

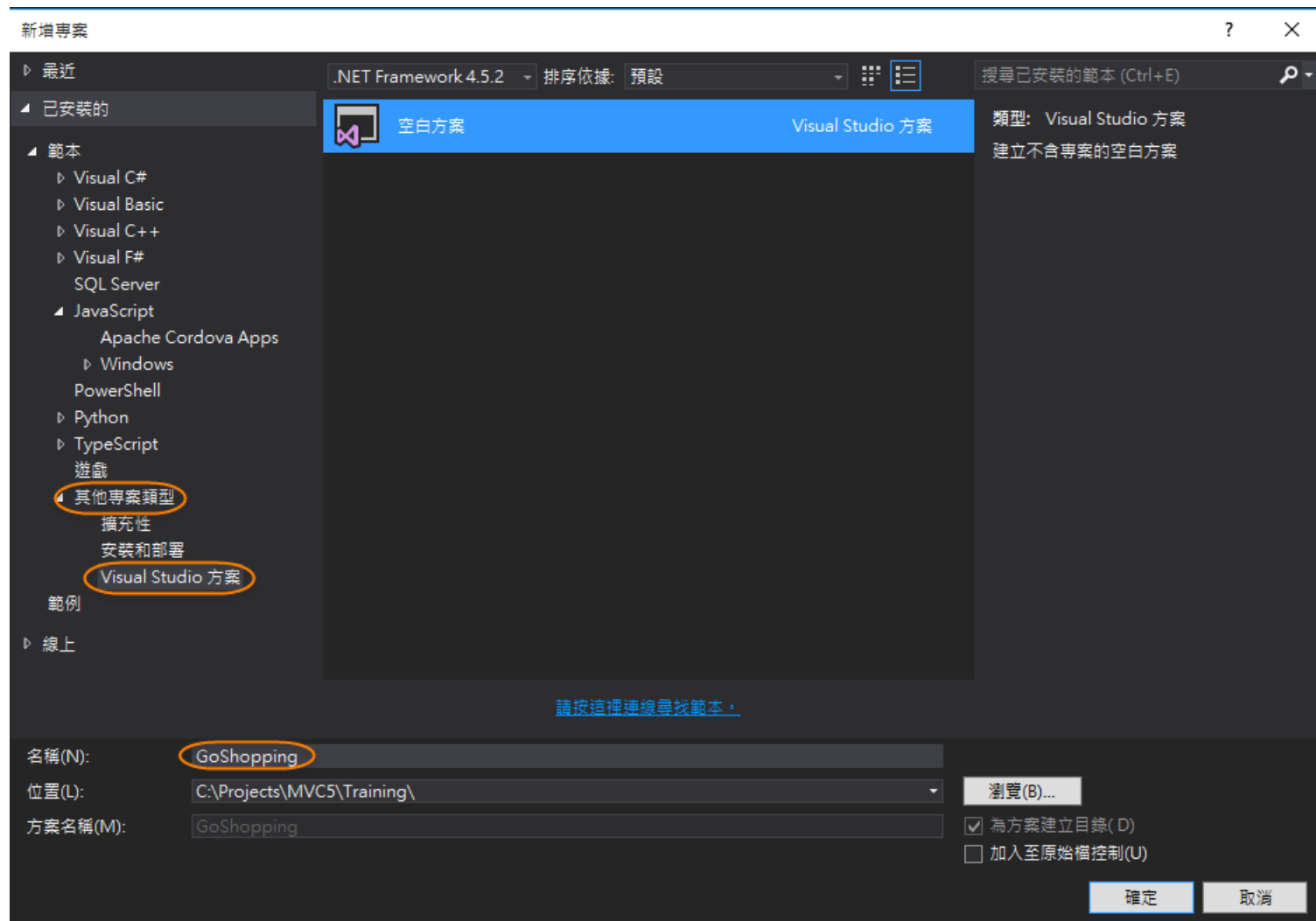
MVC 5 網站實戰篇

Jacky

Web 應用程式專案的架構

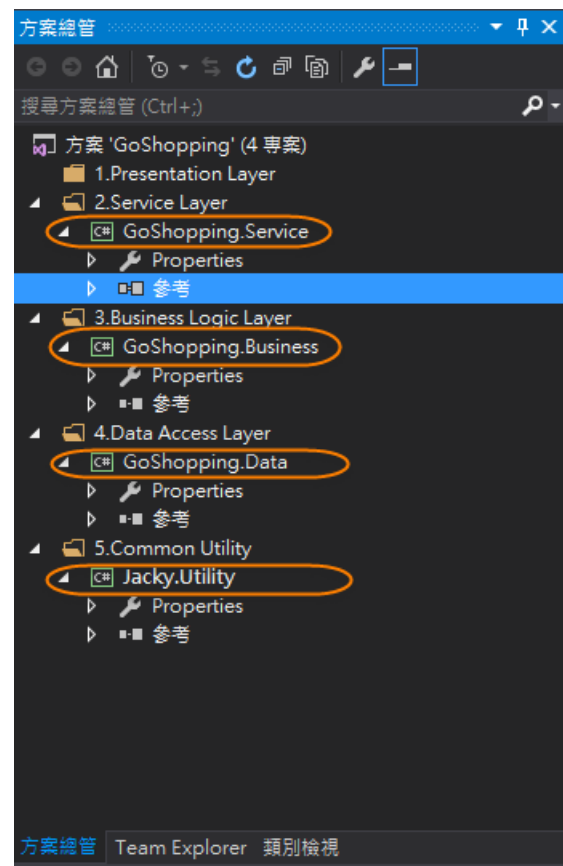
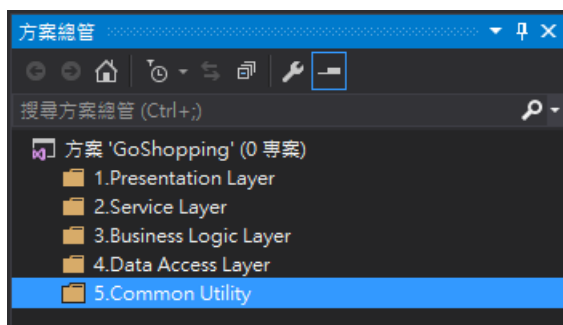


• 方案與方案資料夾

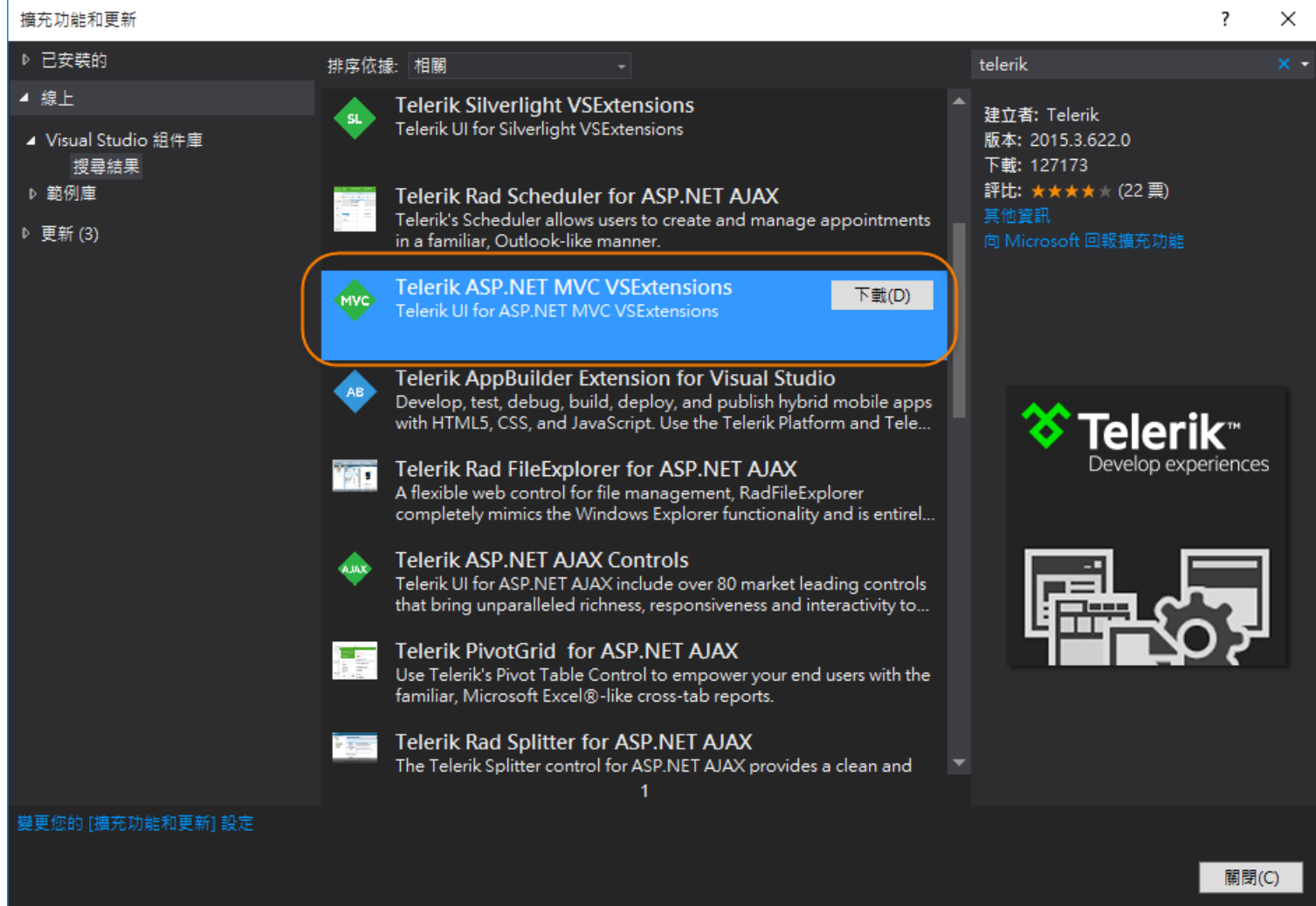
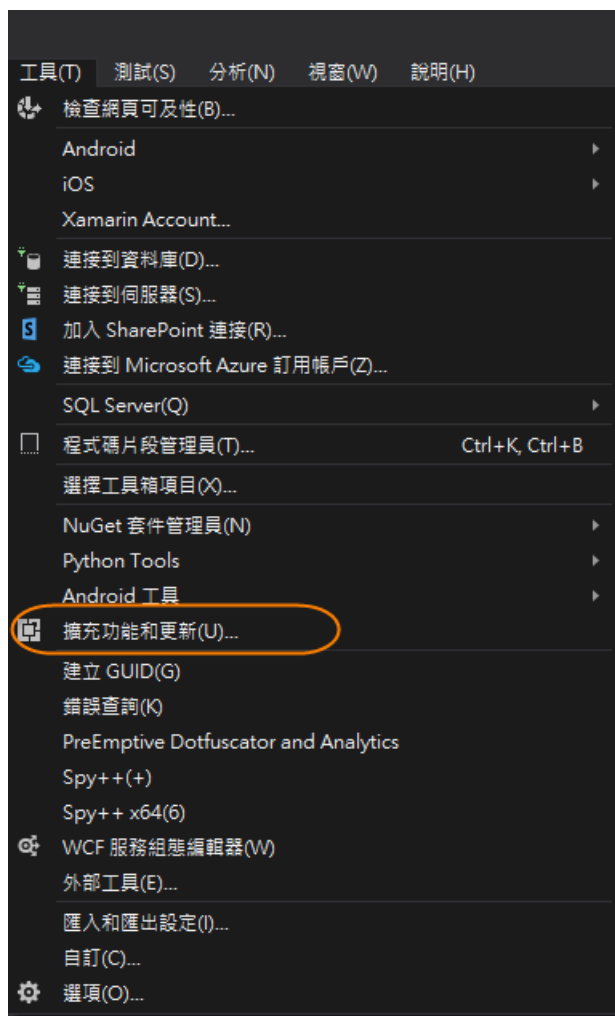


