PART 6

互通性

Q70 如何在 ASP.NET 中引用非 COM 類型的 DLL？

適用範圍： V ASP.NET 1.0 V ASP.NET 1.1 V ASP.NET 2.0 V ASP.NET 3.5

問題

公司有一個應用程式, 是在抓取公司內自動化生產線的機器生產的統計數據的, 這些機器都透過一個 DLL 來進行數據資料的擷取, 公司最近想要把這個應用程式改成 Web 應用程式, 要我在 ASP.NET 中使用這個 DLL, 但這個 DLL 又不是 COM 或 .NET 的 DLL, 無法用加入參考的方式呼叫, 請問我要如何做, 才能使用這個 DLL？

問題說明

以前有這麼一句廣告詞：「斯斯有兩種, 一種治感冒, 一種治咳嗽」, 現在這句話可以改一下, 來說明這個問題：「DLL 有兩種, 一種可以加入參考, 另一種不能加入參考」(這不是廢話？) 。在.NET Framework 中, 可以加入參考的有兩種 DLL, 一種是.NET 的 DLL, 另一種則是以 COM Automation 爲規格開發的 COM DLL 元件, 這種元件透過.NET 的 COM Interoperability 功能, 和.NET 應用程式整合在一起, 所以.NET 應用程式可以取用 COM 元件中的物件。反過來也可以通, .NET 元件也可以開放出來給 COM 應用程式存取, 只要遵照.NET 的 COM Interoperability 規格即可。

小常識 .NET Framework 的 COM Interoperability (互通能力)

COM (Component Object Model) 在 1992 年左右就被發展出來, 當時是以 OLE 爲名, 出現在 Office 系列軟體, 讓 Office 的文件可以支援物件內嵌功能, 後來到 1995 年左右, 原本的 Visual Basic 的 VBX 架構, 被 OCX 架構取代 (VB 4.0) , OCX 就是以 COM 爲基礎打造的, 接著到.NET 出來之前, 微軟平台上的元件幾乎都是用 COM 來發展的, 算算也有六七個年頭, 在這段時間被發展出來的元件少說也有數千到上萬個, 微軟當然不能得罪這些元件的開發廠商, 所以在.NET Framework 中加入了和原有的 COM 架構相通的介面, 讓.NET 和 COM 能夠相互存取, 這也是爲何能在 C# 應用程式中存取 Office 物件模型的原因了。

不過若碰上了非 COM Automation 規格實作的元件, 是不是就完全沒辦法了呢？那倒也不是, 除了對 COM 的互通能力外, 另一個型式的 DLL, 也就是系統層次的 DLL, 這些 DLL 只開放 API 函式, 沒有像 COM 那樣複雜的指標存取, 因此即便 Visual Studio 無法加入非 COM Automation 的系統 DLL, 開發人員也可以在程式碼中宣告這些 DLL 的函式原型 (Prototype) , 然後在程式中取用, 至於取用的細節, 就交給.NET Framework 即可。

小常識　.NET 與系統層次 DLL 函式庫的互通能力

系統層次的 DLL, 例如管理使用者殼層 (Shell) 的 shell32.dll；管理核心功能的 kernel32.dll；負責繪圖子系統的 gdi32.dll；以及管理視窗介面的 user32.dll 等, 它們都以 API 的方式來開放函式, 但並沒有實作 COM 的能力, 所以無法直接存取。其實不只是這幾個函式庫, 坊間一些執行硬體控制或者是具有橋接功能的函式庫, 也大多是以 API 方式開放, 例如 RFID Reader 的控制程式庫, 就多是 API 函式庫, 由於這些函式庫的存取方法和 COM 大不相同 (COM DLL 可以透過型別函式庫來取得類別的資訊) , 所以也不能以相同方法來處理。因此, .NET Framework 利用之前 VB 6.0 呼叫 API 的作法, 發展出一個互通機制, 稱爲平台叫用服務 (Platform Invocation Service; P/Invoke) , 這個服務能夠以 DLL 函式原型宣告的方式, 來和 DLL 開放的 API 做連結。

若要呼叫非 COM 類型的 DLL, 則要先知道函式的原型, 然後用下列的格式來宣告：

程式1：宣告 DLL 函式的格式。

```
// C#
[DllImport("函式庫名稱", EntryPoint="函式實際的進入點")]
public static extern [回傳資料型別] [函式名稱] (參數列);


' VB.NET
Public Declare [函式名稱] Lib [函式庫名稱] Alias  _
  [函式實際的進入點] (參數列)
```

例如, 若想要呼叫 API 提供的 MessageBox API 函式, 則可以這樣宣告：

程式 2：宣告 MessageBox API。

```
// 函式原型：
int MessageBox(HWND hWnd, LPCTSTR lpText,
  LPCTSTR lpCaption, UINT uType);
// 宣告方式：
[DllImport("User32.dll", EntryPoint="MessageBox")]
public static extern int MsgBox (int hWnd, string lpText,
  string lpCaption, uint uType);
```

