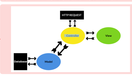
ASP.NET MVC

ASP.NET MVC 5

- 何謂 MVC Pattern
  - 目前最工程中的一種軟體架構模式，把軟體系統分為三個基本部分：模型（Model）、視圖（View）和控制器（Controller）。
    - 

    - Source: https://techaffinity.com/blog/mvc-architecture-benefits-of-mvc/
- ASP.NET MVC 概觀
  - ASP.NET MVC 是一個開發網站應用程式的技術，結合這項技術，我們可以開發出
    - 公開的 Internet 網站
      - 官方形象網站、購物網站、部落格網站、論壇等形式的網站
      - 功能編輯軟 ,例如官方形象網站, 討論板
      - 至於購物網站才會連線紀錄機制
      - 現代論壇用戶不必學習就能直接開始網站各功能
      - 注冊網頁外觀
    - 非公開網站
      - 公司內部使用的網站，例如員工請假系統、文件管理系統、會議室預約系統、甚至是小午訂便當系統。
      - Intranet 網站應用系統
        - 公司內部使用的網站，例如員工請假系統、文件管理系統、會議室預約系統、甚至是小午訂便當系統。
        - 功能編輯軟 ,會有紀錄步驟清單選項
        - 紀錄不注意網頁外觀, 整頓即可
      - extranet 網站應用系統
        - 有些網站並不是供外界任何人存取，例如：電商網站會需要讓會員或供應商，以及供貨商能夠內容訂單部份資訊，以利作業。
        - 比較沙箱測 ,有紀錄步 Web API, 供不同編碼內容之後, 互通資料
        - 通常會鎖定 IP, 不提供任何人都能連結
- .NET 版本
  - ASP.NET MVC 5
    - 使用 .NET Framework 4.8
    - 本課程主要是介紹它
  - ASP.NET Core MVC
    - 使用 .NET Core, 現階段用 3.1, 5, 6
    - 介紹如何使用 VS 新增 MVC 專案, 並執行它
    - 建立新方案
    - 如何新建並找到 MVC 專案, 指定 .NET Framework 版本
      -

        - 如何執行並檢視網頁
    - 如何開啓 Visual Studio 新增 ASP.NET MVC 專案
    - 再加入 ASP.NET MVC project 與空開發一個空專案, 初學者需要多個專案協同運作, 因此建議先建立 solution, 再加入 project!
  - 影片說明
  - 如何開啓 ASP.NET MVC 專案
    -

  - 如何強行使用最新版本的 MVC 專案, 並執行它
  - 如何使用或啟用 https
- 建立第一個 ASP.NET MVC 專案
  - 資料夾說明
    - Controllers
      - 用來存放Controller控制器/檔案
    - Views
      - 用來存放 View Page, 會根據 Controller名稱來分資料夾
      - 例如 HomeController會在 Views/Home/ 之下存放對應的 View Pages
    - Models
      - 存放 Model , classes, 目前非專案沒有用到
  - Controller class 說明
    - 一個class 內含多個method, 在ASP.NET MVC 裡面門稱 method 為 Action動作)
    - 例如 HomeController 下, 包含 Index, Contact, About 等 Actions
  - 新增 /Home/Greeting/
    - 在 HomeController 裡面增 Greeting Action
    - 如何在 Controller 裡頭調用 View Page
      -