新增類別庫專案

- 建立 5 個方案資料夾
- 類別庫專案:除了展示層以外的4個方案資料夾新增一個類別庫專案

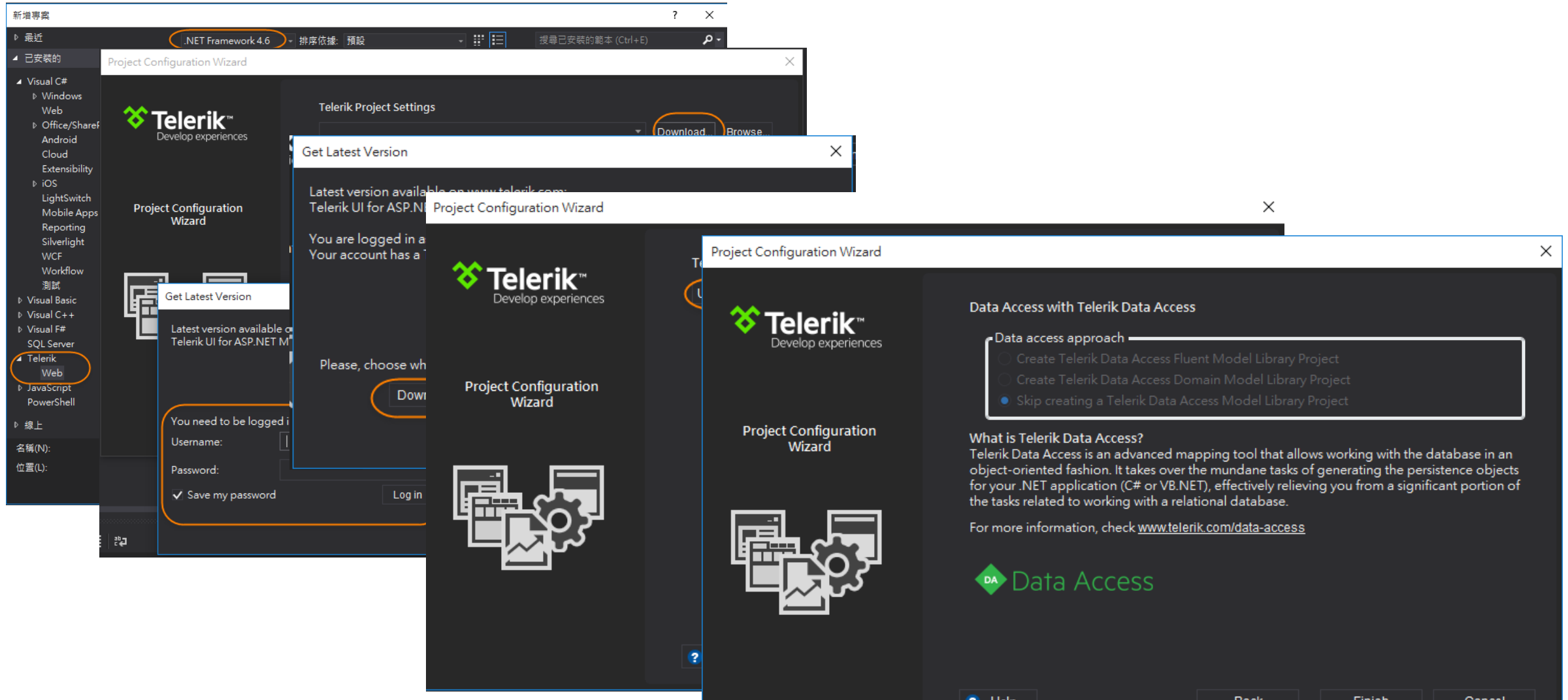


加入:Telerik UI for ASP.NET MVC



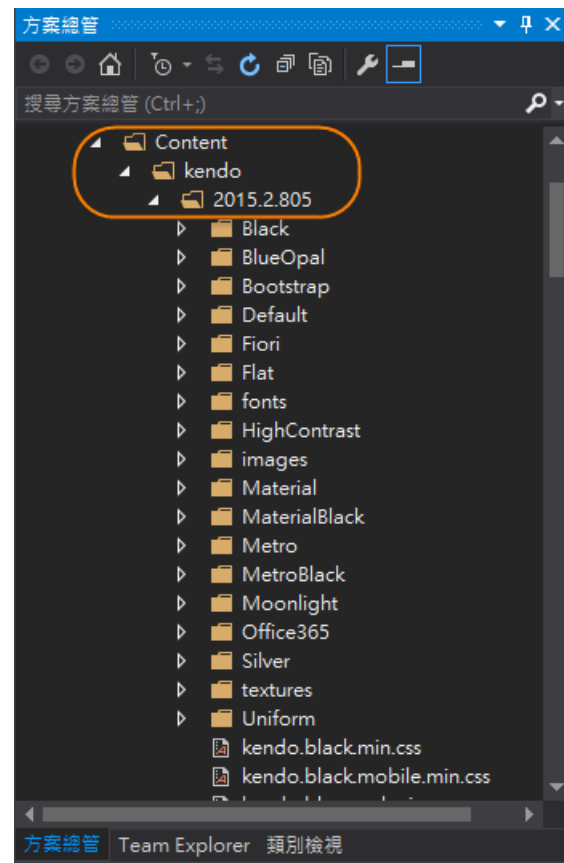
新增:Telerik UI for ASP.NET MVC 專案

- 需登入再下載專案範本才能下一步



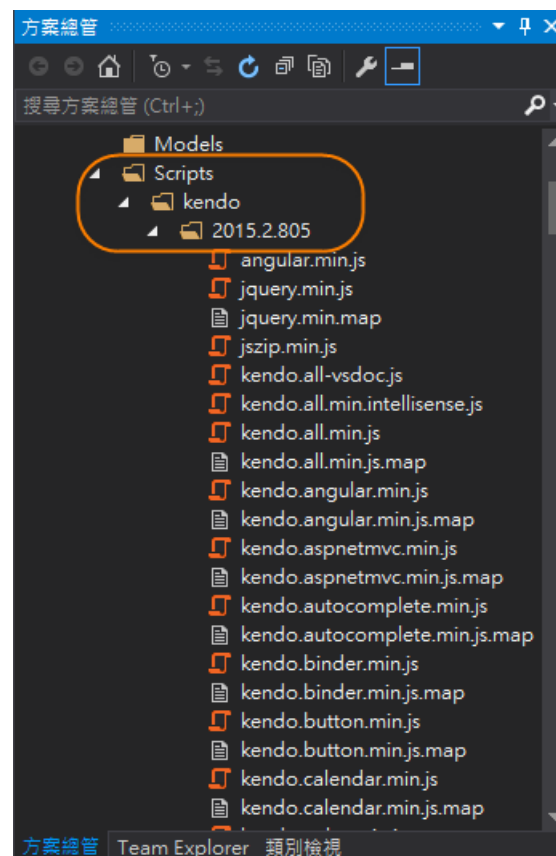
Kendo UI 相關 : CSS

- Content 資料夾: UI 相關的 CSS
 - CSS 檔案中 `kendo.common.css` 是必須使用到的
 - 其他的 `kendo.xxxx.css` (xxxx為某一英文單字，代表某一佈景主題名稱) 可依將來所設計的網頁格調來選擇合適色系的佈景主題。有些佈景主題有其配合的圖片，該些圖片會被放置於與其佈景名稱相同的資料夾底下，例如：
`kendo.bootstrap.css` 必須與 `Bootstrap` 整個資料夾 (包含底下的圖片) 一起被使用。



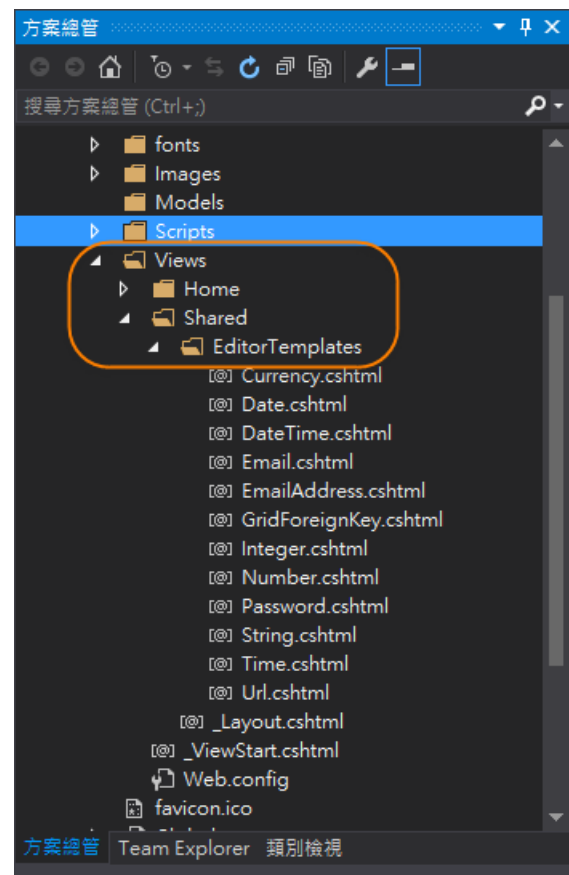
Kendo UI 相關 : Scripts

- Kendo UI 包含了 Web、Mobile，和 DataViz 三個模組，其 JavaScript 檔案分別為：
kendo.web.js、
kendo.mobile.js 和
kendo.dataviz.js 三個檔案，直接選用
kendo.all.js（包含了三個模組）。
- kendo.aspnetmvc.js 為必須
- js\cultures 資料夾底下包含多國語言所需的 JavaScript 檔



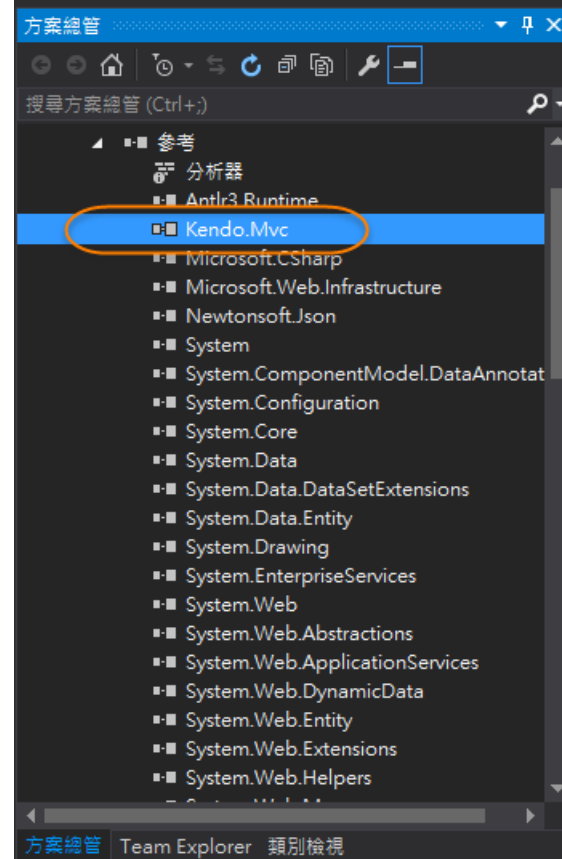
Kendo UI 相關 : Views

- Views\Shared 這個子資料夾都是通用（供所有檢視共用）的檢視元素，在這個資料夾底下
- EditorTemplates 的資料夾，在這個資料夾底下可以放置編輯器（Editor）樣版



Kendo UI 相關dll :

- Kendo.Mvc.dll 加入 Web 專案的參考
- 為了能夠在檢視中直接使用 Kendo UI for ASP.NET MVC Wrappers 的功能，Views 資料夾底下的 Web.config 中加入對 Kendo.Mvc.UI 命名空間的參考



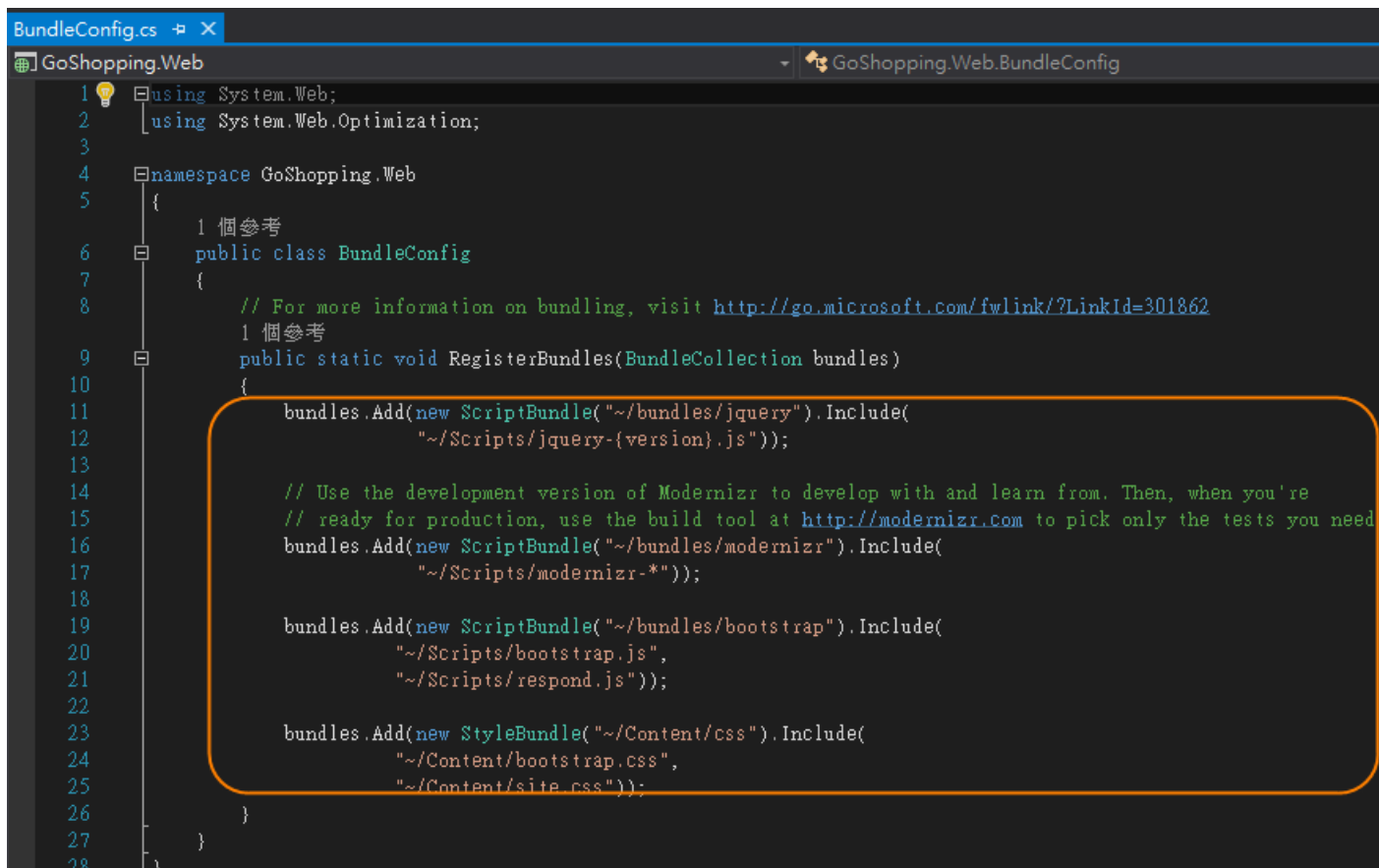
```
Web.config  X
10
11 <system.web.webPages.razor>
12   <host factoryType="System.Web.Mvc.MvcWebRazorHostFactory, System.Web.Mvc, Version=5.0
13   <pages pageBaseType="System.Web.Mvc.WebViewPage">
14     <namespaces>
15       <add namespace="System.Web.Mvc" />
16       <add namespace="System.Web.Mvc.Ajax" />
17       <add namespace="System.Web.Mvc.Html" />
18       <add namespace="System.Web.Optimization" />
19       <add namespace="System.Web.Routing" />
20       <add namespace="GoShopping.Web" />
21       <add namespace="Kendo.Mvc.UI" />
22     </namespaces>
23   </pages>
24 </system.web.webPages.razor>
25
```

Kendo UI 相關

- JavaScript、CSS 和相關圖型檔案加入到 Web 專案中，也說明了檔案名稱中有 .min 和沒有 .min 之間的差別。一般在商業應用程式的開發及除錯過程中，會使用沒有經過壓縮，名稱中沒有 .min 的檔案，以方便除錯。而實際發行到正式營運的伺服器時會使用已經壓縮過，名稱中含有 .min 的檔案。
- 除了在將壓縮過的檔案放置於正式營運的伺服器上，讓使用者端的瀏覽器下載之外，有時也會透過 CDN（Content Delivery Network）內容傳遞網路，讓使用者端的瀏覽器下載。範例在開發的過程會使用未壓縮，正式營運會使用 CDN 的方式

Kendo UI 相關:打包BundleConfig.cs

- Web 專案的 App_Start 資料夾底下的 BundleConfig.cs
 - kendoCommonCssPath、kendoBootstrapCssPath、kendoAllJsPath，與 kendoMvcJsPath 四個常數定義出 kendo.common.min.css、kendo.bootstrap.min.css、kendo.all.min.js，與 kendo.aspnetmvc.min.js 在 CDN 的位置。這樣一來就可以在正式營運的伺服器上使用 CDN 檔案。



The screenshot shows the `BundleConfig.cs` file in a project named `GoShopping.Web`. The file is part of the `GoShopping.Web.BundleConfig` namespace. It defines a `BundleConfig` class and a `RegisterBundles` method. The `RegisterBundles` method registers several bundles, including jQuery, Modernizr, Bootstrap, and the application's CSS and JS files. The bundles are defined as follows:

```
using System.Web;
using System.Web.Optimization;

namespace GoShopping.Web
{
    public class BundleConfig
    {
        // For more information on bundling, visit http://go.microsoft.com/fwlink/?LinkId=301862
        public static void RegisterBundles(BundleCollection bundles)
        {
            bundles.Add(new ScriptBundle("~/bundles/jquery").Include(
                "~/Scripts/jquery-{version}.js"));

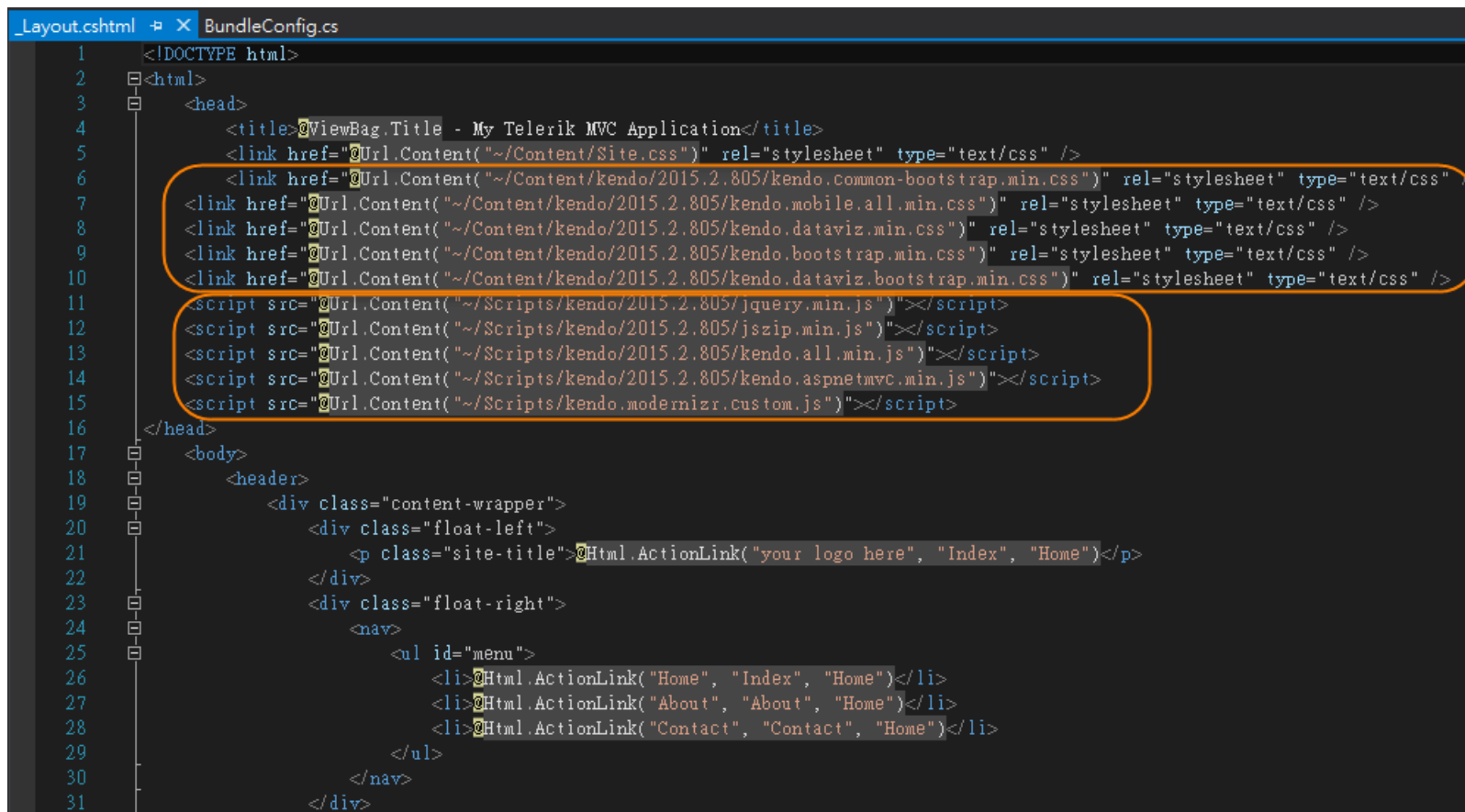
            // Use the development version of Modernizr to develop with and learn from. Then, when you're
            // ready for production, use the build tool at http://modernizr.com to pick only the tests you need
            bundles.Add(new ScriptBundle("~/bundles/modernizr").Include(
                "~/Scripts/modernizr-*"));

            bundles.Add(new ScriptBundle("~/bundles/bootstrap").Include(
                "~/Scripts/bootstrap.js",
                "~/Scripts/respond.js"));

            bundles.Add(new StyleBundle("~/Content/css").Include(
                "~/Content/bootstrap.css",
                "~/Content/site.css"));
        }
    }
}
```

