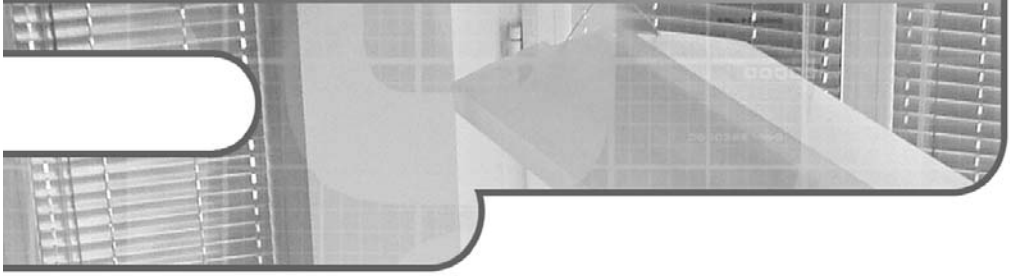


Capítulo 4
Los sitios web MVC

1. El enfoque MVC

Una vez pasada la época en la que se discutía la estructura de una aplicación web, el universo Java ha popularizado el uso de frameworks tales como Struts o Spring. Éste, y Struts en primer lugar, han sentado las bases de una separación de responsabilidades entre los distintos niveles de una aplicación web. Es cierto que las primeras tecnologías web no invitaban a los programadores a organizar sus aplicaciones; el mantenimiento se vuelve muy delicado, al tiempo que el rendimiento es ridículo.

1.1 El patrón de diseño MVC

La expresión MVC se refiere a un enfoque de diseño generalizado, o patrón de diseño. El objetivo consiste en no reinventar la rueda con cada aplicación. Como veremos, el MVC es un patrón bastante simple. No utilizarlo supone, realmente, dirigirse hacia una aplicación complicada y, por tanto, mal hecha, lo que nos recuerda al pasado tal y como veíamos antes.

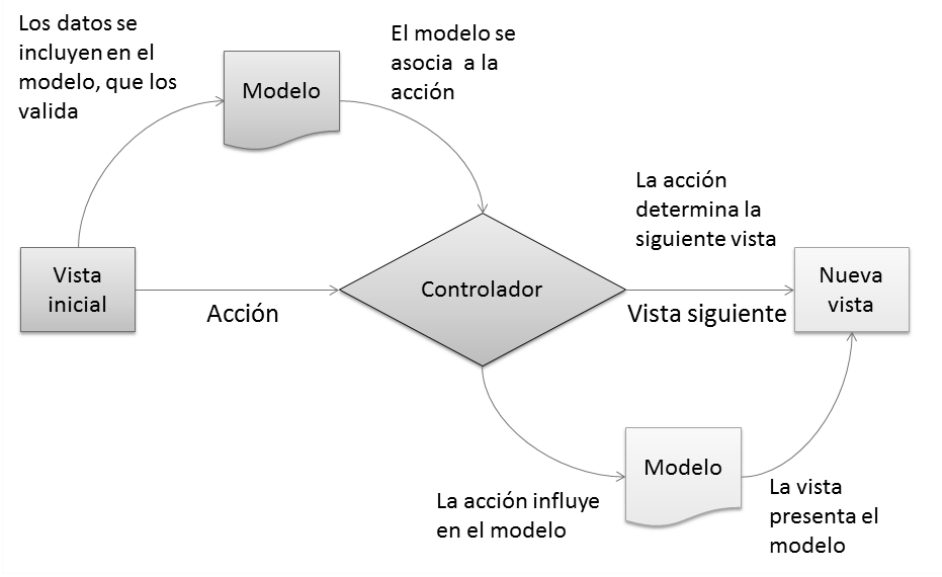
Cada letra del acrónimo MVC se corresponde con un rol bien definido; el modelo, la vista y el controlador.

