

3

Controles Web

Controles HTML

Estos controles son uno de los tipos de controles que podemos encontrar dentro de las páginas ASP .NET. Los controles HTML ofrecen a los desarrolladores de entornos Web la potencia de los **Web Forms** manteniendo la familiaridad y facilidad de uso de las etiquetas HTML que representan los campos de un formulario. Los controles HTML pueden trabajar con código de cliente y con código de servidor.

Estos controles tienen el mismo aspecto que una etiqueta HTML a excepción de que presentan el atributo **runat** con el valor **server**. Otra diferencia es que posee un atributo especial para el tratamiento de eventos del cliente con código del servidor. Si queremos realizar el tratamiento de eventos con código del servidor, utilizaremos el atributo **onserverevent**, a este atributo se le indica como valor el nombre del método que queremos ejecutar.

Para permitir hacer referencia a los controles dentro del código fuente de la página, se debe utilizar el atributo **id**, de esta forma el objeto que se corresponde con el control de servidor se puede utilizar en el código de servidor para utilizar sus métodos o manipular sus propiedades.

Estos controles se ofrecen para mantener la compatibilidad con versiones anteriores de ASP, así si queremos que una etiqueta HTML pase a ser un control HTML simplemente debemos indicar la propiedad runat con el valor server, normalmente se suelen utilizar los controles Web de servidor que encontraremos posteriormente.

```

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></title>
</head>
<body>
<form id="Form1" runat="server">
    <b>Nombre de usuario:</b>
    <br/>
    <input id="nomusuario" type="text" size="30" runat="server" />
    <p>
        <b>Comentarios:</b>
        <br/>
        <textarea id="comentarios" cols="60" rows="10" runat="server"></textarea>
    </p>
    <input id="Submit1" type="submit" value="Añadir comentario" runat="server" />
</form>
</body>
</html>

```

El **espacio de nombres** (NameSpace) en el que podemos encontrar los controles HTML se denomina **System.Web.UI.HtmlControls**, no debemos olvidar que aunque tengan un gran parecido con la sintaxis HTML, los controles HTML, como todos los controles ASP .NET, tienen sus clases correspondientes dentro del .NET Framework, gracias a esto podemos acceder a ellos desde código de servidor.

El elemento clave para trabajar con estos controles de servidor es el atributo **runat="server"**. Cuando se establece este atributo se activan los eventos para su tratamiento en el servidor y el mantenimiento automático del estado de los controles dentro del **Web Form**, es decir, pasa a ser un **control de servidor**. Si no se establece este atributo el control funcionará como un **elemento del lenguaje HTML**, es decir, únicamente como un **control de cliente** al que podemos acceder solamente desde **javascript**.

 <input id="txtiniciar" type="submit" value="Iniciar" onmouseover="iniciar()" />

Esta es una etiqueta de HTML, nos podemos fijar que **nolleva** el atributo **runat="server"** y por tanto sobre ella solo podemos ejecutar código de cliente, como en este caso que llamamos a una **función** escrita en **javascript** denominada **iniciar()** cuando pasamos el ratón por encima de ella (evento **onmouseover**).

<input id="txtenviar" type="submit" value="Enviar" runat="server" onserverclick="enviar" />

A este otro botón se le ha añadido el atributo **runat="server"** y por tanto aparecerá la opción de realizar un **onserverclick** llamando al procedimiento enviar que será ejecutado en el servidor en donde en el archivo **Default.aspx.cs** escribiremos nuestro código en **c#**.