在建立 API 宣告的原型時, 有些地方要特別注意。

1. 部份 API 中會有一些結構 (struct) 的型別, 那在程式中也要宣告這個結構, 才可以套用到原型宣告中, 例如：

程式 3：WNetAddConnection2 的函式宣告。

```
// prototype
DWORD WNetAddConnection2(LPNETRESOURCE lpNetResource,
LPCTSTR lpPassword, LPCTSTR lpUsername, DWORD dwFlags);
// declaration
[DllImport("mpr.dll", CharSet = CharSet.Auto)]
public static extern int WNetAddConnection2(
  NETRESOURCE NetResource, string Password, string UserName,
  UInt32 Flags);
// NETRESORCE prototype
typedef struct _NETRESOURCE
{
  DWORD dwScope;
  DWORD dwType;
```

```
  DWORD dwDisplayType;
  DWORD dwUsage;
  LPTSTR lpLocalName;
  LPTSTR lpRemoteName;
  LPTSTR lpComment;
  LPTSTR lpProvider;
} NETRESOURCE;
// LPNETRESOURCE declaration
[StructLayoutAttribute(LayoutKind.Sequential,
  CharSet = CharSet.Auto)]
public class NETRESOURCE
{
  public int dwScope;
  public int dwRType;
  public int dwDisplay;
  public int dwUsage;
  [MarshalAs(UnmanagedType.LPTStr, SizeConst = 200)]
  public string lpLocalName;
  [MarshalAs(UnmanagedType.LPTStr, SizeConst = 200)]
  public string lpRemoteName;
  [MarshalAs(UnmanagedType.LPTStr, SizeConst = 200)]
  public string lpComment;
  [MarshalAs(UnmanagedType.LPTStr, SizeConst = 200)]
  public string lpProvider;
}
```

2. 若 API 函式具有回呼功能 (Callback) 時, 則程式中必須要宣告一個與回呼函式相同參數的委派 (delegate) 物件, 將委派物件交給函式宣告使用, 在實際呼叫函式時, 給予要實作回呼處理的函式位置即可。

程式碼 4：回呼函式的處理 (取自 .NET Framework SDK 文件的回呼範例)。

```
public delegate bool CallBack(int hwnd, int lParam);
public class EnumReportApp {

  [DllImport("user32")]
  public static extern int EnumWindows(CallBack x, int y);

  public static void Main()
  {
      CallBack myCallBack = new CallBack(EnumReportApp.Report);
      EnumWindows(myCallBack, 0);
  }

public static bool Report(int hwnd, int lParam) {
      Console.Write("Window handle is ");
      Console.WriteLine(hwnd);
      return true;
  }
}
```

3. 若函式中有使用指標, 則要視指標參數的宣告是否可由內建功能處理, 若無法處理, 則需要以 Marshal 封送類別的方式來處理, 例如：

程式碼 5：帶有 void*參數的 API 函式的宣告法 (取自.NET Framework SDK 文件範例)

```
// 函式原型：
void SetData(DataType type, void* data);

// 宣告方式：
[DllImport( "..\\LIB\\PinvokeLib.dll" )]
 public static extern void SetData( DataType t,
   [ MarshalAs( UnmanagedType.AsAny )] Object o );
```

解決方案

利用.NET 的平台叫用功能, 宣告 DLL 中的 API 的函式原型 (以及必要的結構與參數) 後, 就能夠在程式中直接叫用了。

但要注意 DLL 的路徑, 最好是和可執行檔放在一起, 否則就要放到 Windows 的系統目錄, 或是在宣告時設定 DLL 所在的路徑, 不過這些都不是微軟所建議的應用程式發展與部署方法, 因此和可執行檔放在相同的路徑才是比較好的作法。

小常識　叫用 DLL 函式時常見的兩個錯誤

1. 系統找不到指定的檔案：

   發生在.NET Framework 使用 LoadLibrary()載入 DLL 時, 因爲路徑不對或檔案不存在, 才會發生的錯誤, 修正方法爲將 DLL 放到與應用程式可執行檔相同的路徑, 或是 Windows 系統目錄 (%SYSTEM%, 例如 C:\Windows\System32) 中。

2. 函式進入點在指定的 DLL 中找不到：

   發生在.NET Framework 使用 GetProcAddress()取得函式指標時, 因爲設定的進入點函式名稱不對所致, 這個錯誤較容易發生在 Windows NT 以後的系統, 因爲 Windows NT 支援 Unicode 以及寬字元 (Wide character) , 有許多函式的名稱都會加上"A"或"W"來識別函式是否適用於寬字元或是 Ansi 字元。修正方法爲將進入點名稱設定正確。

考生停看聽

本問題所討論的內容, 可用來準備 Exam 70-536: TS: Microsoft .NET Framework 2.0 Application Development Foundation 的下列主題：

- 在.NET Framework 應用程式中實作互通性。
  - 在.NET Framework 應用程式中, 呼叫 Unmanaged DLL 中的函式, 並且控制資料的封送。
    - 在 Managed 程式碼中建立 DLL 函式的原型。
    - 叫用 DLL 函式。
    - 在特殊案例中叫用 DLL 函式, 像是傳遞結構與實作回呼函式。
    - 預設的封送行爲。
    - 以平台叫用方式封送資料。

Q71 如何在 ASP.NET 中執行本機上的程式？

適用範圍： ☑ASP.NET 1.0 ☑ASP.NET 1.1 ☑ASP.NET 2.0 ☑ASP.NET 3.5

問題

我有一支程式 (主控台型應用程式) , 是用來產生分析數據用的, 現在我將應用程式移到 Web, 並且用 ASP.NET 來發展, 我想要在 ASP.NET 中直接呼叫這支程式來產生分析資料, 然後用我寫的 HTTP Handler 來繪出圖表, 請問我要如何做, 才可以呼叫這個程式？

