LESSON 4

用 Razor 語法提升 View 的智慧與戰鬥力

前三章範例中，View 給人的印象是忠實顯示 Controller 傳來的資料，貌似樸拙而無突出才能。但事實上 View 也可以很智慧，因為 View 使用的是 Razor 語法，Razor 語法不但能宣告變數、集合，也能用判斷式、迴圈做出智慧型決策及顯示，遠比你想像中要來得強大。

4-1 Razor 概觀

什麼是 Razor？Razor 又稱 Razor Syntax，是用來將 Server Side 的 C#程式嵌入到 HTML 中的標記語法（Markup Syntax）。Syntax 字面上透露它是語法，而非 Language 語言。

❖ Razor 標記語法

所謂的「將 Server Side 的 C#程式嵌入到 HTML 中的標記語法」是何意？以下是一段 Razor 程式：

```
@{
      var Name = "Kevin";
      var Height = 180;
      var Weight = 75;
  }

<p>我的名字: @Name </p>
 <p>我的身高: @Height </p>
 <p>我的體重: @Weight </p>
```

1. C#程式區塊
C#變數
HTML 的<p> element
2. Razor Inline 表達式

說明：

1. Razor 中只有 HTML 及 C#兩種元素，二者的結合就形成了 Razor 語法。
2. C#程式區塊（Razor Code Block）是以@{...}包覆，裡面是一般 C#程式。
3. Razor Inline 表達式是指「C#變數穿插在 HTML 中」的式子。而 Razor 中預設是 HTML 語言，若遇到@符號，表示它後面接的是 C#指令。
4. Razor 會依不同的規則或符號在 HTML 和 C#之間做切換。

❖ Razor 是語法而非語言

那既然有 Razor 程式，似乎它就是語言，但又為何說 Razor 不是語言？因為 Razor 中只包含 HTML 和 C#，這些都不是 Razor 自己的，一個沒有變數、判斷式和 element tag 的東西，要稱為語言是有些勉強。同時 Razor 程式最後還會被轉換成 C#類別程式來執行，而不是真的有 Razor Language。

❖ Razor 支援的保留關鍵字

而「Razor 中只包含 HTML 和 C#」這句話是有但書的，雖然 HTML 全數可用，但不是所有 C# 指令或關鍵字都能在 Razor 中使用，下圖是 Razor 支援的保留關鍵字，分為兩大類：

1. Razor 關鍵字：section、model、helper、inherits 和 functions 五個關鍵字是 Razor 創造的，用來支持 Razor 語法所需功能。
2. C#關鍵字：這是源自既有的 C#，不是 Razor 所創造。Razor 支援常用的 C#關鍵字，但有些如 namespace 及 class 關鍵字就不支援。

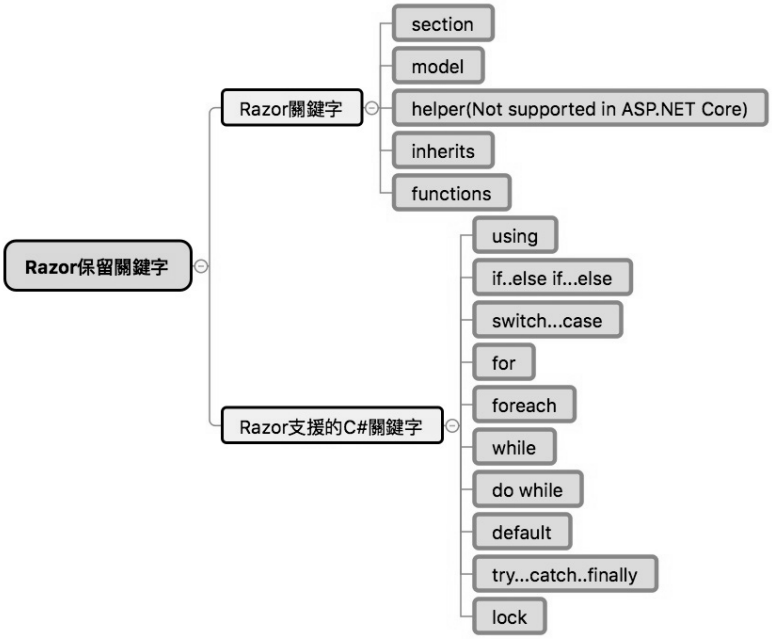

圖 4-1 Razor 保留關鍵字

最後，Razor 語法只能在 View 檢視（.cshtml）中使用，而不能在.html 中使用，所以 View 也稱為 Razor View 或 View Template。

4-2 Razor 語法規則

第一次接觸 Razor 的人，對於 Razor 究竟能做什麼常有疑問，簡言之，除了宣告 HTML 外，Razor 能用 C#宣告：變數、陣列、集合、判斷式、迴圈，也能使用 LINQ 語法。

但 Razor 有其語法規則與風格，若想駕馭，就得了解它的語法形式。請參考 MvcRazor 專案的 Views\Razor\RazorRules.cshtml，以下用 15 個規則解說 Razor 語法。

❖ 規則 1：以@符號作為 C#的開頭

Razor 語法包含：❶HTML 語法，❷ C#語法兩部分，遇到 HTML 的 Markup，就解析為 HTML（HTML Parser），遇到單一@符號開頭，就解析為 C#語法（C# Parser）。

❖ 規則 2：以@{...}宣告單行 C#程式

Razor 用@{...}包覆單行 C#程式，@{...}包含的區塊也稱為 Code Block 程式區塊，每行程式的結尾需加上;分號：

Views\Razor\RazorRules.cshtml

```
@{ var City = "Taipei"; }
@{ var PostalCode = 110; }

@*以下用明確型別宣告變數也可以*@
@{ string city = "Taoyuan"; }
@{ int postalCode = 334; }
```