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

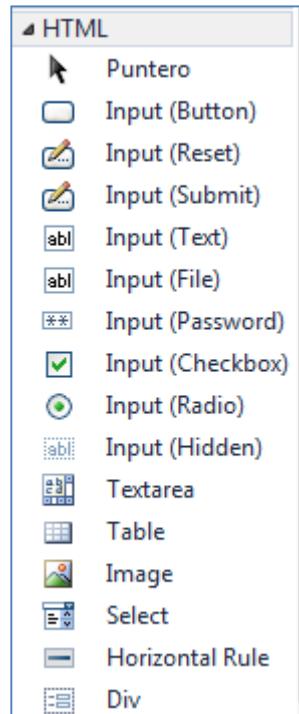
public partial class _Default : System.Web.UI.Page
{
    protected void pulsame(object sender, EventArgs e)
    {
        Response.Redirect("http://www.google.es");
    }
}

```

De todas formas, dentro de un Web Form podemos tener controles de servidor de ASP .NET y controles de cliente pertenecientes a elementos HTML.

Algunas de las clases que se corresponden con los **controles HTML** y que se encuentran dentro del espacio de nombres **System.Web.UI.HtmlControls** son las que aparecen en el cuadro de Herramientas bajo el grupo HTML.

Estas clases ofrecen al desarrollador el acceso en el servidor a las distintas etiquetas HTML que finalmente se enviarán al cliente Web como el resultado de la ejecución de la página ASP .NET correspondiente.



Controles Web

Estos controles Web de ASP .NET, el otro gran grupo de controles, cumplen la misma funcionalidad que los anteriores, los controles HTML, en este caso muestran una sintaxis basada en XML, y no existe una correlación directa entre los controles y las etiquetas HTML que se generan, tal como si sucedía con los controles HTML.

Estos controles pertenecen al NameSpace **System.Web.UI.WebControls**, y a diferencia con el espacio de nombres dónde encontramos los controles HTML, el espacio de nombres que contiene los controles Web presenta un mayor número de controles. Este espacio de nombre deriva del espacio de nombres **System.Web.UI.Control**.

Como se puede comprobar existen algunas diferencias a la hora de utilizar controles HTML y controles Web. Además de las más evidentes relativas a la distinta sintaxis a la hora de declarar los controles, algunos nombres de las propiedades cambian. Por ejemplo, el control **HTML Button** posee la propiedad **Value**, y sin embargo el **control Web equivalente** posee la propiedad **Text**.

A la hora de indicar los métodos para el tratamiento de eventos de la pulsación de un botón del ratón en los **controles HTML** utilizamos la propiedad **onserverclick**, y en los **controlesWeb** utilizamos la propiedad **OnClick**.

Cuando la página se carga en el navegador, el **control Web** determina el tipo de navegador que ha realizado la petición, y de acuerdo con esta información genera el **código HTML** apropiado, podemos decir que en este aspecto se trata de controles inteligentes.

Los **controles Web**, además de ofrecernos los controles estándar, (como hacían también los **controles HTML**), incluyen también otro tipo de controles:

- Controles de Validación
- Controles de Navegación
- Controles de Inicio de Sesión
- Controles de Datos
- WebParts

Controles Web Estándar

Para poder acceder a los controles Web desde el código debemos referirnos a ellos a través de la propiedad ID.

Todos los **controles Web** derivan de las clases **Control** y **WebControl**, en donde podemos encontrar las propiedades, métodos y eventos comunes a todos ellos.

Clase Control

Las propiedades de la clase Control son:

Propiedad	Comentario
ID	Obtiene o establece el identificador de programación asignado al control de servidor.
Page	Obtiene una referencia a la instancia Page que contiene el control de servidor.
Site	Obtiene información sobre el sitio Web al que pertenece el control de servidor.
Visible	Obtiene o establece un valor que indica si un control de servidor se debe procesar como interfaz de usuario en la página.

Clase WebControl

Las propiedades de la clase **WebControl** son:

Propiedad	Comentario
AccessKey	Obtiene o establece la clave de acceso (la letra subrayada) que permite desplazarse rápidamente al control de servidor Web.
BackColor	Obtiene o establece el color de fondo del control de servidor Web.