問題說明

在 Web 應用程式上, 有時候會因爲特殊需要, 而要額外的呼叫外部應用程式, 像是一些 DOS 時代撰寫的應用程式, 或是針對額外的需求撰寫的工具程式, 例如資料轉換或是抓取數據的小程式等, 這些程式可能已年代久遠, 無法再重新撰寫, 或者原始碼已遺失, 這樣的話就不能直接將程式碼轉移到 Web 平台, 只能夠用呼叫的方法來處理。

所幸, ASP.NET 可直接利用 .NET Framework 中的類別庫, System.Diagnostics 命名空間中的 Process 類別可處理對本機作業系統中的行程管理 (Process Management) 工作, 除了可以列舉出目前作業系統中正在執行中的行程清單外, 也可以啓動執行檔與批次檔等可執行碼, 並且也可直接取回來自於應用程式所產生的資料流。

Process.Start() 是用來啓動應用程式的, 用法很簡單, 只要傳入要啓動的執行檔名稱和參數就行了。

```
using System.Diagnostics;
...
Process.Start("File Name", "Arguments");
```

若想要做到更進階的控制, 則可以透過 ProcessStartInfo 這個資料類別來設定, 然後將它傳入 Process.Start()作爲參數即可：

```
ProcessStartInfo startInfo = new ProcessStartInfo();

startInfo.Arguments = "/c dir C:\\"; // 參數
startInfo.FileName = "cmd.exe"; // 執行檔名稱
startInfo.UseShellExecute = false; // 是否使用 Shell API
startInfo.RedirectStandardOutput = true; // 是否重新導向資料流

Process p = Process.Start(startInfo);
```

小常識 ProcessStartInfo.UseShellExecute 的差別

ProcessStartInfo.UseShellExecute 是指示是否要使用 Windows Shell (殼層) 的 ShellExecuteEx() API 來啓動執行檔的參數, 如果設爲 true, 就呼叫 ShellExecuteEx() API, 否則就呼叫 CreateProcess() API, 這兩者有一些地方不同：

功能	ShellExecuteEx()	CreateProcess()
直接操作檔案指令	支援，將指令傳給檔案相關的應用程式處理，例如傳入 Word 的 doc 檔案, 並且傳入 print 指令，即可自動列印指定的 Word 檔案。	不支援，要直接設定應用程式執行檔並下參數。
重新導向資料流	不支援	支援, 透過 STARTINFO 設定 dwFlag 爲 STARTF_USESTDHANDLES 來啓用 StandardInput,StandardOutput 及 StandardError 三種資料流
使用特定使用者帳戶執行	不支援	支援，透過 CreateProcessWithLogon() API 來執行。

接下頁

功能	ShellExecuteEx()	CreateProcess()
輸出入資料流	不支援	支援，透過設定 StandardOutputEncoding 與 StandardErrorEncoding 來支援。

因此, 若想要將外部程式的執行結果輸出到自己的應用程式中, 就要利用 I/O Redirection 的方式, 此時就要將 UseShellExecute 設爲 false, 並且設定 RedirectStandardOutput=true, 若要自己寫入資料到輸入資料流, 則要設定 RedirectStandardInput=true。

反之, 若只是操作檔案之類的小工作, 沒有用到進階功能的話, 則利用 UseShellExecute=true 即可。

例如, 筆者可以應用 Process 搭配命令列工具 (Command Line Tool) cmd.exe 來列舉出磁碟機中的資料夾清單, 列舉清單的指令是 dir, 所以 cmd.exe 是執行檔, 參數要送入 dir 指令, 然後也要配合輸出資料流的重導向 (Steam Redirection), 這樣才能將執行結果重導向到外部的資料流, 也才可以在 ASP.NET 中讀取 (如程式 1) 。

程式 1：在 ASP.NET 中列舉 C:\的目錄。

```
protected void Page_Load(object sender, EventArgs e)
{
  ProcessStartInfo startInfo = new ProcessStartInfo();

  startInfo.Arguments = "/c dir C:\\";
  startInfo.FileName = "cmd.exe";
  startInfo.UseShellExecute = false;
  startInfo.RedirectStandardOutput = true;

  Process p = Process.Start(startInfo); // 啓動程式

  // 讀取轉向的輸出資料流
  string output = p.StandardOutput.ReadToEnd();
```

```
    this.T_Dir.Text = output;
    p.Dispose();
}
```

讀者可以再多試一些指令, 像是將 startInfo.Arguments 設爲/all, 然後將 startInfo.FileName 改爲 ipconfig.exe, 就可以輸出 ipconfig /all 的網路詳細資料。

解決方案

利用 System.Diagnostics 的 Process.Start() 來啓動可執行檔, 若是需要額外的控制的話, 則可以利用 ProcessStartInfo 來設定進階的屬性, 以控制執行時的一些功能, 例如資料流重導向等。

小常識　在 ASP.NET 中, 應小心使用 Process.Start()

