02 ASP.NET MVC CRUD 初體驗

學習目標

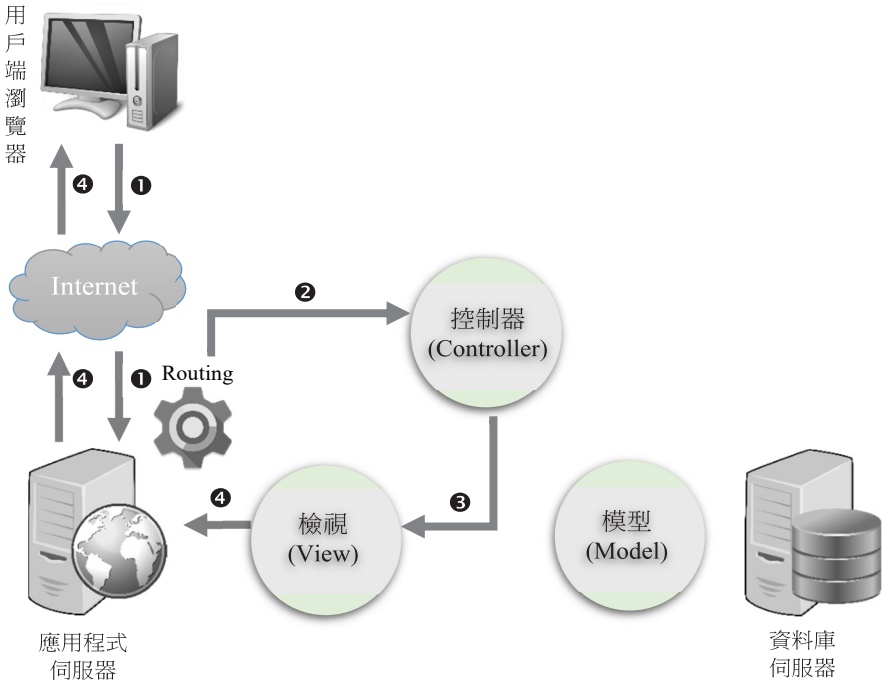
新增、修改、刪除、查詢是 Web 應用程式常見的功能。本章將帶領初學者由建立資料庫開始，學習如何設計具有新增、修改、刪除、查詢作業的待辦事項 ASP.NET MVC 應用程式。並瞭解 ASP.NET MVC 的運作模式、專案架構與網址路由設定。

起初，神創造天地。地是空虛混沌，淵面黑暗；神的靈運行在水面上。神說：「要有光，就有了光。」(創世記 1:1-3)

2.1 ASP.NET MVC 的運作模式

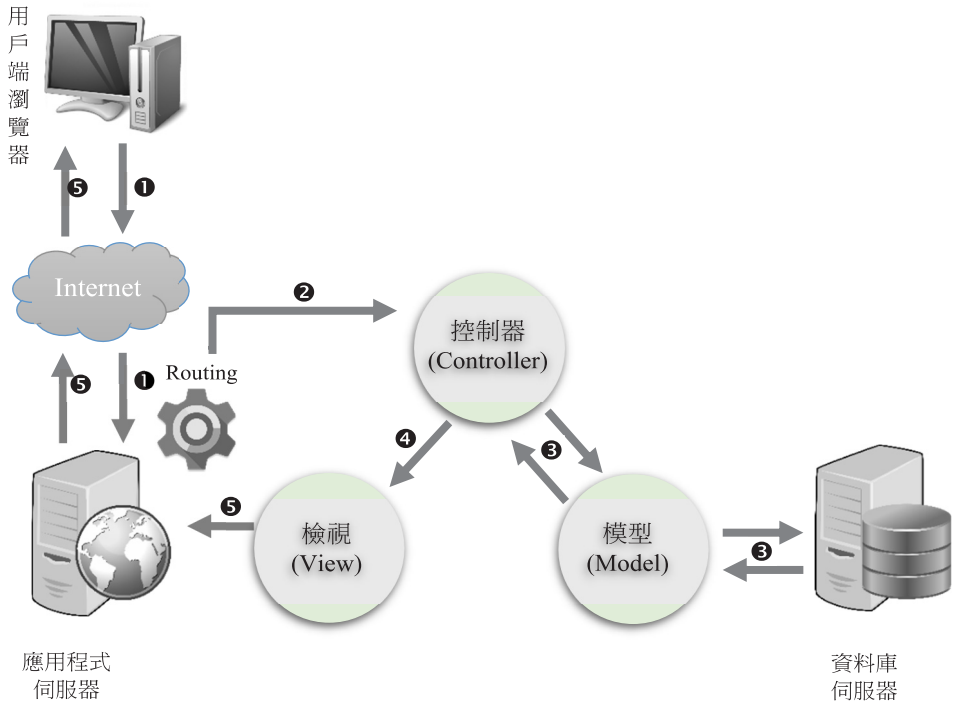
開發 ASP.NET MVC 應用程式之前，有必要瞭解其運作模式。(以下說明示範不包含 Model 的使用)

1. 當用戶端使用者由瀏覽器送出一個請求(Request)，這個請求可能是輸入一段網址、點按超連結或是按下表單的傳送按鈕。
2. 用戶端所送出的請求會傳送到應用程式伺服器的 ASP.NET MVC 的路由表 Routing 進行比對，比對完成 Routing 會執行對應的控制器(Controller)中的動作方法(Action Method)。
3. Controller 控制器處理完使用者的請求之後，接著會將結果傳送給指定的檢視 View。
4. 最後再將 View 檢視以 HTML 網頁呈現在使用者的瀏覽器上。



若使用者進行資料存取，即會使用到 Model 模型，則 ASP.NET MVC 運作模式如下：

1. 用戶端使用者由瀏覽器送出一個請求(Request)。
2. 用戶端所送出的請求傳送到應用程式伺服器的 ASP.NET MVC 的 Routing 進行比對，比對完成 Routing 會執行對應的控制器(Controller)中的動作方法(Action Method)。
3. Controller 控制器透過 Model 模型來存取資料來源。
4. Controller 控制器將取得 Model 模型的資料結果傳送至指定的 View 檢視。
5. 最後將 View 檢視與 Model 模型結果編譯成 HTML 網頁，並呈現在使用者的瀏覽器上。



2.2 ASP.NET MVC 的 CRUD 作業

2.2.1 何謂 CRUD 作業

在資料庫程式設計中，CRUD 字母縮寫即是 Create、Read、Update 以及 Delete，它代表的是操作資料庫應用程式的新增、讀取(查詢)、修改和刪除四項作業，字母縮寫會對應到 SQL 語法和 HTTP 方法(RESTful API)的操作，如下表：

操作	SQL	HTTP
Create	INSERT	PUT / POST
Read(Retrieve)	SELECT	GET
Update(Modify)	UPDATE	PUT / POST / PATCH
Delete(Destroy)	DELETE	DELETE

透過下面範例一步步帶領讀者，由建立專案開始，體會開發一個待辦事項功能的 ASP.NET MVC 網站有多麼容易，此網站可讀取、新增、刪除待辦事項 tToDo 資料表。

2.2.2 開啟與執行 ASP.NET MVC Web 專案

在開始練習製作 ASP.NET MVC 專案時，先學會如何開啟與執行本書範例的專案，將有助於初學者瞭解範例的功能與學習的重點。

上機練習

Step 01 複製專案檔

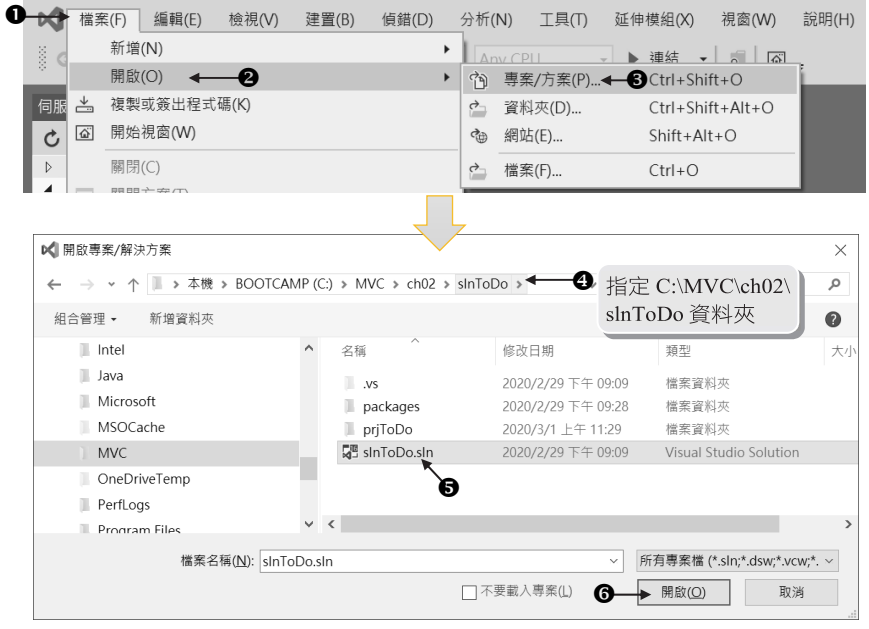
將本書範例 ch02 資料夾複製到「C:\MVC」資料夾下。

Step 02 進入 Visual Studio 2019 整合開發環境

啟動 Visual Studio 2019 出現下圖視窗，請按下「不使用程式碼繼續(W)➔」連結按鈕進入 Visual Studio 2019(本書之後簡稱 VS)整合開發環境。

