
Controles de Validación

RequiredFieldValidator
CompareValidator
RangeValidator
RegularExpressionValidator
CustomValidator

ValidationSummary

Controles de Validación

Los controles de validación siempre validan el control de entrada asociado en el servidor. Los controles de validación también tienen una implementación completa en el cliente que permite a los exploradores compatibles con DHTML (como Internet Explorer versión 4.0 o posterior) realizar la validación en el cliente. La validación en el cliente mejora el proceso de validación, ya que se comprueba la entrada del usuario antes de enviarla al servidor. De este modo se pueden detectar los errores en el cliente antes de enviar el formulario y se evita el recorrido de ida y vuelta de la información necesaria para la validación en el servidor.

Se pueden utilizar varios controles de validación con un control de entrada individual para validar diferentes criterios.

Si un control de entrada no supera la validación, se mostrará en el control de validación el texto

Es posible mostrar un texto diferente para el control de validación y el control **ValidationSummary**. Utilice la propiedad **Text** para especificar el texto que se ha de mostrar en el control de validación. El texto especificado por la propiedad **ErrorMessage** se utiliza siempre en el control **ValidationSummary**.

Control de validación	Descripción
<code>CompareValidator</code>	Compara el valor de una entrada con el de otra o un valor fijo.
<code>CustomValidator</code>	Permite implementar un método cualquiera que maneje la validación del valor introducido.
<code>RangeValidator</code>	Comprueba que la entrada esté entre dos valores dados.
<code>RegularExpressionValidator</code>	Valida el valor de acuerdo a un patrón establecido como una expresión regular.
<code>RequiredFieldValidator</code>	Hace que un valor de entrada sea obligatorio.

Las propiedades que se aplican a todos los controles de validación son las siguientes:

ControlToValidate	Identificativo del control de salida que evaluará el control de validación. Si no es un identificador legítimo, se producirá una excepción.
Display	Modo en que se muestra el control de validación especificado. Esta propiedad puede ser uno de los siguientes valores: None : el control de validación jamás se muestra en línea. Utilice esta opción cuando desee mostrar el mensaje de error sólo en un control ValidationSummary . Static : el control de validación muestra un mensaje de error si se produce un error en la validación. Se asigna un espacio en la página Web para el mensaje de error incluso si el control de entrada supera la validación. No cambia el diseño de la página cuando el control de validación muestra su

	<p>mensaje de error. Como el diseño de página es estático, si hay varios controles de validación para el mismo control de entrada, éstos deberán ocupar distintas ubicaciones en la página.</p> <p>Dynamic: el control de validación muestra un mensaje de error si se produce un error en la validación. El espacio para el mensaje de error se asigna dinámicamente en la página cuando se produce un error en la validación. De este modo, varios controles de validación pueden compartir la misma ubicación física en la página.</p>
	<p>Nota</p> <p>Como el espacio para el control de validación se crea dinámicamente, cambia el diseño físico de la página. Para impedir que cambie el diseño de la página cuando aparece un control de validación, el elemento HTML que contiene el control de validación debe ser lo suficientemente grande para que quepa el control de validación de mayor tamaño.</p>
EnableClientScript	Indica si está habilitada la validación en el cliente. Para deshabilitar la validación en el cliente en los exploradores que admitan esta función, establezca el valor de la propiedad EnableClientScript en false .
Enabled	Indica si está habilitado el control de validación. Para impedir que el control de validación valide un control de entrada, establezca el valor de esta propiedad en false .
ErrorMessage	<p>Mensaje de error que se va a mostrar en el control ValidationSummary si se produce un error de validación. Si no está establecido el valor de la propiedad Text del control de validación, también se muestra este texto en el control de validación cuando se produce un error de validación. Se utiliza normalmente la propiedad ErrorMessage con el fin de proporcionar diferentes mensajes para el control de validación y el control ValidationSummary.</p>
	<p>Nota</p> <p>Esta propiedad no convierte caracteres especiales en entidades HTML. Por ejemplo, el carácter menor que (<) no se convierte en &lt;. Esto permite incrustar elementos HTML, como un elemento , en el valor de esta propiedad.</p>
ForeColor	Especifica el color en el que se va a mostrar el mensaje en línea cuando se produce un error de validación.
IsValid	Indica si el control de entrada especificado por la propiedad ControlToValidate se determina como válido.
SetFocusOnError	Indica si el foco está establecido en el control especificado por la propiedad ControlToValidate cuando se produce un error de validación.
Text	Si esta establecida esta propiedad, se muestra este mensaje en el control de validación cuando se produce un error en la validación. Si no está establecida esta propiedad, el texto especificado en la propiedad ErrorMessage se muestra en el control.
ValidationGroup	Especifica el nombre del grupo de validación al que pertenece este control de validación.

