

# ASP.NET 網頁程式設計（使用 C#）

CSIE NTU

# LinkButton 控制項

2

- **LinkButton** 的外觀上為一個超連結，實值上的功能和 **Button** 控制項一模一樣
- 常用屬性
  - ▣ **Text** – 設定要顯示的文字
  - ▣ **Visible** – 是否顯示控制項
- 常用事件
  - ▣ **Click** – 按一下時發生

# HyperLink 控制項

3

- HyperLink 控制項可用來建立文字或圖片超連結
- 常用屬性
  - ▣ Text – 設定要顯示的文字
  - ▣ ImageUrl – 設定要顯示的圖片(較高優先權)
  - ▣ NavigateUrl – 設定要連結的網頁
  - ▣ Target – 設定連結網頁顯示位置
    - \_blank – 網頁顯示於無框架的新視窗中
    - \_parent – 網頁顯示於包含作用的 frameset 標記框架中
    - \_search – 網頁顯示於搜尋視窗
    - \_self – 網頁顯示於超連結所在框架中
    - \_top – 網頁顯示於無框架的完整視窗中

# 實例探討 sample2-a1

4

- 程式功能
  - ▣ 簡易超連結
- 程式內容

```
protected void Page_Load(object sender,  
    System.EventArgs e)  
{  
    yahooHL.NavigateUrl = "http://tw.yahoo.com";  
    yahooHL.Target = "_blank";  
}
```

# Image 控制項

5

- Image 控制項主要用來顯示圖片
- 常用屬性
  - ▣ ImageUrl – 設定要顯示的圖片
  - ▣ AlternateText – 無法顯示圖片時的替代文字
- 圖片大小顯示設定
  - ▣ 固定大小：設定 Height 及 Width 屬性
  - ▣ 原始大小：Height 及 Width 屬性設為空白

# 課堂練習 sample2-b1

6

- 程式功能
  - ▣ 圖片瀏覽
- 基本概念
  - ▣ 利用 `ImageUrl` 屬性來更改 `Image` 控制項顯示的圖片

# ImageButton 控制項

7

- **ImageButton** 控制項使用圖片作為外觀顯示，其於功能與 **Button** 控制項非常相像
- 常用屬性
  - ▣ **ImageUrl** – 設定要顯示的圖片
  - ▣ **AlternateText** – 無法顯示圖片時的替代文字
- 常用事件
  - ▣ **Click** – 按一下時發生  
(事件處理程式的第二個參數為滑鼠指標按下時的座標位置，因此可作為影像地圖使用)

# 實例探討 sample2-a2 (1)

8

- 程式功能
  - ▣ 影像地圖
  - ▣ 利用 `onmouseover` 與 `onmouseout` 改變圖片

- 程式內容 - HTML 模式部份

```
<asp:ImageButton onmouseover="this.src='carLogo.jpg' "  
onmouseout="this.src='carText.jpg' " ...
```



# 實例探討 sample2-a2 (2)

9

## □ 程式內容 – WebForm1.aspx

```
protected void carLB_Click(object sender,
    System.Web.UI.ImageClickEventArgs e)
{
    if(e.X<=100)
        textLB.Text = "原來您喜歡BMW!!";
    else
        textLB.Text = "原來您喜歡BENZ!!";
}
```

# 課堂練習 sample2-b2

10

- 程式功能
  - ▣ 圖型式密碼輸入
- 基本概念
  - ▣ 可先將使用者按下的數字存於 **Label** 中，並將 **Label** 的 **Visible** 屬性設定為 **false**
  - ▣ 可先將使用者按下的數字儲存於 **Cookie** 或 **Session** 當中

# DropDownList 控制項

11

- DropDownList 控制項提供下拉式選單功能
- 常用屬性
  - ▣ Items – 設定下拉式選單內容  
(items 為 ListItemCollection 類別物件)  
(選單項目為 ListItem 類別物件)
  - ▣ AutoPostBack – 設定選取項目變更時是否通知伺服器
- 常用事件
  - ▣ SelectedIndexChanged – 當選取項目變更時發生  
(需將 AutoPostBack 屬性設定為 true)

# ListItemCollection 類別

12

- **DropDownList** 的 **items** 屬性為 **ListItemCollection** 類別物件 (為一集合)
  - ▣ 可直接透過索引取得集合中指定物件
  - ▣ ex : `DropDownList1.Items[1]`
- 常用屬性
  - ▣ **Count** – 取得集合中 **ListItem** 物件的數目
  - ▣ **SelectedItem** – 取得選取項目物件
  - ▣ **SelectedIndex** – 取得選取項目物件索引值
- 常用方法
  - ▣ **Add** – 新增項目至集合中 (**ListItem**、**string**)
  - ▣ **Clear** – 清除所有項目
  - ▣ **Remove** – 移除指定項目 (**ListItem**、**string**)
  - ▣ **RemoveAt** – 移除指定索引項目 (**int**)

# ListItem 類別

13

- **ListItemCollection** 類別物件為一集合，集合中每一元素為 **ListItem** 類別物件
- 常用屬性
  - ▣ **Text** – 設定項目顯示文字
  - ▣ **Value** – 設定項目對應的關聯值
  - ▣ **Selected** – 項目是否已被選取

# 實例探討 sample2-a3 (1)

14

- 程式功能
  - ▣ 藝人簡介
- 程式內容

```
struct star
{
    public string name;
    public string country;
    public string imgUrl;
}
```

# 實例探討 sample2-a3 (2)

15

## □ 程式內容

```
star[] myStar = new star[3];  
protected void Page_Load(object sender, EventArgs  
    e)  
{  
    myStar[0].name = "S.H.E";  
    myStar[0].country = "台灣";  
    myStar[0].imgUrl = "she.jpg";  
    myStar[1].name = "宋慧喬";  
    myStar[1].country = "韓國";  
    myStar[1].imgUrl = "宋慧喬.jpg";
```

# 實例探討 sample2-a3 (3)

16

## □ 程式內容

```
myStar[2].name = "孫燕姿";  
myStar[2].country = "新加坡";  
myStar[2].imgUrl = "孫燕姿.jpg";
```

```
if(!IsPostBack)  
{  
    foreach(star starItem in myStar)  
        starDDL.Items.Add(starItem.name);  
    starDDL.AutoPostBack = true;  
    changeInfo();  
}  
}
```



# 實例探討 sample2-a3 (4)

17

## □ 程式內容

```
protected void starDDL_SelectedIndexChanged(object sender,
    System.EventArgs e)
{
    changelInfo();
}
protected void changelInfo()
{
    nameLB.Text = myStar[starDDL.SelectedIndex].name;
    countryLB.Text = myStar[starDDL.SelectedIndex].country;
    pictureImg.ImageUrl = myStar[starDDL.SelectedIndex].imgUrl;
}
```

# Listbox 控制項

18

- **Listbox** 控制項與 **DropDownList** 控制項相似，提供清單方塊功能
- 常用屬性
  - ▣ **Items** – 設定清單內容  
(同為 **ListItemCollection** 類別物件)
  - ▣ **SelectionMode** – 設定選取模式
    - **Single** – 單選
    - **Multiple** – 多選  
(多選情況可利用迴圈檢查每一項目的 **Selected** 屬性是否為 **true**，以判斷已被選取項目)

# 實例探討 sample2-a4 (1)

19

- 程式功能
  - ▣ 購物清單
- 程式內容

```
protected void purchaseBTN_Click(object sender,
    System.EventArgs e)
{
    shoppingLB.Items.Add(itemLB.SelectedItem);
    itemLB.Items.Remove(itemLB.SelectedItem);
    shoppingLB.SelectedIndex = -1;
    totalLB.Text = total().ToString();
}
```

# 實例探討 sample2-a4 (2)

20

## □ 程式內容

```
protected void cancelBTN_Click(object sender,  
    System.EventArgs e)  
{  
    itemLB.Items.Add(shoppingLB.SelectedItem);  
    shoppingLB.Items.Remove(shoppingLB.SelectedItem);  
    itemLB.SelectedIndex = -1;  
    totalLB.Text = total().ToString();  
}
```

# 實例探討 sample2-a4 (3)

21

## □ 程式內容

```
protected int total()
{
    int sum = 0;
    foreach(ListItem item in shoppingLB.Items)
    {
        sum += int.Parse(item.Value);
    }
    return sum;
}
```