說明：程式區塊中的 C#變數、集合或陳述式僅做設定或運算用途，而不做 HTML 輸出。而變數型別除了用 var 宣告外，亦可使用明確型別。

❖ 規則 3：以@{...}宣告多行 C#程式

多行程式也是用@{...}來包覆，只不過將程式分成多行：

```
@{
    var Name = "Kevin";
    var Height = 180;
    var Weight = 75;
}
```

❖ 規則 4：C#的 Inline 表達式

若 C#變數穿插在 HTML 中則為 Inline 表達式，以下用@Name、@Height 等 Inline 表達式將規則 2 和 3 的變數做顯示：

```
<p>我的名字：@Name </p>
<p>我的身高：@Height </p>
<p>我的體重：@Weight </p>
<p>居住城市：@City  </p>
<p>郵遞區號：@PostalCode </p>
```

HTML 輸出結果為：

```
我的名字：Kevin
我的身高：180
我的體重：75
居住城市：Taipei
郵遞區號：110
```

若以@符號顯示變數值，ASP.NET 一律將變數值做 HTML 編碼，輸出成純文字，例如@("<h1>MVC</h1>")會輸出<h1>MVC</h1>。

❖ 規則 5：C#程式區塊中的 HTML 隱式轉換

@{...}程式區塊中預設語言是 C#，但若夾雜了 HTML 語法，Razor 會自動做隱式轉換，將該部分輸出成 HTML：

```
@{
    var LeapYear = DateTime.IsLeapYear(DateTime.Now.Year);
    <p>今年是否為閏年：@LeapYear </p>
}
```

結果為：

```
今年是否為閏年：False
```

❖ 規則 6：C#關鍵字嚴格區分大小寫（但 VB 不分）

若 C#變數名稱相同，僅大小寫不同，Razor 仍視為兩個獨立變數：

```
@{
    var MyName = "聖殿祭司";
    var myName = "奚江華";
}
<p>
    筆名: @MyName <br />
    姓名: @myName <br />
</p>
```

結果為：

```
筆名: 聖殿祭司
姓名: 奚江華
```

❖ 規則 7：單行註解－@*...*@

單行註解用**@*...*@**表示：

```
@*這是單行註解*@
```

❖ 規則 8：多行註解－@*...*@

多行註解也是用**@*...*@**表示，只不過分成多行：

```
@*多行註解
也有支援*@
```

❖ 規則 9：Razor 隱性表達式－@符號

Razor 隱性表達式是由**@**符號開頭，系統會自動解析為 C#語法：

```
<p>現在的時間是：@DateTime.Now </p>
```

結果為：

```
現在的時間是：2017/11/5 下午 03:21:09
```

❖ 規則 10：Razor 明確表達式－@(...)符號

Razor 明確表達式是由**@(...)**所包覆，明確指出括號內是 C#運算式：

```
<p>兩週前我出國去玩，出發日期是：@((DateTime.Now - TimeSpan.FromDays(14)).
ToShortDateString()) </p>
<p>3+7 的結果是: @(3 + 7) </p>
```

出發日期是今天日期減 14 天，推算出兩週前的日期，結果為：

```
兩週前我出國去玩，出發日期是：2017/10/22
3+7 的結果是: 10
```

❖ 規則 11：以文字顯示@符號，需用@@表示

在 HTML 中顯示@文字，需加上第二個@做跳脫，也就是**@@**。例如在 HTML 顯示頭昏的表情符號@_@，語法為：

```
<p>
    @@_@@ <br />
<p>
<p>
```

```
    但 Email 和超連結例外<br />
    我的電子郵件: dotnetcool@gmail.com <br />
    <a href="mailto:service@domain.com">Service@domain.com</a>
</p>
```

結果為：

```
@_@
但 Email 和超連結例外
我的電子郵件: dotnetcool@gmail.com
Service@domain.com
```

❖ 規則 12：字串變數中的雙引號顯示

如果字串變數想顯示雙引號，在最前頭先加上@，字串內再用連續兩個""雙引號表示：

```
@{ var word = @"子曰："三人行，必有我師焉"...";}
<p>@word</p>
```

結果為：

```
「子曰："三人行，必有我師焉"...」。
```

❖ 規則 13：用@(...)將 HTML 或 JS 編碼成純文字

@(...)內除了做運算外，還可將表達式做 HTML 編碼，例如將 HTML 或 JavaScript 的表達式編碼成 HTML 文字：

```
@{ var msg = @"<button type='button' onclick='alert(""Hi JavaScript"")'> Raw
原始字串,不做 HTML 編碼</button>"; }
<p>@(msg)</p>  ◄──── 用@(...)進行 HTML 編碼
@("<span>Hello MVC!</span>") <br />
```

結果為：

```
<button type='button' onclick='alert("Hi JavaScript")'> Raw 原始字串, 不做 HTML 編
碼</button>
<span>Hello MVC!</span>
```

說明：無論 HTML 或 JavaScript 都會被編碼成純文字，目的是增加網頁安全性，不被注入網頁攻擊程式。

❖ 規則 14：用@Html.Raw()顯示原始字串, 不做 HTML 編碼

同一個 msg 字串變數，若想顯示原始值，不讓 HTML 或 JavaScript 被編碼，可用@Html.Raw(...)指令顯示：

```
@{ var msg = @"<button type='button' onclick='alert(""Hi JavaScript"")'> Raw 原
                始字串,不做 HTML 編碼</button>"; }
<p>@Html.Raw(msg)</p>
```

說明：

1. 結果會產生一個 HTML <button>按鈕，而不是文字，按下會有 JavaScript Alert 警告訊息。
2. 但若沒做 HTML 編碼可能會有潛在安全性問題。

