Capítulo 3
Los Web Forms

1. Presentación de los Web Forms

Los formularios web (Web Forms) representan la parte más visible de los sitios web ASP.NET y, en consecuencia, la más popular. Se basan en un reparto de responsabilidades de tipo **MVC**: modelo, vista, controlador. Cuando se escribe un formulario utilizando el estilo **código independiente**, la página HTML .aspx se encarga de la representación (vista), la clase C# gestiona los datos y los cálculos realizados con ellos (modelo), mientras que el servidor de aplicaciones ASP.NET coordina el conjunto (controlador). Este análisis resultará familiar, sin duda, a los desarrolladores Java en lo relativo a la organización de sitios web ASP.NET.

Por otro lado, los formularios web son el resultado de la transposición que realiza Microsoft del modelo Visual Basic 6, y una forma original y productiva de desarrollar interfaces gráficas para Internet. El éxito de este modelo ha sido tal, que Sun lo ha replicado por su cuenta en la tecnología de desarrollo web JSF (*Java Server Faces*).

1.1 Estructura de una página ASPX

En el capítulo Los sitios web ASP.NET nos hemos puesto al día con la estructura de una página ASPX desde el punto de vista de la compilación. Ahora se trata de comprender su estructura lógica.

Estudiemos el código que aparece en una página Default.aspx:

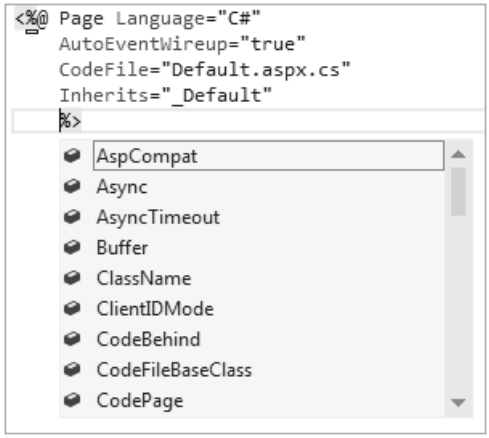
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default.aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

    </div>
    </form>
</body>
</html>
```

Este código está formado por tres partes: una directiva page, una declaración de DTD y código XHTML.

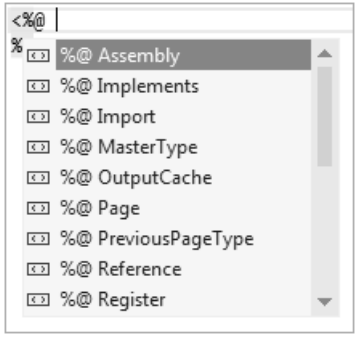
La directiva Page

Las directivas organizan la lectura de una página ASPX en el servidor de aplicaciones. En la página Default.aspx, el atributo **Language** define el lenguaje –C#, VB.NET, C++– utilizado para escribir los scriptlets. Hay otros atributos presentes, que sirven para la comunicación con la página de code behind (**AutoEventWireup**, **CodeFile**, **Inherits**), para aplicar temas, para la gestión de trazas... Descubriremos el uso de estos atributos conforme avance nuestro estudio.

Por suerte, Visual Studio proporciona distintos atributos aplicables utilizando la combinación de teclas [Ctrl][Espacio].



Existen otras directivas disponibles para incluir recursos en el entorno de la página: estrategia de caché, componentes, ensamblados, tipos de página maestra...



Las DTD

Las definiciones de tipo de documento (*Document Type Definition*) las establece el consorcio W3C. Se trata de una norma aplicable a los documentos SGML, XML y HTML que fija las reglas sintácticas y semánticas de la construcción de un documento basado en tags (marcadores).

Los navegadores son bastante tolerantes en lo que respecta a las DTS. Con la versión ASP.NET 1.X, el flujo HTML de salida es compatible con la DTD **HTML transicional de nivel 4**. Salvo el atributo MS_POSITIONNING que no estaba filtrado, el código HTML era completamente estándar. Es cierto que una página ASPX contiene etiquetas especiales (`<asp:label>`, por ejemplo) que se traducen por una secuencia HTML accesible desde el navegador.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

La versión 2.0 aporta una conformidad con **XHTML**, una declaración mucho más estricta del lenguaje HTML. Los puristas pueden dirigirse al sitio web de W3C e introducir una página ASP.NET en el motor de verificación ubicado en la dirección http://validator.w3.org. Las páginas deben estar en conformidad con la DTD correspondiente.

Observación

Preste atención, para realizar esta prueba, es preciso guardar el flujo HTML en un bloc de notas abierto mediante el comando ver código fuente. La función Guardar como - Página HTML del navegador Internet Explorer modifica el archivo y desvirtualiza la prueba.

Para el desarrollador de páginas web, la conformidad con una versión específica del lenguaje HTML no es suficiente para garantizar que una página tenga la misma presentación sea cual sea el navegador. De entrada, los navegadores tienen la responsabilidad de interpretar las reglas de representación tal y como ellos las entiendan. El lenguaje HTML describe el contenido, pero no la representación. Además, las páginas incluyen código JavaScript y estilos CSS, que difieren en su interpretación en función del navegador.

El servidor ASP.NET 2.0 ha introducido otro cambio: desaparece la noción de esquema de navegador de destino. Es cierto que esta directiva no ha podido estar a la par con la evolución de los navegadores, sin contar con la aparición de otros dispositivos de navegación. En su lugar, los sitios web ASP.NET poseen una carpeta **App_Browsers** que considera las características de cada navegador. Este aspecto se estudiará cuando aparezcan los componentes personalizados.

Para ciertos navegadores y programas JavaScript que intervienen en el DOM y que no sean compatibles con la norma XHTML, el servidor de aplicaciones puede configurarse para utilizar el modo HTML transicional. La directiva se ubica en el archivo Web.config:

```
<xhtmlConformance mode="Legacy"/>
```

El atributo mode acepta tres valores:

Legacy	Antiguo formato HTML transicional
Strict	XHTML strict
Transitional	XHTML transicional

El código XHTML

Si bien es cierto que el servidor de aplicaciones ASP.NET 1.X emitía un flujo conforme a la DTD HTML 4 transicional, la propia sintaxis de las páginas ASPX mezclaba secuencias HTML con secuencias XML. Visual Studio 2003 se encargaba de controlar la coherencia del conjunto y generar advertencias cuando era necesario, y el servidor de aplicaciones debía realizar una lectura más atenta (y costosa) para separar las secuencias HTML de las secuencias XML.

Ahora, el elemento <html> contiene una referencia al espacio de nombres XHTML:

```
<html xmlns="http://www.w3.org/1999/xhtml" >
```

Dicho de otro modo, las etiquetas de una página ASPX deben respetar la sintaxis XHTML. De este modo, las etiquetas que comienzan por `asp` (controles web), `uc` (controles de usuario) o `cc` (controles personalizados) no forman parte del vocabulario XHTML. Pero, al menos, la sintaxis es mucho más próxima y más precisa. Y el flujo de salida permanece, en cualquier caso, conforme a la DTD declarada.

Por último, Visual Studio hace todo lo posible para validar de antemano las secuencias HTML que figuran en una página ASPX. Se generan mensajes de advertencia para llamar la atención del desarrollador acerca de las no conformidades.

1.1.1 Estilo anidado, en línea y separado

La organización de una página dinámica es una simple cuestión de estilo. Según la naturaleza de la secuencia HTML que se quiera describir, es preferible optar por la versión anidada o por la versión en línea (inline). Solo el estilo separado (code behind) supone un cambio radical y aporta una distinción neta entre la presentación y el cálculo. Éste es el motivo por el que se le da privilegio en Visual Studio.

El estilo anidado

Son las primeras generaciones de las páginas dinámicas (ASP, PHP) las que imponen el estilo anidado. Con los modelos de componentes web ASP.NET, deja de tener tanto sentido, aunque sigue siendo aplicable. También puede servir en el caso de controles a base de modelos tales como los Repeater o los Data List.

He aquí un ejemplo de código basado en este estilo:

```
<body>
    <form id="form1" runat="server">
    <ul>
    <%
      int i;
       string[] dias = { "lunes", "martes", "miércoles", "jueves",
"viernes", "sábado", "domingo" };
      for(i=0; i< dias.Length; i++)
        {
    %>
            <li><%= dias[i] %></li>
    <%  } %>
    </ul>
    </form>
</body>
```

El desarrollador debe trabajar de la mejor forma para alinear su código como si se tratase de un programa escrito completamente en C#.

El estilo en línea (inline)

El estilo anidado se utiliza, principalmente, en la presentación. No conviene utilizarlo cuando se planifica el procesamiento. La versión en línea separa el código C# y el código HTML en dos partes del mismo archivo .aspx. Las etiquetas `<script runat="server">` indican al compilador que se trata de código C#, aunque puedan reemplazarse por scriptlets `<% %>`.

```
<%@ Page Language="C#" %>
<script runat="server">
    // contiene el código de procesamiento de eventos
    void procesar_click(object sender, EventArgs e)
    {
        mensaje.Text = "¡Ha hecho clic!";
    }

</script>

<!-- límite entre el código C# y el código HTML -->

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Estilo en línea</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="mensaje" runat="server"></asp:Label>
        <asp:Button ID="cmd" runat="server" Text="Haga clic aquí"
```