Process 雖然是個很好用的工具類別, 不過它在 ASP.NET 上應用, 卻有個開發人員常忽略的盲點, 尤其是初學者。

任何使用 Process.Start()啓動的應用程式, 若是有使用者介面的, 則瀏覽器是看不到使用者介面的, 也無法做任何互動, 除非透過 ASP.NET 來執行指令。另外, 如果應用程式是會中斷的 (例如 MessageBox) , 如果 ASP.NET 沒有處理這種情況, 那麼應用程式就會持續佔用資源, 若使用者數很多的話, 伺服器資源很快就被吃光了。

因此, 要在 ASP.NET 上呼叫的外部應用程式, 最好是全自動化, 以及主控台型 (Console) 的應用程式, 並且要檢查行程是否已經執行完畢, 以避免過多的行程佔用記憶體資源。

考生停看聽

本問題所討論的內容, 可用來準備 Exam 70-536: TS: Microsoft .NET Framework 2.0 Application Development Foundation 的下列主題：

- 在.NET Framework 應用程式中嵌入組態、診斷、管理與安裝功能。
  - 使用.NET Framework 2.0 診斷功能, 管理系統行程以及監控.NET Framework 2.0 的應用程式效能。
    - 以使用或不使用命令列參數的方式來啓動行程。

Q72 如何在 ASP.NET 顯示 Microsoft Office 的圖表？

適用範圍： V ASP.NET 1.0 V ASP.NET 1.1 V ASP.NET 2.0 V ASP.NET 3.5

問題

公司想要在網站上的報表中加入統計圖表, 我查了網站上的文章, 知道可以用 Office Web Component (OWC) 來實作, 請問我要如何做才能使用 OWC 來產生圖表？

問題說明

在報表上使用圖表 (Chart) , 對於使用報表的使用者來說, 馬上就可以對數值的意義一目瞭然, 不必去面對一堆數字, 然後還要自己用 Excel 才能轉換成圖表, 尤其是在高階人員所使用的資訊系統中, 圖表所佔的地位又更重了, 甚至還有可以和使用者互動的圖表, 這些功能都要靠開發人員的圖表與互動性設計的功力。

不過筆者認爲平常會有多餘的時間來設計圖表的開發人員, 可能是屈指可數吧 ☺, 能夠有馬上使用的圖表元件的話就用了, 哪來這麼多時間去研究要怎麼撰寫圖表的程式, 尤其在台灣 MIS 要分飾多角的情況下, 更不可能擠出額外的時間了, 除非不睡覺 ☺。

在網路上可以找到很多圖表元件的供應商, 像是 Aspose, ComponentOne, dotnet Charting 等等公司, 都有提供支援 MFC/VB6/.NET 等圖表元件, 不過通常價格都不低, 而且像是 Site License (依站台計價的授權) 或是 Server/Unlimited Distributed License (以 Server 或是不受限制的轉散布) 授權, 都要數千美元之譜, 部份公司還提供原始碼的授權, 但那可就更貴了 (不過圖表的效果通常都蠻棒的, 有些比 OWC 還棒) 。

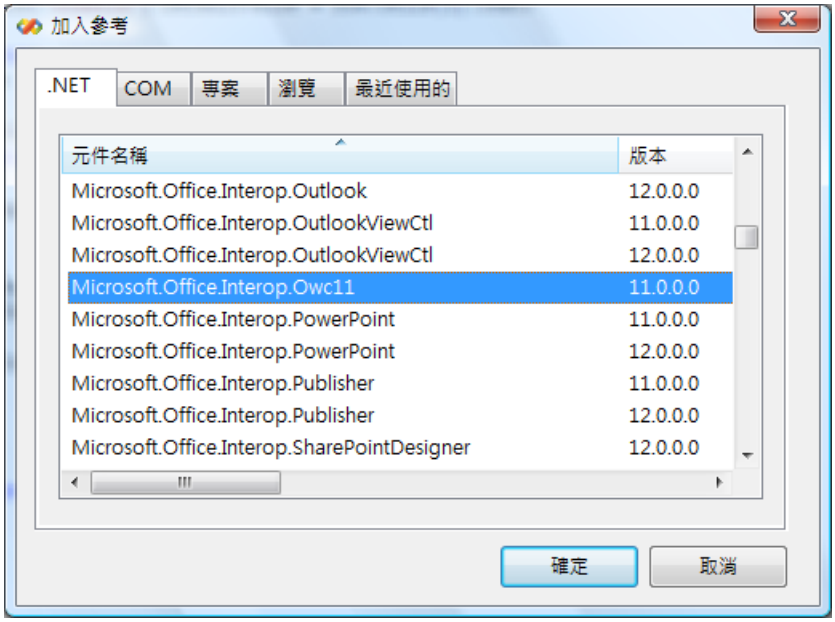
還好, 微軟知道開發人員會有使用圖表的需求, 特別把 OWC (Office Web Component) 留給了開發人員, 可以在沒有 Office 授權的情況下, 使用它產生靜態且唯讀的圖表。開發人員可以在微軟的下載網站中搜尋 OWC 即可找到, 目前的最新版本是 11 版, 也是微軟發布獨立的 OWC 版本的最後一版, 未來它不會再獨立發布, 而是整合到 SharePoint Server 2007 中。