Step 03 開啟 Visual C# 的 ASP.NET Web 應用程式專案

在 VS 整合開發環境執行【檔案(F)/開啟(O)/專案/方案(P)…】開啟「開啟專案/解決方案」視窗，接著請選取「C:\MVC\ch02\slnToDo」資料夾下的「slnToDo.sln」方案檔，並按下 開啟(O) 鈕開啟該 Web 方案。

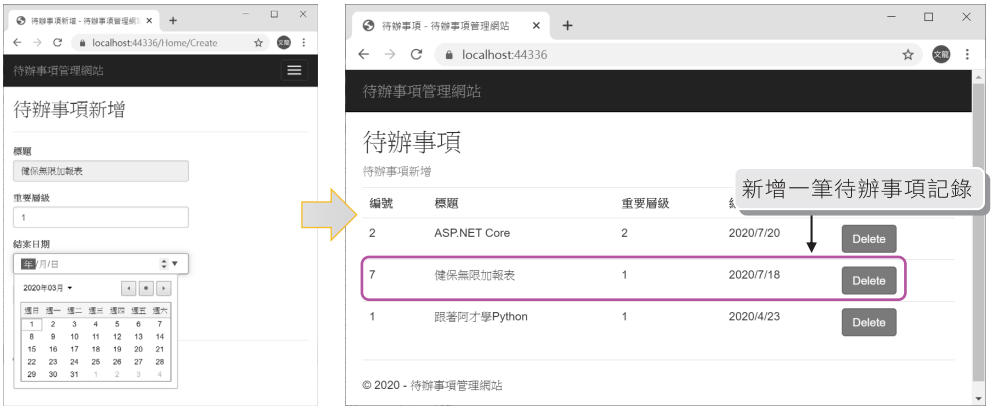


Step 04 執行結果

本例 ASP.NET MVC 待辦事項管理系統提供讀取、新增、刪除功能。請按下執行程式 ▶ 鈕測試本例功能。

1. 網站執行時出現待辦事項列表記錄，待辦事項有編號、標題、重要層級、結案日期等四個欄位，待辦事項列表以結案日期進行遞減排序，每一筆記錄可使用刪除(Delete)記錄的功能；如下圖若按下 Delete 鈕會出現對話方塊再次詢問是否刪除該筆記錄。

2. 按下「待辦事項新增」連結會連結至待辦事項新增的頁面，在此網頁可輸入待辦事項的標題、重要層級、結案日期的資料，編號欄位為自動標號所以不用輸入資料，至於標題、重要層級(以 1~3 表示重要、普通、不重要層級)、結案日期皆為必填欄位，且結案日期使用日期清單呈現；當欲新增的待辦事項記錄輸入完成後再按下 新增 鈕，此時即會返回列表頁面觀看新增後的結果。