Kendo UI 相關:如何在檢視 (View) 中使用

- Web 專案的 Views\Shared 資料夾底下的 _Layout.cshtml
- CSS 檔案是用來設定網頁內的 HTML 元件的樣式，所以必須在 HTML 元件被載入前，先被載入，所以一般會將其放置於 `<head> </head>` 區段之內。而 JavaScript 檔案一般會放置於所有 HTML 元件之後，所以會放置於 `</body>` 之前，但是 jQuery 較為特殊，因為 Kendo UI for ASP.NET MVC 會使用到 jQuery 的功能，所以必須先被載入，因此也是會放置於 `<head> </head>` 區段之內。



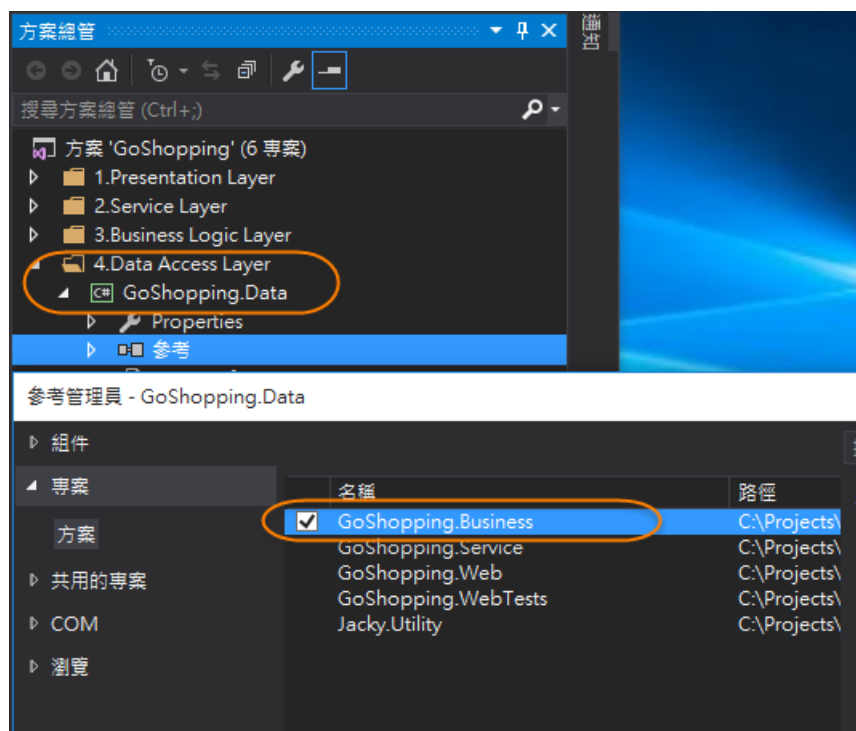
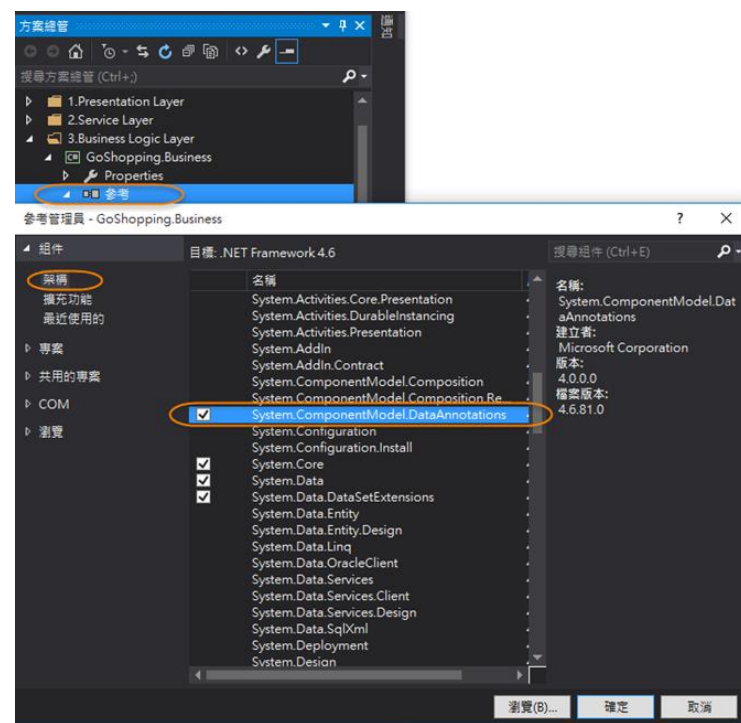
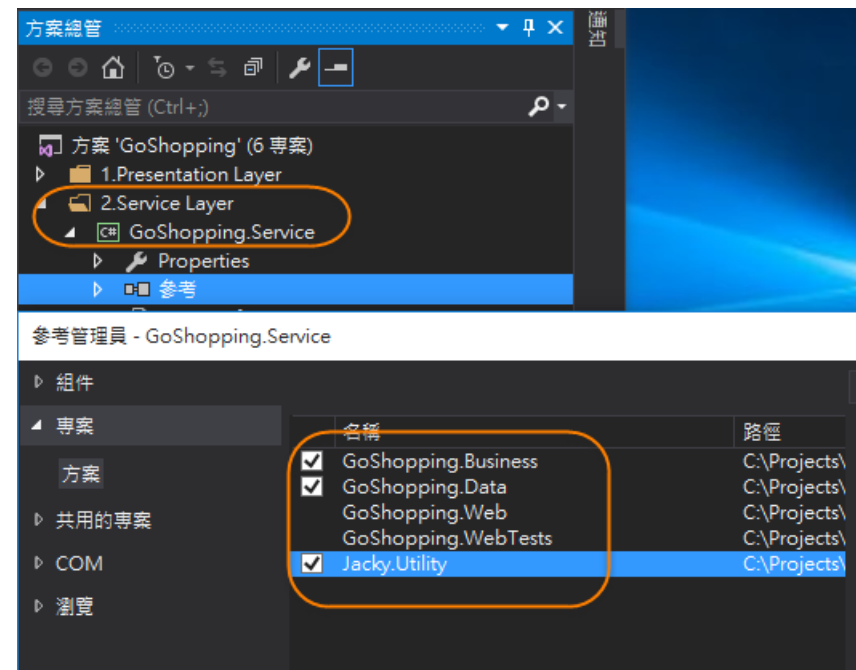
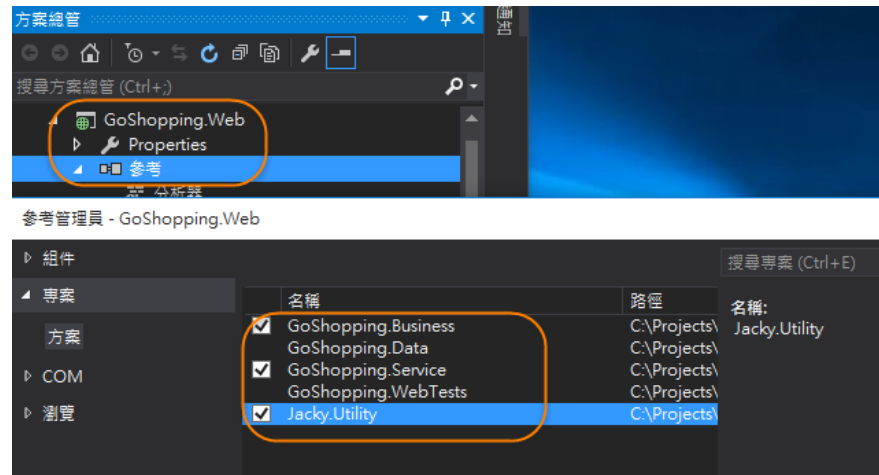
```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>@ViewBag.Title - My Telerik MVC Application</title>
5 <link href="@Url.Content("~/Content/Site.css")" rel="stylesheet" type="text/css" />
6 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.common-bootstrap.min.css")" rel="stylesheet" type="text/css" />
7 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.mobile.all.min.css")" rel="stylesheet" type="text/css" />
8 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.dataviz.min.css")" rel="stylesheet" type="text/css" />
9 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.bootstrap.min.css")" rel="stylesheet" type="text/css" />
10 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.dataviz.bootstrap.min.css")" rel="stylesheet" type="text/css" />
11 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/jquery.min.js")"></script>
12 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/jszip.min.js")"></script>
13 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/kendo.all.min.js")"></script>
14 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/kendo.aspnetmvc.min.js")"></script>
15 <script src="@Url.Content("~/Scripts/kendo.modernizr.custom.js")"></script>
16 </head>
17 <body>
18 <header>
19 <div class="content-wrapper">
20 <div class="float-left">
21 <p class="site-title">@Html.ActionLink("your logo here", "Index", "Home")</p>
22 </div>
23 <div class="float-right">
24 <nav>
25 <ul id="menu">
26 <li>@Html.ActionLink("Home", "Index", "Home")</li>
27 <li>@Html.ActionLink("About", "About", "Home")</li>
28 <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
29 </ul>
30 </nav>
31 </div>
```

加入 Entity Framework 和專案之間的參考



專案加入參考

1. 為Web專案加入專案參考
 2. 為Service專案加入專案參考
 3. 為Business專案加入參考
 4. 為Data專案加入專案參考
- 整個開發環境就
已經準備完成
(可簽入當一範本)

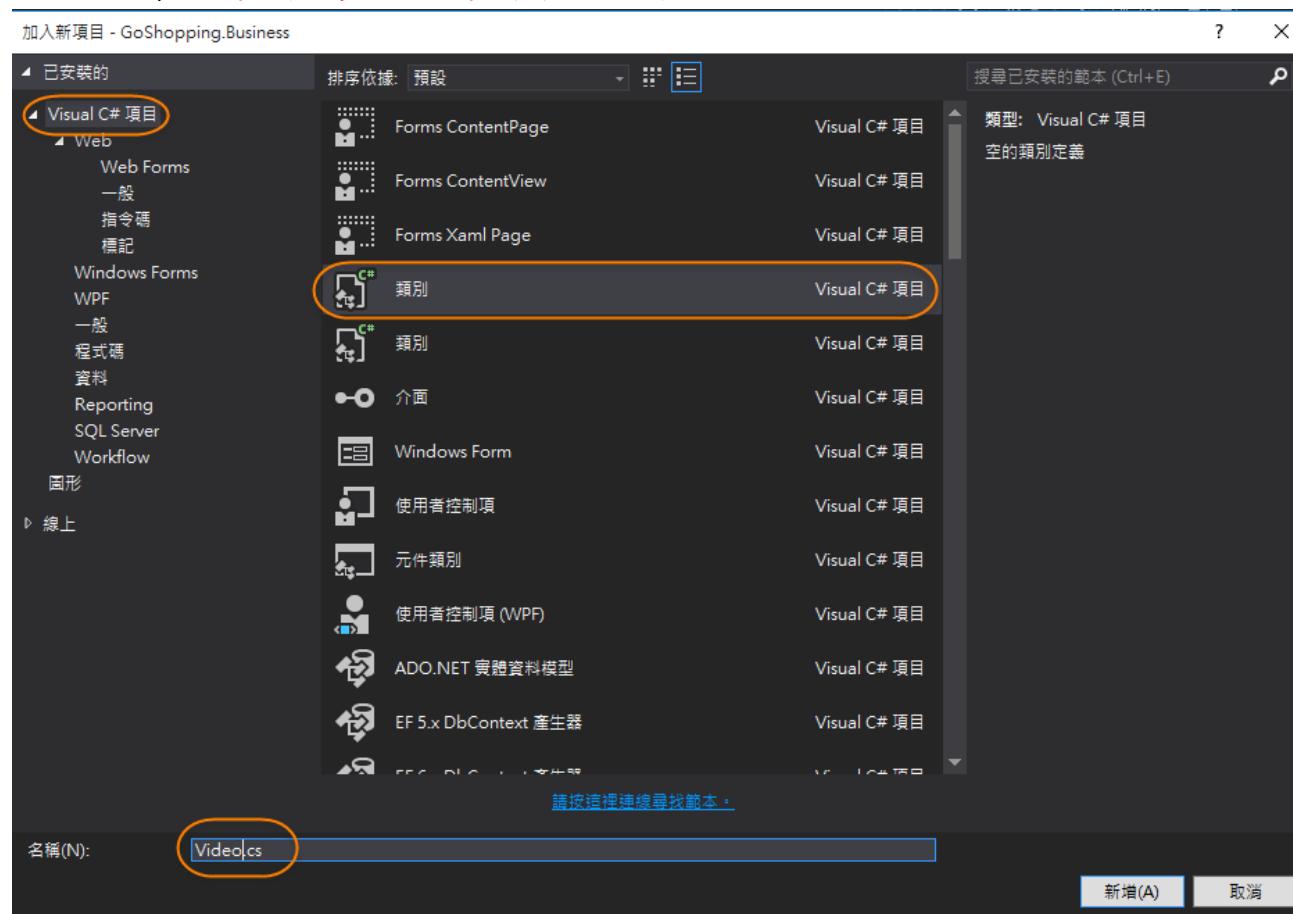
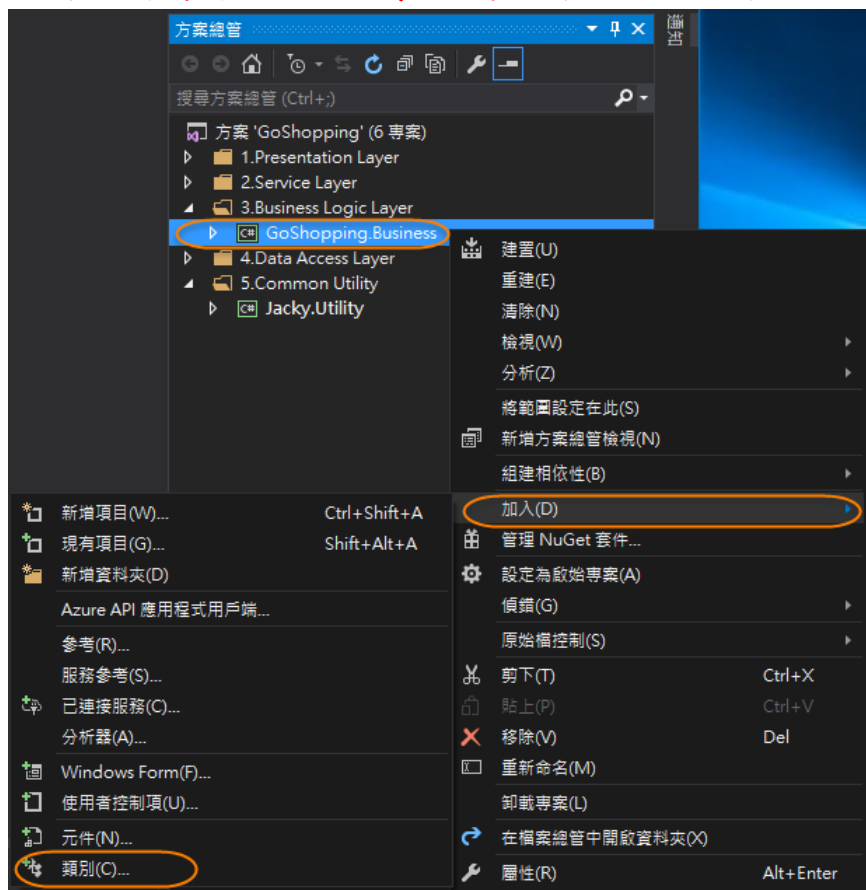