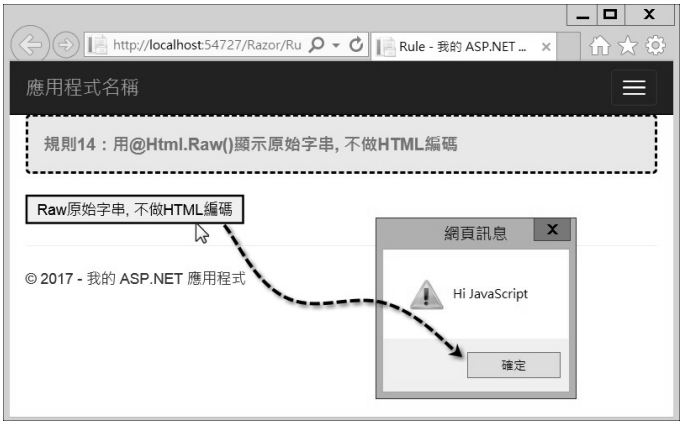

圖 4-2 顯示原始字串不做 HTML 編碼

4-4 以 Razor 語法判斷成績高低並標示不同顏色之實例

前面兩節介紹了 Razor 規則及流程控制，但這種單點式的指令介紹，很難讓人體會 Razor 實際妙用，故本節用幾個範例展現 Razor 的不同之處。第一個範例先製作學生成績列表的原型程式，也就是先不用 Razor 強化 View，後續範例再用 Razor 來改造，以便有清楚的對比。

範例 4-1 製作學生考試成績列表

在此製作學生成績列表，於 Controller 建立學生成績 model 資料，再將 model 傳給 View 作顯示，步驟如下：

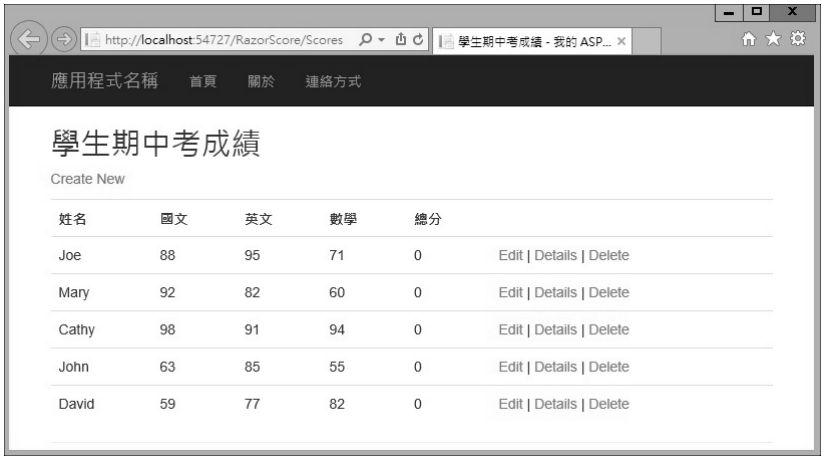

圖 4-4 學生考試成績列表

step01 建立 Model 模型。在 Models 資料夾加入 Student.cs 類別模型及六個屬性，完成後儲存，並按 Shift+Ctrl+B 建置專案。

Models\Student.cs

```
public class Student
{
```

```
    [Display(Name="學號")]
    public int Id { get; set; }
    [Display(Name = "姓名")]
    public string Name { get; set; }
    [Display(Name = "國文")]
    public int Chinese { get; set; }
    [Display(Name = "英文")]
    public int English { get; set; }
    [Display(Name = "數學")]
    public int Math { get; set; }
    [Display(Name = "總分")]
    public int Total { get; set; }
}
```

step**02** 建立 Controller。在 Controllers 資料夾新增 RazorScore 控制器，並以 List 泛型集合建立學生成績資料。

Controllers\RazorScoreController.cs

```
using MvcRazor.Models;
public class RazorScoreController : Controller
{
    //學生考試成績 Model 資料
    protected List<Student> students = new List<Student>
    {
        new Student { Id =1, Name="Joe", Chinese=88, English=95, Math=71 },
        new Student { Id =12, Name="Mary", Chinese=92, English=82, Math=60 },
        new Student { Id =23, Name="Cathy", Chinese=98, English=91, Math=94 },
        new Student { Id =34, Name="John", Chinese=63, English=85, Math=55 },
        new Student { Id =45, Name="David", Chinese=59, English=77, Math=82 }
    };
    ...
}
```

說明：以上是在 Class 類別層級建立 students 欄位，而 students 是 List 泛型集合，裡面包含所有學生資料。

step**03** 建立 Action 方法。在 RazorScore 控制器加入 Scores()方法，將 model 傳給 View。

Controllers\RazorScoreController.cs

```
public ActionResult Scores()
{
    return View(students);
}
```

step04 建立 View 檢視。在 Scores ()方法按滑鼠右鍵→【新增檢視】→範本選擇【List】→模型類別選擇「Student (MvcRazor.Models)」→【加入】。

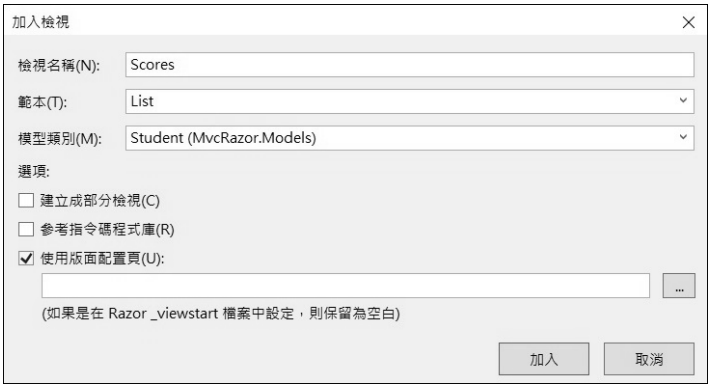

圖 4-5 以 Scaffolding 產生 View

step05 修改 View 的標題文字。

Views\RazorScore\Scores.cshmtl

```
@model IEnumerable<MvcRazor.Models.Student>
@{
    ViewBag.Title = "學生期中考成績";
}
<h2>學生期中考成績</h2>
...
```

