

## ASP.NET CORE CON C#

### 6. LAYOUT Y VISTAS PARCIALES

Una aplicación puede contener partes comunes en la interfaz de usuario que son iguales en toda la aplicación, el logotipo, el encabezado, las barras de navegación o la sección del pie de página. También puede ocurrir que alguna vista se nos quede muy grande, es mejor trabajar con archivos más pequeños.

ASP.NET MVC introdujo una vista de Diseño(layout) que contiene estas partes comunes, de modo que no tenemos que escribir el mismo código en cada página.

Un layout se puede heredar en varias vistas para proporcionar una apariencia coherente en varias páginas de una aplicación. En estos casos se almacenan en la subcarpeta Shared de la carpeta View. Si no es así se suelen crear en la misma carpeta que la vista.

Tiene la misma extensión que otras vistas, .cshtml o .vbhtml.

El layout contiene el Doctype html, la cabeza y el cuerpo como html normal, la única diferencia es la llamada a los métodos `RenderBody()` y `RenderSection()`.

`RenderBody` actúa como un marcador de posición para otras vistas. Por ejemplo, `Index.cshtml` en la carpeta de inicio se inyectará y representará en la vista de diseño, donde se llama al método `RenderBody()`.

La diferencia entre un layout y una vista parcial es que las primeras son contenedoras de otras vistas. Por ejemplo en nuestra aplicación queremos que todas las páginas tengan la misma cabecera y el mismo pie. Lo diseñamos en un layout que será una vista estática (no cambia). Las demás vistas se incluirán en la parte central del layout y son dinámicas, las vamos cambiando.

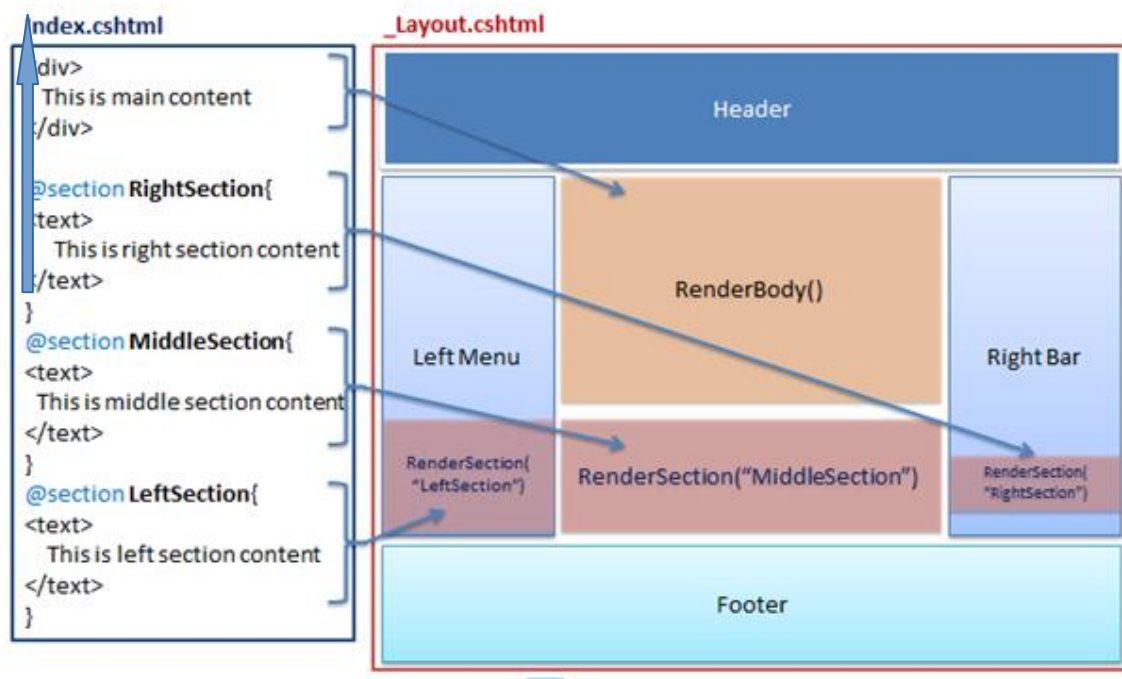
Cuándo las vistas “están en igualdad de condiciones”, es decir una no incluye a la otra y ambas son dinámicas, se utilizan vistas parciales.