開始專案:需求分析

- 在購物網站的首頁，我們想放一個影片，因為影片的內容有季節性，或是因為某項促銷活動而希望能在某一段特定期間內撥放，且又因為伺服器空間和網路流量的問題，因而希望這些影片是透過 **YouTube** 來播放。
- 為了達成這項需求，首先想到的可能是先要有一個類似後台維護介面，讓購物網站的管理者可以新增、編輯、刪除來管理欲顯示在首頁的影片。

資料模型Model-1

- 一個類別（Class）並擁有特定的屬性（Property），而各個屬性擁有特定的限制規則（也可以說是商業邏輯）而已



資料模型Model-2

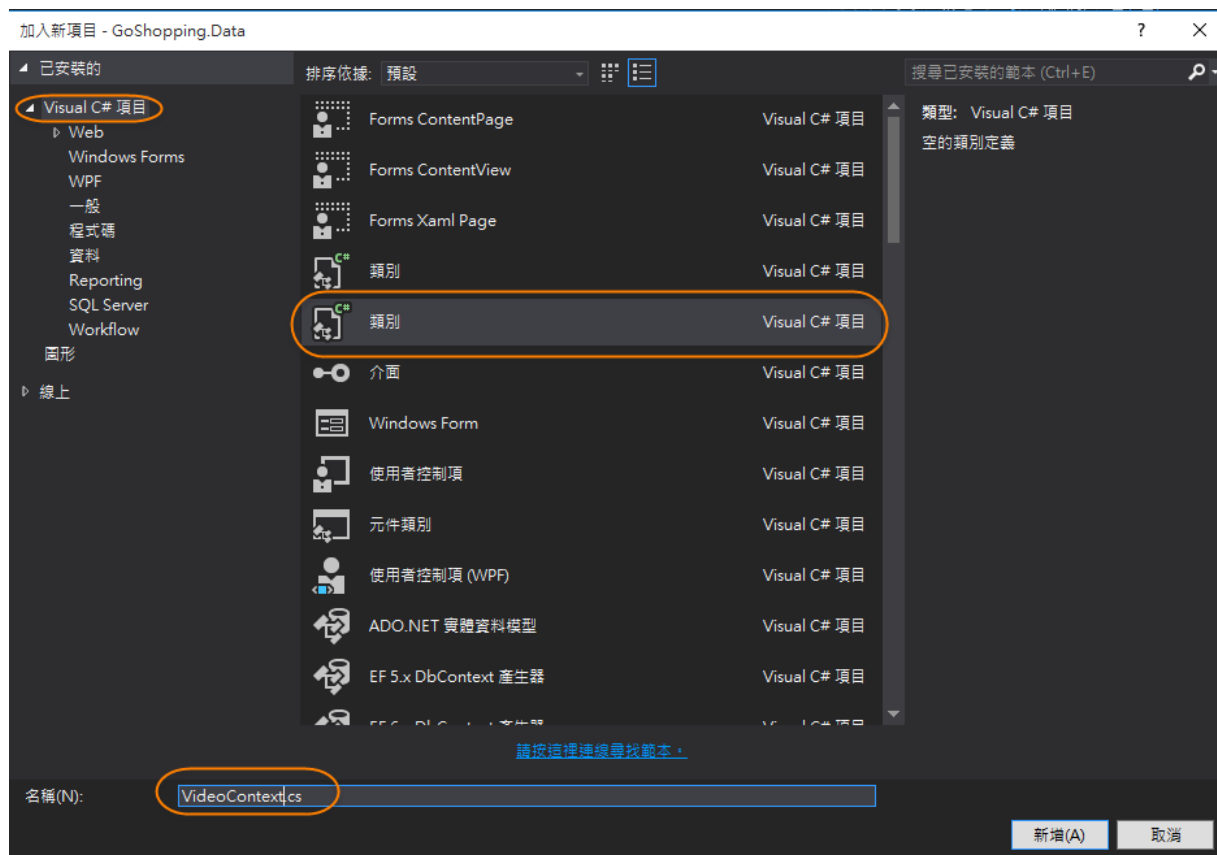
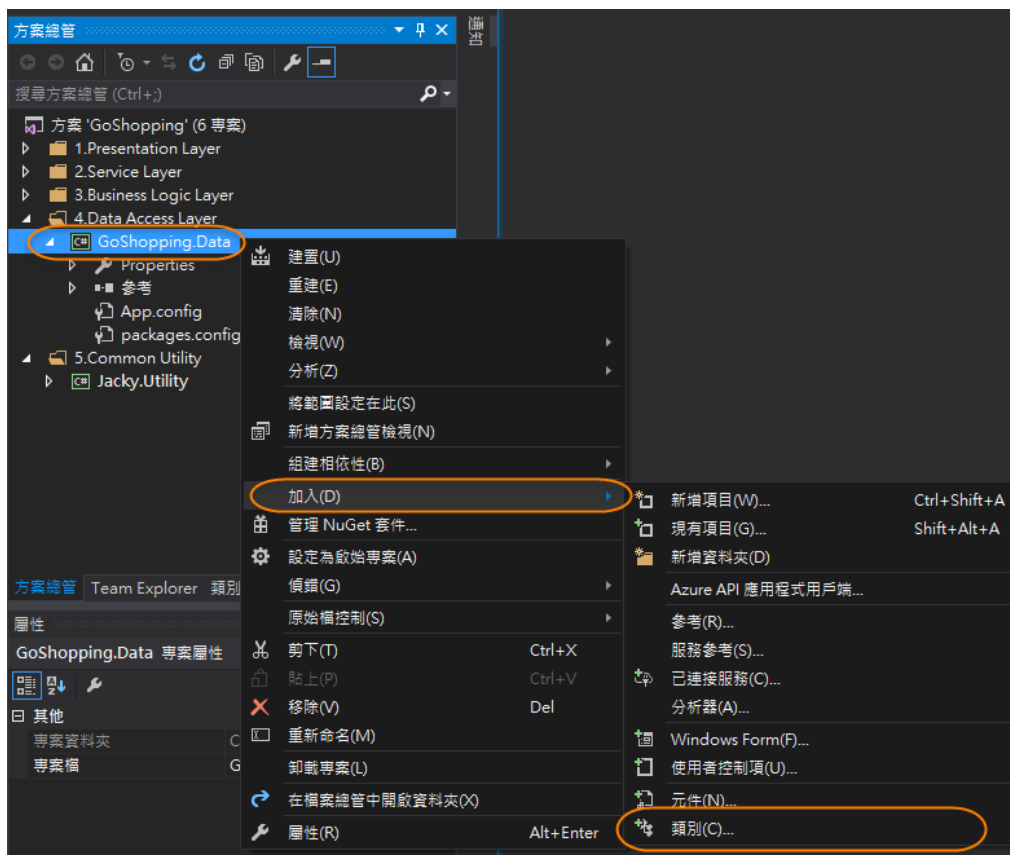
如下的屬性;如下的限制商業邏輯

```
Video.cs*  X
GoShopping.Business  GoShopping.Business.Video
5  using System.Threading.Tasks;
6
7  namespace GoShopping.Business
8  {
9      0 個參考
10     class Video
11     {
12         0 個參考
13         public string Id { get; set; }
14
15         0 個參考
16         public string Title { get; set; }
17
18         0 個參考
19         public DateTime StartDate { get; set; }
20
21         0 個參考
22         public DateTime EndDate { get; set; }
23     }
24 }
```

```
Video.cs  X  GoShoppingDbContext.cs
GoShopping.Business  GoShopping.Business.Video  Id
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6  using System.ComponentModel.DataAnnotations;
7
8  namespace GoShopping.Business
9  {
10     1 個參考
11     public class Video
12     {
13         [Display(Name = "影片代碼")]
14         [Required(ErrorMessage = "{0}不可空白")]
15         [StringLength(11, MinimumLength = 11, ErrorMessage = "{0}必須是{1}個字元")]
16         [UIHint("Video")]
17         public string Id { get; set; }
18
19         [Display(Name = "影片標題")]
20         [Required(ErrorMessage = "{0}不可空白")]
21         [MaxLength(20)]
22         public string Title { get; set; }
23
24         [UIHint("Date")]
25         [Display(Name = "開始日期")]
26         public DateTime StartDate { get; set; }
27
28         [UIHint("Date")]
29         [Display(Name = "結束日期")]
30         public DateTime EndDate { get; set; }
31     }
32 }
```

資料存取-1

- 自動幫我們建立資料庫和資料表
- Entity Framework Code First
- 資料存取層專案新增一個類別



資料存取-2

```
GoShoppingDbContext.cs
GoShopping.Data
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6 using GoShopping.Business;
7 using System.Data.Entity;
8
9 namespace GoShopping.Data
10 {
11     0 個參考
12     public class GoShoppingDbContext : DbContext
13     {
14         0 個參考
15         public DbSet<Video> Videos { get; set; }
16     }
17 }
```

物件總管

連接 ▾

KEIGEN\SQLEXPRESS (SQL Server 11.0.3128 - K)

- 資料庫
 - 系統資料庫
 - GoShopping.Data
 - GoShoppingDbContext
 - 資料庫圖表
 - 資料表
 - 系統資料表
 - FileTable
 - dbo._MigrationHistory
 - dbo.Videos

KEIGEN\SQLEXPRESS...ext - dbo.Videos

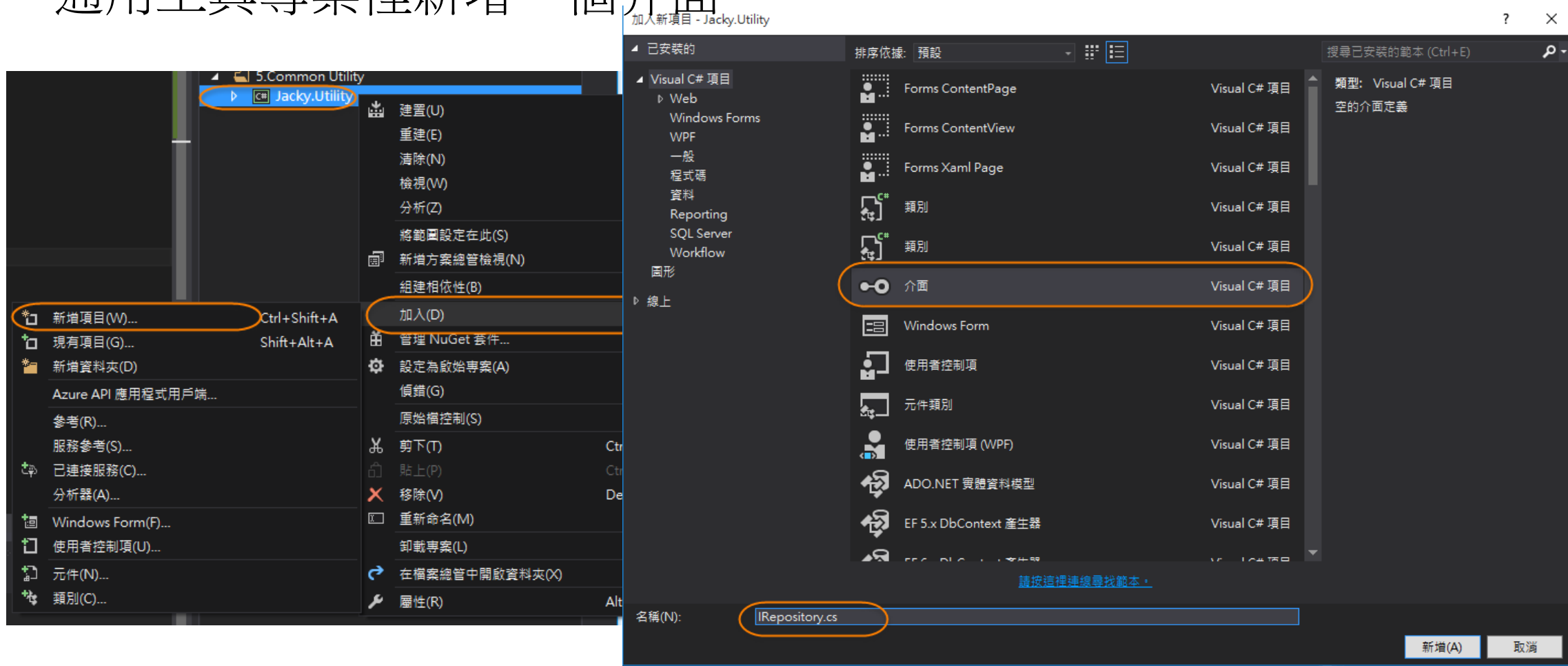
資料行名稱	資料類型	允許 Null
Id	nvarchar(11)	<input type="checkbox"/>
Title	nvarchar(20)	<input type="checkbox"/>
StartDate	datetime	<input type="checkbox"/>
EndDate	datetime	<input type="checkbox"/>

```
public class GoShoppingDbContext : DbContext
{
    public DbSet<Video> Videos { get; set; }
}

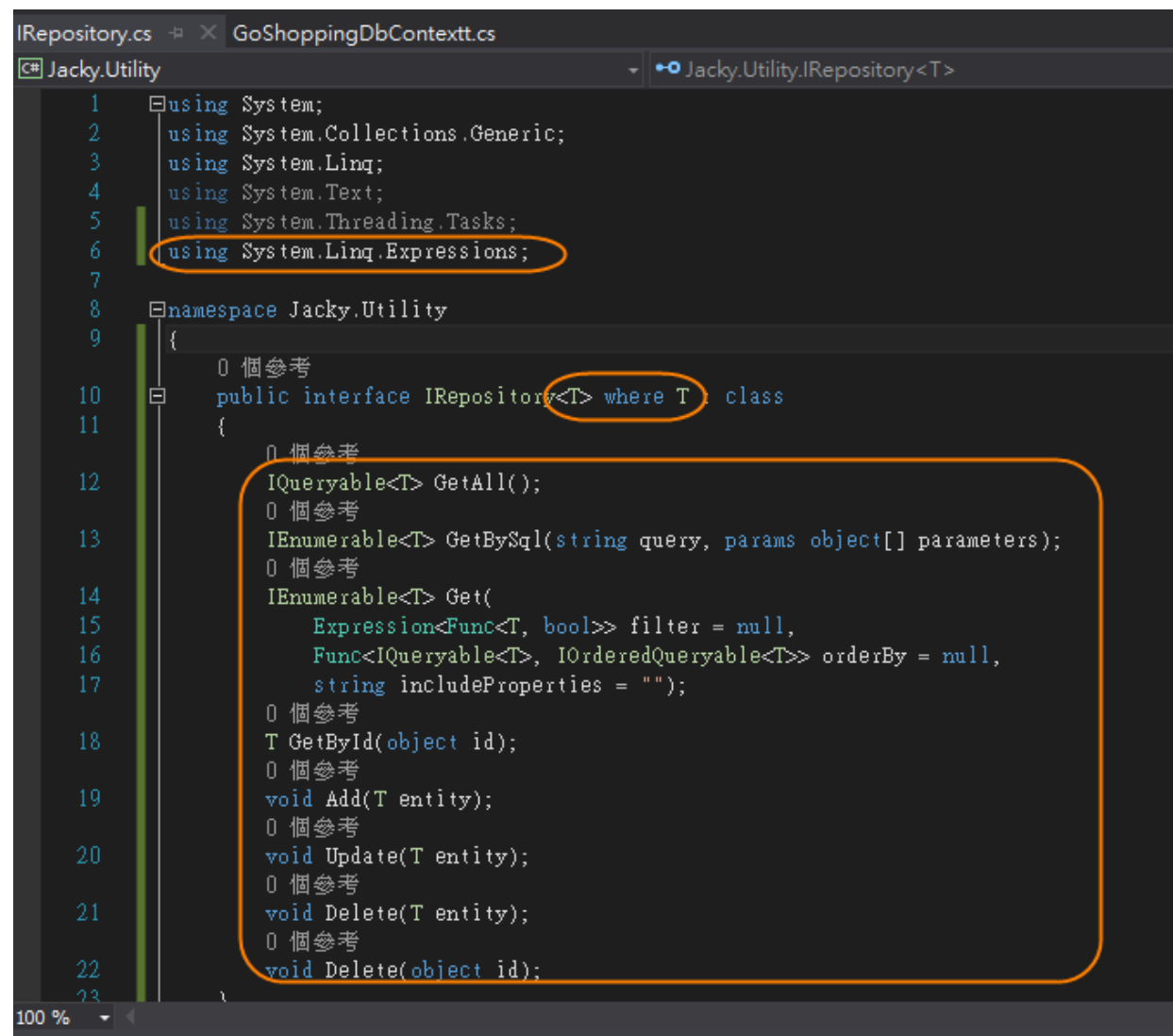
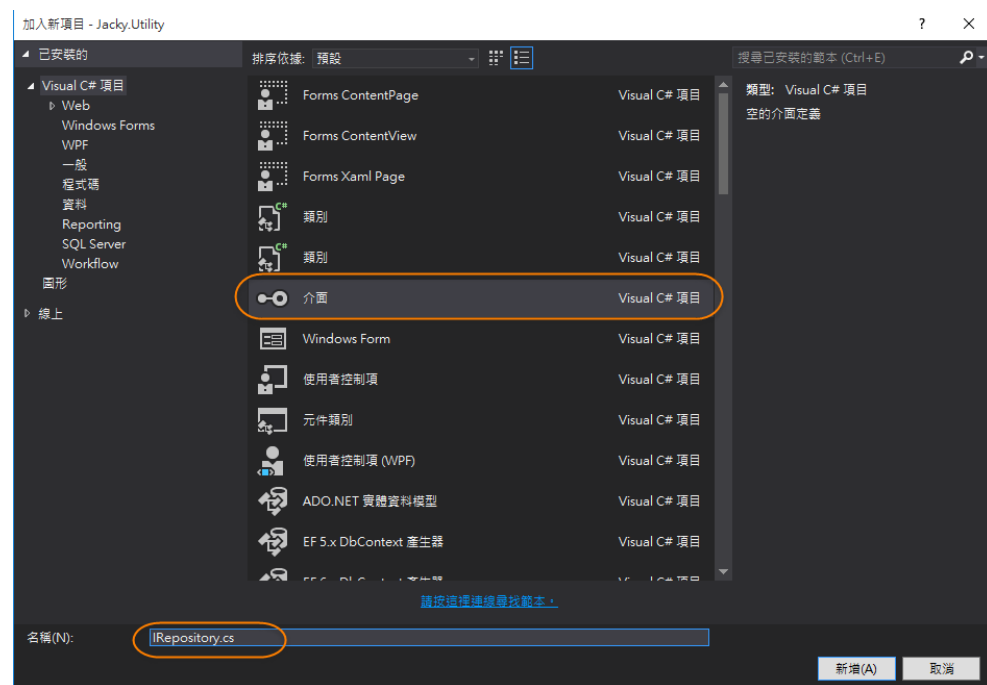
public class Video
{
    public string Id { get; set; }
    public string Title { get; set; }
    public DateTime StartDate { get; set; }
    public DateTime EndDate { get; set; }
}
```