# 課堂練習 sample2-b3

22

- 程式功能
  - ▣ 書籍分類清單
  - ▣ 根據使用者所選擇的下拉式選單項目，改變書籍清單內容

小說	射鵬英雄傳	神鵬俠侶	倚天屠龍記
電腦	C# 程式設計入門	十天學會 JAVA	資料庫程式設計
食譜	韓式料理	日式料理	台式料理

# CheckBox 控制項

23

- **CheckBox** 控制項提供核對方塊功能
- 常用屬性
  - ▣ **Text** – 設定標題文字
  - ▣ **TextAlign** – 設定標題文字顯示位置
  - ▣ **Checked** – 核對方塊是否已被勾選
- 常用事件
  - ▣ **CheckedChanged** – 當勾選狀態改變時發生  
(需將 **AutoPostBack** 屬性設定為 **true**)

# CheckBoxList 控制項

24

- **CheckBoxList** 控制項提供多重選取核對方塊群組
- 常用屬性
  - ▣ **Items** – 設定清單內容  
(同為 **ListItemCollection** 類別物件)
  - ▣ **RepeatColumns** – 設定資料欄位數
  - ▣ **RepeatDirection** – 設定配置方向順序
- 常用事件
  - ▣ **SelectedIndexChanged** – 當選取項目變更時發生  
(需將 **AutoPostBack** 屬性設定為 **true**)



# RadioButton 控制項

25

- **RadioButton** 控制項提供單選選項按鈕功能
- 常用屬性
  - ▣ **Text** – 設定標題文字
  - ▣ **TextAlign** – 設定標題文字顯示位置
  - ▣ **Checked** – 核對方塊是否已被勾選
  - ▣ **GroupName** – 設定所屬群組
- 常用事件
  - ▣ **CheckedChanged** – 當勾選狀態改變時發生  
(需將 **AutoPostBack** 屬性設定為 **true**)

# RadioButtonList 控制項

26

- **RadioButton** 控制項可用來建立單選選項按鈕群組
- 常用屬性
  - ▣ **Items** – 設定清單內容  
(同為 **ListItemCollection** 類別物件)
  - ▣ **RepeatColumns** – 設定資料欄位數
  - ▣ **RepeatDirection** – 設定配置方向順序
- 常用事件
  - ▣ **SelectedIndexChanged** – 當選取項目變更時發生  
(需將 **AutoPostBack** 屬性設定為 **true**)

# 實例探討 sample2-a5 (1)

27

- 程式功能

- ▣ 會員資料輸入

- 程式內容 – WebForm1.aspx

```
protected void submitBTN_Click(object sender,  
    System.EventArgs e)  
{  
    Session.Add("name",nameTB.Text);  
    Session.Add("sex",sex1RB.Checked?"先生":"小姐");  
    Session.Add("education",  
        educationRBL.SelectedItem.Text);
```

# 實例探討 sample2-a5 (2)

28

## □ 程式內容 - WebForm1.aspx

```
string hobby = "";  
foreach(ListItem item in hobbyCBL.Items)  
    if(item.Selected) hobby += item.Text + " ";  
  
if(hobby == "") hobby = "無";  
Session.Add("hobby", hobby);  
  
Response.Redirect("WebForm2.aspx");  
}
```

# 實例探討 sample2-a5 (3)

29

## □ 程式內容 – WebForm2.aspx

```
protected void Page_Load(object sender, EventArgs e)
{
    Response.Write(Session["name"].ToString());
    Response.Write(Session["sex"].ToString() + "您好!!<br>");
    Response.Write("您的學歷爲  

        "+Session["education"].ToString() + "<br>");
    Response.Write("您的興趣爲    "+Session["hobby"].ToString());
}
```

# 課堂練習 sample2-b4

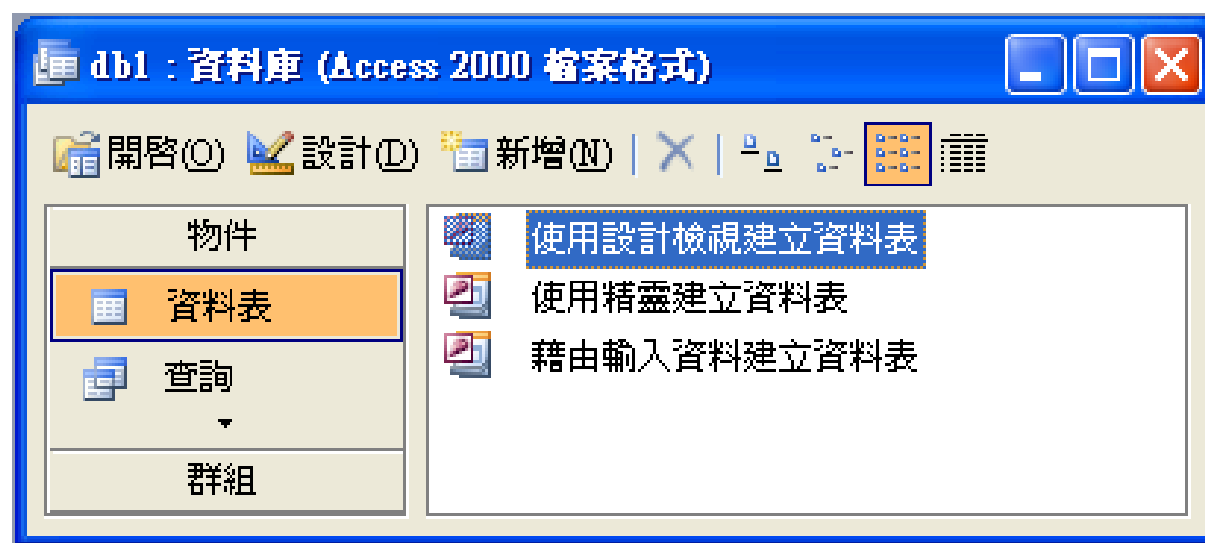
30

- 程式功能
  - ▣ 電腦採購
- 基本概念
  - ▣ 可將選項價格儲存於 `ListItem` 的 `Value` 屬性中

# Access 資料庫 (1)

31

- 建立資料庫
  - ▣ 選擇 [檔案] 中的 [開新檔案] 會跳出開新檔案視窗，在視窗新增部份選取 [空白資料庫]
  - ▣ 設定完檔案儲存位置後會跳出下圖視窗



# Access 資料庫 (2)

32

- 建立資料表
  - 選擇資料庫視窗中的 [使用設計檢視建立資料表]

The screenshot shows the 'Table Design' view in Microsoft Access. The title bar reads '資料表1 : 資料表'. The design grid has three columns: '欄位名稱' (Field Name), '資料類型' (Data Type), and '描述' (Description). The first two rows are populated: 'id' with data type '數字' (Number) and 'name' with data type '文字' (Text). Below the grid is the '欄位內容' (Field Properties) section, which is divided into '一般' (General) and '查閱' (Queries) tabs. The '一般' tab is active, showing various properties for the selected field. The '欄位大小' (Field Size) property is set to '長整數' (Long Integer). The '格式' (Format) property is set to '自動' (Automatic). The '小數位數' (Decimal Places) property is set to '0'. The '預設值' (Default Value) property is set to '0'. The '驗證規則' (Validation Rule) property is set to '是' (Yes). The '驗證文字' (Validation Text) property is set to '是(不可重複)' (Yes (cannot repeat)). The '必須有資料' (Required) property is set to '是' (Yes). The '索引' (Index) property is set to '是(不可重複)' (Yes (cannot repeat)). The '智慧標籤' (Smart Tags) property is empty.