RequiredFieldValidator

Este control permite comprobar si un control tiene un valor.

Ejemplo:

No deja realizar el envío de la página al servidor si no se han introducido el nombre y los apellidos.

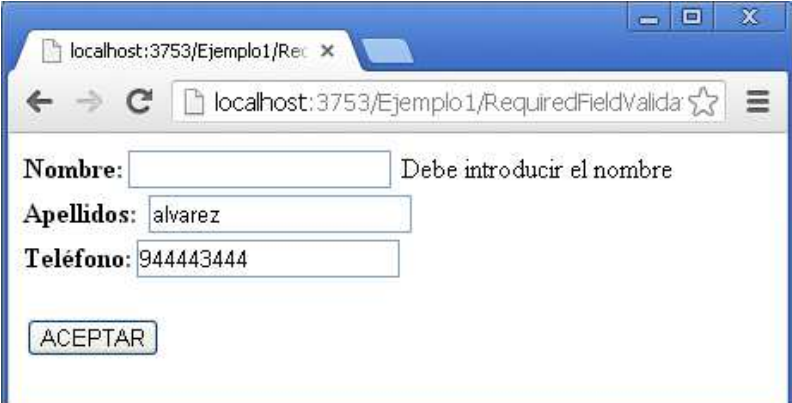
RequiredFieldValidator.aspx

```
<form id="form1" runat="server">
  <div>
    <b>Nombre:</b><asp:TextBox ID="txtnombre" runat="server"></asp:TextBox>
    <asp:RequiredFieldValidator ID="rfvnombre" runat="server"
      ErrorMessage="Debe introducir el nombre"
      ControlToValidate="txtnombre"
      Display="Dynamic"
      EnableClientScript="False"></asp:RequiredFieldValidator>
    <br />
    <b>Apellidos:</b><asp:TextBox ID="txtapellidos" runat="server"></asp:TextBox>
    <asp:RequiredFieldValidator ID="rfvapellidos" runat="server"
      ErrorMessage="Debe introducir los apellidos"
      ControlToValidate="txtapellidos"
      Display="Dynamic"
      EnableClientScript="False"></asp:RequiredFieldValidator>
    <br />
    <b>Teléfono:</b><asp:TextBox ID="txttfno" runat="server"></asp:TextBox>
    <br />
    <br />
    <asp:Button ID="Button1" runat="server" onclick="Button1_Click"
      Text="ACEPTAR" />
  </div>
</form>
```

RequiredFieldValidator.aspx.cs

```
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 Button1_Click(object sender, EventArgs e)
    {
        if (rfvnombre.IsValid)
        {
            Response.Write("PRUEBA");
        }
    }
}
```



CompareValidator

Este control permite comparar los datos introducidos entre un campo de un formulario y un valor dado (que puede ser un valor fijo o el valor introducido en otro control).

Ejemplo Comparación de campo con algún valor:

La edad del usuario debe ser igual o mayor que 18.