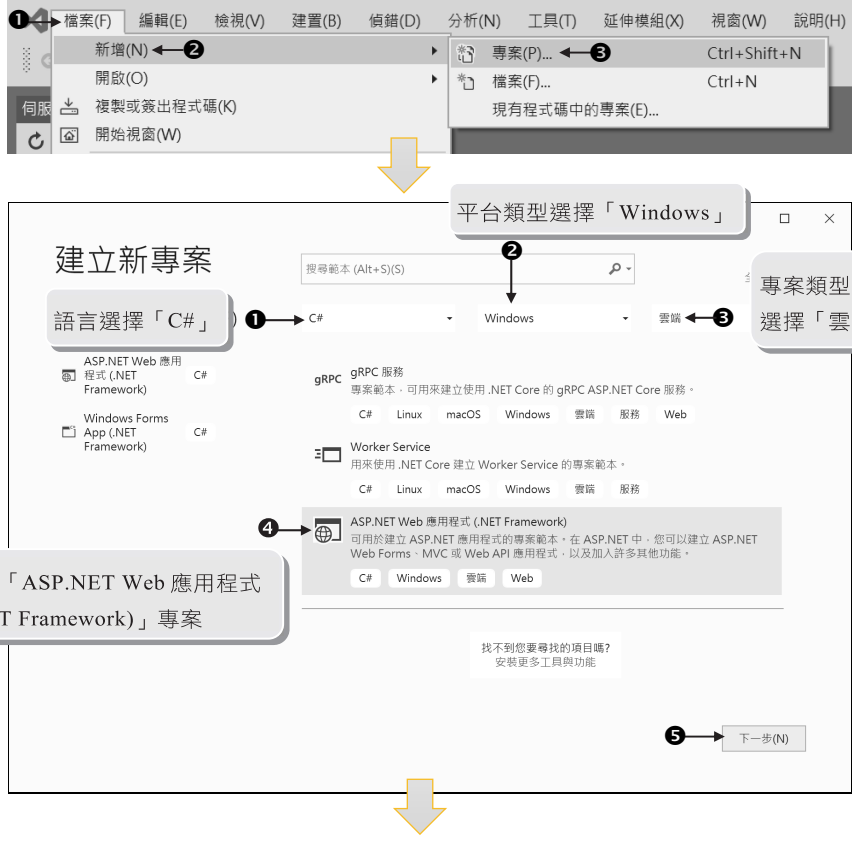


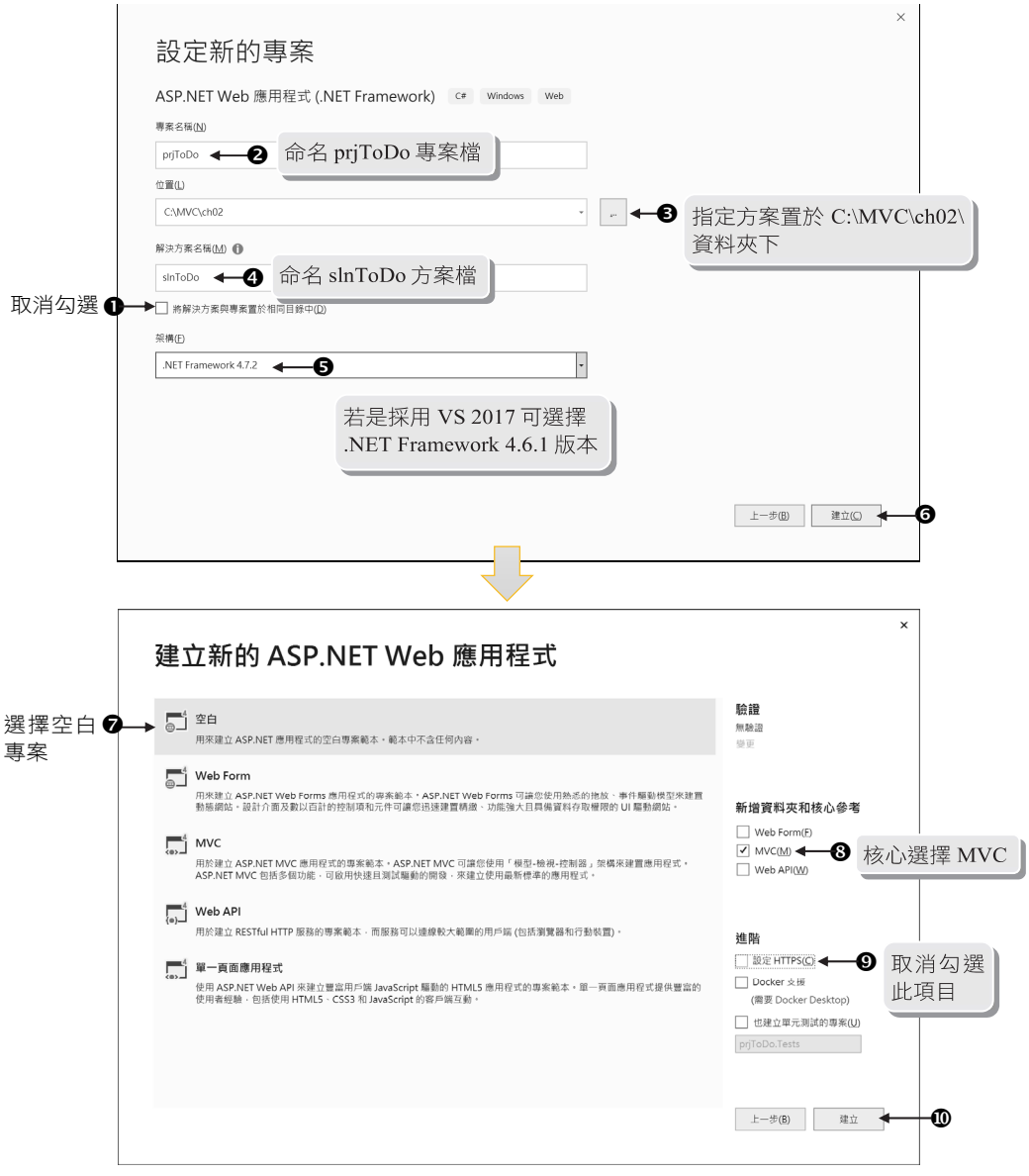
2.2.3 建立 ASP.NET MVC Web 專案與資料庫

上機練習

Step 01 建立 Visual C# 的 ASP.NET Web 應用程式專案

進入 VS 整合開發環境執行【檔案(F)/新增(N)/專案(P)…】開啟下圖「新增專案」視窗，接著依下圖操作在「C:\MVC\ch02」資料夾下建立名稱為「slnToDo」方案，專案名稱命名為「prjToDo」，專案範本為「空白」，核心參考為「MVC」。





Step 02 建立專案使用的 dbToDo.mdf 代辦事項資料庫

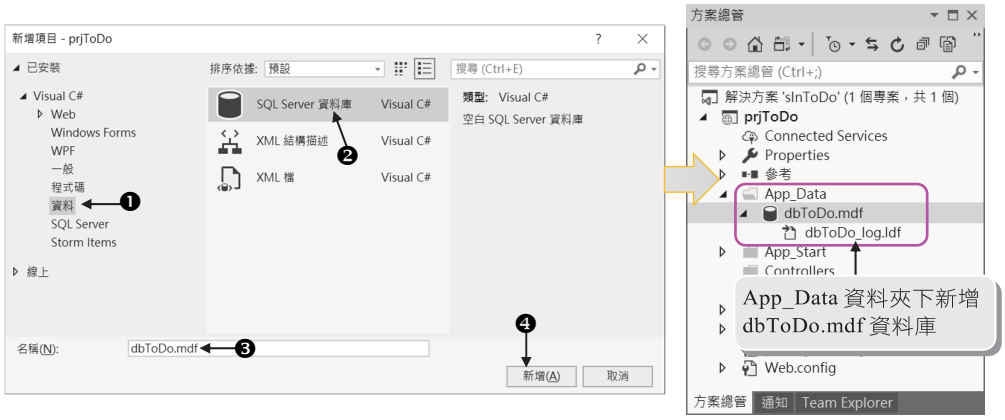
1. Model 模型說明：

   本例 Model 模型可存取 dbToDo.mdf 資料庫的 tToDo 資料表，請依下列步驟建立 dbToDo.mdf 資料庫內含 tToDo 資料表，該資料表的欄位如下：

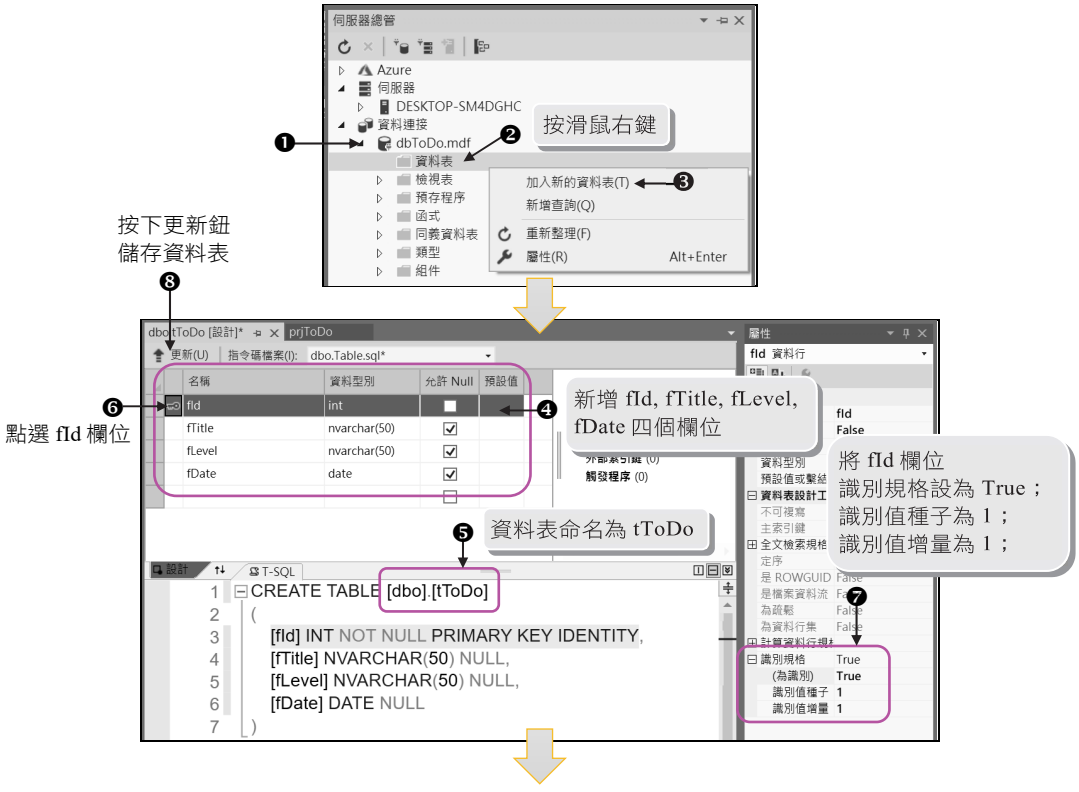
資料表名稱	tToDo				
主鍵值欄位	fId				
欄位名稱	資料型態	長度	允許 null	預設值	備註
fId	int		否		編號 識別規格設為 True 識別值種子為 1 識別值增量為 1
fTitle	nvarchar	50	是		標題
fLevel	nvarchar	50	是		待辦事項重要層級 1：重要 2：普通 3：不重要
fDate	date		是		結案日期(期限日期)

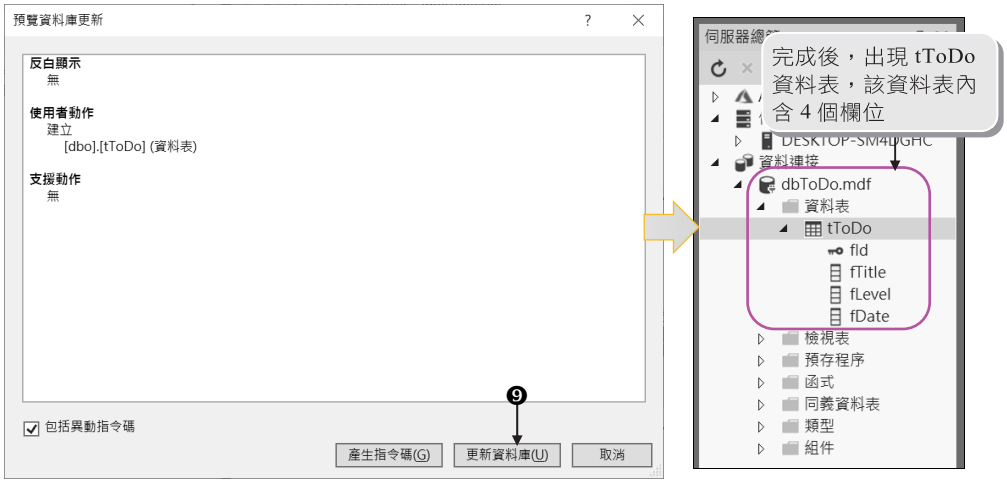
2. 在方案總管的 App_Data 資料夾按滑鼠右鍵，由快顯功能表執行【加入(D)/新增項目(W)…】指令。

3. 接著開啟下圖「加入新項目」視窗，請依圖示操作新增 dbToDo.mdf 資料庫。完成之後方案總管 App_Data 資料夾下會出現 dbToDo.mdf 資料庫。



