1. DTO 跟第三方API接觸 對接時需要注意的內容及特性
2. Debug模式下, debug = “true” ; Release模式下, debug = “false” (?

組態檔運行方式, 編譯出的各種東西的用處

1. View是動態頁面, 如何分成動態頁面/靜態頁面

DTO需要注意甚麼

DTO是一個定義如何在應用程序之間發送數據的對象

內容就是model

需要注意：

1. 只傳遞需要的資料, 不能暴露實體
2. 應該只包含需要的欄位減少資料傳輸的數量
3. DTO 通常需要進行序列化（將物件轉換為 JSON 或其他格式）和反序列化（將 JSON 轉換回物件）。確保你的 DTO 類別是序列化友好的，通常需要使用一些特性來控制序列化過程，尤其是當使用 JSON 或 XML 作為資料格式時。
4. 有需要可以用註解驗證資料
5. 如果未來API升級的話, 可以在DTO進行版本控制, 就能保證向後兼容

當想前後端分離的時候使用，可以有效解耦!

Debug vs Release

**組態檔 (Configuration Files)** 是用來配置應用程式行為的文件。最常見的配置文件是 appsettings.json 和 web.config。

Debug模式會生成.pdb 文件（這些是調試符號，允許你在 Debug 模式下進行調試）。Releas則是會省略.pdb文件或者選擇性生成

**1.1 Debug 模式**

* **用途**：這個模式主要用於開發過程中的調試。編譯器會生成未經優化的程式碼，並包含額外的除錯資訊。這使得開發者能夠進行步驟執行、變數監控等除錯操作。
* **性能**：由於包含除錯符號和未優化的程式碼，**Debug 模式** 的性能較差，運行速度比 **Release 模式** 慢。
* **組態檔**：通常，在 **Debug 模式** 下，會使用開發或本地環境的配置，這些配置可以包含資料庫連接字串、日誌等與開發環境相關的資訊。

**1.2 Release 模式**

* **用途**：此模式是用於生產環境中的編譯。編譯器會對程式碼進行優化，去除不必要的除錯資訊，從而提高應用程式的執行效率。
* **性能**：**Release 模式** 編譯的程式碼經過優化，性能更佳，適合部署到生產環境。
* **組態檔**：在 **Release 模式** 下，您可能會使用與生產環境相關的配置，例如正式的資料庫連接字串、真實的 API 金鑰等。

**1.3 使用方式**

 **在 Visual Studio 中創建 Web.config 轉換檔案**：

* 在解決方案中，右鍵單擊 web.config 文件，然後選擇 **"Add Config Transforms"**，這樣 Visual Studio 會為您創建 web.debug.config 和 web.release.config 文件。

 **編輯 web.debug.config 和 web.release.config**：

* 在 **web.debug.config** 中，您可以為 **Debug 模式** 設定專用的配置，通常會包含開發環境的資料庫連接字串和日誌設定。
* 在 **web.release.config** 中，您可以為 **Release 模式** 設定生產環境的配置，這會指向正式的資料庫連接字串和其他生產環境相關設定。

編譯檔案的用處

### . obj ****資料夾****

obj 資料夾是 **中間檔案**（Intermediate Output）的儲存位置，包含了編譯過程中間生成的檔案，這些檔案不是最終的應用程式執行檔，而是編譯過程的中間產物。具體來說，obj 目錄中會包含以下幾種檔案：

#### 1.1 ****目的：****

* **編譯過程中的中間結果**：當編譯專案時，obj 資料夾會儲存中間檔案，例如編譯器生成的 .obj 檔案（包含原始代碼的機器碼）和 **編譯配置**（例如 Debug 或 Release）。
* **生成臨時檔案**：這些檔案用來支持最終的執行檔案生成，並且在編譯的過程中會被使用。

#### 1.2 ****內容：****

#### **.dll** 或 **.exe**：編譯的中間檔案，這些檔案通常是臨時產物，用來生成最終的輸出檔案。

* **obj\Debug 或 obj\Release 資料夾**
* **.pdb (符號檔)**: 這些是程序調試檔案，包含了源代碼映射，方便調試時追蹤錯誤。
* **\*.dll**：這些是編譯過的中間語言（IL）檔案，是 C# 代碼編譯而來的
* **\*.csproj.FileListAbsolute.txt**：這些檔案記錄了項目中的所有檔案清單，幫助編譯器識別哪些檔案需要編譯。

### 2. bin ****資料夾****

bin 資料夾是 **最終輸出檔案**的儲存位置，包含了應用程式執行所需的實際檔案。這些檔案通常是在專案編譯後可以直接使用和部署的檔案。

#### 2.1 ****目的：****

* **最終的可執行檔案（.exe）或庫檔案（.dll）**：這些是編譯完成後的檔案，包含應用程式的執行代碼。
* **依賴的第三方程式庫（Assemblies）**：專案中所引用的外部程式庫（如 .NET 程式庫或 NuGet 套件）經過編譯後生成的 DLL 檔案。

#### 2.2 ****內容：****

在 bin 資料夾中，您會看到以下類型的檔案：

* **\*.dll**：這些是編譯過的程序集檔案，包含了應用程式的執行邏輯。例如， MVC 控制器、視圖模型、業務邏輯等都會編譯成 DLL 檔案並存放在 bin 資料夾中。
* **\*.pdb**：符號檔案，通常與 DLL 檔案一起存在，用於調試。
* **第三方程式庫（Assemblies）**：引用的外部庫或組件，例如來自 NuGet 的 DLL 檔案，這些檔案會被放在 bin 資料夾中。

動態頁面/靜態頁面

靜態頁面

**定義：** 靜態頁面是預先創建並存儲在伺服器上的 HTML 文件。每當用戶請求該頁面時，伺服器直接返回該頁面的內容，不會進行任何額外的處理或生成。

**特點：**

* **內容固定**
* **簡單性**
* **快速加載**

動態頁面

**定義：** 動態頁面是通過伺服器端的程式碼（如 PHP、ASP.NET、Java、Python 等）根據用戶的請求和後端資料庫的內容動態生成的頁面。

**特點：**

* **內容動態生成**：根據用戶的請求，伺服器會運行程式碼，並生成適合該用戶的頁面。
* **數據庫支持**
* **可擴展性強**：由於內容是動態生成的，可以輕鬆地對網站進行擴展或更新，而不需要手動修改每個頁面。
* **交互性**：動態頁面可以根據用戶操作進行交互式更新

分別：主要看每次請求是否都返回相同內容