BorderColor	Obtiene o establece el color de borde del control Web.
BorderStyle	Obtiene o establece el estilo del borde del control de servidor Web. Opciones: NotSet No se ha establecido el estilo de borde. None Sin borde Dotted Borde de línea punteada. Dashed Borde de línea discontinua. Solid Borde de línea sólida. Double Borde de línea sólida doble. Groove Borde estriado para un aspecto de bajo relieve. Ridge Borde con resalte para un aspecto de borde elevado. Inset Borde con efecto de insertado para un aspecto de control de bajo relieve. Outset Borde de alto relieve para un aspecto de control elevado.
BorderWidth	Obtiene o establece el ancho del borde del control de servidor Web.
Enabled	Obtiene o establece un valor que indica si el control de servidor Web está habilitado.
Font	Obtiene las propiedades de fuente asociadas al control de servidor Web.
ForeColor	Obtiene o establece el color de primer plano (normalmente el color del texto) del control de servidor Web.
Height	Obtiene o establece el alto del control de servidor Web.
TabIndex	Obtiene o establece el índice de tabulación del control de servidor Web.
ToolTip	Obtiene o establece el texto que se muestra cuando el puntero del ratón se desplaza sobre el control de servidor Web.
Width	Obtiene o establece el ancho del control de servidor Web.

Etiquetas (Label)

Las etiquetas permiten añadir texto a nuestras páginas Web que no va a poder ser editado, ni modificado por el usuario. Generalmente se utilizan para informar sobre lo que debe introducir el usuario o para mostrar resultados. Su propiedad más importante es **Text**, propiedad que nos permitirá leer el texto escrito en ella o escribir algún resultado.

Tamaño

Cuadro de Texto (TextBox)

Los cuadros de texto son controles que permiten la introducción de información por parte del usuario. Podemos representar con este mismo control, **cuadros de texto de una única línea**, **múltilínea** (textarea) y **cuadros de contraseña**. Sus propiedades son las siguientes:

Propiedad	Comentario
AutoPostBack	Obtiene o establece un valor que indica si se producirá una devolución automática al servidor siempre que el usuario cambie el texto del control TextBox y salga del control con el tabulador.
Columns	Obtiene o establece el ancho de pantalla del cuadro de texto en caracteres.
MaxLength	Obtiene o establece el número máximo de caracteres que se permiten en el cuadro de texto.
ReadOnly	Obtiene o establece un valor que indica si se puede cambiar el contenido del control TextBox.
Rows	Obtiene o establece el número de filas que se muestran en un cuadro de texto multilínea.
Text	Obtiene o establece el contenido de texto del control TextBox.
TextMode	Obtiene o establece el modo de comportamiento (una sola línea, multilínea o con contraseña) del control TextBox. Se pueden emplear los valores de la enumeración TextBoxMode: MultiLine Modo de entrada multilínea. Password Modo de entrada con contraseña. SingleLine Modo de entrada de una línea.
Wrap	Obtiene o establece un valor que indica si el contenido de texto se ajusta dentro del cuadro de texto multilínea. (Valores posibles: True y false)

El método principal del cuadro de texto:

Método	Comentario
OnTextChanged	Provoca el evento TextChanged. Esto permite controlar directamente el evento.

Y su evento correspondiente es:

Evento	Comentario
TextChanged	Se produce cuando el contenido del cuadro de texto cambia entre cada envío al servidor.

Una nota a tener en cuenta es que el estado de página no se guarda en los **cuadros de contraseña** por razones de seguridad. En los de texto de una única línea y de múltiples líneas si se guarda el estado.