說明：按 F5 執行 RazorScore/Scores 網頁，畫面如圖 4-4，總分目前為 0，後續範例會說明如何做動態加總。

範例 4-2 以 Razor 語法判斷學生成績高低及找出總分最高者

於此改造前一範例，以 Razor 語法做成績判斷，找出❶低於 60 分、❷高於 95 分、❸總分第一名，並以不同顏色標示，步驟如下：

學號	姓名	國文	英文	數學	總分
1	Joe	88	95	71	254
12	Mary	92	82	60	234
23	Cathy	98	91	94	283 (總分排名第一)
34	John	63	85	55	203
45	David	59	77	82	218

圖 4-6 以 Razor 做成績判斷

step01 在 RazorScore 控制器加入 ScoresRazor()方法，先計算出每位學生總分，再找出總分最高者。

Controllers\RazorScoreController.cs

```
public ActionResult ScoresRazor()
{
    //找出總分最高者之 Id
    int topId = 0;
    int topScore = 0;

    foreach (var student in students)
    {
        //計算總分
        student.Total = student.Chinese + student.English + student.Math;

        //判斷總分最高者
        if (student.Total > topScore)
        {
            topScore = student.Total;
            topId = student.Id;
        }
```

```
    }

    //將最高分學生 Id 儲存到 ViewBag，傳遞給 View
    ViewBag.TopId = Convert.ToInt32(topId);

    return View(students);
}
```

step02 在 ScoresRazor()按滑鼠右鍵→【新增檢視】→範本選擇【List】→模型類別選擇「Student (MvcRazor.Models)」→【加入】。

step03 在 ScoresRazor.cshtml 的<tbody>區段，以 Razor 判斷每科成績高低及顯示總分。

Views\RazorScore\ScoresRazor.cshtml

```
...
<table class="table table-bordered table-striped">
    <thead>
        <tr>
            <th>@Html.DisplayNameFor(m => m.Id)</th>
            <th>@Html.DisplayNameFor(m => m.Name)</th>
            <th>@Html.DisplayNameFor(m => m.Chinese)</th>
            <th>@Html.DisplayNameFor(m => m.English)</th>
            <th>@Html.DisplayNameFor(m => m.Math)</th>
            <th>@Html.DisplayNameFor(m => m.Total)</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var m in Model)
        {
            var total = m.Chinese + m.English + m.Math;
            <tr>
                <td>@Html.DisplayFor(x => m.Id)</td>
                <td>@Html.DisplayFor(x => m.Name)</td>
```

用 if 判斷中文成績

```
<!--中文-->
@if (m.Chinese < 60)
{
    <td class="poor">@Html.DisplayFor(x => m.Chinese)</td>
}
else if (m.Chinese >= 95)
{
    <td class="excellent">@Html.DisplayFor(x => m.Chinese)</td>
```

低於 60 賦予的 CSS 樣式

高於 95 賦予的 CSS 樣式

```
                }
                else
                {
                    <td>@Html.DisplayFor(x => m.Chinese)</td>
                }

                <!--英文-->
                @if (m.English < 60)
                {
                    <td class="poor">@Html.DisplayFor(x => m.English)</td>
                }
                else if (m.English >= 95)
                {
                   <td class="excellent">@Html.DisplayFor(x => m.English)</td>
                }
                else
                {
                    <td>@Html.DisplayFor(x => m.English)</td>
                }

                <!--數學-->
                @if (m.Math < 60)
                {
                    <td class="poor">@Html.DisplayFor(x => m.Math)</td>
                }
                else if (m.Math >= 95)
                {
                    <td class="excellent">@Html.DisplayFor(x => m.Math)</td>
                }
                else
                {
                    <td>@Html.DisplayFor(x => m.Math)</td>
                }

                <!--顯示總分-->
                @if (m.Id == ViewBag.topId)
                {
                    <!--總分最高者-->
                    <td class="top1">@Html.DisplayFor(x => m.Total)</td>
                }
                else
                {
                    <td>@Html.DisplayFor(x => m.Total)</td>
                }
            </tr>
        }
    </tbody>
</table>
```

用 if 判斷英文成績

用 if 判斷數學成績

顯示總分

說明：View 中除了 Razor 判斷式外，還做了兩類改造：

1. 在 Table 表格宣告明確加上<thead>及<tbody>。
2. 在 DisplayNameFor()及 DisplayFor()方法中，以更精簡的變數名稱替代。

■ 改造前語法

```
@Html.DisplayNameFor(model => model.Id)
@Html.DisplayNameFor(model => model.Name)
...
@Html.DisplayFor(modelItem => item.Id)
@Html.DisplayFor(modelItem => item.Name)
```

■ 改造後語法

```
@Html.DisplayNameFor(m => m.Id)
@Html.DisplayNameFor(m => m.Name)
...
@Html.DisplayFor(x => m.Id)
@Html.DisplayFor(x => m.Name)
```

精簡變數名稱除了讓宣告變得更簡潔外，另一個用意是點出，View 的 model 及 Lambda 參數名稱是可隨需求做更動的。

step04 在 View 的末端加入自訂 CSS。

Views\RazorScore\ScoresRazor.cshtml

```
@section topCSS{
    <style type="text/css">
        /*設定 Table 欄位標題顏色*/
        th {
            color: white;
            background-color: black;
            text-align: center;
        }

        /*設定 Table 資料列 Hover 時的光棒效果*/
```

```
        .table > tbody > tr:hover {
            background-color: antiquewhite !important;
        }

        /*成績不及格之 CSS*/
        .poor {
            color: white !important;
            background-color: red !important;
        }

        /*成績優秀之 CSS*/
        .excellent {
            background-color: aqua !important;
        }

        /*總分第一名之 CSS*/
        .top1 {
            background-color: yellow !important;
            border: 2px dashed black !important;
            font-weight:900;
            font-size:1.2em;
        }

        .top1::after {
            content: ' (總分排名第一)';
        }
    </style>
}
```

step05 在_Layout.cshtml 也要配合加入 topCSS、topJS、endCSS 及 endJS 四個 RenderSection()宣告，細節可回頭參考範例 3-4。

