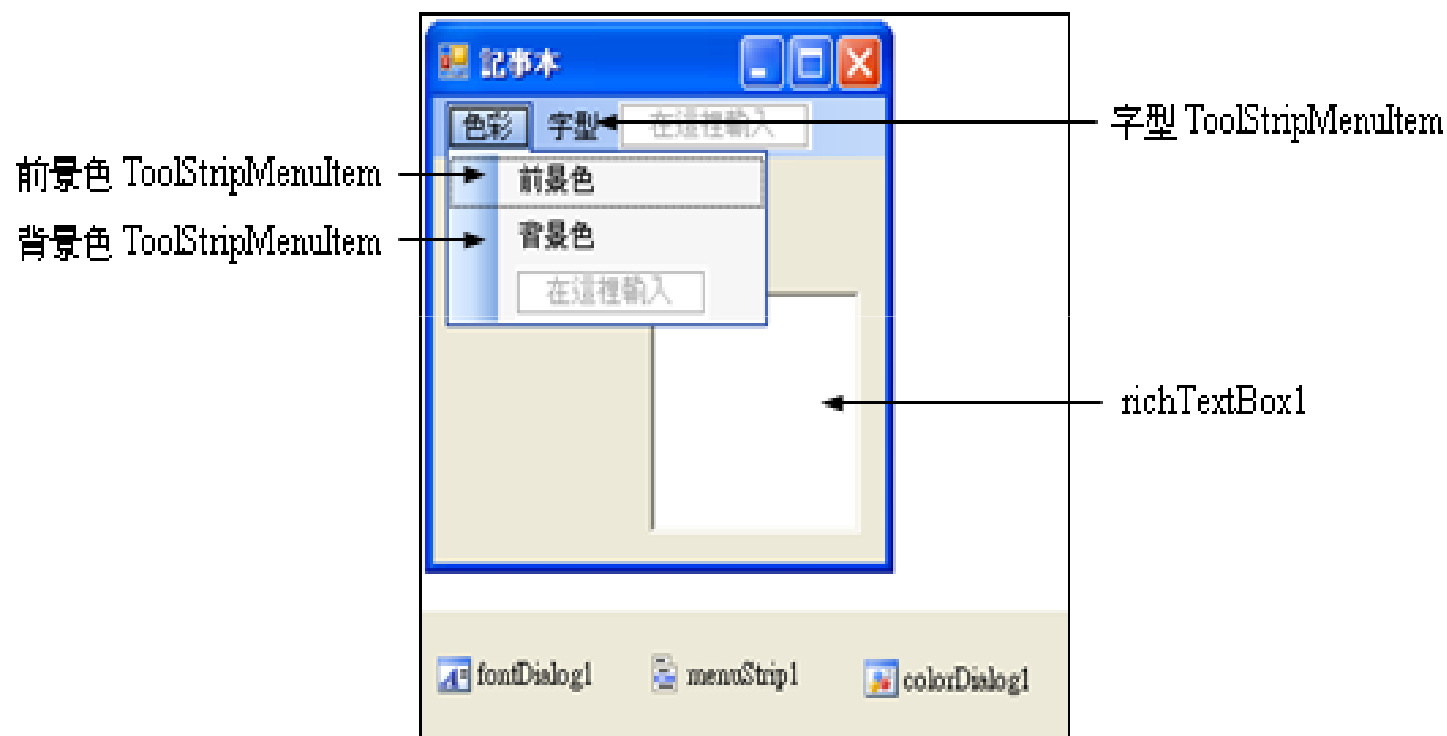
範例演練

使用豐富文字方塊及 MenuStrip 功能表製作下圖的記事本程式。
色彩功能項目擁有「前景色」及「背景色」功能項目可開啟色彩對話方塊來設定所選取文字的前景色及背景色；按下「字型」功能項目可開啟字型對話方塊來設定所選取文字的字型。