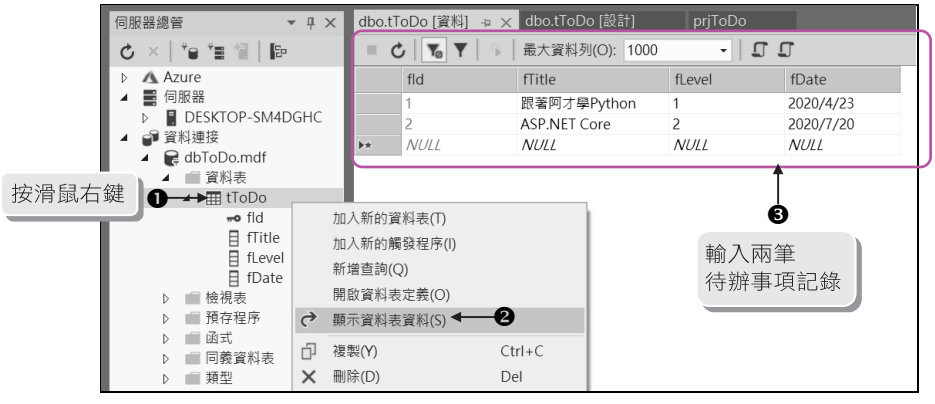
4. 在上圖 dbToDo.mdf 快按滑鼠左鍵兩下開啟伺服器總管。請在伺服器總管點按 dbToDo.mdf，並在「資料表」按滑鼠右鍵由快顯功能表執行【加入新的資料表(T)...】指令，接著依圖示操作新增 tToDo 資料表有 fId、fTitle、fLevel、fDate 四個欄位。





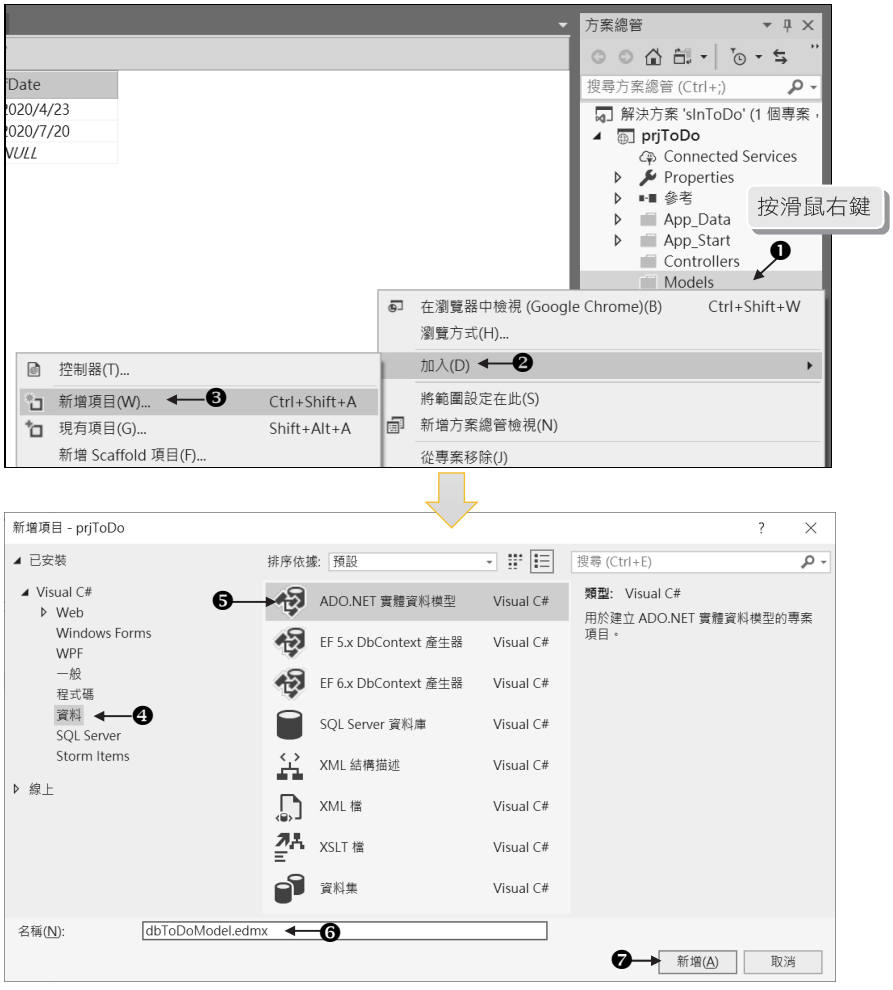
[註] 欄位前面若有 圖示，表示該欄位為主索引鍵。若要將欄位指定為主索引鍵，只要點選該欄位並執行【設定主索引鍵(K)】指令即可，如左下圖；若要移除主索引鍵，只要點選該欄位並執行【移除主索引鍵(K)】指令即可，如右下圖。

5. 在 tToDo 資料表上按滑鼠右鍵由快顯功能表執行【顯示資料表資料(S)】指令，接著請在 tToDo 資料表窗格內輸入兩筆待辦事項記錄。



Step 03 建立可存取 dbToDb.mdf 資料庫的 Model(ADO.NET 實體資料模型)

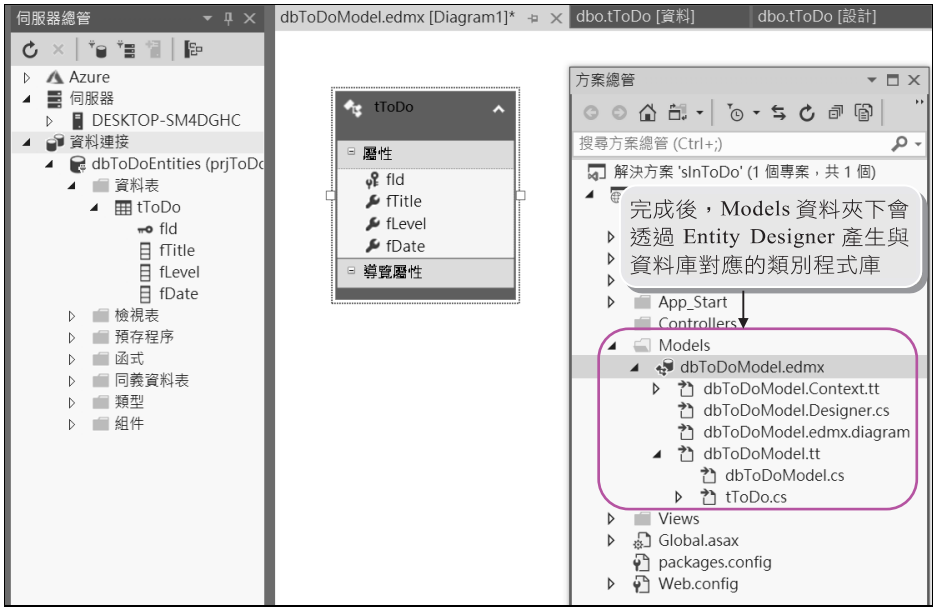
1. 在方案總管的 Models 資料夾按滑鼠右鍵，並執行快顯功能表的【加入(D)/新增項目(W)...】指令新增「ADO.NET 實體資料模型」，將該檔名設為「dbToDoModel.edmx」。(.edmx 副檔名可省略不寫，該檔是用來記錄資料庫所對應的實體模型)



2. 當新增 dbToDoModel.edmx 的 ADO.NET 實體資料模型後，即會開啟「實體資料模型精靈」視窗，該視窗會一步步指引使用者完成模型。


實體資料模型精靈
選擇您的版本
您要使用哪一個 Entity Framework 版本?(W)
7
Entity Framework 6.x
Entity Framework 5.0
此外，您也可以安裝並使用其他 Entity Framework 版本。
進一步了解
8
< 上一步(P)
下一步(N) >
完成(F)
取消
實體資料模型精靈
選擇您的資料庫物件和設定
您的模型中要包含哪些資料庫物件?(W)
資料表
dbo
tToDo
檢視
預存程序和函式
9
勾選 tToDo 資料表，表示依此資料表
建立 tToDo 實體資料模型
將產生的物件名稱複數化或單數化(S)
在模型中包含外部索引鍵資料行(K)
將選取的預存程序和函數匯入實體模型(I)
模型命名空間(M):
dbToDoModel
10
< 上一步(P)
下一步(N) >
完成(F)
取消
安全性警告
執行這個文字範本可能會危害您的電腦。如果這是從不受信任的來源取得，請切勿執行。
按一下 [確定] 執行範本。
按一下 [取消] 停止處理。
不要再顯示此訊息(D)
11
確定
取消(C)

3. 完成上面步驟後，實體資料模型會建立在 Models 資料夾下。接著會進入下圖 Entity Designer 實體資料模型設計工具的畫面，可以發現設計工具內含「tToDo」實體資料模型。