要使用 OWC 來開發圖表應用程式時, 除了安裝 OWC 的元件外, 還要加上 Microsoft.Office.Interop.Owc11 這個主要互聯組件(Primary Interop Assembly), 這個組件包含在 Office 的 Primary Interop Assemblies 安裝套件, 隨著 Office 2003 一起散布, 並且也有網路安裝的版本, 可在微軟下載中心找到, 網址爲 http://www.microsoft.com/downloads, 使用關鍵字「OWC」或「Office Web Component」來搜尋, 即可找到。

做好所有的準備後, 就可以開始使用 OWC 來開發圖表了。

解決方案

首先, 先將 Microsoft.Office.Interop.Owc11.dll 加入專案的參考中。



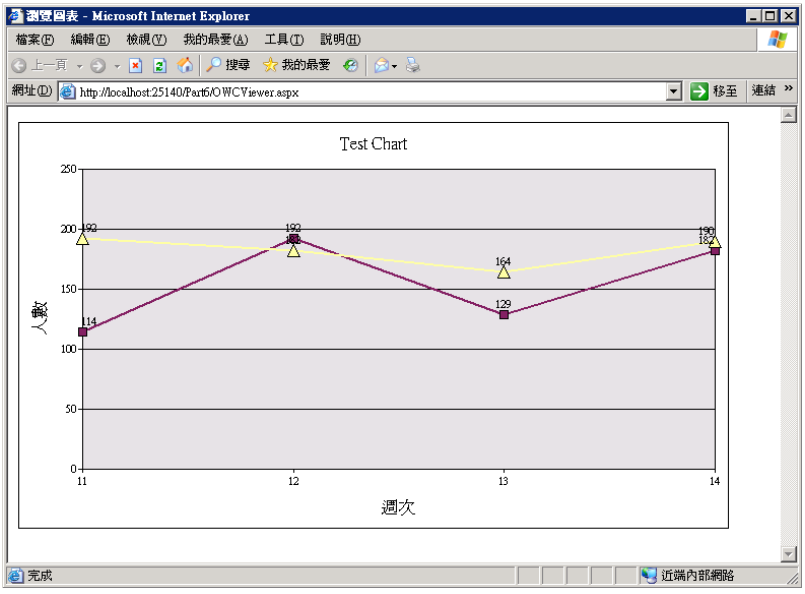
接著, 建立一個 HTTP Handler, 並且在裡面加入產生圖表的程式碼。

```
public void ProcessRequest (HttpContext context) {
// 產生 OWC 圖表空間
ChartSpaceClass chartSpace = new ChartSpaceClass();
// 新增一份圖表
ChChart chart = chartSpace.Charts.Add(0);
// 設定圖表屬性
chartSpace.HasChartSpaceTitle = true; // 是否有圖表空間的標題
chartSpace.ChartSpaceTitle.Caption = "Test Chart"; // 圖表空間的標題
chart.HasTitle = false; // 圖表本身是否有標題
chart.Type = ChartChartTypeEnum.chChartTypeLineMarkers; // 圖表類型
// 圖表的屬性設定, 在二維圖表中, Axes[0] = X 軸, Axes[1] = y 軸
chart.Axes[0].HasTitle = true;
chart.Axes[0].Title.Caption = "週次";
chart.Axes[1].HasTitle = true;
chart.Axes[1].Title.Caption = "人數";
// 設定圖表 X 軸的資料 (分類維度) 。
chart.SetData(ChartDimensionsEnum.chDimCategories,
  (int)ChartSpecialDataSourcesEnum.chDataLiteral, "11\t12\t13\t14");
// 在圖表中新增一個數列
ChSeries series1 = chart.SeriesCollection.Add(0);
// 加入資料到數列
series1.DataLabelsCollection.Add();
series1.DataLabelsCollection[0].HasValue = true;
series1.SetData(ChartDimensionsEnum.chDimValues,
  (int)ChartSpecialDataSourcesEnum.chDataLiteral,
  "114\t192\t129\t182");
// 在圖表中新增一個數列
ChSeries series2 = chart.SeriesCollection.Add(1);
// 加入資料到數列
series2.DataLabelsCollection.Add();
series2.DataLabelsCollection[0].HasValue = true;
```

```
series2.SetData(ChartDimensionsEnum.chDimValues,
  (int)ChartSpecialDataSourcesEnum.chDataLiteral,
  "192\t182\t164\t190");
// 設定圖表輸出的圖片類型
context.Response.ContentType = "image/gif";
// 輸出圖表
context.Response.BinaryWrite(
  (byte[])chartSpace.GetPicture("GIF", 700, 400));

series1 = null;
series2 = null;
chart = null;
chartSpace = null;
GC.Collect();
}
```

最後, 在網頁中加入圖片控制項, 並將 ImageUrl 指向這個 HTTP Handler, 就可以在網頁中顯示出圖表了 (有關於 HTTP Handler 產生圖表的部份可參考「如何動態產生圖片」問答)。其執行的畫面爲：



由於OWC函式庫的範圍實在太大, 筆者無法一一述明, 讀者可以在OWC的安裝目錄 (C:\Program Files\Common Files\Microsoft Shared\Web Components\11\1028\) 中, 找到完整的OWC物件模型說明 (OWCVBA11.chm) , 讀者可參考這份物件模型來設計圖表的應用程式, 或者在網路中也有許多OWC應用的範例程式碼。