    - 如何讓 Controller 傳遞資訊給 View Page
      - 在 layout page 中
        - 加入 viewBag.item "greeting"
    - ASP.NET MVC 的url 並不直接, 會根據規則 / Home/Index 轉
    - 會認以為網站的 Home folder, 會有 index 網頁
    - 但其實不是
    -
  - 操作流程簡介
    - 為何網址沒有 /Home/Index 也能運行 (HomeController.Index)
      - 預設專案會用 global.asax
    - routing 的基本觀念
      - 只要輸入網址的格式符合預設的設定值, 它就能夠在標準內尋找一個 Controller 裡頭的一個 Action
      - 在預設值裡, Controller, Action 都可以不填, 會使用預設值, 分別是 Home , Index
    - 練習
      - 新增 MemberController 並準備好使用權限
      - 在Controller 裡頭增 Create Action
      - 新增 View Page
      - 在 NavBar 加入超連結, "新增會員"
    - 何謂資料模型配置
      - 在專案裡, 有固定的資料夾名稱 Model, View, Controller 各有各預設位置
      - 內容分類清晰工程師日後上手絕佳, 不至於連程式設計師都搞不清楚狀況
    - 何謂類工廠分類
      - view page 只需取用資訊產典資呈現, 至於資訊如何取得? 它不必管
    - 建立 Controller - RegistersController
      - 建立一個可管理或 Database 的模組/ 即為 eventLogin
      - 123
      - Database name : dbEvent
      - Table name : Registers
        - Id, int, PK
        - Name, NVarchar(50), Not Null
        - Email, NVarchar(100), Not Null
        - CreatedTime, DateTime, Not Null
    - 完成 !!!
- 作業
  - 建立 Table: Members
    - Id, PK, int, NOT Null
    - Name , Nvarchar(50), Not Null
    - Account , Nvarchar(30), Not Null , Unique
    - Password, Nvarchar(30), Not Null
    - CellPhone , Varchar(10), Allow Null
- 使用 ScaffoldingMVC 5 快速生成 CRUD 的網頁功能
  - 建立 Database, Table
  - 建立 ASP.NET MVC "Events.Site"
  - 建立 Entity Framework 物件模式
    - 已有一個用來存取 Database 的物件
    - 在 /Models/ 下, 新增 EFModels 資料夾
      - 新增 ADO.NET Entity Model, 預設名稱是 "Model", 它關聯 Database 的角色
        - 勾選 "使用單數化命名", 可以將資料表的名稱生成 class時, 自動改成單數的英文
      - 確認是否正確
        - /Models/EFModels
          - Model.cs
          - Register.cs
        - web.config中, 應該會有新增一個連線字串
      - 查閱檢視 Entity Framework 屬性
      - 建置專案
        - 每次建立 ADO.NET Entity Model 時, 都必須重新建立專案, 否則報告有步驟發生錯誤
        - 它會在背後自動建立 Controller, View Pages, 且在 Controller 中加載相關內容 (P 開頭 DB 的模式)
      - 建立 Controller - RegistersController
        - 如果剛才沒有建置專案, 可以到這步驟會發生錯誤, 請重新建置, 建置專案, 再重來一次即可
        - 完成後, RegistersController 程式已自動寫好
        - View/Registers/ 包括了 Index, Create, Edit, Details, Delete 等 view pages
    - 完成 !!!
  - 由於在開發專案的過程裡, 難免會中途修補影響/更新資料表, 如果有異動, 請重建上方的 EFModels 資料夾建立 DbContext, Entity classes 的步驟, 覆蓋舊有檔案即可, 也能保留舊檔案, 將以前的自訂修改內容手動修改 EFModels 模型模式檔, 避免已做又被覆蓋。
  - Index page 流程說明
    -

    - 1. 輸入 Registers/Index 網址
    - 2. 叫用 RegistersController.Index()
    - 3. 經由 DbContext 取得 Registers 資料表的記錄
    - 4. 傳回 Registers 記錄
    - 5. Index() 傳回 Index View Page, 並將 Registers 記錄傳給此 view page
    - 6. 生成最後的 網頁 html
    - 7. 回傳給 Browser
  - Create 功能的運作流程
    - 第一次運行
      -

      - 1. Browser 輸入 Registers/Create 網址
      - 2. 叫用 RegistersController.Create(), HttpGet 的版本 (函數 class 裡有兩支 Create method)
      - 3. 單純傳回 View page
      - 4. Create view page 生成html
        - Create view page 裡的 LabelFor 用來生成欄位名稱
        - EditorFor 用來生成 textbox控制項
        - ValidationMessageFor 供顯示輸入驗證訊息, 顯示錯誤訊息之用
      - 傳回Browser, 此時 Browser 呈現一個空白的表單, 供使用者輸入
    - 第二次運行
      - 使用者填完表單, 並按下送出, 送出表單
      - Browser 以 post 方式送出表單到 Register/Create
      - 叫用 RegistersController.Create(), HttpPost 的版本
      - 建立一筆紀錄到 database
      - 沒有傳回值
      - Create 轉向到 Index
        - Redirect(ToAction("Index") 表示要轉換到 Registers/Index
      - 產生 Index 的資料庫查詢, 請參考上一個段的時間流程, 傳回 Index view page 的結果
      - 將 index 生成即時 html 傳回Browser, 此時 Browser 呈現已轉名者紀錄
  - Edit功能的運作流程例 Create 相似)
    - 第一次運行
      -