Si queremos que la página sea enviada al servidor al modificar el texto (en el cuadro de texto) y quitar el foco de éste, cuando presionemos la tecla TAB habrá que poner el valor **True** en la propiedad **AutoPostBack**.

Botones (Button)

Limpiar

Este grupo lo forman tres controles, el botón (propriamente dicho **Button**, el botón de imagen **ImageButton** y el botón de enlace **LinkButton**.

Al pulsar el botón se enviará la página al servidor para ser tratada. Podemos contar con las siguientes propiedades, métodos y eventos, que son comunes a los tres tipos de botones.

Las propiedades principales de un control **Button** son las siguientes:

Propiedad	Comentario
CausesValidation	Obtiene o establece un valor que indica si se realiza la validación cuando se hace clic en el control Button. Por defecto True
CommandArgument	Obtiene o establece un parámetro opcional que se pasa al evento Command junto con la propiedad CommandName asociada.
CommandName	Obtiene o establece el nombre de comando asociado al control Button que se pasa al evento Command.
Text	Obtiene o establece el título de texto que se muestra en el control Button.

Los métodos más representativos de la clase **Button** son:

Método	Comentario
OnClick	Provoca el evento Click del control Button.
OnCommand	Provoca el evento Command del control Button.

Los eventos correspondientes a la clase **Button**:

Evento	Comentario
Click	Se produce cuando se hace clic en el control Button.
Command	Similar al anterior, pero además con este evento se pasan los valores de las propiedades CommandName y CommandArgument

Con el evento **Command** podemos aprovecharnos de las propiedades **CommandName** y **CommandArgument** a las que podemos acceder desde el evento a través del parámetro e que es de tipo **CommandEventArgs**.

Botón de Imagen (ImageButton)

Algunas propiedades específicas de este tipo de control son:

Propiedad	Comentario
AlternativeText	Obtiene o establece el texto mostrado cuando el navegador no soporta imágenes.
ImageAlign	Especifica uno de los siguientes valores: AbsBottom, AbsMiddle, Baseline, Bottom, Left, Middle, NotSet, Right, TextTop y Top.
ImageUrl	Especifica el URL de la imagen a utilizar para el botón

Botón de Enlace (LinkButton)

Esta clase representa un control muy similar, en apariencia, al control de la clase **Hyperlink** pero con la funcionalidad de un botón de comando. Es decir, al pulsarlo no se provoca un salto a un hipervínculo, sino la invocación de un método de servidor de respuesta al evento **Click**.

Los **LinkButton** no tienen ninguna propiedad específica.

Clase Image.

Representa un control que permite mostrar una imagen en una página Web. Las propiedades más interesantes son:

- **ImageURL**: contiene la URL de la imagen.
- **ImageAlign**: permite alinear la imagen respecto al resto de elementos de la página Web en la que está.
- **AlternateText**: permite indicar un texto a mostrar cuando la imagen no esté disponible (por el motivo que sea).

Clase Panel

Esta clase representa un control que sirve de contenedor para otros controles.

Para añadir un Panel a una página Web simplemente hay que arrastrarlo desde la Caja de herramientas y dimensionarlo. A partir de ahí se le pueden ir añadiendo controles.

Clase PlaceHolder

Esta clase representa un control similar a **Panel**, con la diferencia de que un **PlaceHolder** puede contener controles (generados dinámicamente) pero no es visible. Es sólo un contenedor.

Botones de Opción (RadioButton)

Trabajador

|

Un botón de opción es un control que se encuentra representado por un círculo relleno en el interior de una circunferencia en la caja de herramientas.