Step1 設計輸出入介面





```
// FileName : FontColorDialogDemo.sln
01 private void Form1_Load(object sender, EventArgs e)
02 {
03     richTextBox1.Dock = DockStyle.Fill; //richTextBox1填滿整個表單
04 }
05
06 private void 前景色ToolStripMenuItem_Click(object sender, EventArgs e)
07 {
08     if (colorDialog1.ShowDialog() == DialogResult.OK)
09     {
10         richTextBox1.SelectionColor = colorDialog1.Color;
11     }
12 }
13}
```




```
15 private void 背景色ToolStripMenuItem_Click(object sender, EventArgs e)
16 {
17
18     if (colorDialog1.ShowDialog() == DialogResult.OK)
19     {
20         richTextBox1.SelectionBackColor = colorDialog1.Color;
21     }
22 }
23
24 private void 字型ToolStripMenuItem_Click(object sender, EventArgs e)
25 {
26
27     if (fontDialog1.ShowDialog() == DialogResult.OK)
28     {
29         richTextBox1.SelectionFont = fontDialog1.Font;
30     }
31 }
```




12.7 檔案對話方塊控制項

 工具箱內提供有關檔案的相關工具有兩個：

①  OpenFileDialog

開檔對話方塊可用來提示使用者開啟指定的檔案；

②  SaveFileDialog

存檔對話方塊可用來提示使用者選取儲存檔案的位置。



一. 檔案對話方塊常用屬性

屬性	說明
CheckFileExists	當使用者指定不存在的檔名，設定對話方塊是否顯示警告訊息。
CheckPathExists	當使用者指定不存在的路徑，設定對話方塊是否顯示警告訊息。
DefaultExt	設定或取得檔案對話方塊預設的副檔名。
FileName	設定或取得檔案對話方塊中所選取檔名。
Filter	<p>設定或取得目前檔案對話方塊的檔名篩選字串，用來設定在對話方塊中 [另存檔案類型] 或 [檔案類型] 方塊的選項。如下寫法，則檔案對話方塊的檔案類型清單會顯示「txt files (*.txt)」及「All files (*.*)」</p> <pre>openFileDialog1.Filter = "txt files (*.txt) *.txt All files (*.*) *.*";</pre> 
FileIndex	設定或取得檔案對話方塊中目前所選取的第 i 個索引。
InitialDirectory	設定或取得檔案對話方塊所顯示的初始檔案目錄。
ShowHelp	設定或取得 [說明] 按鈕是否在檔案對話方塊中顯示。
Title	設定或取得檔案對話方塊的標題名稱。



二. 檔案對話方塊常用方法

ShowDialog 方法

開檔對話方塊控制項和存檔對話方塊一樣是屬於幕後執行的控制項，平時不顯現。程式中欲開啟上述任一檔案對話方塊時，透過 ShowDialog 方法。寫法：

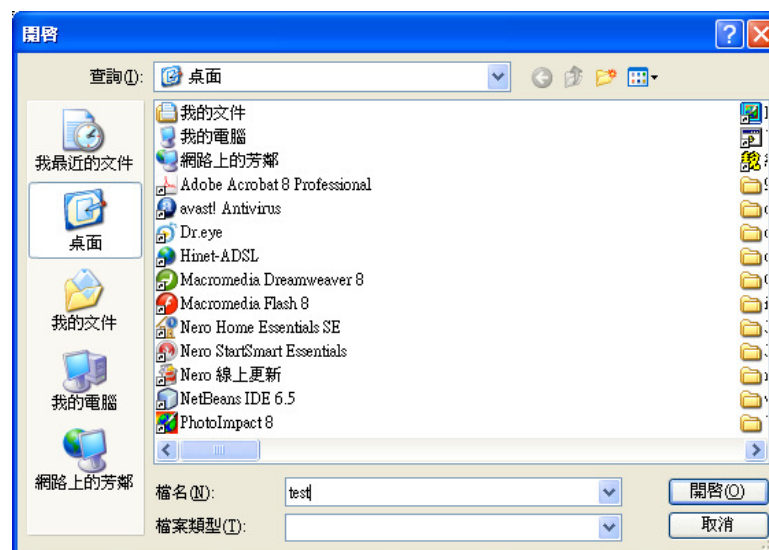
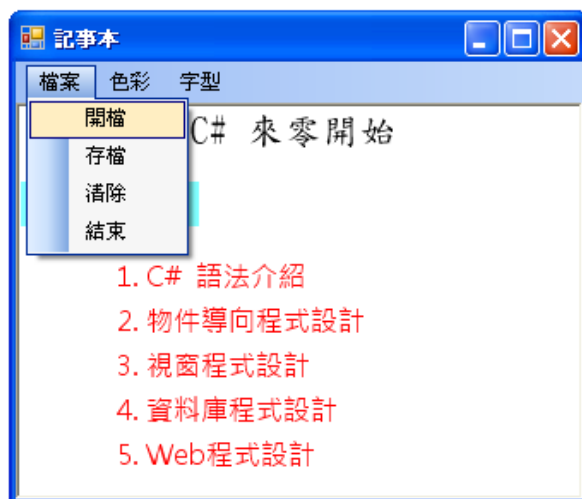
```
openFileDialog1.ShowDialog();
```