4. 請選取方案總管視窗的專案名稱(prjToDo)，並按滑鼠右鍵執行快顯功能表的【建置(U)】指令編譯整個專案(也可以執行【重建(E)】)，此時就可以使用 Entity Framework 來存取 tToDo 資料表。

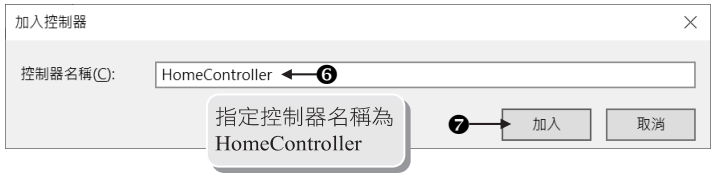
2.2.4 ASP.NET MVC 讀取作業

上節建立好可以連接 dbToDo.mdf 資料庫，以及存取 tToDo 資料表的資料模型(Model)，接著本節練習將 tToDo 資料表內的所有記錄依結案日期遞減排序並顯示於網頁上。

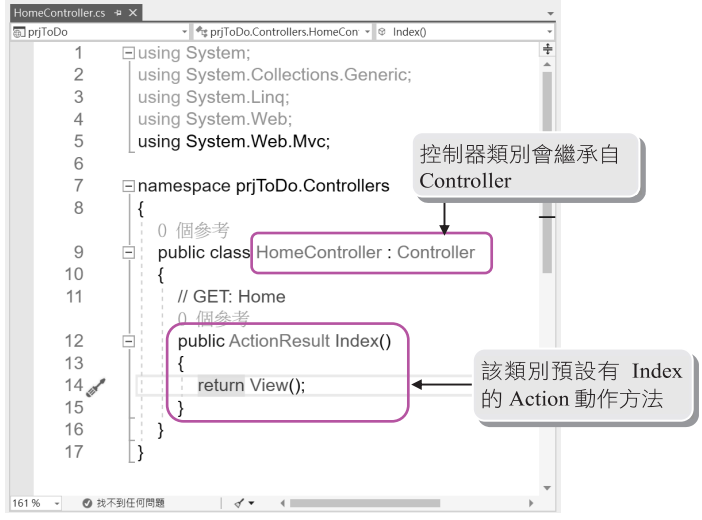
上機練習

Step 01 建立 Home 控制器

在方案總管的 Controllers 資料夾按滑鼠右鍵，並執行快顯功能表的【加入(D)/控制器(T)…】指令新增「HomeController.cs」控制器檔案，控制器類別會繼承自 Controller 類別，該控制器內含 Index()動作方法(Action Method)。



控制器名稱設為「HomeController」是因為 ASP.NET MVC 專案預設會執行 HomeController，且控制器類別名稱必須要使用「Controller」當結尾。



Step 02 撰寫 HomeController 控制器類別的 Index()動作方法

當執行「http://localhost/Home/Index」時會執行 HomeController 的 Index()動作方法，請撰寫如下灰底處的程式碼。

C# 程式碼 FileName: Controllers/HomeController.cs

```
01 using System;
02 using System.Collections.Generic;
03 using System.Linq;
04 using System.Web;
05 using System.Web.Mvc;
06 using prjToDo.Models;
07
08 namespace prjToDo.Controllers
09 {
10     public class HomeController : Controller
11     {
```

```
12          dbToDoEntities db = new dbToDoEntities();
13          // GET: Home
14          public ActionResult Index()
15          {
16              var todos = db.tToDo
17                  .OrderByDescending(m => m.fDate).ToList();
18              return View(todos);
19          }
20      }
21  }
```

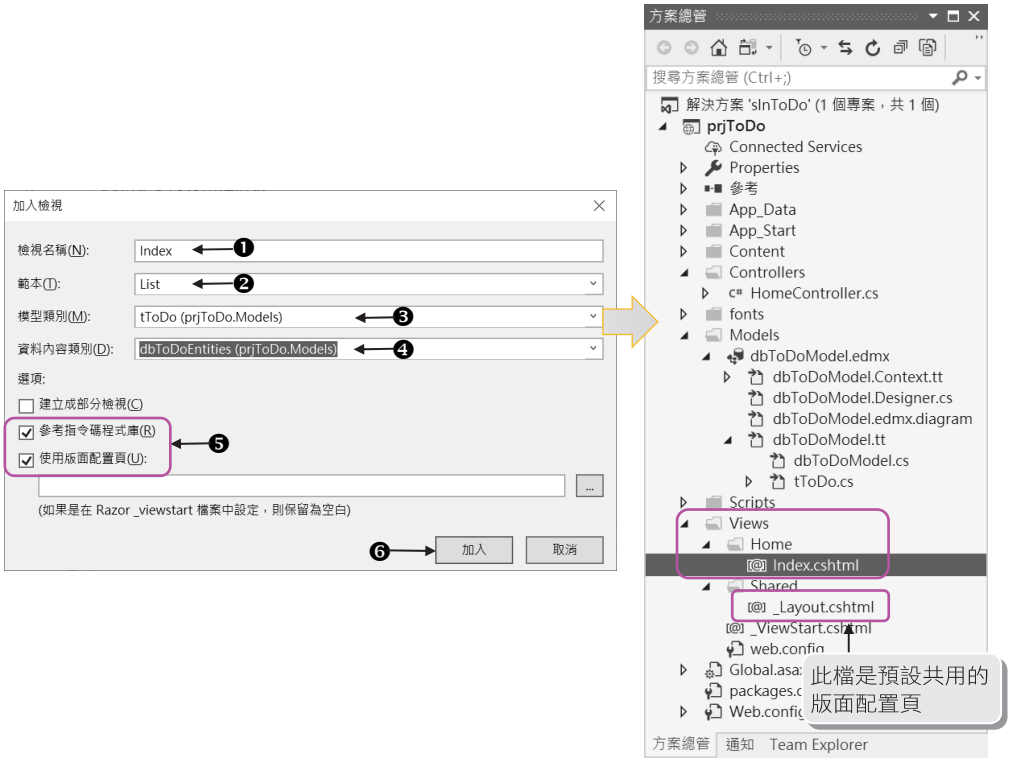
說明

1) 第 1~5 行：預設引用的命名空間。
2) 第 6 行：存取 dbToDo.mdf 資料庫的 dbToDoEntities 類別物件置於 Models 資料夾，因此請引用 prjToDo.Models 命名空間。
3) 第 12 行:建立 dbToDoEntities 類別 db 物件,此物件存取 dbToDo.mdf 資料庫。
4) 第 16-17 行：將 tToDo 資料表內的記錄依 fDate 欄位進行遞減排序並轉成串列，再將結果指定 todos 變數。
5) 第 18 行：將 todos 待辦事項的所有記錄(todos 串列物件)傳到 Index.cshtml 的 View 檢視頁面。

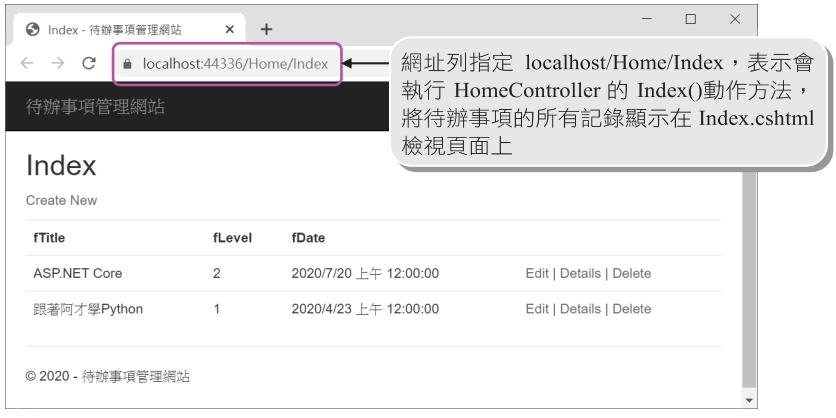
Step 03 建立 Index.cshtml 待辦事項列表 View 檢視頁面

1. 在 Index()動作方法上按滑鼠右鍵，並執行快顯功能表的【新增檢視(D)…】指令。

2. 在加入檢視視窗指定檢視名稱為「Index」；範本(T)為「List」；模型類別(M)為「tToDo」；資料內容類別(D)為「dbToDoEntities」；並勾選 參考指令碼程式庫(R) 與 使用版面配置頁(U) 項目，最後按下 加入 鈕。結果方案總管的 Views/Home 資料夾下會新增 Index.cshtml 檢視頁面。