欄位名稱	資料類型	描述
id	數字	
name	文字	

欄位內容

一般 查閱

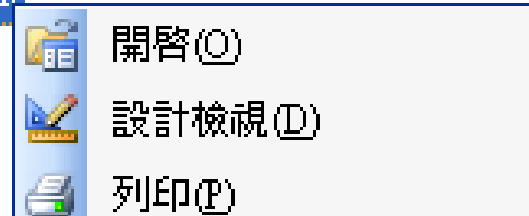
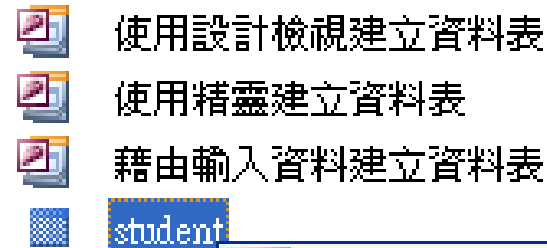
欄位大小	長整數
格式	自動
小數位數	0
輸入遮罩	
標題	
預設值	0
驗證規則	是
驗證文字	是(不可重複)
必須有資料	是
索引	是(不可重複)
智慧標籤	



# Access 資料庫 (3)

33

- 儲存資料表
  - ▣ 當關閉資料表視窗時，系統即會詢問是否要儲存資料表等資訊
- 資料表操作
  - ▣ 開啓
    - 輸入資料
  - ▣ 設計檢視
    - 修改資料表欄位設定



# ADO .NET

34

- ADO.NET 爲 .NET Framework 所提供的類別，專門用來存取『資料儲存體』中的資料
- 『資料儲存體』包含資料庫或非資料庫型式的儲存體
  - ▣ Ex：MSSQL、Access、Excel、XML
- ADO.NET 採用中斷連線的方式，減低系統工作負載

# .NET 資料提供者

35

- .NET 資料提供者為存取資料來源的一組類別
- .NET Data Provider
  - ▣ SQL .NET Data Provider
    - 命名空間：System.Data.SqlClient
    - 可用資料庫：MS-SQL
  - ▣ OLE DB .NET Data Provider
    - 命名空間：System.Data.OleDb
    - 可用資料庫： Dbase、FoxPro、Excel、Access、Oracle、Access...
  - ▣ ODBC .NET Data Provider
    - 需至微軟網站另外下載
    - 可用資料庫： MySQL...

# DataAdapter 物件

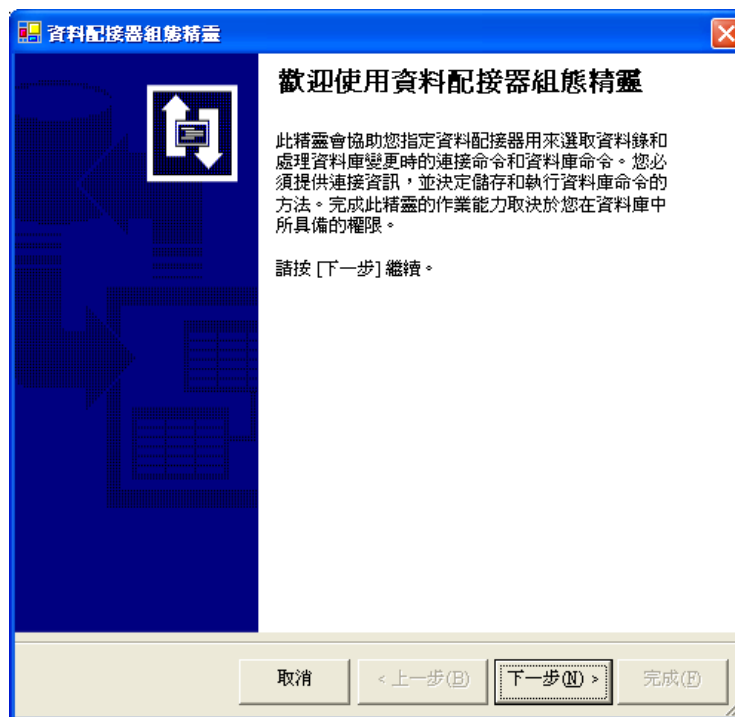
36

- DataAdapter 物件為資料來源與 DataSet 之間的溝通橋樑
- 常用方法
  - ▣ Fill – 將資料來源資料內容填入資料集當中
  - ▣ Update – 更新資料來源中的資料內容

# 利用 DataAdapter 建立連線 (1)

37

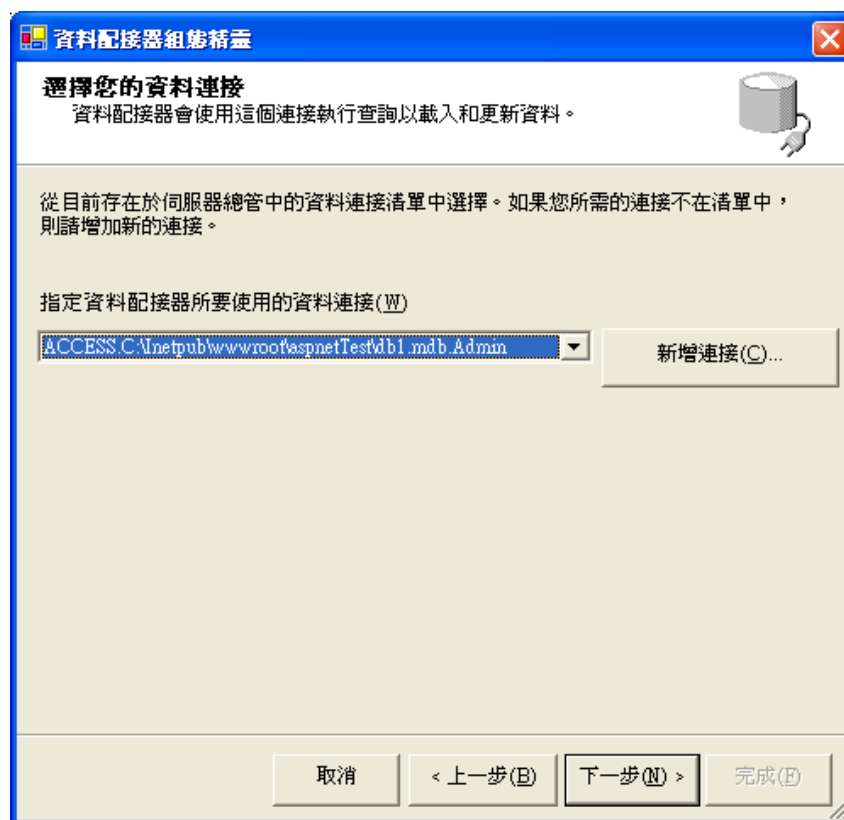
- 當我們從工具箱資料類別分類中拖拉 **DataAdapter** 物件時，即會產生組態設定精靈視窗