範例演練

延續上例，在「檔案」功能項目下新增「開檔」、「存檔」、「清除」、「結束」四個子功能項目，各子功能項目功能說明：

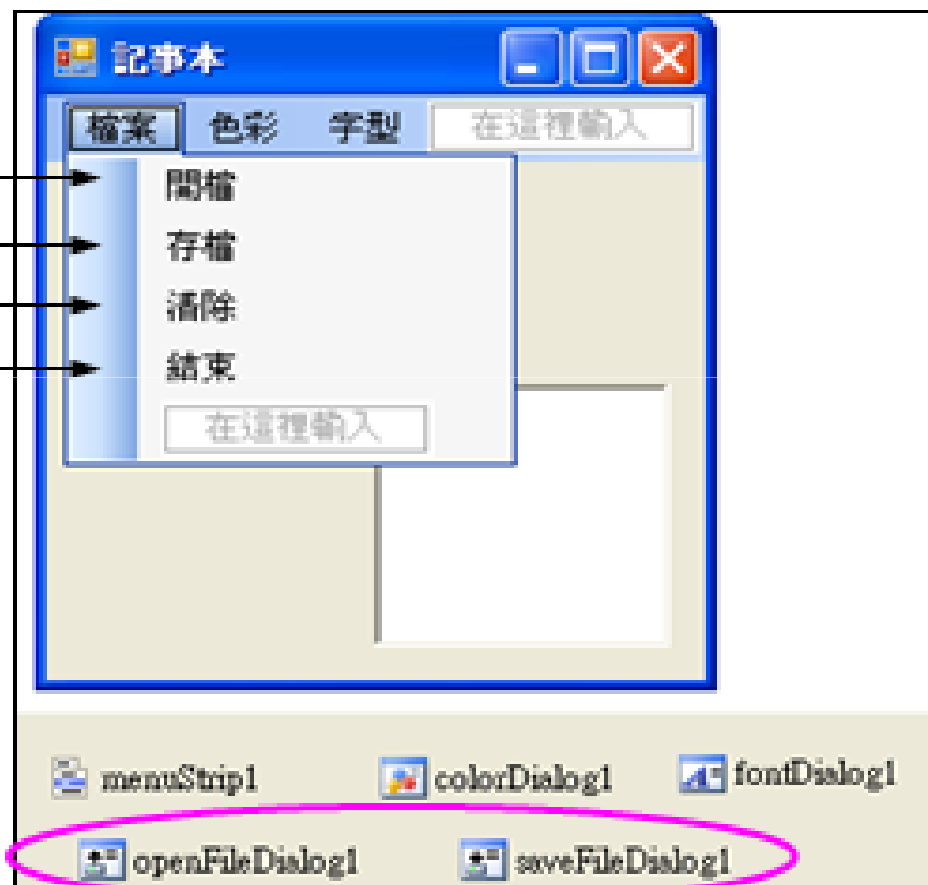
- ① 執行功能表【檔案/開檔】可開啟開檔對話方塊讓使用者選取欲開啟的檔案。
- ② 執行功能表【檔案/存檔】可開啟存檔對話方塊讓使用者進行存檔。
- ③ 執行功能表【檔案/清除】指令會將豐富文字方塊的內容全部清除。
- ④ 執行功能表【檔案/結束】指令會結束程式。

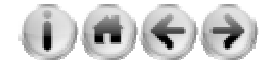




Step1 設計輸出入介面

開檔 ToolStripMenuItem
存檔 ToolStripMenuItem
清除 ToolStripMenuItem
結束 ToolStripMenuItem





```
// FileName : OpenSaveFileDialogDemo.sln
01 private void 開檔ToolStripMenuItem_Click(object sender, EventArgs e)
02 {
03     if (openFileDialog1.ShowDialog() == DialogResult.OK)
04     {
06         richTextBox1.LoadFile(openFileDialog1.FileName, RichTextBoxStreamType.RichText);
07     }
08 }
10 private void 存檔ToolStripMenuItem_Click(object sender, EventArgs e)
11 {
12     if (saveFileDialog1.ShowDialog() == DialogResult.OK)
13     {
15         richTextBox1.SaveFile(saveFileDialog1.FileName, RichTextBoxStreamType.RichText);
16     }
17 }
19 private void 清除ToolStripMenuItem_Click(object sender, EventArgs e)
20 {
21     richTextBox1.Text = "";
22 }
24 private void 結束ToolStripMenuItem_Click(object sender, EventArgs e)
25 {
26     Application.Exit();
27 }
```