CompareValidator1.aspx

```
<form id="form1" runat="server">
    <div>
        <b>Nombre:</b><asp:TextBox ID="txtnombre" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvnombre" runat="server"
            ErrorMessage="Debe introducir el nombre"
            ControlToValidate="txtnombre"
            Display="Dynamic"></asp:RequiredFieldValidator>
        <br />
        <b>Apellidos:</b><asp:TextBox ID="txtapellidos"
            runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvapellidos" runat="server"
            ErrorMessage="Debe introducir los apellidos" ControlToValidate="txtapellidos"
            Display="Dynamic"></asp:RequiredFieldValidator>
        <br />
        <b>Edad:</b><asp:TextBox ID="txtedad" runat="server"></asp:TextBox>
        <asp:CompareValidator ID="cvedad" runat="server"
            ErrorMessage="Debe ser mayor de edad" ControlToValidate="txtedad"
            Display="Dynamic" Operator="GreaterThanEqual"
            ValueToCompare="18"></asp:CompareValidator>
        <br />
        <br />
        <asp:Button ID="Button1" runat="server" onclick="Button1_Click"
            Text="ACEPTAR" />
    </div>
</form>
```

CompareValidator1.aspx.cs

```
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 Button1_Click(object sender, EventArgs e)
    {
        if (rfvnombre.IsValid)
        {
            Response.Write("PRUEBA");
        }
    }
}
```



localhost:3753/Ejemplo1/CompareValidator2.aspx

Nombre: ana

Apellidos: alvarez

Edad: 12 Debe ser mayor de edad

ACEPTAR

Ejemplo Comparación de Campos:

La fecha de nacimiento del usuario debe ser anterior a la fecha del sistema mostrada en el cuadro de texto txtfechasistema.

CompareValidator2.aspx

```
<form id="form1" runat="server">
    <div>
        <b>Nombre:</b><asp:TextBox ID="txtnombre" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvnombre" runat="server"
            ErrorMessage="Debe introducir el nombre"
            ControlToValidate="txtnombre"
            Display="Dynamic"></asp:RequiredFieldValidator>
        <asp:TextBox ID="txtfechasistema" runat="server"></asp:TextBox>
        <br />
        <b>Apellidos: </b><asp:TextBox ID="txtapellidos"
            runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvapellidos" runat="server"
            ErrorMessage="Debe introducir los apellidos"
            ControlToValidate="txtapellidos"
            Display="Dynamic"></asp:RequiredFieldValidator>
        <br />
        <b>Fecha Nacimiento:</b><asp:TextBox ID="txtfecha"
            runat="server"></asp:TextBox>
        <asp:CompareValidator ID="cvfecha" runat="server"
            ErrorMessage="Debe ser anterior a la fecha de hoy"
            ControlToValidate="txtfecha"
            Display="Dynamic" Operator="LessThanEqual"
            ControlToCompare="txtfechasistema"></asp:CompareValidator><br />
        <br />
        <asp:Button ID="Button1" runat="server" onclick="Button1_Click"
            Text="ACEPTAR" />
    </div>
</form>
```

CompareValidator2.aspx.cs

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

public partial class Default2 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!Page.IsPostBack) {
            txtfechasistema.Text = DateTime.Today.ToLongDateString();
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
    }
```

```
Response.Write("PRUEBA")
;
}
```

localhost:3753/Ejemplo1/CompareValidator1.aspx

Nombre: Fecha del Sistema:

Apellidos:

Fecha Nacimiento: Debe ser anterior a la fecha de hoy

RangeValidator

El control **RangeValidator** permite comprobar si el valor de un control se encuentra entre un valor mínimo y otro máximo.

Ejemplo:

La edad del usuario debe estar comprendida entre 18 y 65 años.

RangeValidator.aspx