3. 按下執行程式 ▶ 鈕，觀看網頁執行結果。



4. 上圖標題欄位希望呈現中文名稱，待辦事項的結案日期以日期顯示，因此需要修改 Index.cshtml 檢視頁面的程式碼，請將如下刪除線的程式碼刪除，並修改成粗體字或指定的程式碼。

程式碼 FileName:Views/Home/Index.cshtml

```
01 @model IEnumerable<prjToDo.Models.tToDo>
02
03 @{
04     ViewBag.Title = "~~Index~~";
05 }
06
07 <h2>~~Index~~</h2>
08
09 <p>
10     @Html.ActionLink("~~Create New~~", "Create")
11 </p>
12 <table class="table">
13     <tr>
14         <th>
15             編號
16         </th>
17         <th>
18             ~~@Html.DisplayNameFor(model => model.fTitle)~~
19             標題
20         </th>
21         <th>
22             ~~@Html.DisplayNameFor(model => model.fLevel)~~
23             重要層級
24         </th>
25         <th>
26             ~~@Html.DisplayNameFor(model => model.fDate)~~
27             結案日期
28         </th>
29         <th></th>
30     </tr>
31
32 @foreach (var item in Model) {
33     <tr>
34         <td>
```

待辦事項

待辦事項

待辦事項新增

更改中文標題

```
35              @item.fId
36          </td>
37          <td>
38              @Html.DisplayFor(modelItem => item.fTitle)
39          </td>
40          <td>
41              @Html.DisplayFor(modelItem => item.fLevel)
42          </td>
43          <td>
44              @Html.DisplayFor(modelItem => item.fDate)
45              @DateTime.Parse(item.fDate.ToString()).ToShortDateString()
46          </td>
47          <td>
48              @Html.ActionLink("Edit", "Edit", new { id=item.fId }) |
49              @Html.ActionLink("Details", "Details", new { id=item.fId }) |
50              @Html.ActionLink("Delete", "Delete", new { id=item.fId })
51          </td>
52      </tr>
53  }
54
55  </table>
```

顯示編號欄位

結案日期以日期格式顯示

文字連結

動作方法

刪除 Edit 與 Details 的超連結

說明

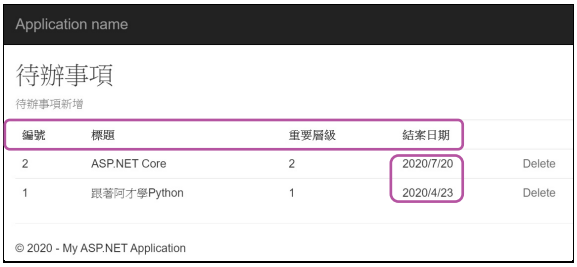
1) 第 1 行：宣告 Index.cshtml 檢視頁面@model 型別為 tToDo 資料模型；表示此檢視面頁使用 tToDo 模型。

2) 第 32~53 行：使用 foreach 迴圈逐一將 Model 模型(@model)中的每一筆記錄顯示出來。

3) 第 50 行：使用 HtmlHelper 的@Html.ActionLink()方法來指定超連結功能。當按下 Delete 文字超連結即會執行 Home 控制器的 Delete()方法，並傳送 URL 參數 id，而 id 參數值是 fId 欄位的資料。

4) 第 48,49 行：執行方式同 50 行，本例不會製作 Edit(修改)與 Details(明細)功能，因此請將此兩行敘述刪除。

5. 按下執行程式 ▶ 鈕觀看網頁執行結果。結果發現標題以中文呈現，待辦事項結案日期以日期格式呈現。



Step 04 _Layout.cshtml 版面配置頁說明

在前一個步驟「加入檢視」視窗設定中若有勾選「使用版面配置頁」，且第一次建立檢視頁面時，VS 會在 Views 資料夾下建立 _ViewStart.cshtml，同時也會在 Views/Shared 資料夾建立_Layout.cshtml 版面配置頁。

_ViewStart.cshtml 預設指定專案的檢視頁面套用_Layout.cshtml 版面配置頁，此版面配置頁預設使用 Bootstrap 前端套件；Bootstrap 是以 HTML、CSS 與 JavaScript 為主，是用來開發自適應與行動優先網站的套件。

_Layout.cshtml 版面配置頁預設會和一般 View 檢視頁面進行合併，View 檢視頁面的內容會置於版面配置頁的 @RenderBody()區域，程式碼如下：

程式碼 FileName:Views/Shared/_Layout.cshtml

```
01 <!DOCTYPE html>
02 <html>
03 <head>
04    <meta charset="utf-8" />
05  <meta name="viewport" content="width=device-width, initial-scale=1.0">
06    <title>@ViewBag.Title - My ASP.NET Application</title>
07    <link href="~/Content/Site.css" rel="stylesheet" type="text/css" />
08    <link href="~/Content/bootstrap.min.css" rel="stylesheet"
09       type="text/css" />
10    <script src="~/Scripts/modernizr-2.8.3.js"></script>
11 </head>
12 <body>
13    <div class="navbar navbar-inverse navbar-fixed-top">
14        <div class="container">
15            <div class="navbar-header">
16                <button type="button" class="navbar-toggle"
17                  data-toggle="collapse" data-target=".navbar-collapse">
18                    <span class="icon-bar"></span>
19                    <span class="icon-bar"></span>
20                    <span class="icon-bar"></span>
21                </button>
22                @Html.ActionLink("Application name", "Index", "Home",
23                    new { area = "" }, new { @class = "navbar-brand" })
24            </div>
25            <div class="navbar-collapse collapse">
26                <ul class="nav navbar-nav">
27                </ul>
28            </div>
29        </div>
30    </div>
31
32    <div class="container body-content">
33        @RenderBody()
34        <hr />
35        <footer>
36            <p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>
```

待辦事項管理網站

待辦事項管理網站

View 檢視頁面會放入此處與版面配置頁進行合併

待辦事項管理網站

```
37          </footer>
38      </div>
39
40      <script src="~/Scripts/jquery-3.4.1.min.js"></script>
41      <script src="~/Scripts/bootstrap.min.js"></script>
42  </body>
43  </html>
```

說明

1) 第 4 行：指定網頁的語系為 utf-8。
2) 第 5 行：指定網頁畫面能隨裝置的螢幕自動做縮放。
3) 第 6 行：@ViewBag.Title 動態屬性可在控制器中設定，此處@ViewBag.Title 在 Index.cshtml 已指定為「待辦事項管理網站」。
4) 第 7~9 行：使用 Bootstrap 的 CSS 樣式。
5) 第 10,40,41 行：使用 Bootstrap 和 jQuery 的 JavaScript 函式。
7) 第 13~30 行：網頁尾首的導覽列。
8) 第 22~23 行：指定網站頁首區的連結；此處使用@Html.ActionLink()方法連結到 Home 控制器的 Index()方法，同時套用 CSS 樣式名稱為 navbar-brand。請將連結文字「Application name」改成「待辦事項管理網站」。
9) 第 32~38 行：網頁內文區域。在第 36 行的@DateTime.Now.Year 會取得系統的年份。
10)第 33 行：View 檢視頁面會放入版面配置頁的 `@RenderBody()` 區域並進行合併。如下圖即是_Layout.cshtml 和 Index.cshtml 合併的結果。

2.2.5 ASP.NET MVC 新增作業

上機練習

Step 01 撰寫 HomeController 控制器的 Create()動作方法

在 HomeController 控制器中撰寫灰底處多載的 Create()動作方法(Action Method)。說明如下：

1. public ActionResult Create()方法
   當連結 Home/Create 即執行 Home 控制器的 Create()方法，此時會傳回 Create.cshtml 的 View 檢視畫面。

2. [HttpPost]
   public ActionResult Create(string fTitle, string fLevel, DateTime fDate)方法
   當在擁有 fTitle、fLevel、fDate 欄位的表單按下 Submit 鈕，此時會執行這個 Create()方法，並將表單欄位資料傳送到 Create()方法對應的虛引數。

C# 程式碼 FileName: Controllers/HomeController.cs

```
01 using System;
02 using System.Collections.Generic;
03 using System.Linq;
04 using System.Web;
05 using System.Web.Mvc;
06 using prjToDo.Models;
07
08 namespace prjToDo.Controllers
09 {
10     public class HomeController : Controller
11     {
12         dbToDoEntities db = new dbToDoEntities();
13         // GET: Home
14         public ActionResult Index()
15         {
16             var todos = db.tToDo
17                 .OrderByDescending(m => m.fDate).ToList();
18             return View(todos);
19         }
20
```

```
21          public ActionResult Create()
22          {
23              return View();
24          }
25
26          [HttpPost]
27          public ActionResult Create
28             (string fTitle, string fLevel, DateTime fDate)
29          {
30              tToDo todo = new tToDo();
31              todo.fTitle = fTitle;
32              todo.fLevel = fLevel;
33              todo.fDate = fDate;
34              db.tToDo.Add(todo);
35              db.SaveChanges();
36              return RedirectToAction("Index");
37          }
38      }
39  }
```