12.8 列印文件控制項

■ 工具箱提供有關列印的相關工具有四個：

- ① **PageSetupDialog** 為列印格式對話方塊工具。
- ② **PrintPreviewDialog** 為預覽列印對話方塊工具。
- ③ **PrintDialog** 為列印對話方塊工具。
- ④ **PrintDocument** 為列印文件工具，

①②③ 對話方塊控制項的資料來源都是 **PrintDocument** 列印文件控制項。



一. PrintDocument 常用屬性

屬性	說明
DocumentName	顯示給使用者的文件名稱，預設值為 document。
DefaultPageSettings	執行階段的屬性值，可以設定列印文件控制項的邊界等屬性值。

二. PrintDocument 常用方法

1. Print 方法

利用 Print 方法可觸動 printDocument1 控制項的 PrintPage 事件列印出文件，語法：

```
printDocument1.Print();
```



三. PrintDocument 常用事件

1. PrintPage 事件

- 是 PrintDocument 控制項最重要的事件
- 當用 Print 方法時會觸動本事件，要將列印程式碼寫在 PrintPage 事件處理函式中。
- 在 PrintPage 事件中要宣告一個繪圖物件，再用 DrawString 方法將文字資料傳給 PrintDocument 控制項。
- 例如要列印以黑色新細明體、大小12、座標(150,120)，列印「快樂」字串，語法：

```
Graphics pg = e.Graphics ;
```

```
Font pf = new Font("新細明體",12) ;
```


```
pg.DrawString("快樂",pf, Brushes.Black,150, 120) ;
```

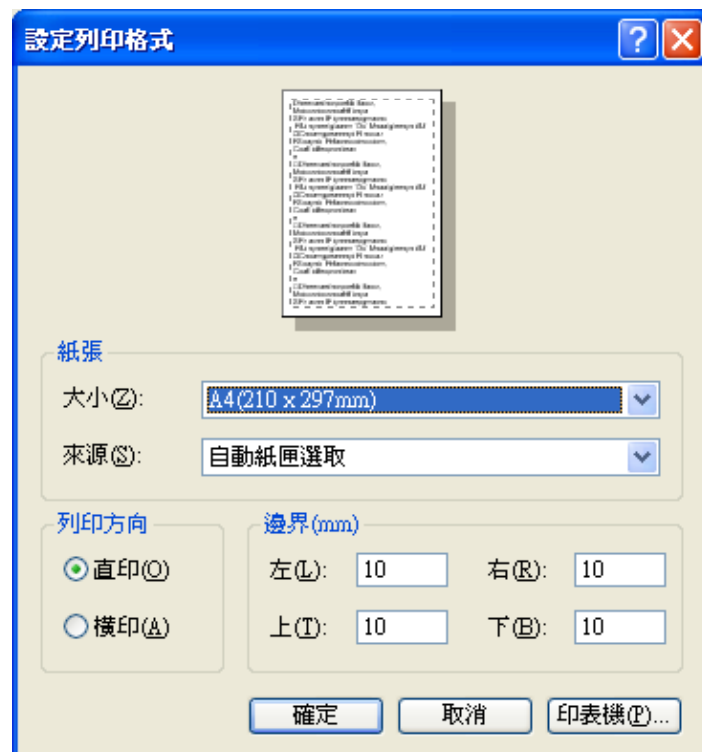


- 另外也可用 Graphics 物件的方法在 PrintDocument 控制項中繪製圖形。
- 例如要繪製一個從左上角座標(100、150)開始，寬度為360，高度為 240 的紅色矩形框，語法：
- 語法：
`e.Graphics.DrawRectangle(Pens.Red, 100, 150, 360, 240);`