# 利用 DataAdapter 建立連線 (2)

38

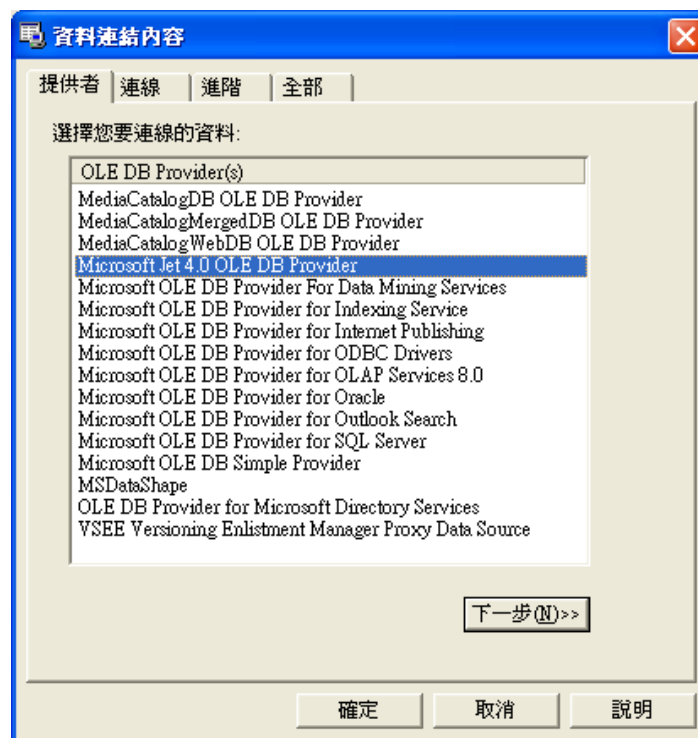
- 選擇新增連接來建立連線



# 利用 DataAdapter 建立連線 (3)

39

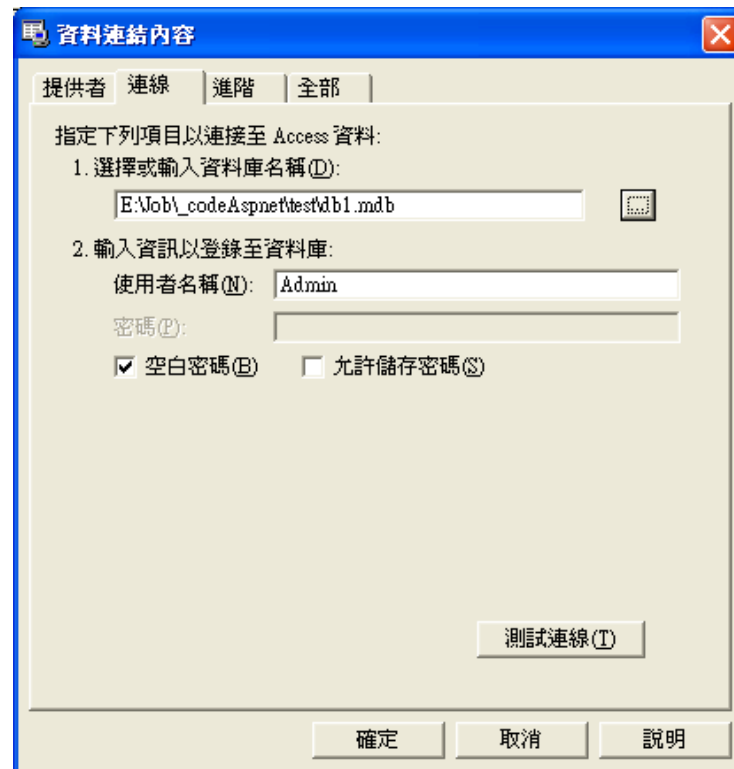
- 我們所採用的是 ACCESS 資料庫，因此提供者必須選取 Microsoft Jet 4.0 OLE DB Provider