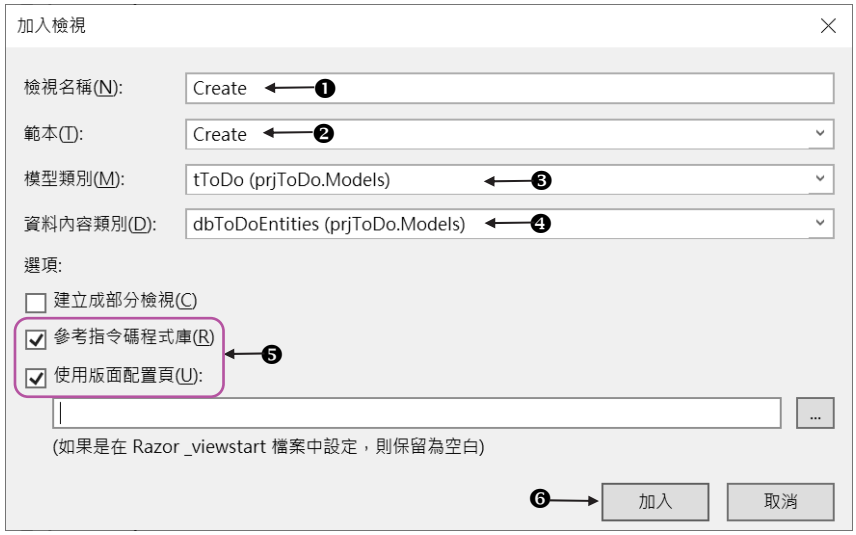
說明

1) 第 21~24 行：連結到 Home/Create 時會執行此方法，接著傳回預設 Create.cshtml 的 View 檢視頁面。
2) 第 26~37 行：在 Create.cshtml 的檢視頁面按下 Submit 鈕會執行此方法。
3) 第 30 行：建立 tToDo 待辦資料型別 todo 物件。
4) 第 31~33 行：將表單 fTitle、fLevel、fDate 欄位的資料逐一指定給 todo 物件的 fTitle、fLevel、fDate 屬性。此處未指定 fId 欄位資料是因為 tToDo 資料表已設定 fId 欄位為自動編號(識別規格)，因此 fId 欄位會採自動編號模式由 1 開始新增。
5) 第 34 行：將 todo 待辦事項物件放入 tToDo 資料表內。
6) 第 35 行：執行 SaveChanges()方法進行異動資料庫，必須執行此方法 todo 待辦事項物件才可寫入 tToDo 資料表內。
7) 第 36 行：重新執行 Home 控制器的 Index()方法，使 todos 待辦事項結果傳給 Index.cshtml 的 View 檢視頁面。

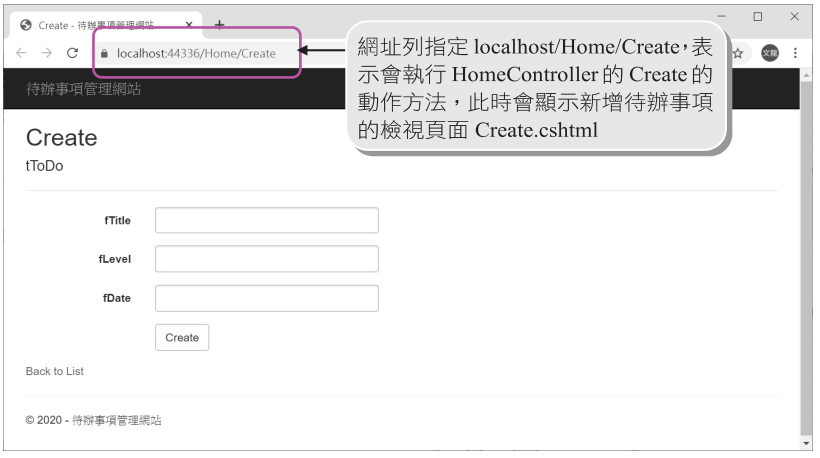
Step 02 建立 Create.cshtml 新增待辦事項 View 檢視頁面

1. 在 Create()動作方法上按滑鼠右鍵，並執行快顯功能表的【新增檢視(D)…】指令。

2. 在加入檢視視窗指定檢視名稱為「Create」；範本(T)為「Create」；模型類別(M)為「tToDo」；資料內容類別(D)為「dbToDoEntities」；並勾選 參考指令碼程式庫(R) 與 使用版面配置頁(U) 的核取方塊，最後按下 加入 鈕。結果方案總管的 Views/Home 資料夾下會新增 Create.cshtml 檢視頁面。



3. 按下執行程式 ▶ 鈕觀看網頁執行結果。



4. 上圖 Create.cshtml 檢視頁面的所有標題欄位希望呈現中文名稱，待辦事項的結案日期能使用日期清單，且有欄位為必填項目，因此需要修改 Create.cshtml 檢視頁面的程式碼，請將如下刪除線的程式碼刪除，並修改成粗體字或指定的程式碼。

程式碼 FileName:Views/Home/Create.cshtml

```
01 @model prjToDo.Models.tToDo
02
03 @{
04     ViewBag.Title = "~~Create~~";
05 }
06
07 <h2>~~Create~~</h2>
08
09 @using (Html.BeginForm())
10 {
11     @Html.AntiForgeryToken()
12
13     <div class="form-horizontal">
14         ~~<h4>tToDo</h4>~~
15         <hr />
16         @Html.ValidationSummary(true,"",new { @class = "text-danger" })
17         <div class="form-group">
18             <label class="control-label col-md-2">標題</label>
```

待辦事項新增

待辦事項新增

```
19              @Html.LabelFor(model => model.fTitle, htmlAttributes:
20                  new { @class = "control-label col-md-2" })
21              <div class="col-md-10">
22                  @Html.EditorFor(model => model.fTitle, new
23                      { htmlAttributes = new { @class = "form-control"
24                      , required="required" } })
25                  @Html.ValidationMessageFor(model => model.fTitle, "",
26                      new { @class = "text-danger" })
27              </div>
28          </div>
29
30          <div class="form-group">
31              <label class="control-label col-md-2">重要層級</label>
32              @Html.LabelFor(model => model.fLevel, htmlAttributes:
33                  new { @class = "control-label col-md-2" })
34              <div class="col-md-10">
35                  @Html.EditorFor(model => model.fLevel, new
36                      { htmlAttributes = new { @class = "form-control"
37                      , required="required" } })
38                  @Html.ValidationMessageFor(model => model.fLevel, "",
39                      new { @class = "text-danger" })
40              </div>
41          </div>
42
43          <div class="form-group">
44              <label class="control-label col-md-2">結案日期</label>
45              @Html.LabelFor(model => model.fDate, htmlAttributes: new
46                  { @class = "control-label col-md-2" })
47              <div class="col-md-10">
48                  @Html.EditorFor(model => model.fDate, new
49                      { htmlAttributes = new { @class = "form-control"
50                      ,type="date", required="required" } })
51                  @Html.ValidationMessageFor(model => model.fDate, "",
52                      new { @class = "text-danger" })
53              </div>
54          </div>
55
56          <div class="form-group">
57              <div class="col-md-offset-2 col-md-10">
58                  <input type="submit" value="Create"
```

必填欄位

必填欄位

欄位為日期清單且為必填

新增

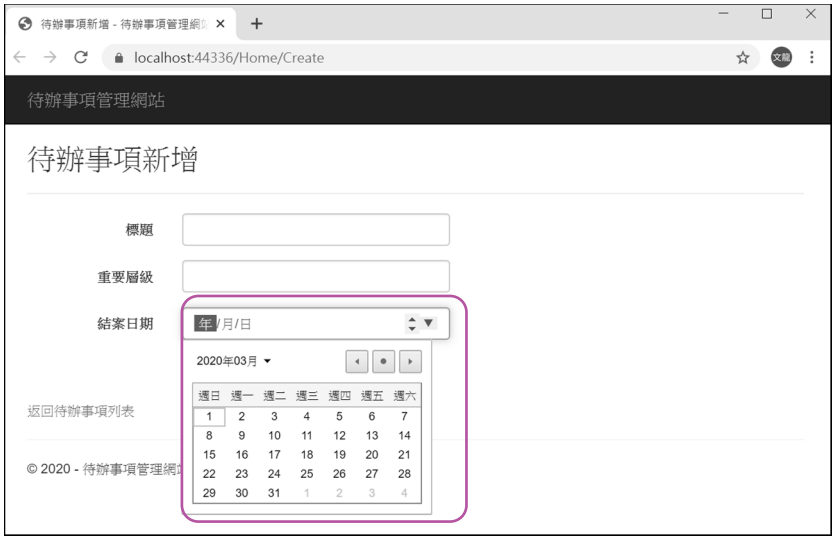
```
59                      class="btn btn-default" />
60              </div>
61          </div>
62      </div>
63  }
64
65  <div>
66      @Html.ActionLink("Back to List", "Index")
67  </div>
68
69  <script src="~/Scripts/jquery-3.4.1.min.js"></script>
70  <script src="~/Scripts/jquery.validate.min.js"></script>
71  <script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>
```