12.9 列印格式對話方塊控制項

- 執行  PageSetupDialog 對話方塊控制項時，出現一個下圖設定列印格式對話方塊
- 供使用者設定紙張大小、邊界以及列印方向等各項設定。





一. PageSetupDialog 常用屬性

屬性	說明
Document	選取要處理的 PrintDocument 控制項，預設值為無，本屬性一定要設定否則對話方塊無效。
PageSettings	執行階段的屬性值，取得或設定當使用者按一下對話方塊中的  按鈕時，要修改的印表機設定。例如將列印文件控制項的設定值設為和列印格式對話方塊相同的語法如下： <pre>printDocument1.DefaultPageSettings = pageSetupDialog1.PageSettings;</pre>
AllowMargins	設定是否顯示「邊界」供使用者輸入，預設值為 True。
AllowOrientation	設定是否顯示「列印方向」(橫向和縱向)供使用者選擇，預設值為 True。
AllowPaper	設定是否顯示「紙張」供使用者選擇，預設值為 True。
AllowPrinter	設定是否顯示「印表機」鈕供使用者選擇印表機，預設值為 True。
MinMargins	設定最小的邊界值，預設值為「0,0,0,0」



二. PageSetupDialog 常用方法

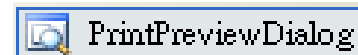
1. ShowDialog 方法

- PageSetupDialog 對話方塊控制項是屬於幕後執行控制項，平時不會顯現出來。
- 當需要顯示列印格式對話方塊控制項供使用者設定時就用 ShowDialog 方法來開啟。語法：

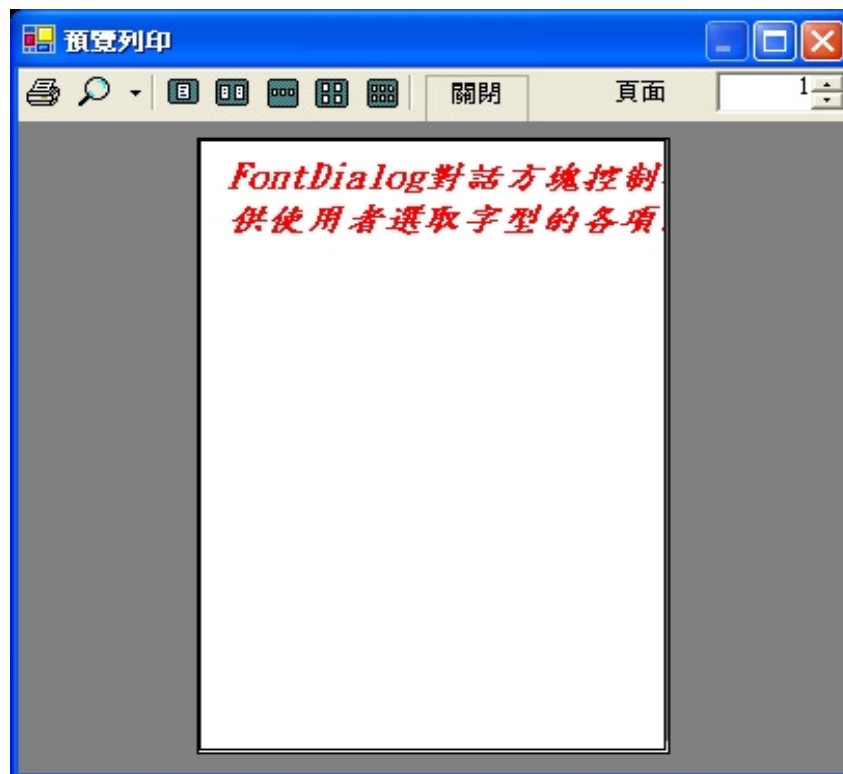
```
pageSetupDialog1.ShowDialog();
```



12.10 預覽列印對話方塊控制項



- 執行預覽列印對話方塊控制項時，會出現下圖「預覽列印」對話方塊，供使用者預覽將要列印的文件及檢查列印該文件檢查是否超版。





一. PrintPreviewDialog 常用屬性

屬性	說明
Document	選取要處理的 PrintDocument 控制項，預設值為無，本屬性一定要設定否則對話方塊無效。

二. PrintPreviewDialog 常用方法

1. ShowDialog 方法

- PrintPreviewDialog 對話方塊控制項是屬幕後執行的控制項，平時不會顯現出來。
- 當要顯示預覽列印控制項對話方塊供使用者設定時，就用 ShowDialog 方法。語法：

```
printPreviewDialog1.ShowDialog();
```



12.11 列印對話方塊控制項



- 執行 列印對話方塊控制項時，會出現下圖 列印對話方塊，供使用者設定和列印相關的各項設定。

列印

印表機

名稱(N): RICOH Aficio 1045 RPCS

狀態: 待機中

類型: RICOH Aficio 1045 RPCS

位置: IP_192.168.0.75

註解:

內容(P)

尋找印表機(F)...