# 利用 DataAdapter 建立連線 (4)

40

- 在連線部份設定好資料來源檔案位置，並測試連線是否正常

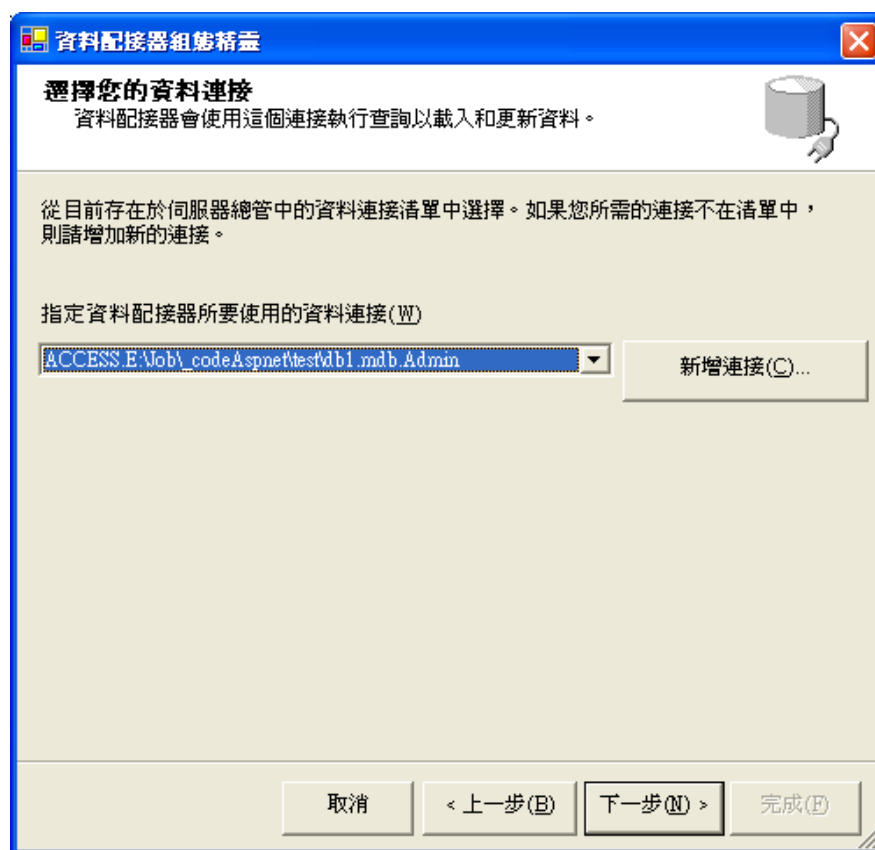




# 利用 DataAdapter 建立連線 (5)

41

- 連線正常後按下確定按鈕即可產生連接設定記錄



# 利用 DataAdapter 建立連線 (6)

42

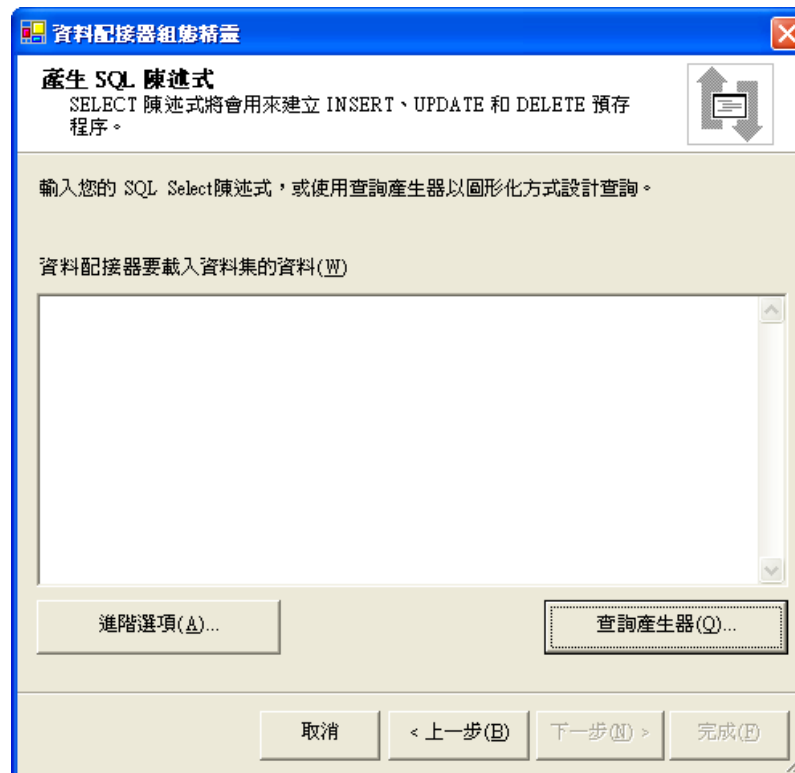
- 設定存取方式部份選擇使用 SQL 陳述式



# 利用 DataAdapter 建立連線 (7)

43

- 在產生 SQL 陳述式設定中，可點選查詢產生器來產生欲執行的 SQL 查詢語法



# 利用 DataAdapter 建立連線 (8)

44

- 選擇欲查詢的資料表



# 利用 DataAdapter 建立連線 (9)

45

- 勾選設定欲查詢的欄位，按下確定後即可產生對應的 SQL 語法

查詢產生器

book

- ☒ \* (所有資料行)
- ☐ 出版社
- ☐ 作者
- ☐ 書名
- ☐ 編號

資料行	別名	資料表	輸出	排序類型	排序次序	準則
*		book	✓			

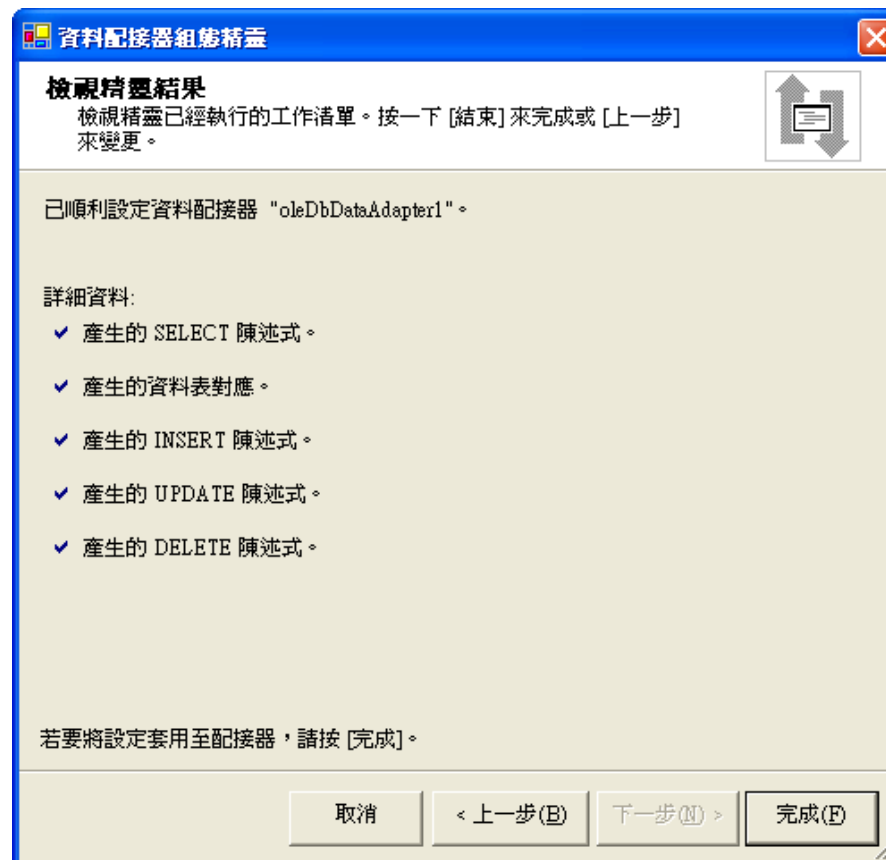
```
SELECT book.*  
FROM book
```

確定 取消

# 利用 DataAdapter 建立連線 (10)

46

## □ 完成 DataAdapter 設定



# 利用 DataAdapter 建立連線 (11)

47

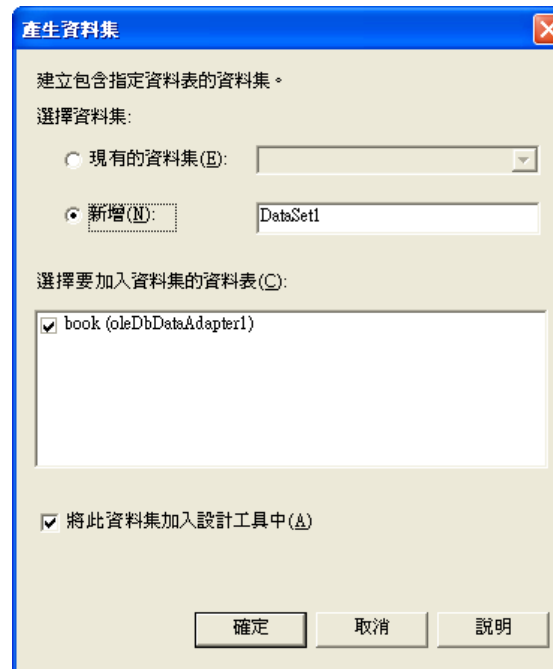
- 在 DataAdapter 屬性視窗中，可建立對應的 DataSet (資料集)



# 利用 DataAdapter 建立連線 (12)

48

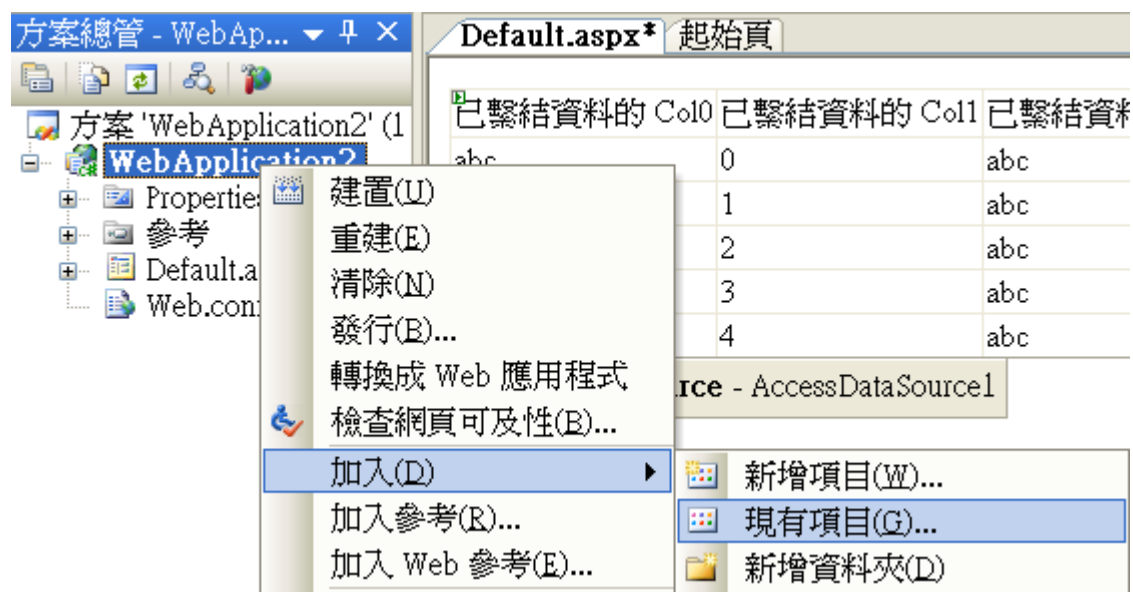
- 按下確定即可產生 DataSet (資料集)，再利用 DataAdapter 中 Fill 方法即可將所設定的查詢內容填入 DataSet 當中





# 新增資料庫至方案總管

- 打開方案總管
- 方案上點右鍵，選取「加入」→「現有項目」
- 選取欲使用的資料庫



# DataGrid 控制項 (1)

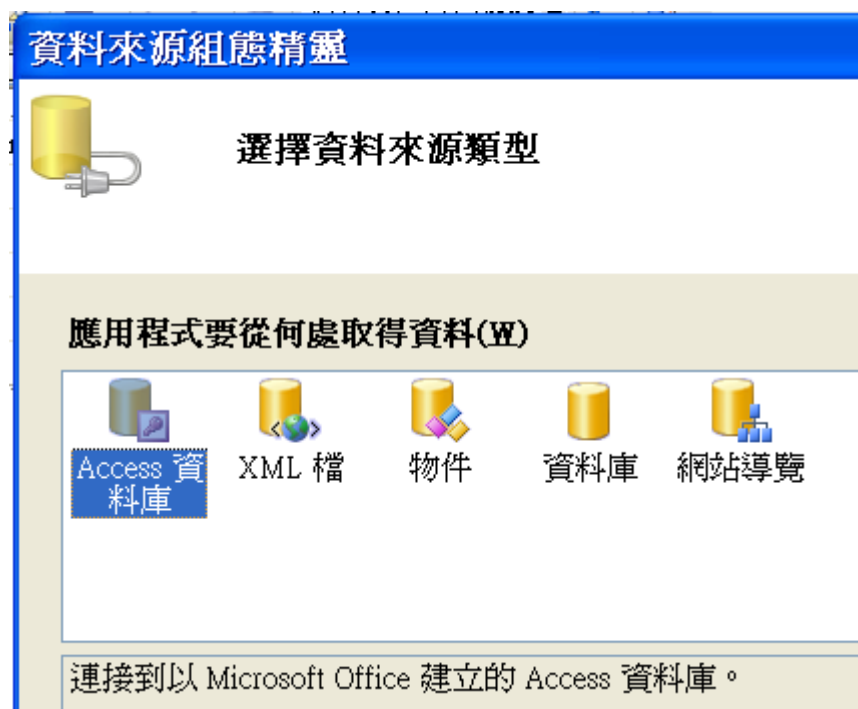
50

- **DataGrid** 控制項以表格方式顯示資料，適用於表現資料庫內容
- 常用屬性
  - ▣ **DataSource** – 設定資料來源
  - ▣ **AutoGenerateColumns** – 自動產生資料欄位
  - ▣ **Columns** – 設定欲包含的欄位
  - ▣ 自動格式化 – 內建色彩配置
  - ▣ 屬性產生器 – 自訂各項屬性內容
- 常用方法
  - ▣ **DataBind** – 進行資料繫結動作

# DataGrid 控制項 (2)

51

- 設定DataSource
  - ▣ 選擇資料來源 <新資料來源...>
  - ▣ 在設定「資料來源視窗」，選擇資料庫。



# 實例探討 sample2-a6

52

- 程式功能
  - ▣ 將資料庫內容顯示於 **DataGrid** 控制項中
- 程式內容

```
protected void Page_Load(object sender,  
    System.EventArgs e)  
{  
    DataGrid1.DataBind();  
}
```