小常識　什麼是 Primary Interop Assemblies (PIAs)？

在Microsoft .NET中, 如果加入了COM元件的參考, .NET的tlbimp.exe (型別函式庫匯入工具) 會產生一組給Managed Code參考使用的組件碼, 這個組件碼在不同的電腦上可能會有不同的結果, 如果在A電腦中和在B電腦中產生的組件碼不同時, 就會發生找不到組件的錯誤, 因此微軟在tlbimp.exe中加入了一個產生固定組件碼的能力, 讓開發人員可以直接使用這個組件碼, 就可以避免因爲加入COM元件所產生的組件碼不同的問題, 這組固定的組件碼就稱爲Primary Interop Assembly, 是跟著COM元件一起部署的, 微軟的Office軟體都有提供相對應的PIAs供開發人員使用。

Q73 如何讓 ASP 和 ASP.NET 共同運作？

適用範圍：VASP.NET 1.0 VASP.NET 1.1 VASP.NET 2.0 VASP.NET 3.5

問題

我的公司是從事線上交易的電子商店, 原本已經有和銀行溝通的 ASP 應用程式, 並且運作正常, 但是最近公司要升級系統到 ASP.NET, 而且老闆說在銀行溝通的 ASP 程式升級到 ASP.NET 之前, 都必須要沿用 ASP 的程式, 請問我要如何在 ASP.NET 中使用 ASP 程式？

問題說明

這個問題在企業升級現有應用程式時很常見, 在 ASP 時代, 就已經有很多的企業將他們的應用程式放到 Web 平台之上, 並且開始運轉, 長久下來若都沒有什麼問題, 倒也相安無事, 如果哪天老闆說要升級或是轉換的話, 開發人員就要開始頭痛了, 因爲在大多數的情況下, 新舊系統是要並行運作的, 再加上如果是核心的系統元件, 在不影響系統運作的考量下, 有可能就不再更新, 而保持現有的環境來處理。因爲如果寫壞了, 讓公司的營運無法順利進行的話, 可能管理階層裡有人的烏紗帽可就不保了 ☺。

在維持企業正常營運的前提下, 現在還有很多的舊系統都還留在當時發展的平台上, ASP 就是因爲如此而留了下來, 在 IIS 7.0 上仍可以看的到 ASP 的影子。所以, 如果公司中有關鍵任務 (critical mission) 的應用程式是用 ASP 開發的, 那麼開發人員所寫的新系統務必也要和 ASP 系統並存, 和 ASP 系統交換資料的工作也就變得必要了。

若要將 ASP.NET 的資料傳遞給 ASP, 可以使用的方法有 Query String、Form 和 Cookies, 例如 ASP.NET 用下列的指令：

```
HttpCookie cookie = new HttpCookie("data");
cookie.Value = "datavalue";
cookie.Expires = DateTime.Now.AddMinutes(5);
Response.Cookies.Add(cookie);
Response.Redirect("Process.asp");
```

在 ASP 就可以用下列指令來接值了：

```
Dim data = Request.Cookies("data")
```

若 ASP 要傳值給 ASP.NET, 也可以用相同的方法, 例如在 ASP 中使用下列指令：

```
Response.Cookies("returnValue") = returnValue
Response.Cookies("returnValue").Expires = DateAdd("n", 1, Now())
Response.Redirect("Process.aspx")
```

接著在 ASP.NET 即可接值。

```
string returnValue = Request.Cookies["returnValue"].Value;
```

舉一個實務上的例子, 例如公司已經有一個寫好的使用者驗證的網頁 (ASP) , 但因爲某些因素, 無法將它改寫爲 ASP.NET 時, 就可以應用前面所說的來處理。

負責登入的 ASP.NET 網頁原始碼如程式 1 所示。

程式 1：登入處理的 ASP.NET 程式碼。

```
protected void Page_Load(object sender, EventArgs e)
{
```

```
  // 取回驗證結果, 若不存在代表尚未處理登入
  if (Request.Cookies["USERAUTH"] != null)
  {
    if (Request.Cookies["USERAUTH"].Value == "true")
      Response.Write("ASP RESPONSE: Passed");
    else
      Response.Write("ASP RESPONSE: FAILED");
  }
}

protected void cmdLogin_Click(object sender, EventArgs e)
{
  // 建立 cookie.
  HttpCookie username_cookie =
    new HttpCookie("UserName", this.T_UserName.Text);
  HttpCookie password_cookie =
    new HttpCookie("Password", this.T_Password.Text);

  username_cookie.Expires = DateTime.Now.AddMinutes(5);
  password_cookie.Expires = DateTime.Now.AddMinutes(5);

  Response.Cookies.Add(username_cookie);
  Response.Cookies.Add(password_cookie);

  Response.Redirect("login.asp"); // 導向到負責處理登入的 ASP 網頁程式
}
```

在ASP.NET網頁輸入登入的使用者名稱與密碼後, 程式會將這些值寫入Cookie中, 然後轉向到負責處理登入的 login.asp 中, 如程式 2。

程式 2：login.asp 程式碼 (VBScript 程式碼)

```
<%
' 檢查 UserName 與 Password 的 Cookie 資料。
if Request.Cookies("UserName") <> "root" and _
   Request.Cookies("Password") <> "rootpassword" then
  Response.Cookies("USERAUTH") = "false"  ' 驗證失敗
  Response.Cookies("USERAUTH").Expires = DATEADD("n", 1, Now())
else
  Response.Cookies("USERAUTH") = "true" ' 驗證成功
  Response.Cookies("USERAUTH").Expires = DATEADD("n", 1, Now())
end if

Response.Redirect("LoginWithASP.aspx") ' 轉向回登入網頁。
%>
```