☐ 列印至檔案(L)

☐ 手動雙面列印(X)

指定範圍

☒ 全部(A)

☐ 本頁(E)

☐ 頁數(G):

☐ 選取範圍(S)

輸入頁碼/文件範圍，並以逗點分隔 (例如: 1,3,5 - 12)。

份數

份數(C): 1

☒ 自動分頁(I)

顯示比例

每張紙所含頁數(H): 1 頁

配合紙張調整大小(Z): 不變更比例

列印內容(W): 文件

列印(R): 範圍內全部頁面

選項(O)...

確定 取消



一. PrintDialog 常用屬性

屬性	說明
Document	選取要處理的 PrintDocument 控制項，預設值為無，本屬性一定要設定否則對話方塊無效。
AllowSomePages	設定是否顯示起始頁數至終止頁數的選項。
AllowSelection	設定是否顯示選擇範圍的選項。



二. PrintDialog 常用方法

1. ShowDialog方法

- PrintDialog 對話方塊控制項是屬於幕後執行的控制項，平時不會顯現出來。
- 當要顯示列印對話方塊控制項供使用者設定時，就用ShowDialog 方法。語法：

```
printDialog1.ShowDialog();
```

2. Reset方法

- 將printDialog1列印對話方塊控制項所有屬性，都還原為預設值。語法：

```
printDialog1.Reset();
```

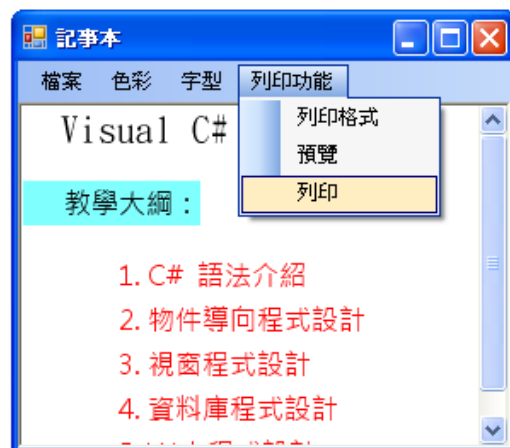


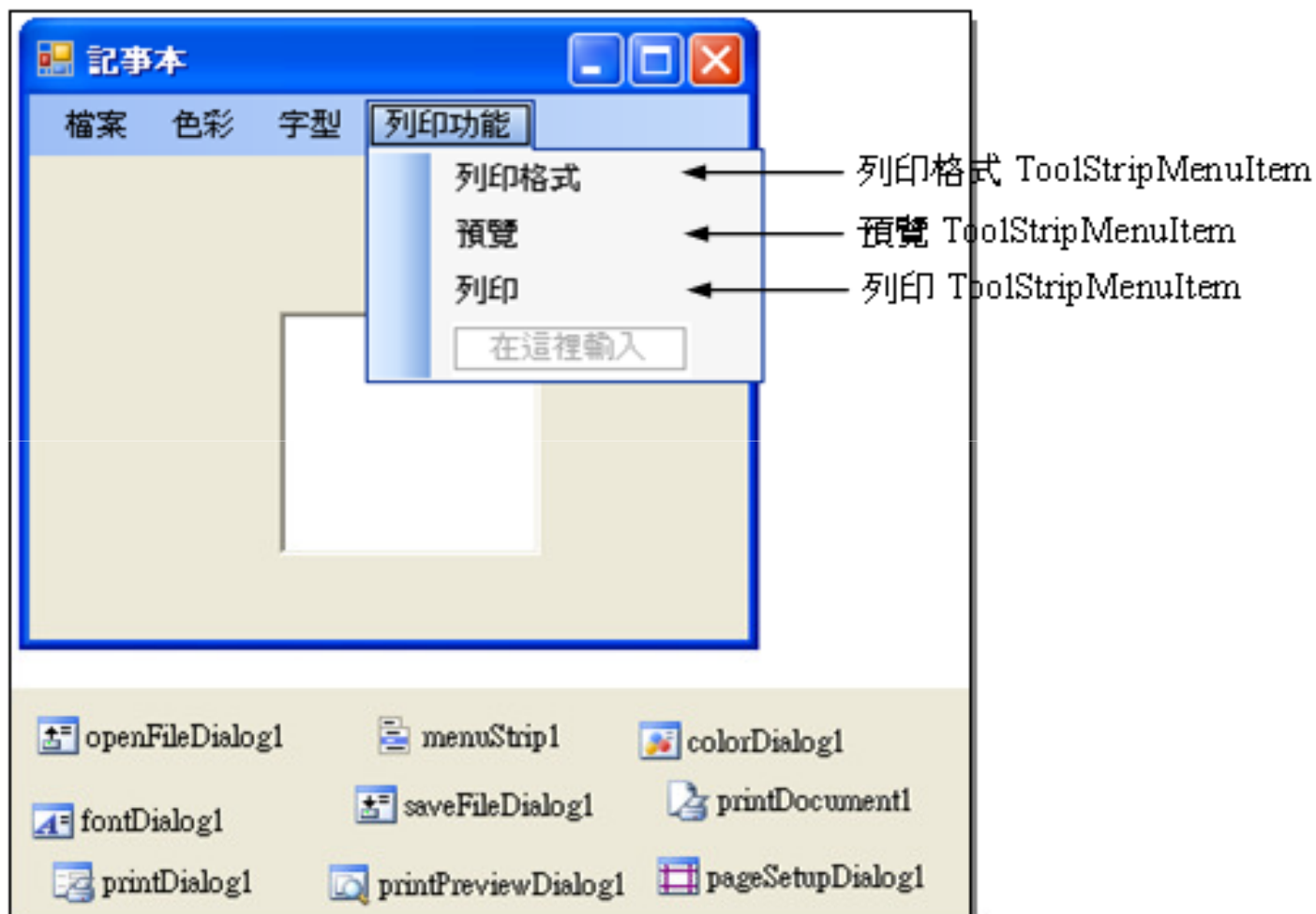
範例演練



延續上例，在 列印功能項目下新增 列印格式、預覽、列印三個子功能項目，各子功能項目的功能說明如下：

- ① 執行功能表【列印功能/列印格式】可開啟設定列印格式對話方塊讓使用者設定列印文件的格式。
- ② 執行功能表【列印功能/預覽】指令可開啟預覽列印對話方塊讓使用者預覽要列印件的文件。
- ③ 執行功能表【列印功能/印列】指令可開啟列印對話方塊讓使用者進行列印文件。



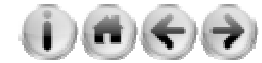




```
// FileName : PrintDialogDemo.sln
01 private void Form1_Load(object sender, EventArgs e)
02 {
03     richTextBox1.Dock = DockStyle.Fill; //richTextBox1填滿整個表單
04
05     // 設定printDialog1, printPreviewDialog1, pageSetupDialog1的
06     // 列印資料來源為printDocument1
07     printDialog1.Document = printDocument1;