# 課堂練習 sample2-b5

53

- 資料庫內容
  - ▣ 建立一會員帳號資料表
  - ▣ 包含帳號及密碼二個欄位
- 程式功能
  - ▣ 將資料庫內容顯示於 **DataGrid** 控制項中
- 基本概念
  - ▣ 利用 **DataAdapter** 建立所需連線

# 資料行類型

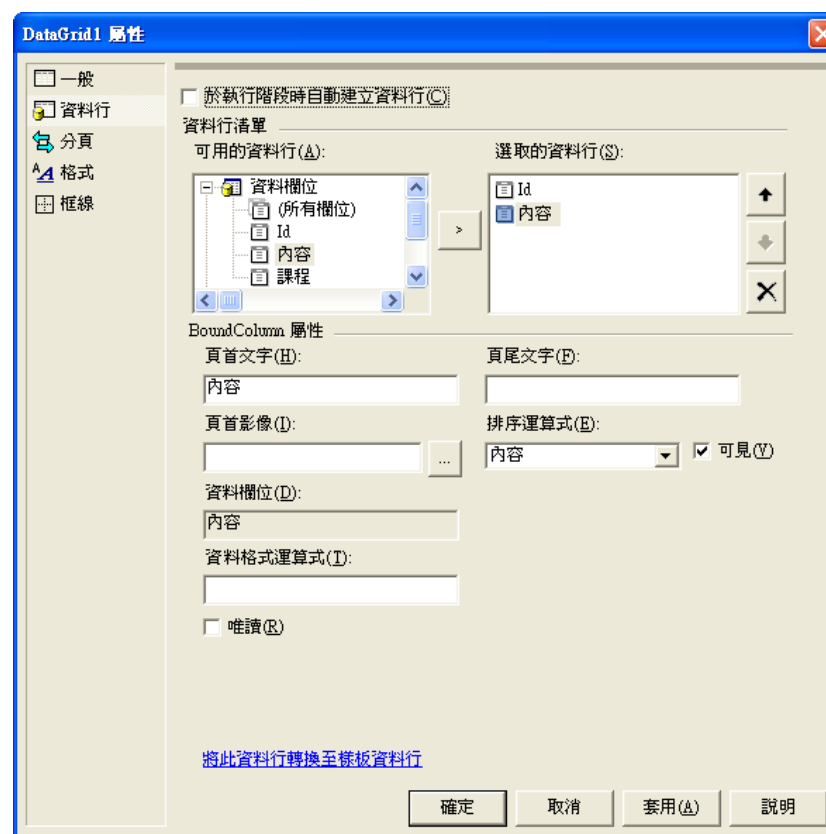
54

- 繫結資料行
  - ▣ 顯示資料來源指定的欄位
- 按鈕資料行
  - ▣ 建立可處理該筆資料的按鈕
- 超連結資料行
  - ▣ 將資料行內容以超連結顯示
- 樣板資料行
  - ▣ 可利用其它控制項來自訂顯示配置樣式

# 繫結資料行

55

- 繫結資料行可用來顯示資料來源指定繫結欄位的資料內容
  - 頁首文字 -  
DataGrid 上所顯示的欄位名稱
  - 資料欄位 -  
指定繫結資料來源的欄位名稱

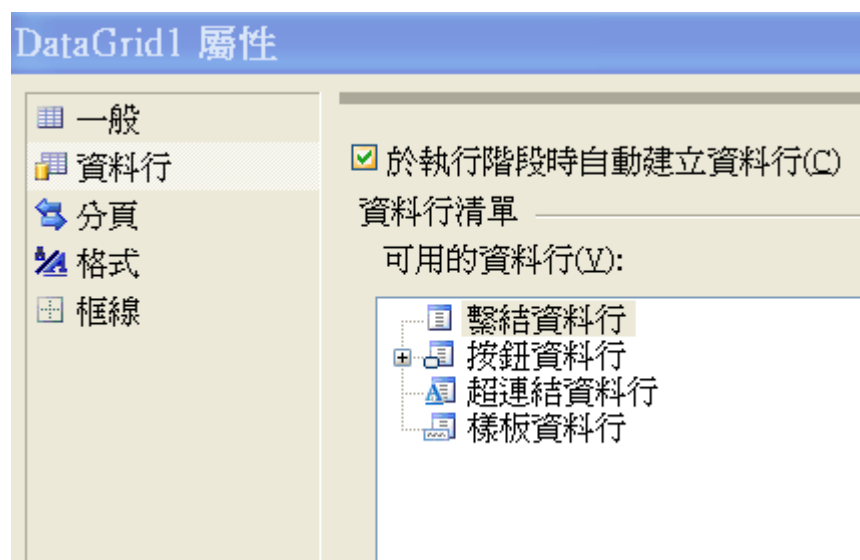


# 編輯DataGrid資料行

- 點選「屬性產生器」



- 點選「資料行」





# 選取按鈕資料行

57

- 按鈕資料行包含選取按鈕、編輯更新取消按鈕及刪除按鈕
- 選取按鈕
  - ▣ 文字 - 選取按鈕上欲顯示的文字 (全一致)
  - ▣ 文字欄位 - 選取按鈕上欲顯示的資料內容  
(繫結)
- 用法
  - ▣ 可透過 **SelectedIndexChanged** 事件捕捉使用者選取狀況
  - ▣ 可利用 **SelectedItem** 屬性取得被選取的資料列

# DataGrid 儲存格資料

58

- 欲取得 DataGrid 某一儲存格資料，可藉由 Items 屬性中的 Cells 屬性來達成
  - ▣ Items 屬性 - 儲存 DataGrid 資料列的集合
  - ▣ Cells 屬性 - 儲存資料列的儲存格的集合
- 用法
  - ▣ 取得控制項中第一列第三欄儲存格的資料 - `DataGrid1.Items[1].Cells[3].Text;`

# 實例探討 sample2-a7

59

- 程式功能
  - ▣ 將購物資訊顯示於 DataGrid 控制項中
  - ▣ 計算總消費金額

- 程式內容

```
protected void Page_Load(object sender, EventArgs e)
{
    DataGrid1.DataBind();
    int total = 0;
    foreach(DataGridItem item in DataGrid1.Items)
        total += int.Parse(item.Cells[1].Text);
    totalLB.Text = total.ToString();
}
```

# 課堂練習 sample2-b6

60

- 資料庫內容
  - ▣ 建立一學生成績資料表
  - ▣ 包含姓名、國文、英文及數學四個欄位
- 程式功能
  - ▣ 計算全班成績總平均
  - ▣ 設計選取欄位可供選取
  - ▣ 列出被選取學生各科成績、總分及平均等資料
  - ▣ 預設選取第一位學生
- 基本概念
  - ▣ 利用選取資料行來顯示特定學生資料
  - ▣ 需將 `AutoGenerateColumns` 設定為 `false`

# 超連結資料行

61

- 超連結資料行能將資料行中各項內容以超連結顯示
  - ▣ 文字 - 欲顯示的文字 (全一致)
  - ▣ 文字欄位 - 欲顯示的資料內容 (繫結)
  - ▣ 目標 - 網頁開啓位置
  - ▣ URL 欄位 - 繫結超連結目標
  - ▣ URL 格式字串 - 可透過此屬性傳遞網址列參數
- 用法
  - ▣ 若將 URL 格式設定為 `WebForm2.aspx?filename={0}` , 則 URL 欄位資料會填入 {0} 位置

# 實例探討 sample2-a8

62

- 程式功能
  - ▣ 我的最愛
- 程式內容

```
protected void Page_Load(object sender,  
    System.EventArgs e)  
{  
    OleDbDataAdapter1.Fill(dataSet1 1);  
    favoriteDG.DataBind();  
}
```

# 課堂練習 sample2-b7

63

- 資料庫內容
  - ▣ 建立一藝人簡介資料表
  - ▣ 包含姓名、國籍及圖片三個欄位
- 程式功能
  - ▣ 加入一超連結資料行
  - ▣ 點選超連結導至 **WebForm2.aspx** 網頁，並於該網頁顯示所點選的藝人照片
  - ▣ 以網址列傳送圖片檔案名稱
- 基本概念
  - ▣ 設定 **URL** 格式字串，使 **URL** 欄位資料能以網址列參數型式傳送