解決方案

運用 Query String、Form 與 Cookie, 來實作資料的交換機制, 若有安全的疑慮時 (像是本範例中所示範傳遞密碼) , 可能需要加密, 但要注意 ASP 端能否能夠解密, 如此便可確保在資料交換時的安全性。

小常識　為何不能在 ASP 和 ASP.NET 之間使用 Session 來交換資料？

這也是常問的問題之一, 其根本原因和處理 ASP 和 ASP.NET 的行程有關係, 在 IIS 中, 每個處理函式庫都是以 ISAPI Filter 或 ISAPI Extension 的方式存在, ASP 的處理函式庫是 asp.dll, 而 ASP.NET 則是 aspnet_isapi.dll 以及 aspnet_wp.exe, 以作業系統的設計來看, 行程本身的資料是只能在自己的行程內部使用, 而無法跨到另一個行程 (若想做到, 要實作行程間通訊功能) , Session 不論是在 asp.dll, 或是 aspnet_isapi.dll 都是一種在內部處理的資料集合而已, 所以 ASP/ASP.NET 的 Session 無法直接交換。

小常識 爲何不用 Server.Execute() 來處理？

如果想要在 ASP.NET 中使用 Server.Execute() 來呼叫 ASP 網頁的話, 通常會看到這個訊息：

```
System.Web.HttpException: 執行 login.asp 的子要求時發生錯誤。
```

這也是因爲資料傳送的邊界問題所導致, ASP.NET 的處理常式無法直接呼叫 ASP 的處理常式, 所以無法使用 Server.Execute() 來處理這件事, 只能利用 Response.Redirect() 來呼叫 ASP 的程式。

詳細的資訊, 可以參考微軟知識庫中的文章：

```
http://support.microsoft.com/kb/320439/en-us
```

Q74 如何動態載入自訂的使用者控制項？

適用範圍： V ASP.NET 1.0 V ASP.NET 1.1 V ASP.NET 2.0 V ASP.NET 3.5

問題

我有一個使用者控制項, 是用來繪製銷售統計圖表的, 它有一個參數是店面的識別碼, 我想要在網頁載入時, 能夠設定這個值, 讓使用者控制項能夠依據店面來顯示銷售的統計圖, 請問我應該如何做？

問題說明

使用者控制項在 Web 應用程式中, 扮演著相當重要的角色, 除了簡化控制項的發展以外, 它還可以動態部署多個相同的控制項, 或者是依照條件參數來產生控制項的內容, 不過由於使用者控制項在預設的情況下是固定的內容, 所以需要一點小小的手段, 才能夠讓它有這樣的能力。

不論是使用者控制項、自訂控制項或是繼承控制項, 它們都不脫一個本質：物件類別 (Class) , 熟悉物件導向程式設計的開發人員都知道, 一個類別可以存在有屬性 (Property) 、方法 (Method) 與事件 (Events) , 所以使用者控制項是可以有屬性的, 開發人員可以透過屬性來設定參數, 當然也可以開放方法或事件出來。另一個特性, 就是執行期型別資訊 (Runtime Type Information；RTTI) , 在.NET Framework 中的 System.Reflection 命名空間, 就是 RTTI 的支援類別, 外部程式可以透過 Reflection 的能力, 對於物件類別做成員的搜尋、呼叫與引動 (Invoke) , 其中亦包含屬性值的擷取與設定, 因此開發人員可以利用這樣的能力, 靈活的操作使用者控制項。

首先, 使用者控制項不能先放到頁面, 而是以 PlaceHolder 控制項先保留位置, 然後在程式中載入使用者控制項, 這個工作可以由 Page.LoadControl()來做到：

```
Control ctl = Page.LoadControl("ctl.ascx");
```

經由 Page.LoadControl()讀入的控制項, 由於是一個未知型別的控制項, 故以基底物件 Control 爲型別, 接著, 開發人員可以利用 System.Reflection 命名空間中的功能來呼叫在控制項中的屬性：

```
ctl.GetType().GetProperty("StoreID").SetValue(ctl, strStoreID, null);
```

當屬性或必要的程序設定完成後, 把使用者控制項放到先前預放的 PlaceHolder 控制項中, 就大功告成了：

```
phHolder.Controls.Add(ctl);
```

使用者控制項中也有一個 Page_Load 事件常式, 這個常式會在控制項被插入到父控制項時, 才會被呼叫, 在 Page.LoadControl()產生時還不會被呼叫, 因此控制項內容產生的程式可以放在 Page_Load 事件常式中, 只要在這之前先把必要程序做好, 那麼 Page_Load 中的事件常式即可正常運作。