Repository Pattern:新增泛用的介面程式-1

- 通用工具專案裡新增一個介面

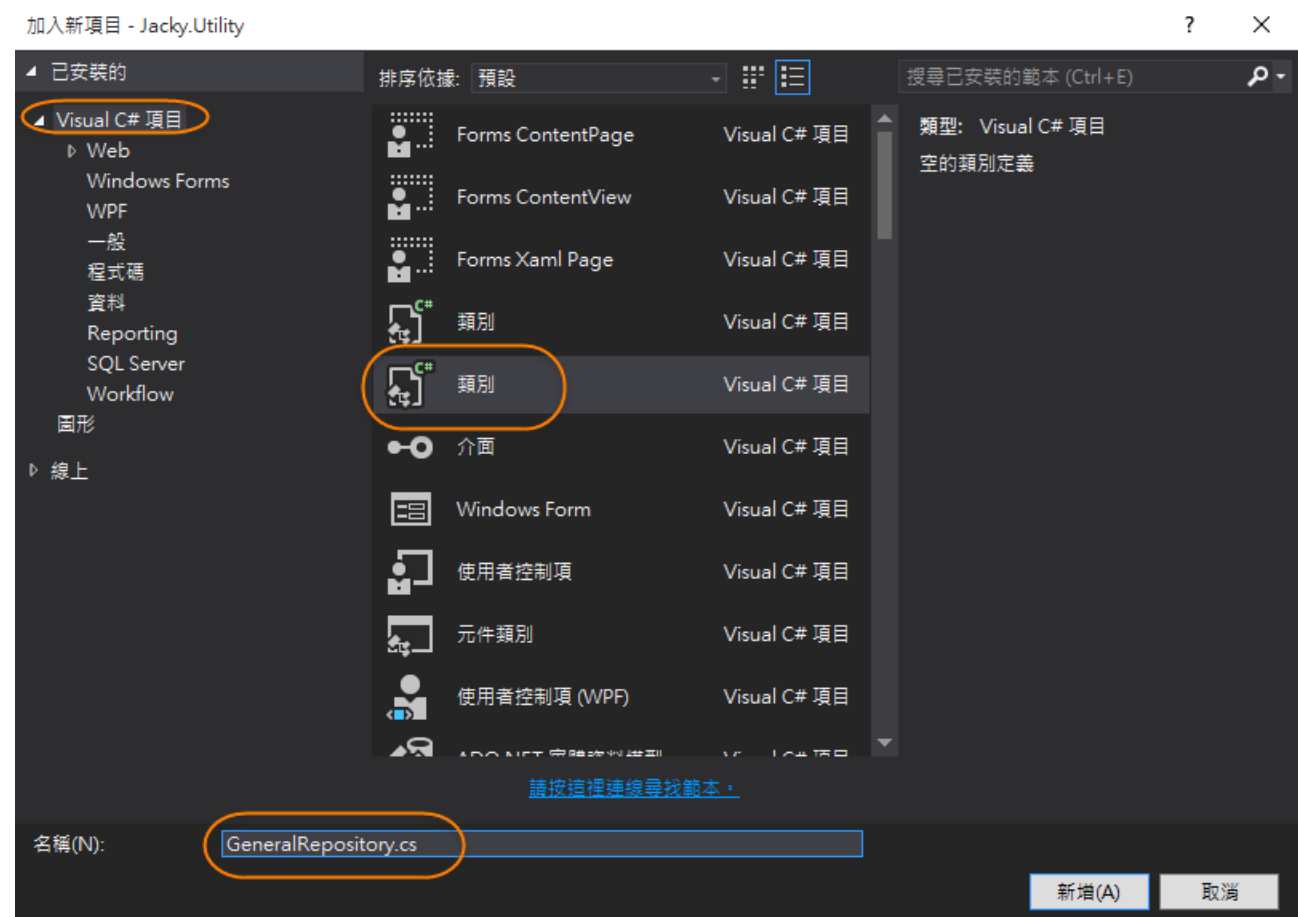
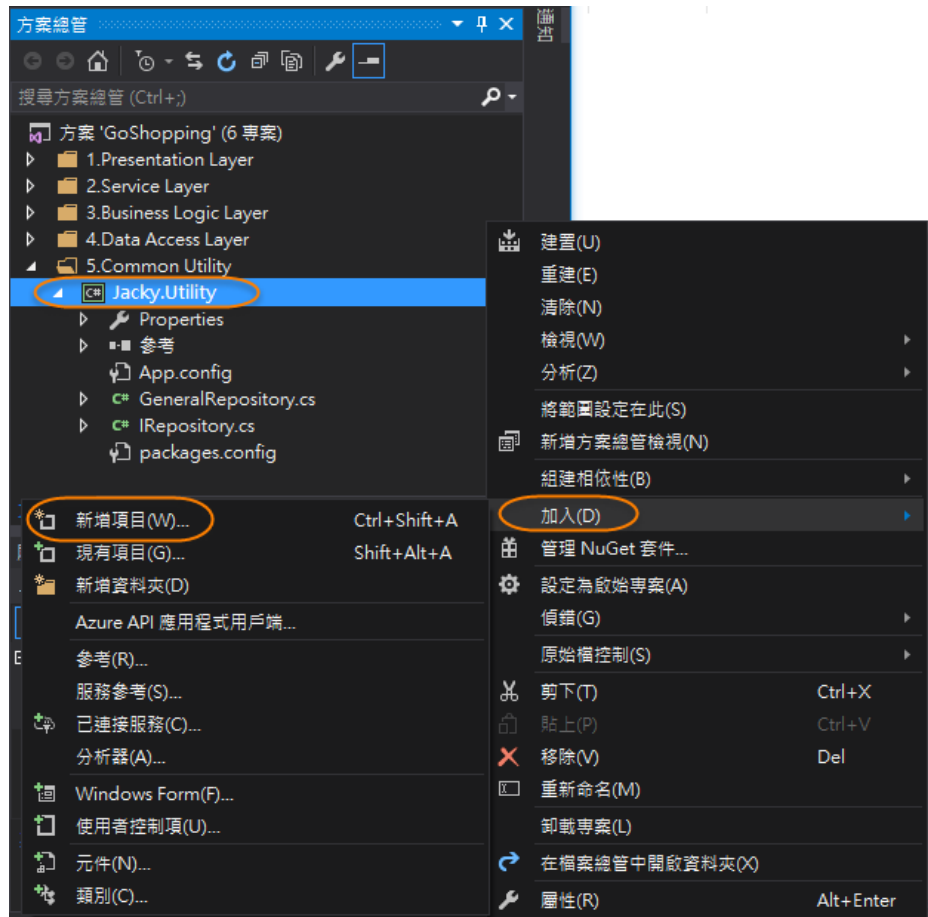


Repository Pattern: 定義介面-2



Repository Pattern:新增介面實作類別及方法-3

- 通用工具專案裡新增類別實作該介面所規範的方法



Repository Pattern:新增介面實作類別及方法-4

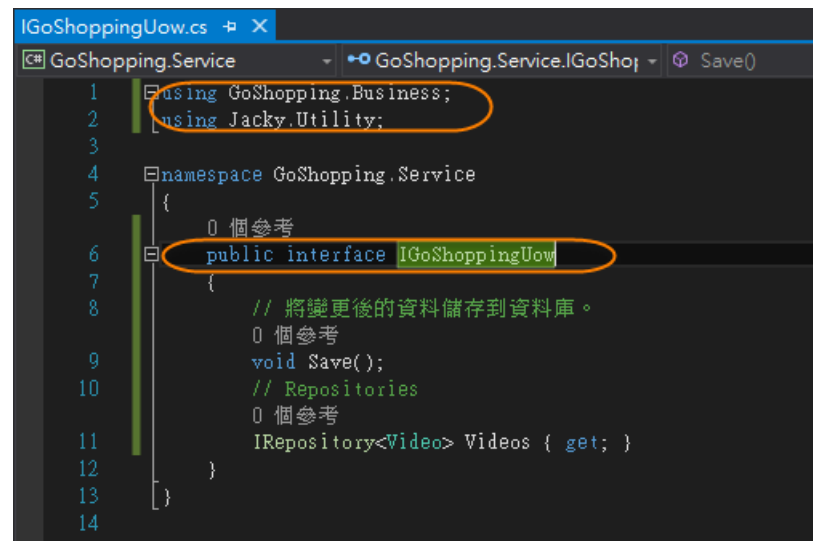
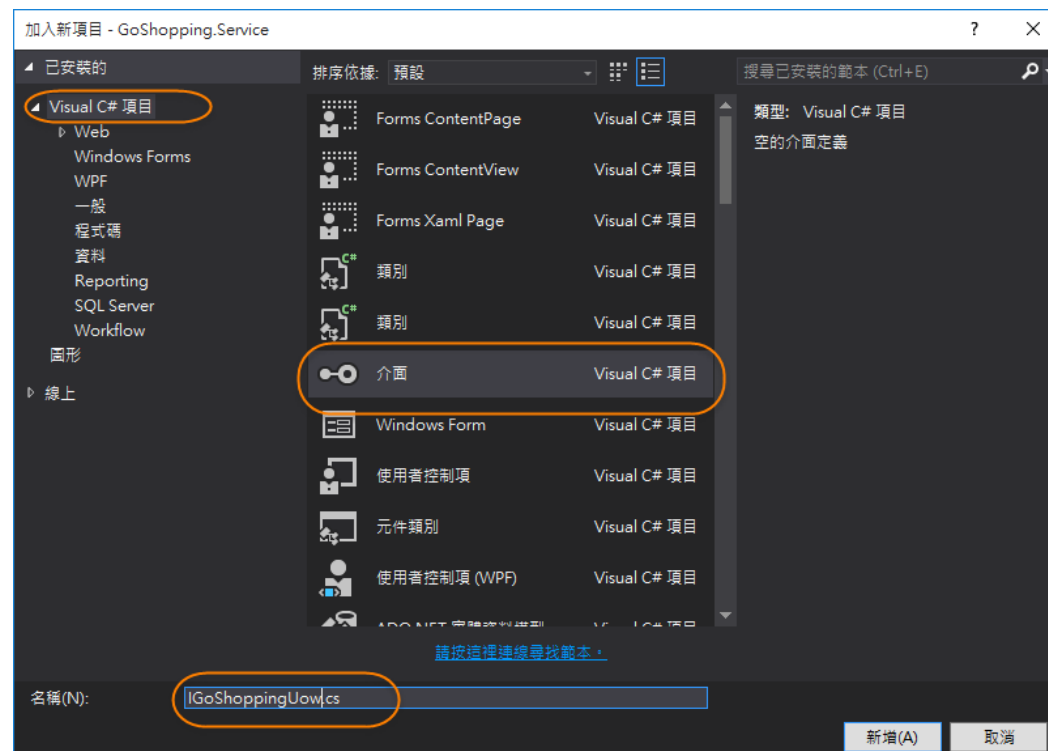
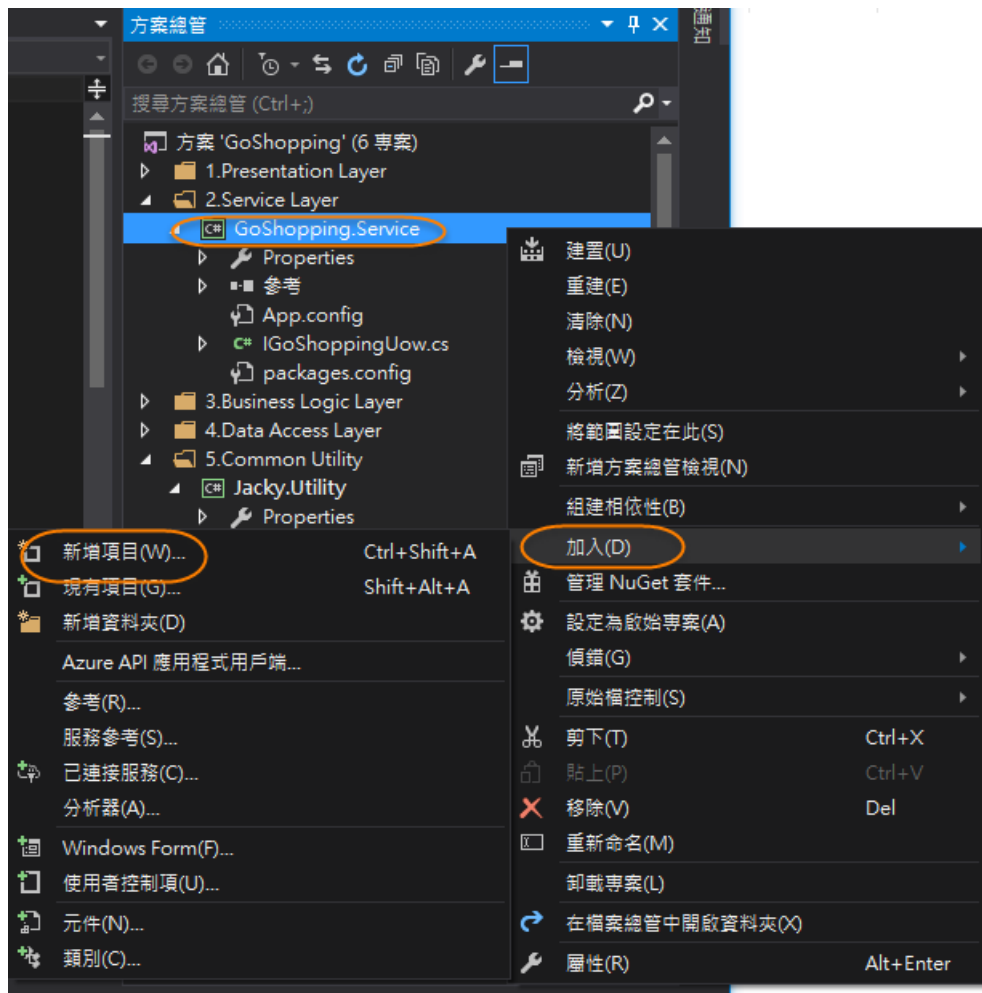
- 首先由建構函式初始化 DbContext 和 DbSet 變數。接著一一實作8個方法

```
GeneralRepository.cs* X
C# Jacky.Utility Jacky.Utility.GenericRepository<T> Get()

6 using System.Data.Entity;
7 using System.Linq.Expressions;
8 using System.Data.Entity.Infrastructure;
9
10 namespace Jacky.Utility
11 {
12     2 個參考
13     public class GenericRepository<T> : IRepository<T> where T : class
14     {
15         internal DbContext Context;
16         internal DbSet<T> DbSet;
17
18         //建構函式初始化 DbContext 和 DbSet 變數
19         1 個參考
20         public GenericRepository(DbContext context) {...}
21
22         //實作GetAll就是傳回整張資料表
23         1 個參考
24         public virtual IQueryable<T> GetAll() {...}
25
26         //實作 GetBySql 是由傳入 SQL 陳述式及參數查詢出符合條件的記錄 (為了防止 SQL Injection 請絕對不要用組字串的方式組出 SQL 陳述式)。
27         1 個參考
28         public virtual IEnumerable<T> GetBySql(string query, params object[] parameters) {...}
29
30         //實作Linq 語法取出符合某些條件的記錄
31         1 個參考
32         public virtual IEnumerable<T> Get(Expression<Func<T, bool>> filter = null, Func<IQueryable<T>, IOOrderedQueryable<T>> orderBy = null,
33             string includeProperties = "") {...}
34
35         //以資料模型的主鍵值取得該筆記錄
36         2 個參考
37         public T GetById(object id) {...}
38
39         //實作新增一筆記錄
40         1 個參考
41         public virtual void Add(T entity) {...}
42
43         //實作更新記錄
44         1 個參考
45         public virtual void Update(T entity) {...}
46
47         //實作刪除一筆記錄
48         2 個參考
49         public virtual void Delete(T entity) {...}
50     }
51 }
```