# 按鈕資料行

64

- 按鈕資料行提供自訂按鈕功能
  - ▣ 文字 - 欲顯示的文字 (全一致)
  - ▣ 文字欄位 - 欲顯示的資料內容 (繫結)
  - ▣ 命令名稱 - 執行的命令 (**CommandName**)
- 用法
  - ▣ 當按鈕資料行中的按鈕被點選時，會引發 **ItemCommand** 事件
  - ▣ **ItemCommand** 事件會將選取到的資料列傳入事件參數 **e** 當中
  - ▣ 在一個 **DataGrid** 控制項中可包含多個按鈕資料行，可藉由命令名稱來判斷出被點選到的按鈕
  - ▣ 設定成按鈕外觀時，需將 **EnableViewState** 設為 **false**



# 實例探討 sample2-a9

65

- 程式功能
  - ▣ 購物網站
- 程式內容

```
protected void Page_Load(object sender, System.EventArgs e)
{
    itemDG.DataBind();
}

protected void itemDG_ItemCommand(object source,
    System.Web.UI.WebControls.DataGridCommandEventArgs e)
{
    shoppingLB.Items.Add(e.Item.Cells[1].Text+ " - "
        +e.Item.Cells[2].Text);
    totalLB.Text = (int.Parse(totalLB.Text) +
        int.Parse(e.Item.Cells[2].Text)).ToString();
}
```

# 課堂練習 sample2-b8

66

- 資料庫內容
  - ▣ 建立一商品資訊資料表
  - ▣ 包含項目及金額二個欄位
- 程式功能
  - ▣ 修改 **sample2-a9** 多加入移除按鈕
- 基本概念
  - ▣ 利用 **CommandName** 來區分被點選的按鈕
  - ▣ 可藉由 **Items** 屬性中的 **FindByText** 等方法檢查是否含有該欲刪除項目

# 分頁

67

- **DataGrid** 控制項允許我們在資料筆數過多時，能利用分頁檢視方式來顯示資料
  - ▣ 利用屬性產生器或屬性視窗將允許分頁 (**AllowPaging**) 屬性設定為 **true**
  - ▣ 頁面大小 (**PageSize**) 可設定每頁資料筆數
  - ▣ 利用 **PageIndexChanged** 事件捕捉使用者選擇其它分頁，可從事件中物件參數 **e** 的 **NewPageIndex** 屬性得知使用者欲前往的分頁
  - ▣ 修改 **DataGrid** 中 **CurrentPageIndex** 屬性，並重新進行繫結動作

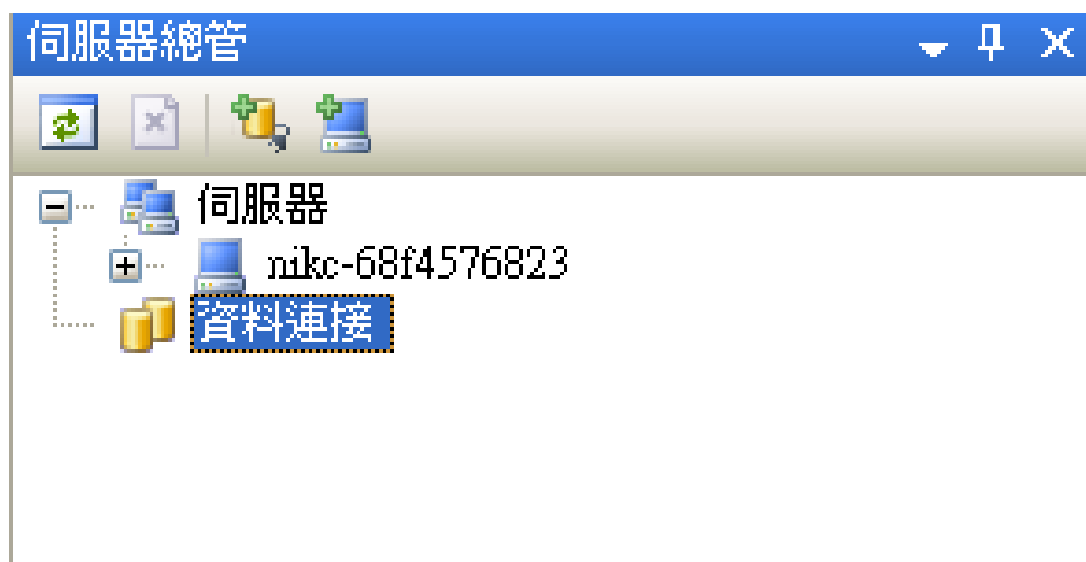
# 課堂練習 sample2-b9

68

- 資料庫內容
  - ▣ 建立一學號資料表
  - ▣ 包含學號及姓名二個欄位
  - ▣ 填入十筆學生資料
- 程式功能
  - ▣ 利用分頁方式檢視資料
  - ▣ 每一分頁顯示四位學生資料
- 基本概念
  - ▣ 利用 `PageIndexChanged` 事件參數來設定 `DataGrid` 控制項的 `CurrentPageIndex` 屬性

# 伺服器總管

- 開啓
  - ▣ 檢視→其它視窗→伺服器總管
- 「資料連接」項目
  - ▣ 設定資料庫連線、資料查詢或管理



# 建立SQL Server資料庫

- 在伺服器總管視窗，點選「資料連接」項目，再按滑鼠右鍵
- 選取「建立新的SQL Server資料庫」



# 設定連線資訊

- 伺服器名稱：
  - ▣ 「(local)\sqlexpress」
- 登入伺服器：
  - ▣ 選擇「使用Windows驗證」
- 新資料庫名稱：
  - ▣ 輸入名稱，例如「MyDB」
- 按「確定」
  - ▣ 自動建立資料庫

建立新的 SQL Server 資料庫

請輸入資訊以連接至 SQL Server，然後指定要建立的資料庫名稱。

伺服器名稱(E): (local)\sqlexpress 重新整理(R)

登入伺服器

☒ 使用 Windows 驗證(W)  
☐ 使用 SQL Server 驗證(Q)

使用者名稱(U):  
密碼(P):  
☐ 儲存我的密碼(S)

新資料庫名稱(D): MyDB

確定 取消

# 連接到SQL Server資料庫

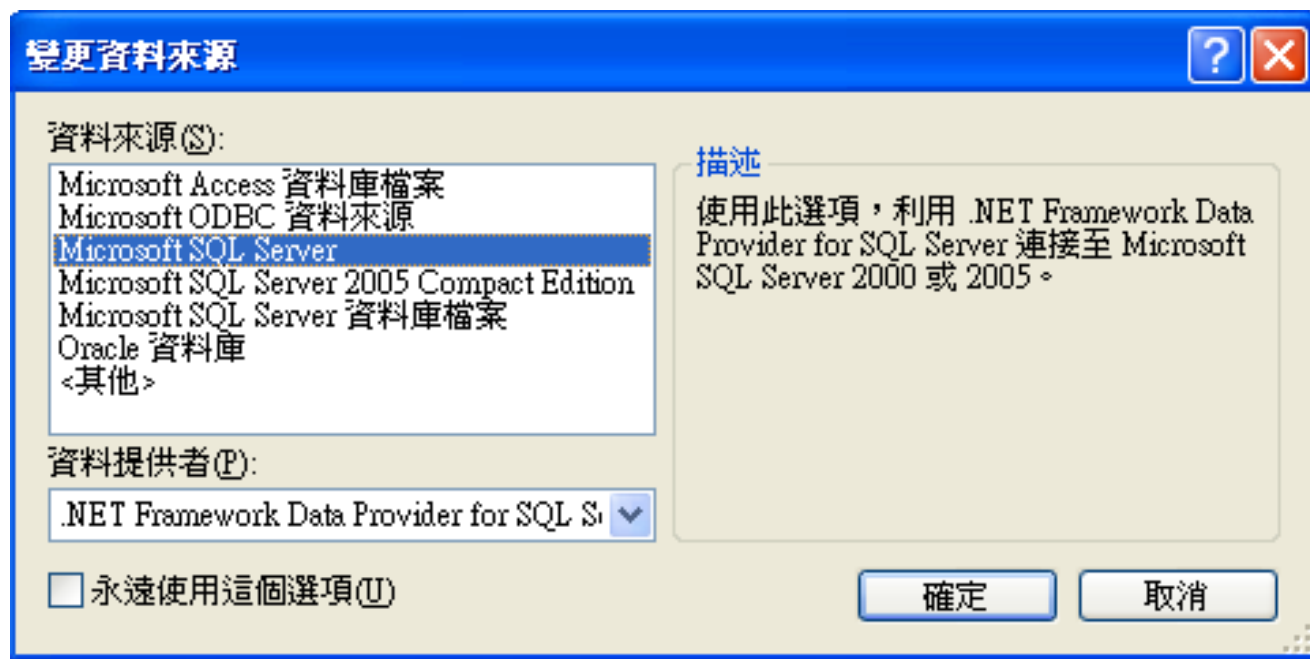
- 開啓「伺服器總管」視窗
- 在「資料連結」選項上按滑鼠右鍵
  - 選擇「加入資料連接」