      - Browser 輸入 Registers/Edit/99 網址(其中99表示 id值)
      - 叫用 RegistersController.Edit(), HttpGet 的版本(函數 class 裡有兩支 Edit method)
      - 去資料庫取得要編輯的紀錄
        - 如果找不到該筆紀錄, 就傳回 HttpNotFound(), 會顯示錯誤訊息
      - 傳回一筆紀錄
      - Edit() 傳回 Edit view page , 並將一筆剛才紀錄帶入 view page
        - HiddenFor 用來生成 hidden tag, 包含了 id 值
      - Edit view page 生成html
      - 傳回Browser, 此時 Browser 呈現一個填好紀錄欄位值的表單, 供使用者更新
    - 第二次運行
      - 使用者更新表單, 並按下更新按鈕, 送出表單
      - Browser 以 post 方式送出表單到 Register/Edit/99
      - 叫用 RegistersController.Edit(), HttpPost 的版本
      - 更新一筆紀錄到 database
      - 沒有傳回值
      - Edit 轉向到 Index
      - 產生 Index 的資料庫查詢, 請參考上一個段的時間流程, 傳回 Index view page 的結果
      - 將 index 生成即時 html 傳回Browser, 此時 Browser 呈現已轉名者紀錄
  - Delete 功能的運作流程
    - 第一次運行
      -

      - Browser 輸入 Registers/Delete/99 網址(其中99表示 id值)
      - 叫用 RegistersController.Delete(), HttpGet 的版本
      - 去資料庫取得要刪除的紀錄
        - 如果找不到, 傳回 HttpStatusCodeResult(HttpStatusCode.BadRequest)
      - 傳回一筆紀錄
      - Delete() 傳回 Delete view page , 並將一筆剛才紀錄帶入 view page
      - Delete view page 生成html
      - 傳回Browser, 此時 Browser 呈現一筆紀錄的值, 供使用者確認是否要刪除
    - 第二次運行
      - Browser 以 post 方式送出表單到 Register/Delete/99
      - 叫用 RegistersController.DeleteConfirmed(), HttpPost 的版本(因為二支參數相同, 無法都取名為 Delete)
      - 從 database 刪除一筆紀錄
      - 沒有傳回值
      - DeleteConfirmed 轉向到 Index
      - 產生 Index 的資料庫查詢, 請參考上一個段的時間流程, 傳回 Index view page 的結果
      - 將 index 生成即時 html 傳回Browser, 此時 Browser 呈現已轉名者紀錄
  - Details 功能的運作流程
    - 由於與 Edit / Delete 的第一個段十分相似, 在此就不再介紹
  - 修改 navbar
    - 修改 layout page, 加入註冊的連結
  - 練習
    - 在資料庫 dbEvent 裡新增一個資料表 "GuestBooks"
欄位名稱	型別	允值	說明
Id	int	N	Identity,primary
Name	nvarchar(50)	Y	姓名
Email	nvarchar(100)	Y	電子郵件
CellPhone	varchar(10)	Y	行動電話
Message	nvarchar(500)	Y	留言內容
CreateDate	DateTime	Y	開放時 getdate() 自動填寫
    - 公司有一個網站, 想加裝一個訪客留言板功能
    - 只需要有新增功能即可
    - 計劃在留言板增加人臉辨識, Email 多欄, 訊息通知功能 (需要公司內部伺服器自有權限)
    - 按下"送出"按鈕, 跳轉到頁 confirm 頁, 顯示"您的留言已內容已送出, 非常感謝您!!!謝謝"
      - Controller : RegistersController
      - View : /Views/Registers/ *.cshtml
        - Index.cshtml
        - Create.cshtml
        - Edit.cshtml
        - Delete.cshtml
        - Details.cshtml
      - Model 角色
        - EFModels 資料夾下的 DbContext , Entity class 就是 Model
- 在 MVC 裡頭裡, Controller, View 的角色定義還是模糊的
  - 修改研討會報名程式 - 加入商業邏輯
  - 無條件再與會的功能及檔案
  - 建立 Service Object
  - 建立 Repository Object
- ASP.NET MVC 開發流程小結
- View的角色, 建立 ViewModel
- 多個 ViewModel 的使用時機
- MVC 的關連主題

ASP.NET Core MVC