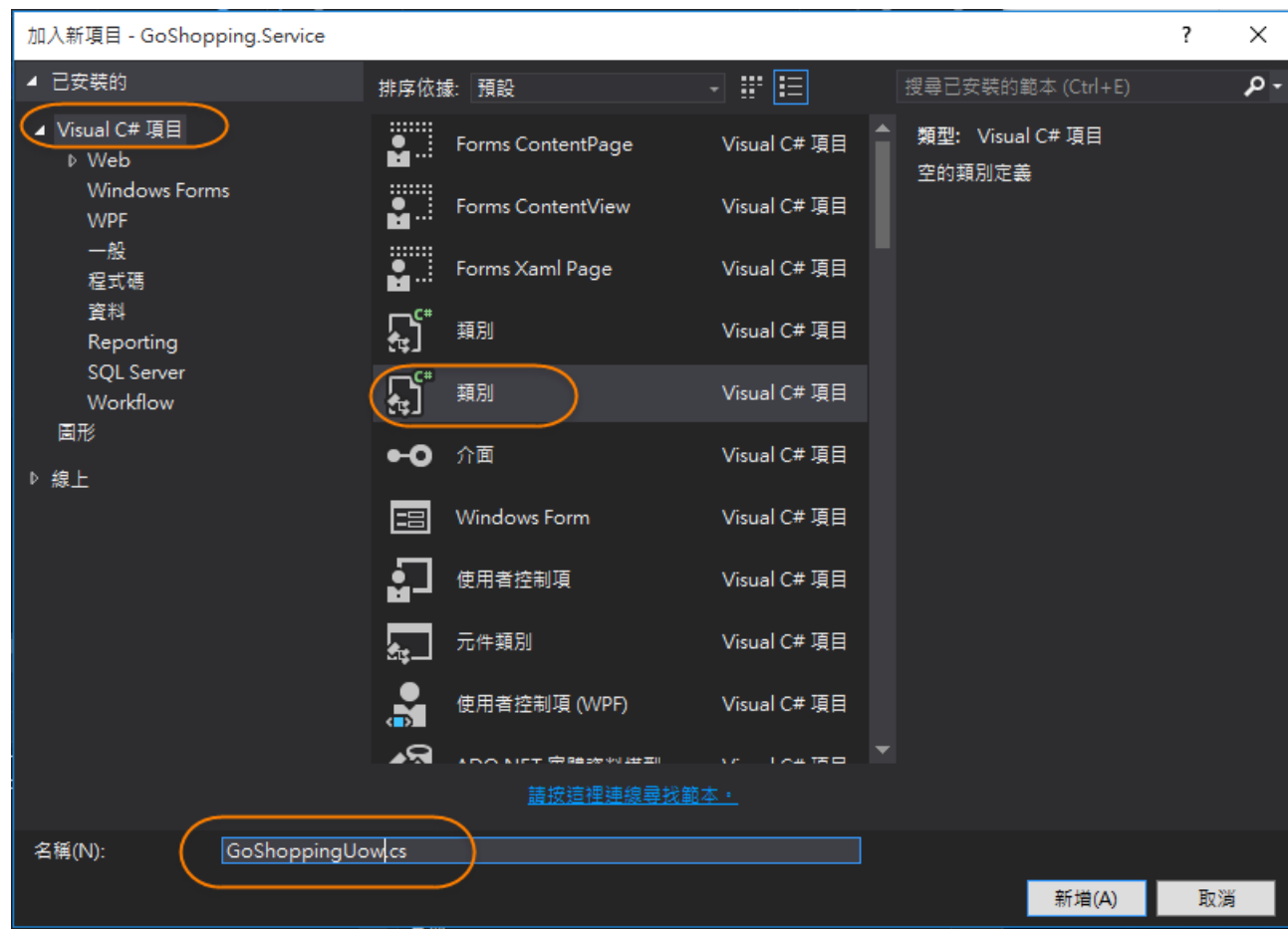
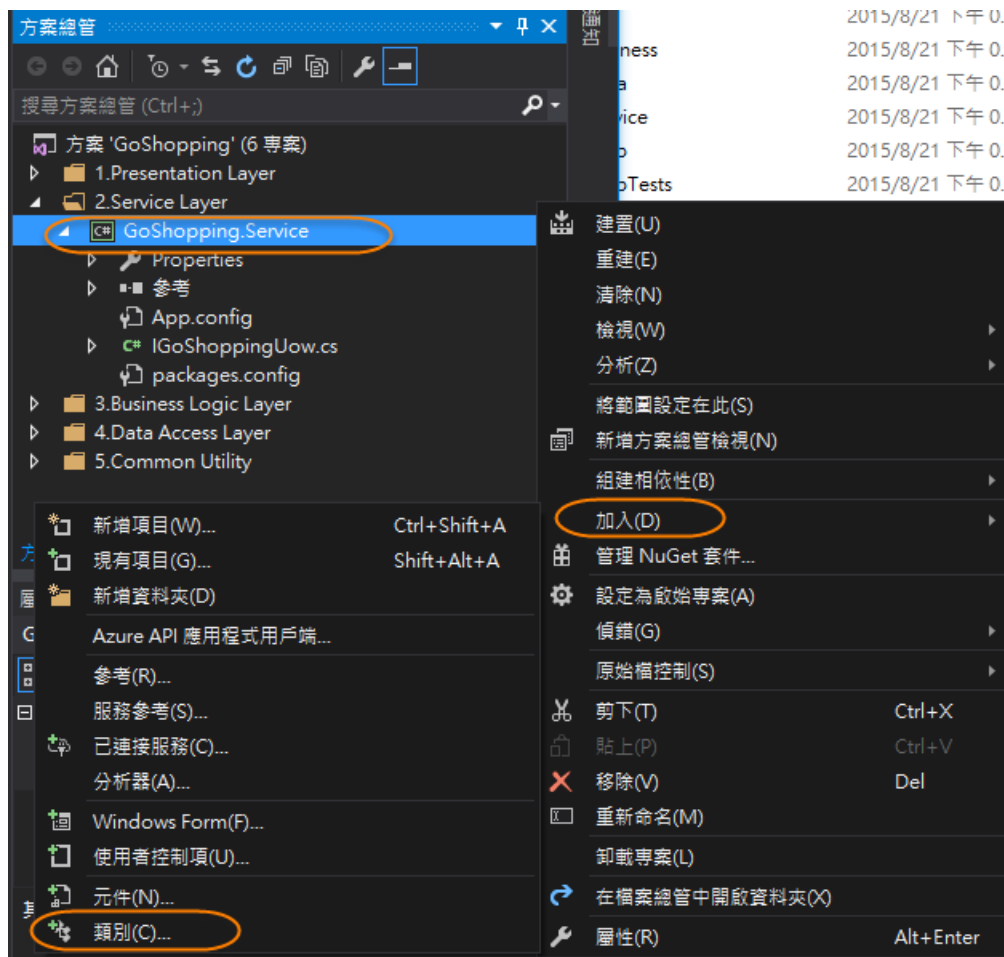

Unit of Work Pattern-1

- 服務層專案裡新增一個介面



Unit of Work Pattern-新增介面實作類別及方法-2

- 同一專案內新增一個類別



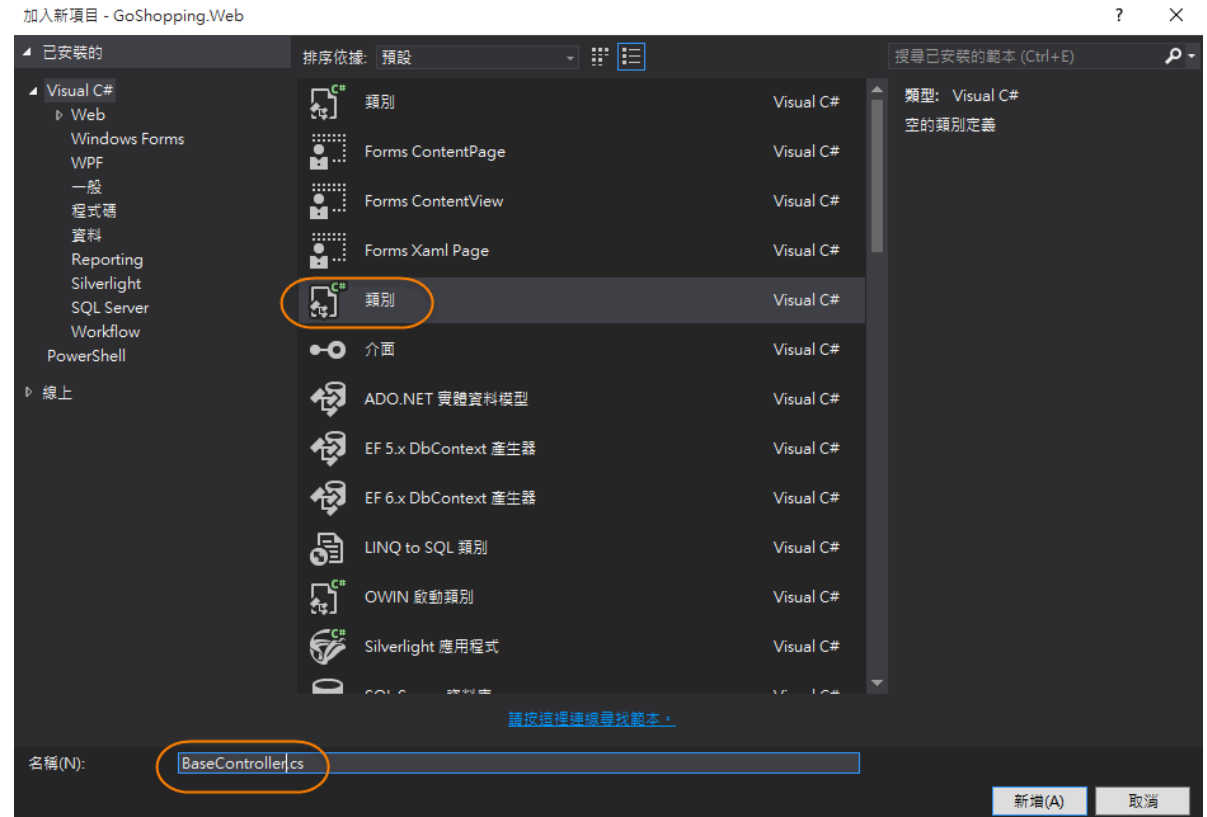
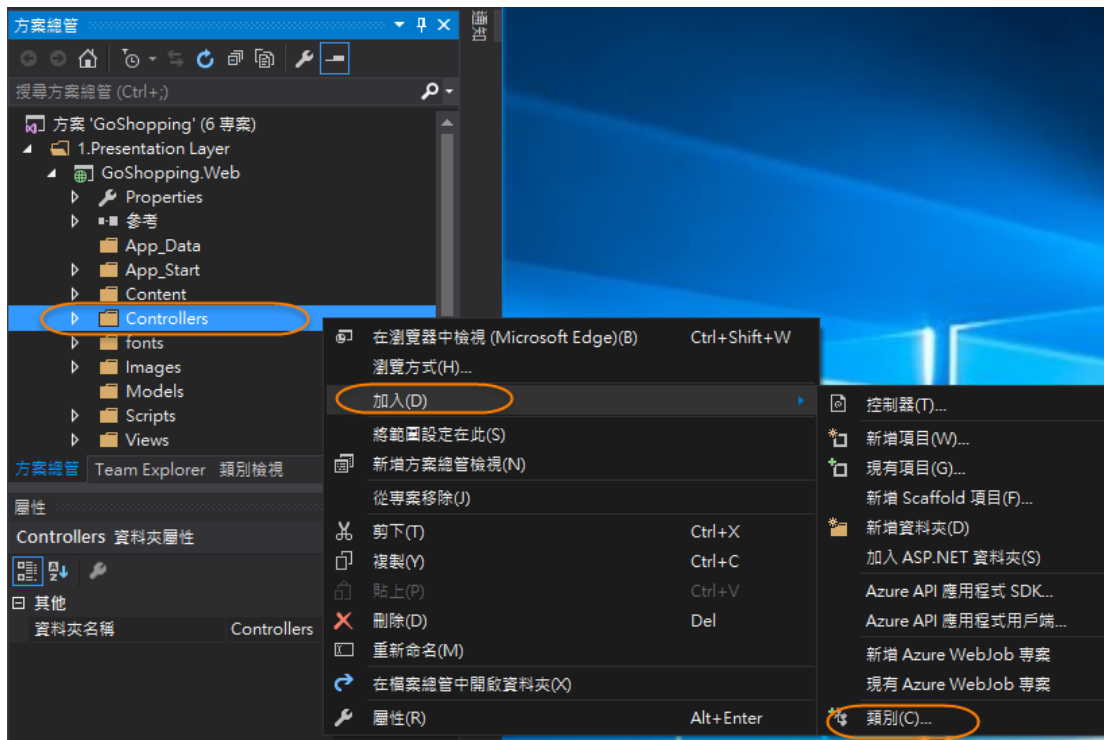
Unit of Work Pattern-新增介面實作類別及方法-3

```
GoShoppingUow.cs  X
GoShopping.Service  GoShopping.Serv

1  using GoShopping.Data;
2  using GoShopping.Business;
3  using Jacky.Utility;
4  using System;
5  using System.Data.Entity;
6
7  namespace GoShopping.Service
8  {
9      1 個參考
10     public class GoShoppingUow : IGoShoppingUow, IDisposable
11     {
12         5 個參考
13         private DbContext DbContext { get; set; }
14         0 個參考
15         public GoShoppingUow()
16         {
17             DbContext = new GoShoppingDbContext();
18             Videos = new GenericRepository<Video>(DbContext);
19         }
20         1 個參考
21         public void Save()
22         {
23             DbContext.SaveChanges();
24         }
25         2 個參考
26         public IRepository<Video> Videos { get; private set; }
27         1 個參考
28         public void Dispose()
29         {
30             Dispose(true);
31             GC.SuppressFinalize(this);
32         }
33         1 個參考
34         protected virtual void Dispose(bool disposing)
35         {
36             if (disposing)
37             {
38                 if (DbContext != null)
39                 {
40                     DbContext.Dispose();
41                 }
42             }
43         }
44     }
45 }
```