範例 4-3 在 View 中以純粹的 Razor 及 LINQ 語法找出總分最高者

前一範例是在 Action 中找出總分最高者 Id，再傳給 View。在這要介紹的是在 View 中，以純粹的 LINQ 語法找出總分最高者，請參考 Scores RazorPure.cshtml：

View\RazorScore\ScoresRazorPure.cshtml

```
@using MvcRazor.Models;
@model IEnumerable<MvcRazor.Models.Student>

@{
    ViewBag.Title = "學生期中考成績";

    //計算所有學生總分
    var Students = from s in Model
         select new Student { Id = s.Id, Name = s.Name, Chinese = s.Chinese,
         English = s.English, Math = s.Math, Total = s.Chinese + s.English +
         s.Math };


    //找出最高總分
    int topScore = Students.Max(s=>s.Total);
}
...
<table class="table table-bordered table-striped">
    <thead>
        <tr>
            <th>@Html.DisplayNameFor(m => m.Id)</th>
            ...
        </tr>
    </thead>
    <tbody>
        @foreach (var m in Students)
        {
            <tr>
                <td>@Html.DisplayFor(x => m.Id)</td>
                ...
                @if (m.Total == topScore)
                {
                    <!--總分最高者-->
                    <td class="top1">@Html.DisplayFor(x => m.Total)</td>
                }
                else
                {
                    <td>@Html.DisplayFor(x => m.Total)</td>
                }
            </tr>
        }
```

```
    </tbody>
</table>

...
```

4-5 以@helper 指示詞建立可重複使用的 Razor Helper

Razor Helper 是將一段 HTML 或 Razor 語法獨立成可重複使用的區塊，方式是在 View 中以@helper 關鍵字來宣告。

例如，前一節 Razor 用 if 判斷國文、英文、數學成績的高低，並給予不同的背景色：

```
<!--中文-->
@if (m.Chinese < 60)
{
    <td class="poor">@Html.DisplayFor(x => m.Chinese)</td>
}
else if (m.Chinese >= 95)
{
    <td class="excellent">@Html.DisplayFor(x => m.Chinese)</td>
}
else
{
    <td>@Html.DisplayFor(x => m.Chinese)</td>
}
```

若將這段判斷式獨立成 Razor Helper－DisplayScore()：

```
@helper DisplayScore(int score)
{
    if (score < 60)
    {
        <td class="poor">@score</td>
    }
    else if (score >= 95)
```

傳入的參數

自訂 Helper 的名稱

```
    {
        <td class="excellent">@score</td>
    }
    else
    {
        <td>@score</td>
    }
}
```

說明：Razor Helper 在結構及使用上，跟 C# Method 很像，只不過前者在 Razor 頁面中宣告，後者在 Class 類別中宣告。

在 View 中呼叫 Razor Helper 只一行程式，且還能❶精簡 View 主程式，❷Razor Helper 重複使用：

```
<!--中文成績-->
@DisplayScore(m.Chinese)
```

範例 4-4 建立可重複使用的 Razor Helper

在此將國文、英文、數學的判斷式改寫成可重複使用的 Razor Helper，請參考 ScoresRazorHelper.cshtml，步驟如下：

step01 在 RazorScore 控制器新增 ScoresRazorHelper()方法，Action 程式照舊。

step02 在 ScoresRazorHelper ()方法按滑鼠右鍵→【新增檢視】→範本選擇「List」→模型類別選擇「Student (MvcRazor.Models)」→【加入】。

step03 在 View 中自訂 Razor Helper -- DisplayScore()，以下為關鍵程式。

Views\RazorScore\ScoresRazorHelper.cshtml

```
<table class="table table-bordered table-striped">
   ...
    <tbody>
```

```
        @foreach (var m in Model)
        {
                                    2.呼叫使用--自訂 Helper--DisplayScore()
            ...
                <!--中文-->
                @DisplayScore(m.Chinese)
                <!--英文-->
                @DisplayScore(m.English)
                <!--數學-->
                @DisplayScore(m.Math)

                <!--計算總分-->
                @if (m.Id == ViewBag.topId)
                {
                    <!--總分最高者-->
                    <td class="top1">@Html.DisplayFor(x => m.Total)</td>
                }
                else
                {
                    <td>@Html.DisplayFor(x => m.Total)</td>
                }
            </tr>
        }
    </tbody>
</table>
                                    1.自訂 Helper -- DisplayScore()
@helper DisplayScore(int score)
{
    if (score < 60)
    {
        <td class="poor">@score</td>
    }
    else if (score >= 95)
    {
        <td class="excellent">@score</td>
    }
    else
    {
        <td>@score</td>
    }
}
```

4-6 將個別頁面中的 Razor Helper 提升到全網站共用

前面 Helper 在個別 View 中定義，但這也會限制於該頁面才能使用，其他 View 頁面無法共用。解決方式是將 Helper 提升到全網站共用，關鍵是：

“將 Helper 程式移至網站的 App_Code 資料夾，檔名一樣用.cshtml。”