El modelo es un objeto "de negocio" que agrupa sus datos, su comportamiento (métodos) y sus reglas de validación. No contiene, por lo general, ninguna lógica técnica (presentación, navegación). Es posible atribuirle aspectos (inyección de servicios tales como persistencia de archivos o SQL, transacciones, seguridad...). En los enfoques menos completos, el objeto de negocio se asocia con una clase de servicios que sirve de interfaz (API).

La vista se encarga de restituir el modelo en el seno de una interfaz gráfica (web, en nuestro caso), y permite al usuario interactuar con el modelo.

El controlador define las reglas de navegación (también llamada la cinemática). El paso de una vista a otra se realiza mediante acciones dirigidas por un controlador. El modelo se interroga, o enriquece, para condicionar el desarrollo de acciones.

La siguiente ilustración describe la secuencia de interacciones entre estos objetos:



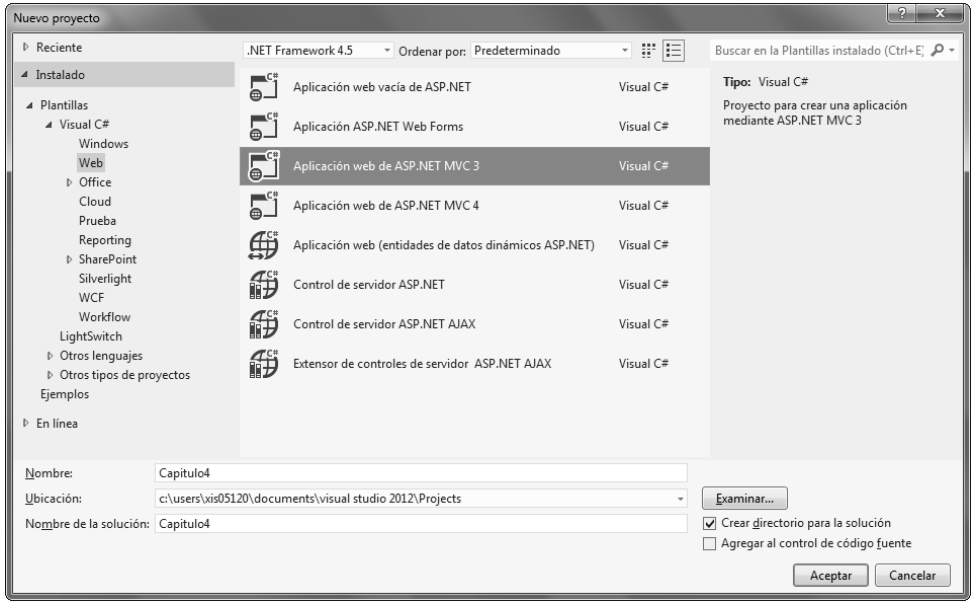
1.2 Evolución de MVC

El enfoque MVC 2 es, principalmente, una evolución del framework; consiste en utilizar únicamente un controlador para varias acciones. Esta evolución reduce considerablemente el esfuerzo en cuanto a programación y a configuración. Por suerte, el framework ASP.NET soporta, en lo sucesivo, el nivel MVC 2 mediante MVC 3 y MVC 4.

2. Los sitios ASP.NET MVC

2.1 Creación de un sitio

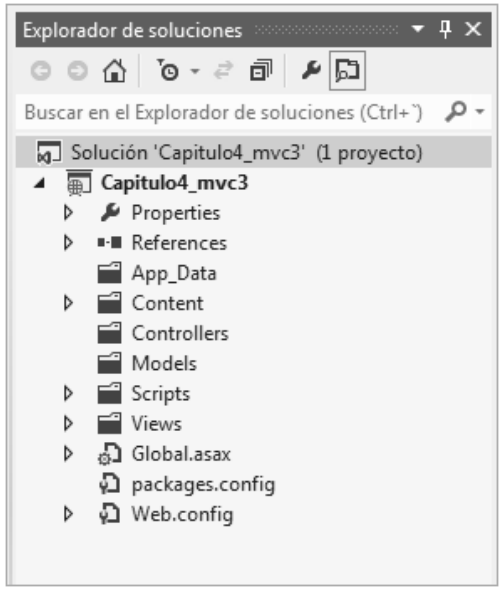
Visual Studio 2012 provee dos modelos de proyecto para MVC según el nivel de soporte MVC 3 o MVC 4. Escogeremos el nivel MVC 3 para nuestro ejemplo:



Como la aplicación MVC requiere el uso de clases que no están en el código subyacente (como con los Web Forms), el modelo Visual Studio está disponible únicamente para un proyecto web, y no para un sitio web.

2.2 Organización de carpetas

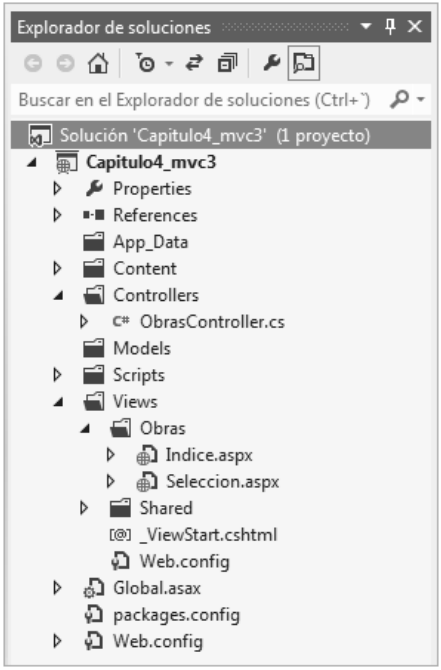
La solución del proyecto web contiene muchas más carpetas que un proyecto Web Forms.



Estas carpetas tienen, como objetivo, guiar al programador:

Content	Contiene las hojas de estilo CSS y demás recursos compartidos.
Controllers	Agrupa los controladores destinados a procesar las acciones.
Models	Agrupa los modelos que son componentes funcionales.
Scripts	Conjunto de módulos JavaScript, jQuery y AJAX.
Views	Páginas .aspx.

Las vistas son páginas ASPX, aunque no tienen código subyacente (como veremos a continuación). Se agrupan, en principio, en carpetas llamadas zonas, las cuales se corresponden con controladores. Esta regla no tiene ningún carácter obligatorio desde un punto de vista técnico, aunque es más sencillo utilizar el framework si nos ceñimos a ella.



2.3 Creación del modelo

Un modelo es una clase cuyas instancias se denominan "objetos de negocio". Esto significa que no contiene ninguna lógica técnica, y que el framework se encarga de gestionar el ciclo de vida del componente de negocio, aportándole servicios técnicos de alto nivel tales como la seguridad, las transacciones, la validación, la persistencia...

```
#region Modelo Obra
    /// <summary>
    /// Objeto de negocio Obra
    /// </summary>
```

```
public class Obra
{
    /// <summary>
    /// Por convención, ID es una clave primaria
    /// </summary>
    public int ID { get; set; }

    /// <summary>
    /// Campo Autor
    /// </summary>
    public string Autor { get; set; }

    /// <summary>
    /// Campo Título
    /// </summary>
    public string Titulo { get; set; }

    public Obra()
    {
        ID = 0;
        Autor= "";
        Titulo = "";
    }

    public Obra(int id, string autor, string titulo)
    {
        ID = id;
        Autor = autor;
        Titulo = titulo;
    }
}
#endregion
```

Hemos asociado una clase de servicio que contiene algunos métodos ineludibles. Las clases de servicio implementan bastante a menudo el patrón CRUD (*Create Read Update Delete*). Nosotros obviaremos esta práctica, sin implementar la persistencia en base de datos SQL, para centrarnos en la arquitectura MVC.

```
#region ObraServicio
/// <summary>
/// Clase de servicio para Obra
/// </summary>
public class ObraServicio
```