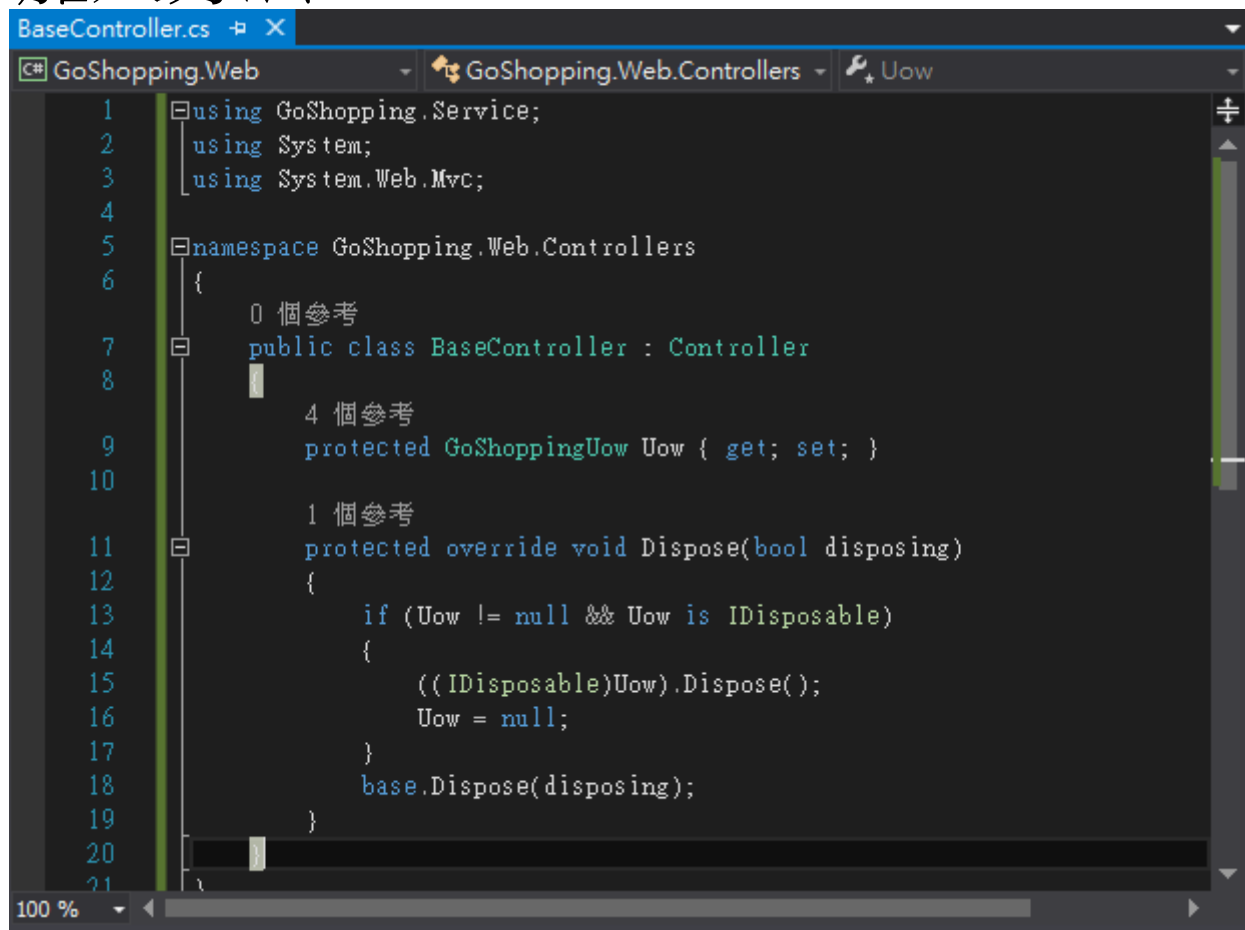
控制器 (Controller) -1

- Web 專案的 Controllers 資料夾底下，加入一個命名為 BaseController



控制器（Controller）-2

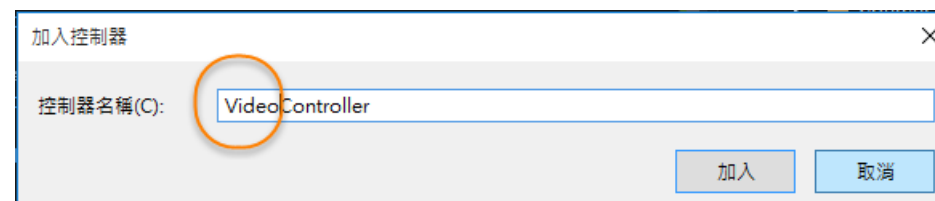
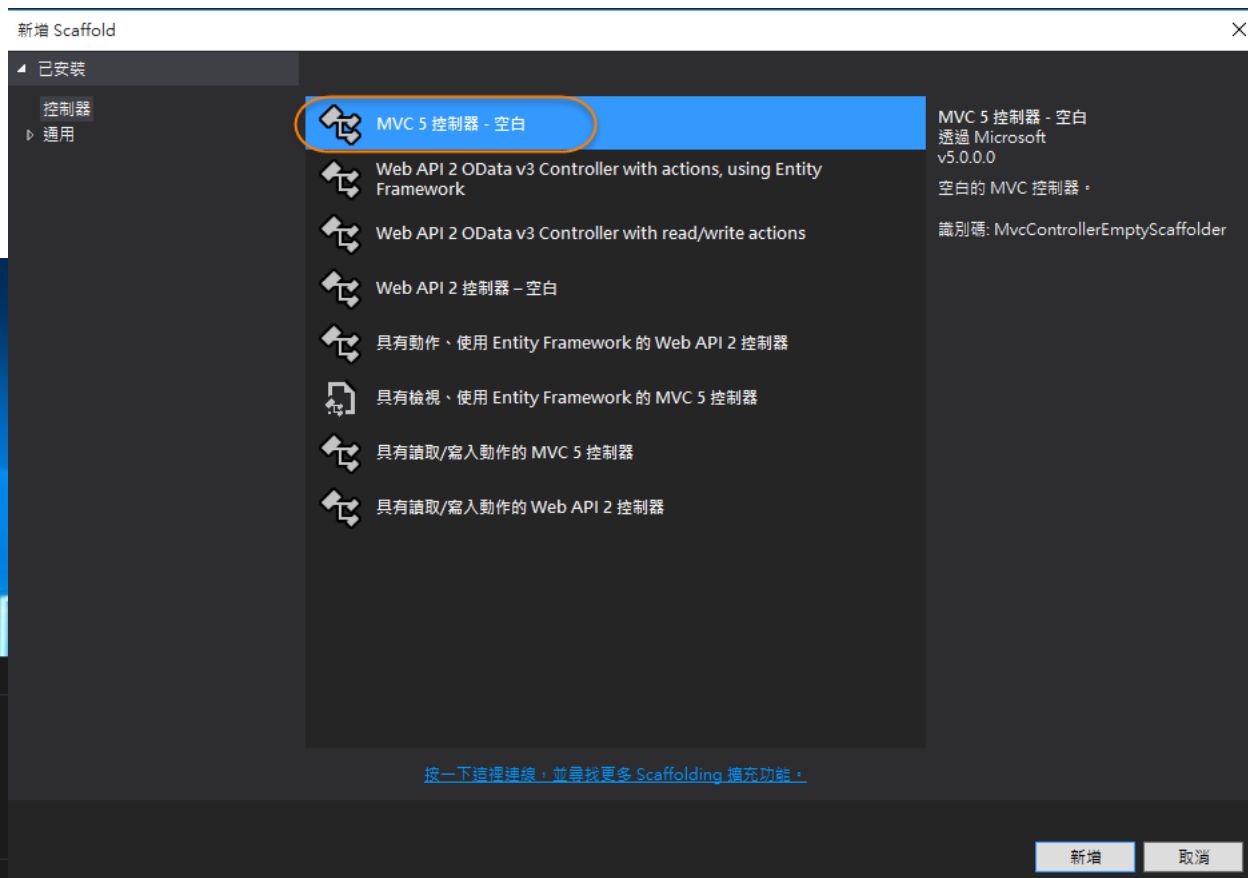
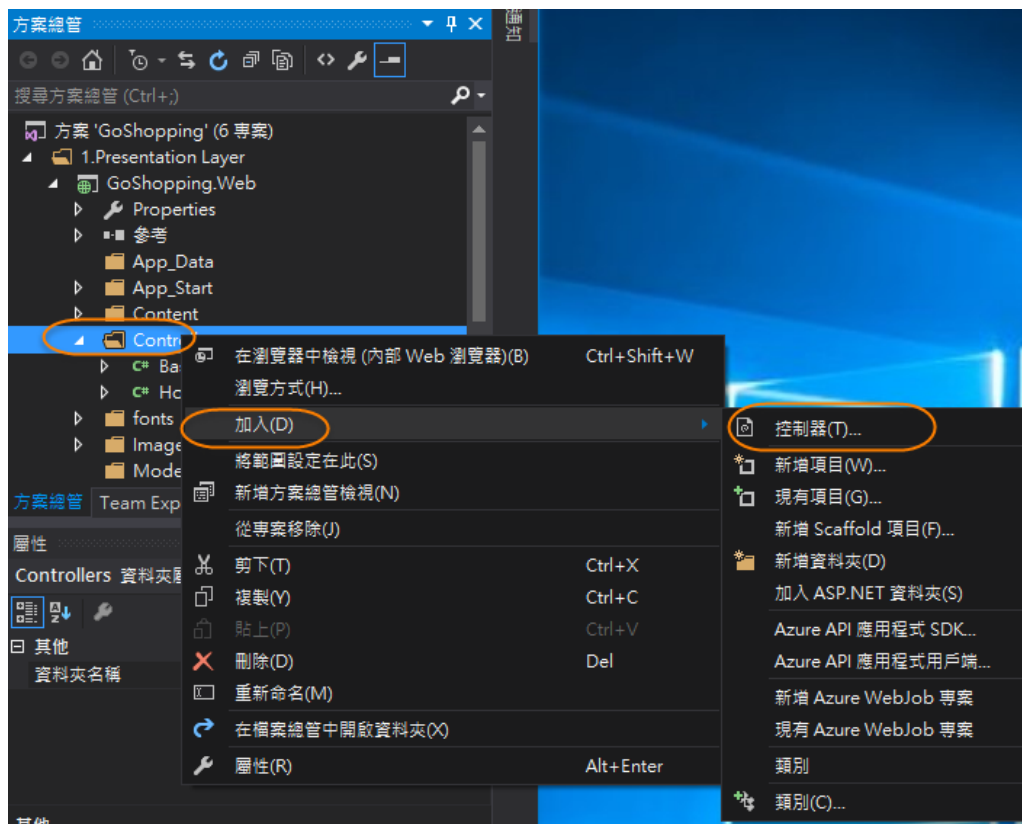
- 加入方法



```
BaseController.cs
[GoShopping.Web] GoShopping.Web.Controllers Uow
1 using GoShopping.Service;
2 using System;
3 using System.Web.Mvc;
4
5 namespace GoShopping.Web.Controllers
6 {
7     0 個參考
8     public class BaseController : Controller
9     {
10         4 個參考
11         protected GoShoppingUow Uow { get; set; }
12
13         1 個參考
14         protected override void Dispose(bool disposing)
15         {
16             if (Uow != null && Uow is IDisposable)
17             {
18                 ((IDisposable)Uow).Dispose();
19                 Uow = null;
20             }
21             base.Dispose(disposing);
22         }
23     }
24 }
```

控制器 (Controller) -3

- 加入空白控制器



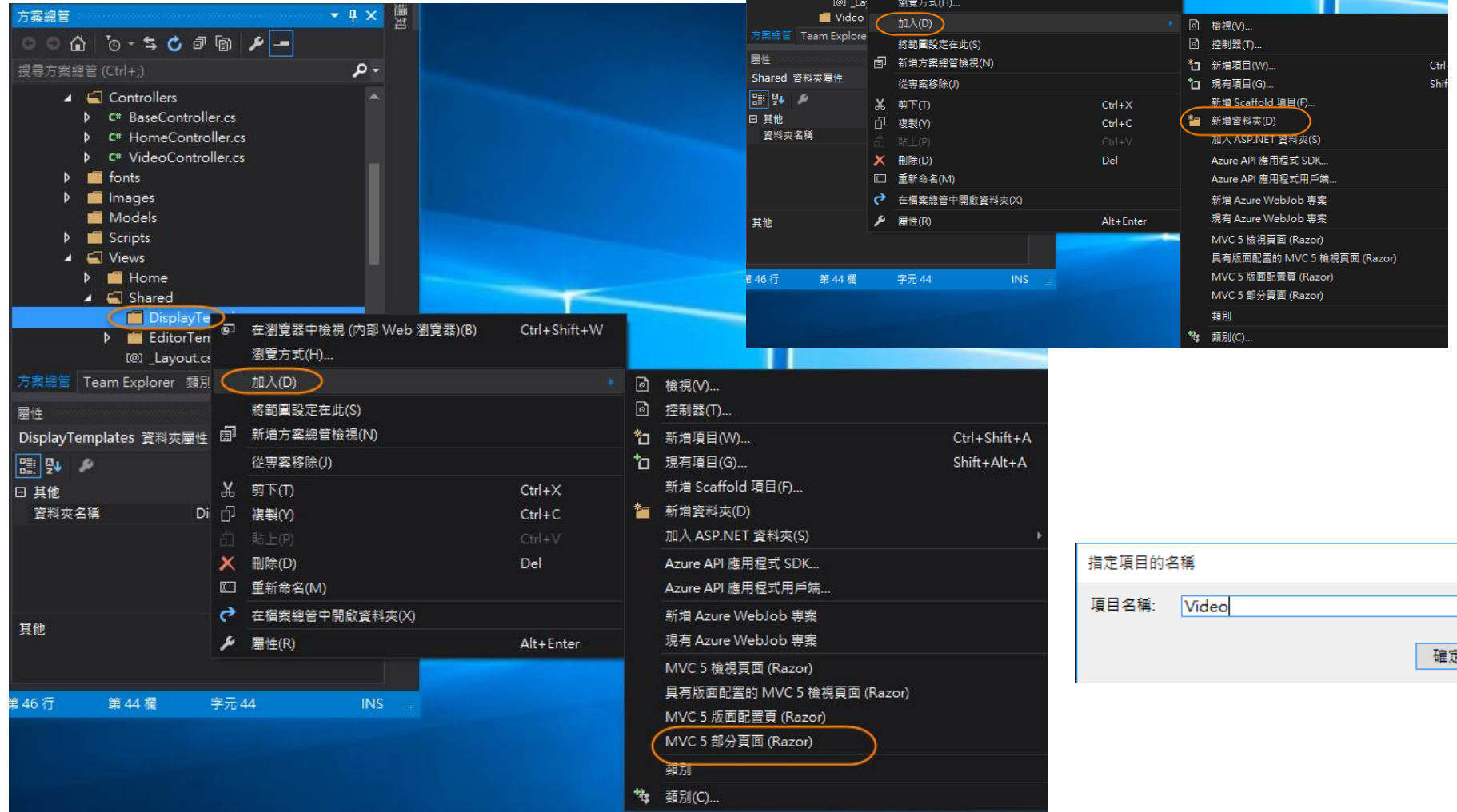
修改控制器

- 繼承了剛剛實作的基底類別
- 建構函式中初始化 UoW 物件之後，該控制器即有使用實作在 UoW 中的對資料庫進行 CRUD 的能力，且也會於控制器使用完後自動關閉資料庫連線。

```
1 using System.Web.Mvc;
2 using GoShopping.Business;
3 using GoShopping.Service;
4 using Kendo.Mvc.Extensions;
5
6 namespace GoShopping.Web.Controllers
7 {
8     1 個參考
9     public class VideoController : BaseController
10     {
11         0 個參考
12         public VideoController()...
13         // GET: Video
14         0 個參考
15         public ActionResult Index()...
16         //接收由瀏覽器 POST 過來的 Video 資料，首先檢查資料內容是否正確，如果正確就在資料庫新增一筆資料，否則就退回重填。
17         [HttpPost]
18         0 個參考
19         public ActionResult Create(Video video) ...
20         //接收由瀏覽器 POST 過來的 Video 資料，首先檢查資料內容是否正確，如果正確就更新資料庫，否則就退回重填。
21         [HttpPost]
22         0 個參考
23         public ActionResult Update(Video video) ...
24         //接收瀏覽器POST過來的Video主鍵值，再依主鍵值刪除該筆記錄。
25         [HttpPost]
26         0 個參考
27         public ActionResult Destroy(int id) ...
28     }
29 }
```