返回待辦事項列表

說明

1) 第 1 行：宣告 View 使用的@model 模型為 tToDo 型別。(有關模型更多用法請參閱第 7 章)
2) 第 22~24,35~37, 48~50 行：Create.cshtml 會產生名稱為 fTitle、fLevel、fDate 表單文字欄位，這三個欄位會和 Create()動作方法的 fTitle、fLevel、fDate 參數進行資料繫結。
3) 第 22~24 行：@Html.EditorFor()方法可用來產生文字方塊欄位，請在此處加入「, required="required"」指定標題文字欄為必填欄位。
4) 第 35~37 行：執行同 24 行，指定重要層級文字欄位為必填欄位。
5) 第 48~50 行：@Html.EditorFor()方法可用來產生文字方塊欄位，請在此行加入「, type="date", required="required"」，將此欄位變更為使用日期清單欄位且為必填，以方便使用者輸入日期資料。

5. 按下執行程式 ▶ 鈕觀看網頁執行結果。結果發現當文字欄無輸入資料即按 新增 鈕會顯示提示訊息，所有欄位必填；且待辦事項的結案日期以日期清單呈現。



2.2.6 ASP.NET MVC 刪除作業

上機練習

Step 01 撰寫 HomeController 控制器的 Delete()動作方法(Action Method)

在 HomeController 控制器中撰寫灰底處的 Delete()動作方法。當連上網址

http://localhost/**Home/Delete/刪除編號**

或

http://localhost/**Home/Delete?id=刪除編號**

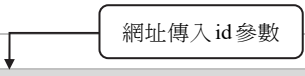
會將「刪除編號」代入 URL 參數 id 並傳給 Delete()動作方法的 id 引數。

Home/Delete/刪除編號 預設未設定 URL 參數，這是因為 ASP.NET MVC 網址路由預設指定 URL 參數為 id，此部份在 2.4 節會做說明。

C# 程式碼 FileName: Controllers/HomeController.cs

```
01 using System;
02 using System.Collections.Generic;
03 using System.Linq;
04 using System.Web;
05 using System.Web.Mvc;
```

```
06 using prjToDo.Models;
07
08 namespace prjToDo.Controllers
09 {
10    public class HomeController : Controller
11    {
12        dbToDoEntities db = new dbToDoEntities();
13        // GET: Home
14        public ActionResult Index()
15        {
16            var todos = db.tToDo
17                .OrderByDescending(m => m.fDate).ToList();
18            return View(todos);
19        }
20
21        public ActionResult Create()
22        {
23            return View();
24        }
25
26        [HttpPost]
27        public ActionResult Create
28           (string fTitle, string fLevel, DateTime fDate)
29        {
30            tToDo todo = new tToDo();
31            todo.fTitle = fTitle;
32            todo.fLevel = fLevel;
33            todo.fDate = fDate;
34            db.tToDo.Add(todo);
35            db.SaveChanges();
36            return RedirectToAction("Index");
37        }
38
39        public ActionResult Delete(int id)
40        {
41            var todo = db.tToDo.Where(m=>m.fId==id).FirstOrDefault();
42            db.tToDo.Remove(todo);
43            db.SaveChanges();
44            return RedirectToAction("Index");
45        }
```



```
46     }
47 }
```

說明

1) 第 39~45 行：連結到 Home/Delete 並傳入 URL 的 id 參數時會執行此方法。
2) 第 41 行：依傳入的 id 參數找出要刪除的待辦事項物件並指定給 todo。
3) 第 42 行：刪除 tToDo 資料表符合 todo 物件的資料。
4) 第 43 行：執行 SaveChanges()方法進行異動資料庫，必需執行此方法才能刪除 tToDo 資料表內符合 todo 物件的記錄。
5) 第 44 行：重新執行 Home/Index()方法。

按下執行程式 ▶ 鈕觀看網頁執行結果，結果發現按下「Delete」超連結文字會執行 Home/Delete 並傳入 URL 的 id 參數，id 參數值為該筆待辦事項的編號，接著會直接刪除掉該筆記錄。

Step 02 修改 Index.cshtml 待辦事項列表 View 檢視頁面

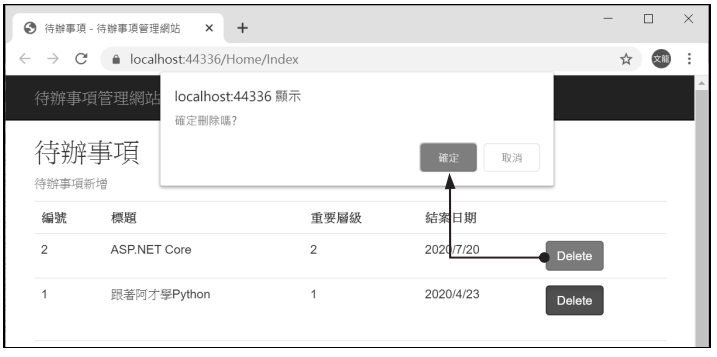
按下 Delete 進行刪除，通常會先以警告視窗方式讓使用者再確認是否真的要刪除，若要達成此功能，請開啟 Views/Home/Index.cshtml 檢視頁面，並修改 Delete 連結灰色程式碼的地方，此處新增確認的 JavaScript 程式碼，同時將 Delete 按鈕套用 Bootstrap 的「btn btn-danger」樣式，使按鈕以 Delete 呈現。

程式碼 FileName:Views/Home/Index.cshtml

```
01 @model IEnumerable<prjToDo.Models.tToDo>
02
03 @{
04     ViewBag.Title = "待辦事項";
05 }
06
07 <h2>待辦事項</h2>
08
09 <p>
10     @Html.ActionLink("待辦事項新增", "Create")
11 </p>
     ............
     ............
```

```
47     <td>
48         @Html.ActionLink("Delete", "Delete", new { id = item.fId }
49                         , new {   onclick = "return confirm('確定刪除嗎?')"
50                         , @class="btn btn-danger"} )
51     </td>
52 </tr>
53 }
54
55 </table>
```

按下執行程式 ▶ 鈕觀看網頁執行結果。結果發現當按下 Delete 鈕時會出現對話方塊詢問「確定要刪除嗎？」，若按下 確定 則刪除該筆記錄，否則取消刪除作業。



2.3 ASP.NET MVC 的專案架構

由上一節範例實作中可以發現：

1. 操作資料庫存取的程式置於 Models 資料夾下。
2. 檢視頁面會對應至控制器的動作方法名稱，且會放在 Views 資料夾下。
3. 控制器的名稱最後面要再加上 Controller，控制器類別檔會放在 Controller 資料夾下。

因此開發人員只要依照此規範來撰寫 MVC 程式，讓後續接手的開發人員有個依據，會比較容易管理且好上手。下圖即是專案架構說明：

1. App_Data
   存放 Web 應用程式的資料庫。
2. App_Start
   存放 Web 服務啟動時所執行的網址路由 Routing 檔案。
3. Content
   存放 CSS 檔。
4. Controllers
   存放控制器類別檔，控制器檔案結尾一定要是「Controller」，不然預設的網址由路會讀不到。
5. fonts
   存放應用程式使用的字型檔。
6. images
   存放網站所使用的圖檔。
7. Models
   存放模型以及存取資料庫相關的類別檔。
8. Scripts
   存放 JavaScript 檔案或 JavaScript 相關函式庫。例如 bootstrap 或 jQuery 函式庫等。
9. Views
   存放檢視的資料夾。若 HomeController 控制器的 Create()方法，則 Create.cshtml 檢視檔會存放在 Views/Home/Create.cshtml。Views 資料夾下的_ViewStart.cshtml 會指定 Views/Shared/_Layout.cshtml 為專案預設的共用版面配置頁。
10. Global.asax
    Web 應用程式層級事件以及全域功能設定檔案。如驗證授權、過濾器…等功能設定。
11. packages.config
    記錄專案所使用套件程式庫的版本。
12. Web.config
    應用程式組態檔。用來記錄資料庫連接字串或使用的 .NET Framework 版本。