範例 4-5 自訂全網站可用的 Razor Helper

在此將自訂的 Helper 移至 App_Code 資料夾，變成全網站可使用，請參考 GlobalHtmlHelper.cshtml，步驟如下：

step01 在 RazorScore 控制器新增 GlobalHtmlHelper ()方法，Action 程式照舊。

step02 在 GlobalHtmlHelper ()方法按滑鼠右鍵→【新增檢視】→範本選擇【List】→模型類別選擇「Student (MvcRazor.Models)」→【加入】。

step03 在專案中加入 App_Code 資料夾。

圖 4-7 專案加入 App_Code 資料夾

step04 在 App_Code 資料夾新增 MyHelper.cshtml，並加入兩個 Helper。

App_Code\MyHelper.cshtml

```
@helper DisplayScore(int score)
{
    if (score < 60)
    {
        <td class="poor">@score</td>
    }
    else if (score >= 95)
    {
        <td class="excellent">@score</td>
    }
    else
    {
        <td>@score</td>
    }
}

@helper DisplayTopScore(int id, int topId, int total)
{
    if (id == topId)
    {
        <td class="top1">@(total)</td>
    }
    else
    {
        <td>@(total)</td>
    }
}
```

說明：第二個 Helper 是將原本找出最高總分者，獨立成 DisplayTopScore()，進一步精簡主程式及提高重複使用性。

step05 在 GlobalHtmlHelper.cshtml 呼叫全域 MyHelper。

Views\RazorScore\GlobalHtmlHelper.cshtml

```
<tbody>
    @foreach (var m in Model)
    {
```

```
            ...
            <!--中文-->
            @MyHelpers.DisplayScore(m.Chinese)
            <!--英文-->
            @MyHelpers.DisplayScore(m.English)
            <!--數學-->
            @MyHelpers.DisplayScore(m.Math)
            <!--總分-->
            @MyHelpers.DisplayTopScore(m.Id, topId, m.Total)
        </tr>
    }
</tbody>
```

說明：此時 MyHelpers 已是全網站可共用，於任何 Razor 頁面中皆可呼叫。

step06 將 GlobalHtmlHelper.cshtml 底部的@section topCSS{...}中，一長串 CSS 移出成 customStyle.css 檔，然後加入<link>參考，做最終簡化。

Views\RazorScore\GlobalHtmlHelper.cshtml

```
@section topCSS{
    <link href="~/Assets/css/customStyle.css" rel="stylesheet" />
}
```

說明：由此可知，以@section 自訂 CSS 或 JavaScript 中，可放原始程式定義，也可以放檔案的超連結。

4-7 建立可重複使用的 Partial View 部分檢視

本節要介紹 Partial View 部分檢視，它是可重複使用的檢視區塊，相較於一般 View，Partial View 著眼的是可重複使用區塊，和製作一個大型完整的 View 出發點是不同的。

4-7-1 Partial View 運作方式與特性

實際上 Partial View 是如何運作？首先設計一個 Partial View 區塊內容（.cshtml），這個區塊內容像是一個小零件，提供給所有 Views 呼叫使用。Partial View 若被一般 Views 呼叫，Partial View 的內容就會 Render 加入到 Parent View 中。

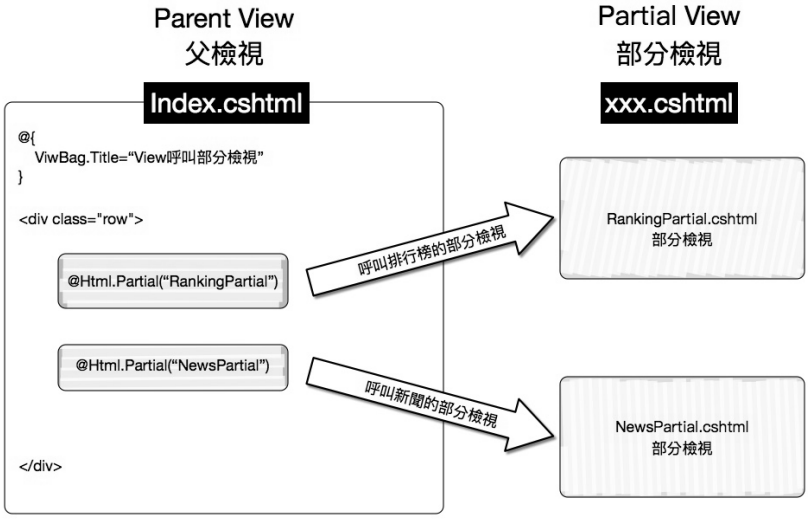

圖 4-8 部分檢視運作關係

以下是 Partial View 的特性說明：

1. 呼叫 Partial View 的 View 就是 Parent View 父檢視。
2. 父檢視呼叫 Partial View 的指令是@Html.Partial("部分檢視.cshtml")。
3. Partial View 延伸檔名與 View 相同，都是.csthml。
4. Partial View 可放在個別的 View 資料夾，同一個資料夾的 Parent View 會優先參考使用。也可放在 Views\Shared 資料夾，讓所有資料夾的 Parent View 共用。

5. 因此 Parent View 尋找 Partial View 的順序，是先在本身所在的資料夾搜尋，若找不到的話，才會到 Shared 資料夾中尋找。
6. 如果是通用佈局設計應該放在_Layout.cshtml 佈局檔中，非佈局類但可重複使用內容，就可以做成 Partial View。

❖ Partial View 和 View 的區別

雖然 Partial View 和 View 延伸檔名皆為.cshtml，但二者仍有些區別：

1. Partial View 不會執行_ViewStart.cshtml。
2. 若在 Partial View 中宣告 Section，Section 不會出現在 Parent View，等於是無法作用。
3. 雖然 Partial View 可加入<style />和<script />，樣式和 JavaScript 會被帶入 Parent View 中。但是若 Parent View 呼叫 Partial View 多次，這些樣式和 JavaScript 也會重複產生多次，需思考是否合適？
4. Partial View 支援鏈狀（chained）呼叫，就是可以再層層呼叫其他的 Partial View。