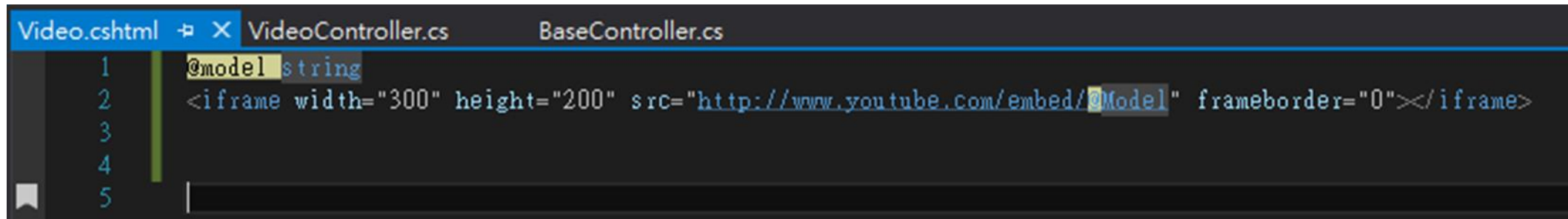

Display Template 和 Editor Template

- 新增一個命名為 DisplayTemplates 的資料夾。
- 加入一個部分頁面



Display Template

- 加入如下的程式碼

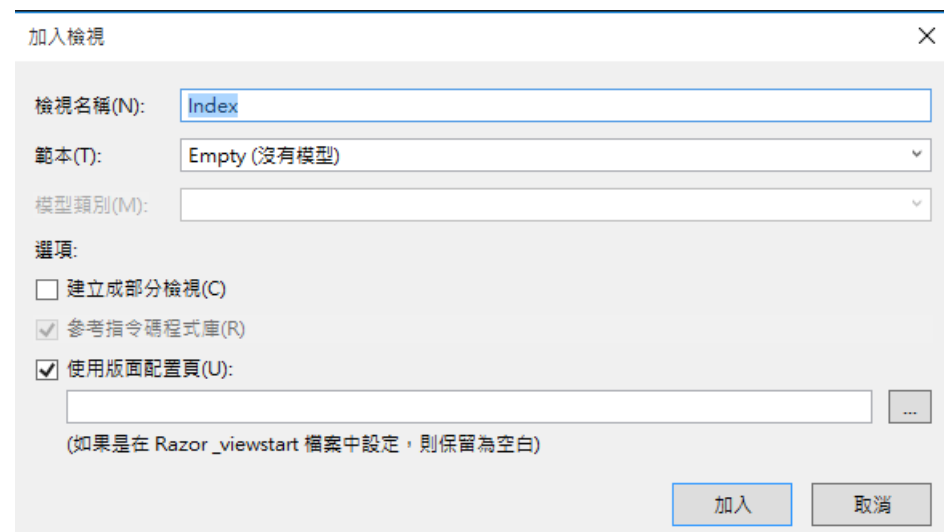
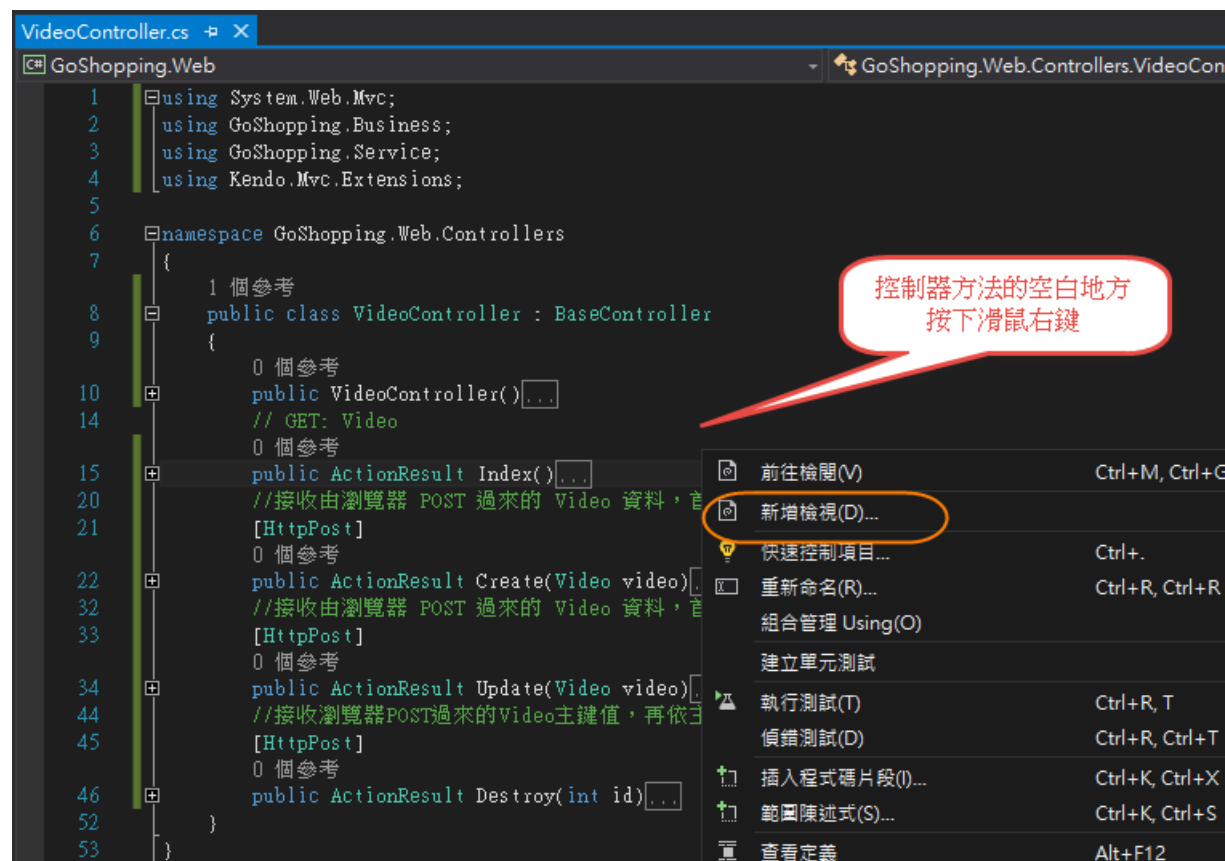


The screenshot shows a code editor with three tabs: 'Video.cshtml', 'VideoController.cs', and 'BaseController.cs'. The 'Video.cshtml' tab is active, displaying the following code:

```
1 @model string
2 <iframe width="300" height="200" src="http://www.youtube.com/embed/@Model" frameborder="0"></iframe>
3
4
5
```

檢視 (View)

- 實作影片後台維護介面



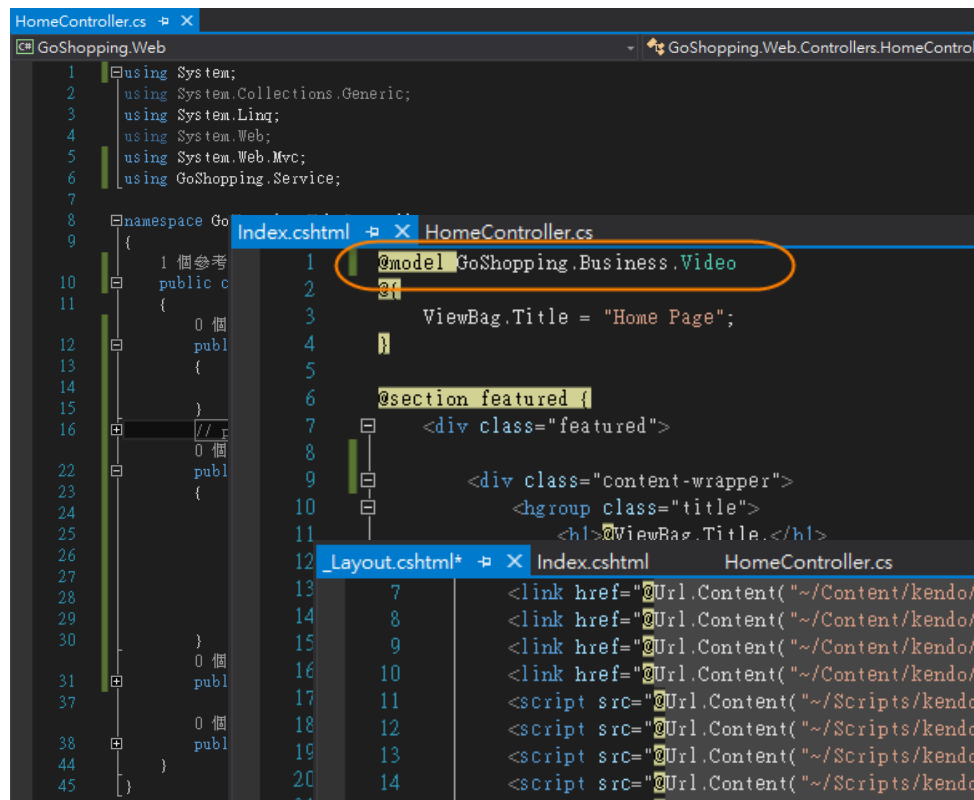
Video View Index.cshtml加入程式碼

- 檢視加入如下所示之程式碼

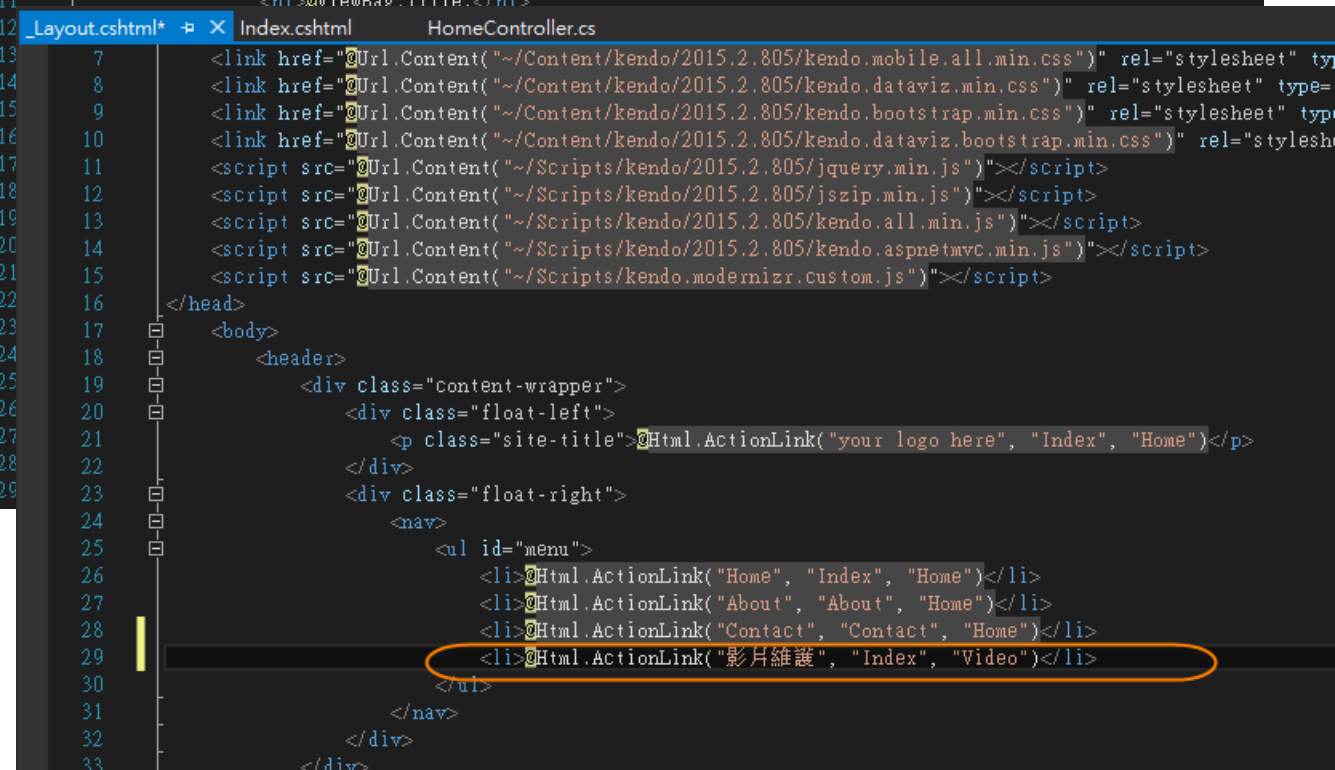
```
Index.cshtml  X VideoController.cs
1  @model IEnumerable<GoShopping.Business.Video>
2  @{
3      ViewBag.Title = "Index";
4  }
5  @(Html.Kendo().Grid(Model)
6      .Name("Videos")
7      .Columns(c =>
8      {
9          c.Command(p => p.Edit()).Text("編輯");
10         c.Bound(p => p.Id).Width(310);
11         c.Bound(p => p.Title);
12         c.Bound(p => p.StartDate);
13         c.Bound(p => p.EndDate);
14         c.Command(p => p.Destroy()).Text("刪除");
15     })
16     .ToolBar(toolbar => toolbar.Create().Text("新增"))
17     .Editable(e => e.Mode(GridEditMode.PopUp))
18     .DataSource(d => d
19         .Server()
20         .Model(m =>
21         {
22             m.Id("Id");
23             m.Field(f => f.StartDate).DefaultValue(DateTime.Today);
24             m.Field(f => f.EndDate).DefaultValue(DateTime.Today.AddDays(15));
25         })
26         .Create("Create", "Video")
27         .Update("Update", "Video")
28         .Destroy("Destroy", "Video")
29     )
30 )
31 @section scripts {
32     @Scripts.Render("~/bundles/jqueryval")
33 }
34
```

前端展示

- 在Home Controller 的 Index 方法了，請開啟 Controllers\HomeController.cs 並加入Index程式碼
- 在 Views\Home\Index.cshtml 檔案裡加入連結的程式碼
- 在 Views\Shared_Layout.cshtml加入維護介面



```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.Mvc;
6 using GoShopping.Service;
7
8 namespace GoShopping.Web.Controllers
9 {
10     public class HomeController : Controller
11     {
12         // GET: /
13         public ActionResult Index()
14         {
15             ViewBag.Title = "Home Page";
16
17             @section featured {
18                 <div class="featured">
19
20                 <div class="content-wrapper">
21                     <hgroup class="title">
22                         <h1>@ViewBag.Title.</h1>
23                     </hgroup>
24
25                     <div class="content">
26
27                     </div>
28                 </div>
29             }
30         }
31     }
32 }
```



```
7 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.mobile.all.min.css")" rel="stylesheet" type="text/css">
8 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.dataviz.min.css")" rel="stylesheet" type="text/css">
9 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.bootstrap.min.css")" rel="stylesheet" type="text/css">
10 <link href="@Url.Content("~/Content/kendo/2015.2.805/kendo.dataviz.bootstrap.min.css")" rel="stylesheet" type="text/css">
11 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/jquery.min.js")"></script>
12 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/jszip.min.js")"></script>
13 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/kendo.all.min.js")"></script>
14 <script src="@Url.Content("~/Scripts/kendo/2015.2.805/kendo.aspnetmvc.min.js")"></script>
15 <script src="@Url.Content("~/Scripts/kendo.modernizr.custom.js")"></script>
16 </head>
17 <body>
18     <div class="content-wrapper">
19         <div class="float-left">
20             <p class="site-title">@Html.ActionLink("your logo here", "Index", "Home")</p>
21         </div>
22         <div class="float-right">
23             <nav>
24                 <ul id="menu">
25                     <li>@Html.ActionLink("Home", "Index", "Home")</li>
26                     <li>@Html.ActionLink("About", "About", "Home")</li>
27                     <li>@Html.ActionLink("Contact", "Contact", "Home")</li>
28                     <li>@Html.ActionLink("影片維護", "Index", "Video")</li>
29                 </ul>
30             </nav>
31         </div>
32     </div>
33 </body>
```