Cada botón de opción es independiente de los demás, ya que cada uno de ellos tiene su propio nombre (**propiedad ID**). El número de opciones representadas de esta forma puede ser cualquiera, y de ellas el usuario sólo puede seleccionar una cada vez.

Si deseamos crear varios grupos de botones de opción, lo que tenemos que hacer es utilizar la propiedad **GroupName** para agrupar los botones de opción de cada grupo.

Propiedades:

Propiedad	Comentario
AutoPostBack	Obtiene o establece un valor que indica si el estado del control RadioButton se devuelve automáticamente al servidor cuando se hace clic en él.
Checked	Obtiene o establece un valor que indica si el control RadioButton está activado.
GroupName	Obtiene o establece el nombre del grupo al que pertenece el botón de opción.
Text	Obtiene o establece la etiqueta de texto asociada al control RadioButton.
TextAlign	Obtiene o establece la alineación de la etiqueta de texto asociada al control RadioButton. Los valores pueden ser Left y Right (el predeterminado es right)

Métodos:

Método	Comentario
OnCheckedChanged	Provoca el evento CheckedChanged del control RadioButton. Esto permite controlar directamente el evento.

Eventos:

Evento	Comentario
CheckedChanged	Se produce cuando el valor de la propiedad Checked cambia de un envío a otro envío en el servidor.

Uno Dos Tres

Lista de Botones de Opción (RadioButtonList)

Representa un control de lista que encapsula un grupo de controles de botón de opción. Este control proviene de un control base **ListControl** y, por consiguiente, funciona como los controles de servidor Web **ListBox**, **DropDownList**, **BulletedList** y **CheckBoxList**, que veremos. Por esta razón, muchos de los procedimientos utilizados al trabajar con el control **RadioButtonList** son los mismos que se utilizan con otros controles de lista de servidor Web.

Cada tipo de control tiene sus ventajas. Los controles **RadioButton** individuales ofrecen más control sobre el diseño del grupo de botones de opción. Por ejemplo, puede incluir texto (texto que no pertenezca a botones de opción) entre los botones de opción.

El control **RadioButtonList** no permite insertar texto entre los botones, pero es mucho más fácil de utilizar si se desea enlazar los botones a un origen de datos. También resulta algo más fácil escribir código que determine qué botón se ha seleccionado.

Comparándolos con el control **RadioButton**, los elementos de lista del control se agrupan automáticamente.

El control **RadioButtonList** proporciona a los desarrolladores de páginas un grupo de botones de opción de selección única que se pueden generar dinámicamente mediante el enlace de datos. Contiene una colección **Items** con miembros que se corresponden con elementos individuales de la lista. Para determinar cuál es el elemento seleccionado, pruebe la propiedad **SelectedItem** de la lista.

Puede especificar la representación de la lista con las propiedades **RepeatLayout** y **RepeatDirection**. Si **RepeatLayout** se establece en **RepeatLayout.Table** (el valor predeterminado), la lista se representa en una tabla. Si se establece en **RepeatLayout.Flow**, la lista se representa sin estructura con forma de tabla. De forma predeterminada, **RepeatDirection** se establece en **RepeatDirection.Vertical**. Si establece esta propiedad en **RepeatDirection.Horizontal**, la lista se representa en formato horizontal.

Otras Propiedades son las siguientes:

Propiedad	Comentario
AutoPostBack	Obtiene o establece un valor que indica si el estado del control RadioButtonList se devuelve automáticamente al servidor cuando se hace clic en él.
CellPadding	Obtiene o establece la distancia (en píxeles) entre el borde y el contenido de la celda de la tabla.
CellSpacing	Obtiene o establece la distancia (en píxeles) entre celdas contiguas.