4-7-2 Partial View 的資料傳遞方式

Parent View 傳遞資料給 Partial View，可用 ViewData 或 model 物件，分述如下。

❖ 以 ViewData 傳遞資料給 Partial View

Parent View 在呼叫 Partial View 時，一併傳遞 ViewData 的語法：

```
@Html.Partial("PartialView 名稱", ViewData)
```

說明：

1. Partial View 初始時，它會得到 Parent View 的 ViewDataDictionary 複本。
2. 但 Partial View 更動 ViewData 資料，它不會更新回 Parent View。
3. Partial View 回傳時，它的 ViewData 便會消失。

❖ 以 model 物件傳遞資料給 Partial View

Parent View 傳遞 model 物件給 Partial View 的語法：

```
@Html.Partial("PartialName", model 物件)
```

說明：傳遞 model 的方式，同時適用 Partial、PartialAsync、RenderPartial 及 RenderPartialAsync 方法。

範例 4-6 將人物牌卡製作成 Partial View，供所有 View 呼叫使用

在此將人物牌卡製作成 Partial View，讓所有 View 呼叫使用，步驟如下：

圖 4-9 將人物牌卡製作成 Partial View

step01 在 Views\Shared 資料夾按滑鼠右鍵→【加入】→【檢視】→【建立成部分檢視】，命名為「_SimpleCardPartial」。或用【加入】→【MVC5 部分頁面(Razor)】。

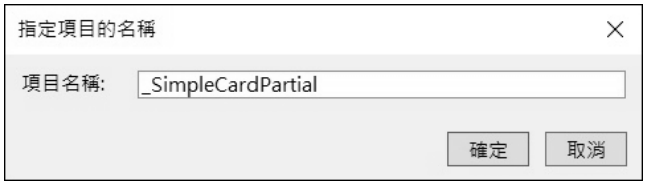

圖 4-10 加入 Partial View 部分檢視檔

step02 Partial View 建立之初內容空無一物，以下加入 HTML 及 Bootstrap 宣告後，成為圖 4-9 的人物牌卡。

Views\Shared_SimpleCardPartial.cshtml

```
<div class="col-xl-3 col-lg-4 col-md-6 col-sm-12">
    <div class="card">
        <div class="headshot">
            <img class="card-img-top" src="~/Assets/images/MarkZuckerberg.jpg"
              alt="">
        </div>
        <div class="card-block">
            <h3 class="card-title">Mark Zuckerberg</h3>
            <p class="card-text">Facebook 創辦人 馬克· 祖伯克</p>
            <a href="https://goo.gl/BktGGA" class="btn btn-primary">Wiki</a>
        </div>
    </div>
</div>
```

step03 在 Controllers 資料夾新增 PartialView 控制器，並加入 SimpleCard()。

Controllers\PartialViewController.cs

```
public ActionResult SimpleCard()
{
    return View();
}
```

step04 在 SimpleCard ()方法按滑鼠右鍵→【新增檢視】→範本【Empty (沒有模型)】→【加入】，並加入以下程式，用@Html.Partial ("_SimpleCardPartial")呼叫_SimpleCardPartial.cshtml 部分檢視。

Views\Shared\SimpleCard.cshtml

```
@{
    ViewBag.Title = "SimpleCard";
}
<div class="jumbotron">
    <h2>人物牌卡(簡單型)</h2>
</div>
                    Bootstrap 的 row
<div class="row">
    <!--以下連續呼叫 Partial View 九次-->
    @Html.Partial("_SimpleCardPartial")   ← Parent View 呼叫 Partial View
    @Html.Partial("_SimpleCardPartial")
    @Html.Partial("_SimpleCardPartial")
    @Html.Partial("_SimpleCardPartial")
    @Html.Partial("_SimpleCardPartial")
    @Html.Partial("_SimpleCardPartial")
    @Html.Partial("_SimpleCardPartial")
    @Html.Partial("_SimpleCardPartial")
    @Html.Partial("_SimpleCardPartial")

    <!--等同以下用迴圈執行九次-->
    @for (int i = 0; i < 9; i++)
    {
        @Html.Partial("_SimpleCardPartial");
    }
</div>

<!--以下是牌卡的樣式定義-->
@section topCSS{
    <style>
        .jumbotron {
            text-align: center;
            background-color: lightcoral;
        }

        .card {
            border: 1px solid black;
```

```
            margin-bottom: 30px;
        }

        .card img {
            width: 100%;
        }

        .card:hover .card-block {
            background-color: lightgreen;
        }

        .card, .headshot {
            overflow: hidden;
        }

        .card-block {
            border-top: 1px solid black;
            background-color: #ffbf67;
            padding: 10px;
        }

        .card-title {
            color: white;
            background-color: black;
            display: inline-block;
            border-radius: 5px;
            padding: 5px;
        }
    </style>
}
```

說明：

1. 前面提過，Partial View 不能使用 Section 加入 css 或 js，因為不會出現在 Parent View 中。
2. 同時也不能用<style />將 css 加入到 Partial View 中，技術上雖沒問題，但 Parent View 一連呼叫九次，這段 css 就會重複出現九次。
3. 故將牌卡 css 定義在 Parent View 中才是正確的。

最後瀏覽 PartialView/SimpleCard 頁面，由於用 Html.Partial()呼叫了九次 Partial View，便出現九張人物牌卡。

圖 4-11 Partial View 生成後的九張牌卡

這範例簡單扼要，可快速理解 Partial View 運作方式。但重複出現九張相同牌卡沒有太大意義，倘若 Partial View 能結合資料，動態產生不同的牌卡內容，就會有更高的實用價值，程式只需稍做調整即可，且看以下範例。