08     printPreviewDialog1.Document = printDocument1;
09     pageSetupDialog1.Document = printDocument1;
10 }
11 .....
.....
//其他功能項目的Click事件處理函式省略
```



```
12 private void 列印格式ToolStripMenuItem_Click(object sender, EventArgs e)
13 {
14     pageSetupDialog1.ShowDialog();
15     printDocument1.DefaultPageSettings = pageSetupDialog1.PageSettings;
16 }
18 private void 預覽ToolStripMenuItem_Click(object sender, EventArgs e)
19 {
20     printPreviewDialog1.ShowDialog();
21 }
22
23 private void 列印ToolStripMenuItem_Click(object sender, EventArgs e)
24 {
25     if (printDialog1.ShowDialog() == DialogResult.OK)
26     {
27         printDocument1.Print();
28     }
29 }
30
```

```
31 private void printDocument1_PrintPage(object sender,  
    System.Drawing.Printing.PrintPageEventArgs e)  
32 {  
33     Graphics pg = e.Graphics;  
34     Font pf = new Font(richTextBox1.Font.Name, richTextBox1.Font.Size,  
        richTextBox1.Font.Style);  
35     SolidBrush pb = new SolidBrush(richTextBox1.ForeColor);  
36     Single x = printDocument1.DefaultPageSettings.Margins.Left;  
37     Single y = printDocument1.DefaultPageSettings.Margins.Top;  
38     pg.DrawString(richTextBox1.Text, pf, pb, x, y);  
39 }
```