DataMember	Obtiene o establece la tabla específica de DataSource para enlazarla al control.
DataSource	Obtiene o establece el origen de los datos que rellenan los elementos del control de lista.
DataTextField	Obtiene o establece el campo del origen de datos que proporciona el contenido de texto de los elementos de lista.
DataTextFormatString	Obtiene o establece la cadena de formato que sirve para controlar cómo se muestran los datos enlazados al control de lista.
DataValueField	Obtiene o establece el campo del origen de datos que proporciona el valor de cada elemento de lista.
Items	Obtiene la colección de elementos del control de lista.
RepeatColumns	Obtiene o establece el número de columnas que se muestran en el control RadioButtonList.
RepeatDirection	Obtiene o establece la dirección en la que se muestran los botones de radio del grupo. Los valores pueden ser Horizontal o Vertical
RepeatLayout	Obtiene o establece la disposición de los botones de radio dentro del grupo. Valores posibles: Table (por defecto) y Flow
SelectedIndex	Obtiene o establece el índice ordinal inferior de los elementos seleccionados en la lista.
SelectedItem	Obtiene el elemento seleccionado con el índice inferior en el control de lista.
SelectedValue	Obtiene el valor del elemento seleccionado en el control de lista o selecciona el elemento en el control de lista que contiene el valor especificado.
TextAlign	Obtiene o establece la alineación del texto de los botones de radio dentro del grupo. Valores: Left y Right

Método:

Métodos	Comentario
OnSelectedIndexChanged	Provoca el evento SelectedIndexChanged. Esto permite incluir un controlador personalizado para el evento.

Evento:

Eventos	Comentario
SelectedIndexChanged	Se produce cuando la selección del control de lista cambia entre cada envío al servidor.

Métodos de la Colección **Items** de una Lista:

Nombre	Descripción
Add	Agrega un objeto al final de la lista.
AddRange	Agrega los elementos de la colección especificada al final de la lista.
Clear	Elimina todos los elementos de la lista.
Contains	Determina si un elemento se encuentra en la lista. Devuelve un bool, true si se encuentra, false si no se encuentra.
CopyTo	Sobrecargado. Copia el objeto lista o una parte del mismo en una matriz.
Equals	Sobrecargado. Determina si dos instancias de Object son iguales. (Se hereda de Object).
FindByName	Busca un elemento de la lista por texto.
FindByValue	Busca un elemento de la lista por Valor.
IndexOf	Sobrecargado. Devuelve el índice de base cero de la primera aparición de un valor en la lista.
Insert	Inserta un elemento en la lista, en el índice especificado.
Remove	Quita la primera aparición de un objeto específico de la lista.
RemoveAt	Quita el elemento en el índice especificado de la lista.
ToString	Devuelve una clase String que representa la clase Object actual. (Se hereda de Object).

Lista de Casillas de Verificación (CheckList)

Deporte Música Informática Viajes

Crea un grupo de casillas de verificación de selección múltiple. Las propiedades, métodos y eventos son los mismos que los de la lista de botones de opción (RadioButtonList). La única diferencia es que son, como ya se ha dicho, de múltiple selección, es decir, que podemos seleccionar todos los elementos que queramos a la vez. Y que una pulsación selecciona la casilla de verificación y otra pulsación deseleccionaría la casilla de verificación. Por lo tanto se suelen emplear sentencias condicionales para comprobar si está seleccionada o no la casilla de verificación (propiedad Selected, en vez de Checked (como era en la lista de botones de opción)). Y para recorrer la lista se emplea una sentencia repetitiva como **foreach** o **for**:

```
lblresultado2.Text = "";  
  
foreach (ListItem hobbie in chkhobbies.Items)  
{  
    if (hobbie.Selected == true)  
    {  
        lblresultado2.Text += hobbie.Text;  
    }  
}
```

DROPODOWNLIST



La **dropdownlist** es una lista desplegable, que puede ser empleada para seleccionar un elemento dentro de un conjunto. Los elementos de una **dropdownlist** son de tipo **ListItem** y serán introducidos a través de la colección de **Items** de la lista. Al igual que los controles **RadioButtonList** y **CheckBoxList** cuenta con una serie de métodos para añadir, insertar, borrar, seleccionar elementos,...

Las propiedades de una **dropdownlist** son las siguientes:

Propiedad	Comentario
AutoPostBack	Obtiene o establece un valor que indica si el estado del control DropDownList se devuelve automáticamente al servidor cuando se hace clic en él.
DataMember	Obtiene o establece la tabla específica de DataSource para enlazarla al control.
DataSource	Obtiene o establece el origen de los datos que rellenan los elementos del control de lista.
DataTextField	Obtiene o establece el campo del origen de datos que proporciona el contenido de texto de los elementos de lista.
DataTextFormatString	Obtiene o establece la cadena de formato que sirve para controlar cómo se muestran los datos enlazados al control de lista.
DataValueField	Obtiene o establece el campo del origen de datos que proporciona el valor de cada elemento de lista.
Items	Obtiene la colección de elementos del control de lista.
SelectedIndex	Obtiene o establece el índice ordinal inferior de los elementos seleccionados en la lista.
SelectedItem	Obtiene el elemento seleccionado con el índice inferior en el control de lista.
SelectedValue	Obtiene el valor del elemento seleccionado en el control de lista o selecciona el elemento en el control de lista que contiene el valor especificado.

Método

Métodos	Comentario
OnSelectedIndexChanged	Provoca el evento SelectedIndexChanged. Esto permite incluir un controlador personalizado para el evento.

Evento

Eventos	Comentario
SelectedIndexChanged	Se produce cuando la selección del control de lista cambia entre cada envío al servidor.

LISTBOX



La **lista** es un control en el que el usuario puede seleccionar uno o varios elementos dentro de un conjunto. La propiedad **SelectionMode** es la que determinará si se puede seleccionar 1 o varios elementos de la lista. Al igual que el control **DropDownList** y **RadioButtonList** o **CheckBoxList** cuenta con una serie de métodos para introducir, borrar, seleccionar elementos,...

Las propiedades de la lista son las siguientes:

Propiedad	Comentario
AutoPostBack	Obtiene o establece un valor que indica si el estado del control ListBox se devuelve automáticamente al servidor cuando se hace clic en él.
DataMember	Obtiene o establece la tabla específica de DataSource para enlazarla al control.
DataSource	Obtiene o establece el origen de los datos que rellenan los elementos del control de lista.
DataTextField	Obtiene o establece el campo del origen de datos que proporciona el contenido de texto de los elementos de lista.
DataTextFormatString	Obtiene o establece la cadena de formato que sirve para controlar cómo se muestran los datos enlazados al control de lista.
DataValueField	Obtiene o establece el campo del origen de datos que proporciona el valor de cada elemento de lista.
Items	Obtiene la colección de elementos del control de lista.
Rows	Indica el número de filas a mostrar en el cuadro de lista (por defecto 4).
SelectedIndex	Obtiene o establece el índice ordinal inferior de los elementos seleccionados en la lista.
SelectedItem	Obtiene el elemento seleccionado con el índice inferior en el control de lista.
SelectedValue	Obtiene el valor del elemento seleccionado en el control de lista o selecciona el elemento en el control de lista que contiene el valor especificado.
SelectionMode	Determina si el usuario puede seleccionar más de un elemento de lista a la vez. Valores: Multiple y Single (por defecto)

Métodos

Métodos	Comentario
OnSelectedIndexChanged	Provoca el evento SelectedIndexChanged. Esto permite incluir un controlador personalizado para el evento.

Evento

Eventos	Comentario
SelectedIndexChanged	Se produce cuando la selección del control de lista cambia entre cada envío al servidor.

Clase Calendar

Representa un control que muestra un calendario de mes, que permite al usuario seleccionar fechas y moverse al anterior y/o posterior mes.

septiembre de 2013						
lun	mar	mié	jue	vie	sáb	dom
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	1	2	3	4	5	6

Por defecto, un control de tipo **Calendar** muestra los días del mes, cabeceras para los días de la semana, un título con el nombre del mes e hipervínculos para moverse al mes anterior y posterior.

Las propiedades más interesantes de un control **Calendar** son:

- **SelectedDate**: esta propiedad indica la fecha seleccionada. Un objeto de esta propiedad es de la clase **DateTime** y tiene por tanto todas sus propiedades (Day, Month, Year, Date, etc...).
- **SelectedDates**: esta propiedad contiene una colección con todos los días (o semanas..., depende de **SelectionMode**) seleccionados.
- **SelectionMode**: permite indicar si la selección será de un día, una semana o un mes.
- **FirstDayofWeek**: esta propiedad permite indicar qué día se pondrá en la primera columna de la semana (lunes, domingo...). Al seleccionar un día (o una semana...) en el control Calendar se provoca un evento
- **SelectionChanged**, en el método asociado. Es donde se ha de meter el código de respuesta al evento.

El siguiente código muestra cómo hacer que se visualice la información sobre un día seleccionado en cuatro cajas de texto: la fecha, el año, el mes y el día.

```
protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
    txtfecha.Text = Calendar1.SelectedDate.ToString();
    txtaño.Text = Calendar1.SelectedDate.Year.ToString();
    txtmes.Text = Calendar1.SelectedDate.Month.ToString();
    txtdia.Text = Calendar1.SelectedDate.Day.ToString();
}
```