範例 4-7 傳遞 model 資料到 Partial View，動態生成不同的牌卡

在這以前一個範例為基礎，讓 Partial View 接受 model 物件資料，以動態生成不同的牌卡，步驟如下：

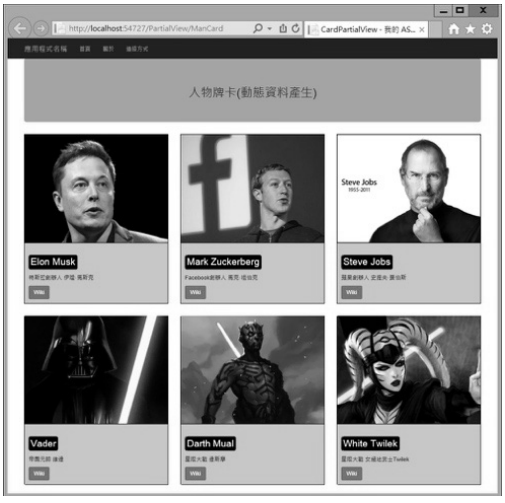

圖 4-12 將 model 資料傳入 Partial View 動態產生人物牌卡

step01 在 Models 資料夾新增一 Card 類別模型。

Models\Card.cs

```
public class Card
{
    public int Id { get; set; }
    public string Name { get; set; }     //名字
    public string Brief { get; set; }    //簡介
    public string Photo { get; set; }    //照片
    public string WikiUrl { get; set; } //Wiki 的 Url
}
```

step02 在 PartialView 控制器新增 ManCard()程式，裡面建立人物牌卡資料。

Controllers\PartialViewController.cs

```
public ActionResult ManCard()
{
    List<Card> cards = new List<Card>
    {
        new Card{ Name = "Elon Musk", Brief="特斯拉創辦人 伊隆· 馬斯克",
             Photo="ElonMusk.jpg", WikiUrl="https://goo.gl/46xeXx" },
```

```
        new Card{ Name = "Mark Zuckerberg", Brief="Facebook 創辦人 馬克· 祖伯克",
             Photo="MarkZuckerberg.jpg", WikiUrl="https://goo.gl/BktGGA" },
        new Card{ Name = "Steve Jobs", Brief="蘋果創辦人 史提夫· 賈伯斯",
             Photo="SteveJobs.jpg", WikiUrl="https://goo.gl/nAiX0y" },
        new Card{ Name = "Vader", Brief="帝國元帥  維達", Photo="Vader.jpg",
             WikiUrl="https://en.wikipedia.org/wiki/Darth_Vader" },
        new Card{ Name = "Darth Mual", Brief="星際大戰 達斯摩", Photo="DarthMual.jpg",
             WikiUrl="https://goo.gl/5obLhX"},
        new Card{ Name = "White Twilek", Brief="星際大戰 女絕地武士 Twilek",
             Photo="WhiteTwilek.jpg", WikiUrl="https://goo.gl/reKzAu" }
    };

    return View(cards);
}
```

說明：這裡初始一個 List 泛型集合，每筆資料都有 Name、Brief、Photo 和 WikiUrl 設定值，然後將 cards 這個 model 物件傳給 View。

step**03** 在 ManCard ()方法按滑鼠右鍵→【新增檢視】→範本【Empty(沒有模型)】→【加入】，並建立父檢視程式。

Views\PartialView\ManCard.cshtml

```
@model IEnumerable<MvcRazor.Models.Card>
@{
    ViewBag.Title = "CardPartialView";
}
<div class="jumbotron">
    <h2>人物牌卡(動態資料產生)</h2>
</div>

<div class="row">
    @foreach (var man in Model)
    {
        @Html.Partial("_CardPartial", man);
    }
</div>

@section topCSS{
   <link href="~/Assets/css/card.css" rel="stylesheet" />
}
```

呼叫 Partial View，同時傳入 model 物件 man

將原本的 css 獨立成 card.css

說明：Parent View 逐一取出 model 中的項目，在以 Html.Partial() 呼叫 Partial View 時，一併將 man 物件傳遞給 Partial View 使用。

step04 在 Views\Shared 資料夾按滑鼠右鍵→【加入】→【MVC5 部分頁面(Razor)】，命名為「_CardPartial」。

Views\Shared_CardPartial.cshtml

```
@model MvcRazor.Models.Card
<div class="col-xl-3 col-lg-4 col-md-6 col-sm-12">
    <div class="card">
        <div class="headshot">
           <img class="card-img-top" src="~/Assets/images/@Model.Photo" alt="">
        </div>
        <div class="card-block">
            <h3 class="card-title">@Model.Name</h3>
            <p class="card-text">@Model.Brief</p>
            <a href="@Model.WikiUrl" class="btn btn-primary">Wiki</a>
        </div>
    </div>
</div>
```

將 model 的屬性值帶入 HTML 中

將 model 的屬性值帶入 HTML 中

說明：執行 PartialView/ManCard 頁面，即可看到動態產生的不同人物牌卡。

❖ 其他章節的 Partial View 應用

除本章外，其他章節專案亦有用到 Partial View，如：

1. MvcBootstrap 專案

   在 Views\Shared 資料夾中有以下 Partial View：

   - ArticlePartial.cshtml
   - BreakpointPartial.cshtml
   - ColumnHeaderPartial.cshtml
   - MoviePartial.cshtml

2. MvcJqueryMobile 專案

   Views\Pages\MultiPagesPartial.cshtml 和 MultiPagesPartialDB.cshtml 也呼叫 Views\Shared\HeroPartial.cshtml 部分檢視。

若想知道哪些父檢視呼叫哪個 Partial View，可在 Visual Studio 按 Ctrl + F，以「@Html.Partial」關鍵字對整個方案做「全部尋找」，就會全部列出。

4-8 結論

Razor 語法是設計 View 檢視頁必定會用到的，對於宣告變數、陣列或集合，甚至是 model 資料讀取與顯示，都需 Razor 語法的幫助。當然 Razor 還可以做進階邏輯判斷與迴圈處理，再輔以 Helper 及 Partial View 的運用，設計 View 時，工作可以更輕鬆、更有智慧。