Por ejemplo, en nuestro ejercicio del portfolio, podríamos hacer varias vistas parciales: una para nuestra presentación, otra para nuestros conocimientos.....

**Para añadir un layout** se añade una vista Razor y nos aseguramos de que no están marcadas las opciones de vista parcial o layout

Usaremos con frecuencia dos métodos: `RenderBody()` , `RenderSection()`.

1. `RenderBody()`: método que utilizaremos para indicarle al layout que en esa parte inyectaremos las vistas asociadas al mismo. Es como un marcador de posición
2. `RenderSection()`: método para indicar que sección de las vistas asociadas se inyecta en esa posición



Al crear la vista especificaremos que a que layout la asociamos

También podemos escribir mediante código Razor la instrucción directamente en la vista o podemos configurar un archivo

`_ViewStart.cshtml`

en el que especificamos el layout asociado a todas las vistas que están contenidas en la misma carpeta que dicho archivo. En cualquiera de las dos primeras formas mencionadas, si cambiamos el nombre del layout estamos obligados a revisar todas las vistas para

cambiar ahí la asociación. Si utilizamos el fichero ViewStart, esto no pasará.

Añadimos este fichero en la carpeta View. Cuando se renderiza una vista, MVC buscará este archivo de manera automática y el contenido de este archivo será tratado como si estuviera incluido en la vista. De esta manera podremos omitir de nuestro código la línea que establece la propiedad Layout y MVC usará el archivo `_viewstart` que hemos creado, para establecer el layout de nuestra vista.

Hay que tener en cuenta:

- El contenido de este archivo se usará en todas las vistas de la aplicación que no tengan establecida la propiedad Layout. Si queremos que en una de las vistas se use otro layout distinto al del archivo `_viewstart`, deberemos establecerlo explícitamente. La propiedad layout de la vista **tiene precedencia respecto al archivo `_viewstart`**.
- Hay una diferencia entre omitir la propiedad layout y establecerla a null. Si en la vista no queremos usar un layout, entonces estableceremos la propiedad layout a nulo, de lo contrario MVC **generará el archivo ViewStart y asociará su layout por defecto**.

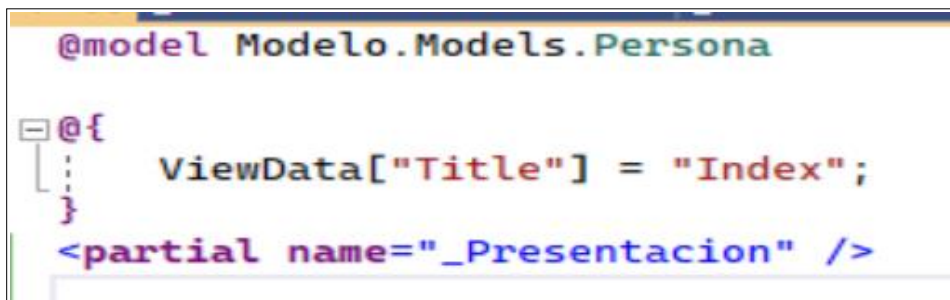
Para añadir una vista parcial, se añade desde la carpeta del controlador que cargará dicha vista parcial o desde shared si es que esta vista parcial va a ser compartida por varias vistas

Por convención el nombre también lleva un \_ porque también es un elemento compartido

En ese archivo coloco el html

Para indicar en mi vista que tiene que incluir una vista parcial

`<partial name="-----" />`



```
@model Modelo.Models.Persona

@{
    ViewData["Title"] = "Index";
}

<partial name="_Presentacion" />
```

Los modelos a las vistas parciales se pasan igual que a una página

Veamos un ejemplo de cómo pasar una lista de objetos, operación bastante frecuente.

1. Generamos una clase Alumno
2. Generamos una clase para definir el modelo que le pasamos a la vista Index, dicho modelo es una lista de alumnos.
3. En el controlador generamos la lista y el modelo anterior con dicha lista
4. En la vista usamos código razor para comprobar que no está vacía y la recorremos con foreach

Cuando necesitemos pasar varios modelos (distintas clases) a una vista, tendremos que hacer lo mismo que en el ejemplo, es decir, generar una clase cuyas propiedades sean objetos de esas clases.

Si necesitamos pasar a una vista parcial sólo un tipo de objeto que forma parte del modelo completo

`<partial name="-----" model="Model.propiedad">`