例如, 這是一個紅綠燈的控制項：

程式碼1：紅綠燈使用者控制項。

```
<%@ Control Language="C#" AutoEventWireup="true"
  CodeFile="Light.ascx.cs"
  Inherits="CorporationContent_Controls_Light" %>
<table width="90px" border="0" cellspacing="1" cellpadding="1">
  <tr>
    <td style="width: 30px">
      <asp:Image ID="G_Icon" ImageUrl="~/images/GYR/G_Gray_30.png"
        ImageAlign="AbsMiddle" runat="server"
        Width="30" Height="30" />
```

```
    </td>
    <td style="width: 30px">
      <asp:Image ID="Y_Icon" ImageUrl="~/images/GYR/Y_Gray_30.png"
        ImageAlign="AbsMiddle" runat="server"
        Width="30" Height="30" />
    </td>
    <td style="width: 30px">
      <asp:Image ID="R_Icon" ImageUrl="~/images/GYR/R_Gray_30.png"
        ImageAlign="AbsMiddle" runat="server"
        Width="30" Height="30" />
    </td>
  </tr>
</table>
```

程式碼2：紅綠燈使用者控制項的程式碼

```
public partial class LightControl : System.Web.UI.UserControl
{
  public enum DisplayLightFlag
  {
    Green, Yellow, Red
  }

  private DisplayLightFlag _displayFlag = DisplayLightFlag.Green;

  protected void Page_Load(object sender, EventArgs e)
  {
```

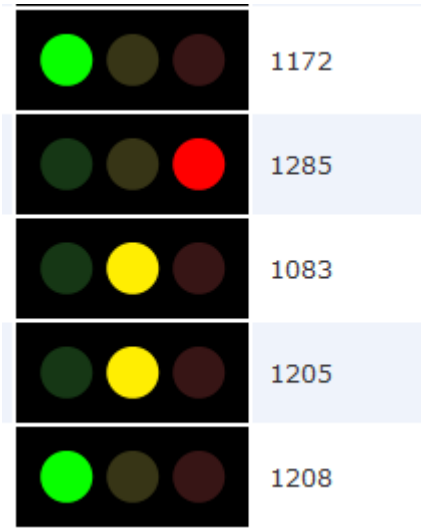
```
    switch (this._displayFlag)
    {
      case DisplayLightFlag.Green:
        this.G_Icon.ImageUrl = "~/images/GYR/G_Light_30.png";
        this.R_Icon.ImageUrl = "~/images/GYR/R_Gray_30.png";
        this.Y_Icon.ImageUrl = "~/images/GYR/Y_Gray_30.png";
        break;

      case DisplayLightFlag.Red:
        this.G_Icon.ImageUrl = "~/images/GYR/G_Gray_30.png";
        this.R_Icon.ImageUrl = "~/images/GYR/R_Light_30.png";
        this.Y_Icon.ImageUrl = "~/images/GYR/Y_Gray_30.png";
        break;

      case DisplayLightFlag.Yellow:
        this.G_Icon.ImageUrl = "~/images/GYR/G_Gray_30.png";
        this.R_Icon.ImageUrl = "~/images/GYR/R_Gray_30.png";
        this.Y_Icon.ImageUrl = "~/images/GYR/Y_Light_30.png";
        break;
    }
  }

  public DisplayLightFlag Flag
  {
    set { this._displayFlag = value; }
  }
}
```

它顯示起來是這個樣子的：



由於要視給定的參數, 來決定紅綠燈的狀態, 因此就需要用到前面所提的方法, 所以在放置紅綠燈的頁面中, 會對紅綠燈控制項來設定屬性, 決定它所要顯示的燈號, 也就是下列的程式碼：

程式 3：使用紅綠燈控制項的網頁中, 設定紅綠燈屬性的程式碼。

```
switch (Convert.ToInt32(row["State"]))
{
  case 0:
    ctl = Page.LoadControl("~/Controls/Light.ascx");
    ctl.GetType().GetProperty("Flag").SetValue(ctl,
      Control.DisplayLightFlag.Green, null);
    phCtl.Controls.Add(ctl);
    break;
```

```
  case 1:
    ctl = Page.LoadControl("~/Controls/Light.ascx");
    ctl.GetType().GetProperty("Flag").SetValue(ctl,
      Control.DisplayLightFlag.Yellow, null);
    phCtl.Controls.Add(ctl);
    break;

  case 2:
    ctl = Page.LoadControl("~/Controls/Light.ascx");
    ctl.GetType().GetProperty("Flag").SetValue(ctl,
      Control.DisplayLightFlag.Red, null);
    phCtl.Controls.Add(ctl);
    break;

  default:
    ctl = Page.LoadControl("~/Controls/Light.ascx");
    ctl.GetType().GetProperty("Flag").SetValue(ctl,
      Control.DisplayLightFlag.Red, null);
    phCtl.Controls.Add(ctl);
    break;
}
```

解決方案

首先, 先在使用者介面預留一個放置控制項的位置, 並在使用者控制項中開放一個屬性做爲控制參數, 然後利用 Reflection 的功能來動態設定控制項中的屬性, 最後再將控制項插入預留的位置即可。

考生停看聽

本問題所討論的內容, 可用來準備 Exam 70-528: TS: Microsoft .NET Framework 2.0 Web Client Development 的下列主題：

- 建立 Web 的自訂控制項
  - 建立複合式 Web 應用程式控制項
    - 建立使用者控制項。
    - 操作使用者控制項屬性。
    - 以程式建立使用者控制項的實體。
    - 於 Code-Behind 方式發展使用者控制項。

相關問題

- Q7：我需要的功能內建的控制項做不到, 我該怎麼辦？