# 連接到SQL Server資料庫

- 在「加入連接」視窗，選擇適用的資料來源
  - ▣ 預設使用「Microsoft SQL Server」
  - ▣ 按下「變更」按鈕，跳出「變更資料來源」視窗



# 設定「加入連接」視窗

- 伺服器名稱
  - ▣ (local)\sqlexpress
- 登入伺服器
  - ▣ 點選「使用Windows驗證」
- 資料庫
  - ▣ 選取 ADO\_NET\_DB
- 按一下「測試連接」
  - ▣ 測試連線是否成功
- 按「確定」離開

修改連接

請輸入資訊，以連接至選取的資料來源，或者按一下 [變更]，選擇不同的資料來源及/或提供者。

資料來源(S):  
Microsoft SQL Server (SqlClient) 變更(C)...

伺服器名稱(E):  
(local)\sqlexpress 重新整理(R)

登入伺服器

☒ 使用 Windows 驗證(W)  
☐ 使用 SQL Server 驗證(Q)

使用者名稱(U):  
密碼(P):  
☐ 儲存我的密碼(S)

連接至資料庫

☒ 選取或輸入資料庫名稱(D):  
ADO\_NET\_DB  
☐ 附加資料庫檔案(H):  
瀏覽(B)...

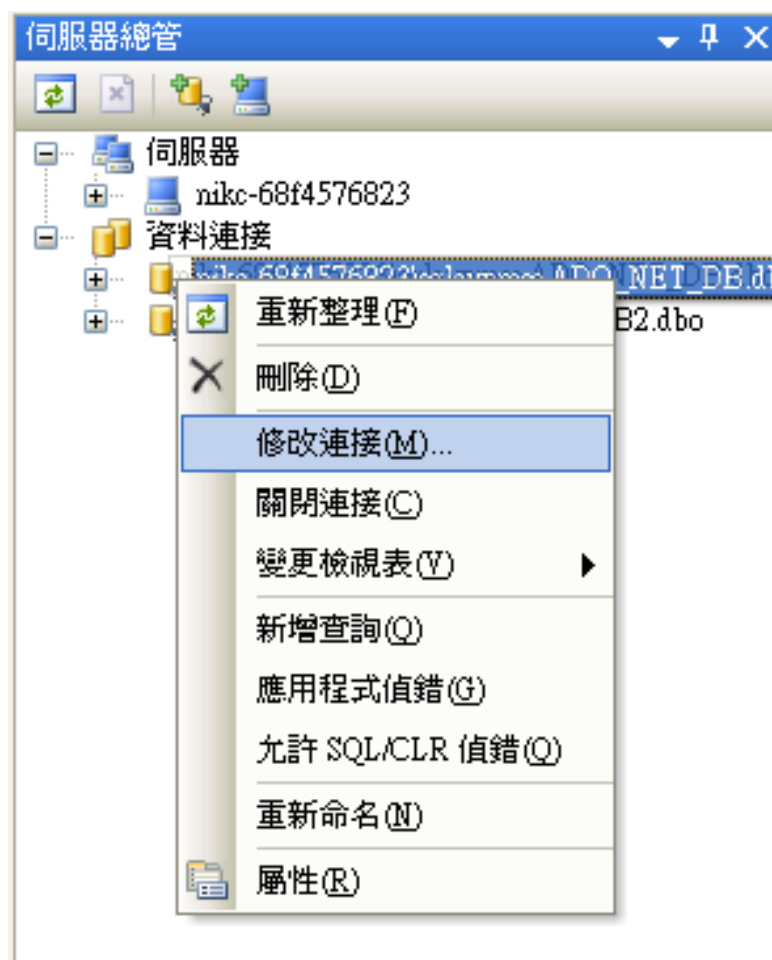
邏輯名稱(L):

進階(V)...

測試連接(T) 確定 取消

# 修改資料連結項目

- 進入「伺服器總管」視窗
- 點選「資料連結」下的  
的連接項目



# 建立資料表與查詢

- 「伺服器總管」→「資料連結」
  - ▣ 展開ADO\_NET\_DB的连接
- 對「資料表」項目按右鍵
  - ▣ 「加入新的資料表」
  - ▣ 「新增查詢」

# 建立MySQL資料庫

- 下載

- <http://www.mysql.com>

- 建立資料庫與資料表

- 下載並安裝

- MySQL Connector/NET

- <http://www.mysql.com/products/connector/net>

- MySQL Connector/ODBC

- <http://www.mysql.com/products/connector/odbc>

# 使用MySQL Connector/NET連接

- 建立Windows視窗程式專案。
- 加入MySQL.Data.dll參考。
  - ▣ C:\Program Files\MySQL\MySQL Connector Net 1.0.7\bin
- 在程式中匯入以下命名空間：
  - ▣ Imports MySql.Data.MySqlClient
  - Imports System.Data

# 使用MySQL Connector/NET連接

- **Button**的**click**事件，加入以下程式碼，開啓資料庫連線

- ▣ Private Sub Button1\_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles Button1.Click  
Dim cn As MySqlConnection  
Dim da As MySqlDataAdapter  
Dim ds As New DataSet  
cn = New MySqlConnection("server=localhost; user id=root;  
password=mysql; database=mysql ")  
cn.Open()  
MessageBox.Show("連線的狀態爲" + \_  
cn.State.ToString())  
cn.Close()  
End Sub

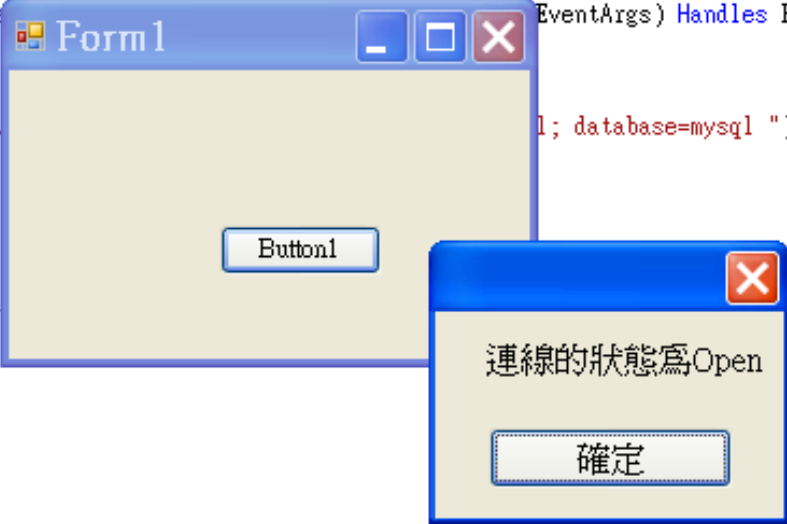
# 使用MySQL Connector/NET連接

- 開始偵錯，並按下button鍵

```
Imports MySql.Data.MySqlClient
Imports System.Data

Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As EventArgs) Handles Button1.Click
        Dim cn As MySqlConnection
        Dim da As MySqlDataAdapter
        Dim ds As New DataSet
        cn = New MySqlConnection("server=localhost; database=mysql ")
        cn.Open()
        MessageBox.Show("連線的狀態為" & cn.State.ToString())
        cn.Close()
    End Sub
End Class
```



The screenshot illustrates the execution of the provided C# code. The code defines a class `Form1` with a `Button1_Click` event handler. This handler creates a `MySqlConnection` object with the connection string `"server=localhost; database=mysql "`, opens the connection, displays a message box showing the connection state (`cn.State.ToString()`), and then closes the connection. The running application window, titled `Form1`, contains a button labeled `Button1`. When the button is clicked, a message box appears with the text `連線的狀態為Open` (Connection state is Open) and an `確定` (OK) button.