```
<form id="form1" runat="server">

    <div>
        <b>Nombre:</b><asp:TextBox ID="txtnombre" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvnombre" runat="server"
            ErrorMessage="Debe introducir el nombre"
            ControlToValidate="txtnombre"
            Display="Dynamic"></asp:RequiredFieldValidator>
        <br />
        <b>Apellidos: </b><asp:TextBox ID="txtapellidos"
            runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvapellidos" runat="server"
            ErrorMessage="Debe introducir los apellidos"
            ControlToValidate="txtapellidos"
            Display="Dynamic"></asp:RequiredFieldValidator>
        <br />
        <b>Edad:</b><asp:TextBox ID="txttfno" runat="server"></asp:TextBox>
        <asp:CompareValidator ID="cvedad" runat="server"
            ErrorMessage="Debe ser mayor de edad" ControlToValidate="txttfno"
            Display="Dynamic" Operator="GreaterThanEqual"
            ValueToCompare="18"></asp:CompareValidator>
        <asp:RangeValidator ID="RangeValidator1" runat="server"
            ControlToValidate="txttfno" Display="Dynamic"
            ErrorMessage="La edad debe estar comprendiida entre 18
            años y 65 años."
            MaximumValue="65" MinimumValue="18"></asp:RangeValidator>
        <br />
        <br />
        <asp:Button ID="Button1" runat="server" onclick="Button1_Click"
            Text="ACEPTAR" />
    </div>
</form>
```

```
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 Button1_Click(object sender, EventArgs e)
    {
        if (Page.IsValid)
        {
            Response.Write("PRUEBA");
        }
    }
}
```



RegularExpressionValidator

El control **RegularExpressionValidator** permite comparar el valor introducido en un campo del formulario con una expresión regular.

Las expresiones Regulares son cadenas de caracteres formadas por elementos que representan los valores permitidos en el control de validación. Algunos de los elementos utilizados son:

- \t — Representa un tabulador.
- \r — Representa el "retorno de carro" o "regreso al inicio" o sea el lugar en que la línea vuelve a iniciar.
- \n — Representa la "nueva línea" el carácter por medio del cual una línea da inicio. Es necesario recordar que en [Windows](#) es necesaria una combinación de \r\n para comenzar una nueva línea, mientras que en [Unix](#) solamente se usa \n y en [Mac OS](#) clásico se usa solamente \r.
- \a — Representa una "campana" o "beep" que se produce al imprimir este carácter.
- \e — Representa la tecla "Esc" o "Escape"
- \f — Representa un salto de página
- \v — Representa un tabulador vertical
- \x — Se utiliza para representar caracteres [ASCII](#) o ANSI si conoce su código. De esta forma, si se busca el símbolo de derechos de autor y la fuente en la que se busca utiliza el conjunto de caracteres Latin-1 es posible encontrarlo utilizando "\xA9".
- \u — Se utiliza para representar caracteres [Unicode](#) si se conoce su código. "\u00A2" representa el símbolo de centavos. No todos los motores de Expresiones Regulares soportan Unicode. El .Net Framework lo hace, pero el EditPad Pro no, por ejemplo.
- \d — Representa un dígito del 0 al 9.

- \w — Representa cualquier carácter alfanumérico.
- \s — Representa un espacio en blanco.
- \D — Representa cualquier carácter que no sea un dígito del 0 al 9.
- \W — Representa cualquier carácter no alfanumérico.
- \S — Representa cualquier carácter que no sea un espacio en blanco.
- \A — Representa el inicio de la cadena. No un carácter sino una posición.
- \Z — Representa el final de la cadena. No un carácter sino una posición.
- \b — Marca el inicio y el final de una palabra.
- \B — Marca la posición entre dos caracteres alfanuméricos o dos no-alfanuméricos.

Repetitivas

- {númeromínimo,númeromáximo} Indica que el símbolo anterior a los corchetes se repetirá un número de veces comprendido entre el mínimo y el máximo expresado entre corchetes.

\d{2,3} Pueden aparecer 2 ó 3 dígitos.

\d{3} Aparecen 3 dígitos

Alternativas

- Una barra vertical separa las alternativas. Por ejemplo, "marrón|castaño" casa con *marrón* o *castaño*.

Cuantificativas

- Un cuantificador tras un carácter especifica la frecuencia con la que éste puede ocurrir. Los cuantificadores más comunes son +, ? y *:

+

- El signo más indica que el carácter al que sigue debe aparecer al menos una vez. Por ejemplo, "ho+la" describe el conjunto infinito *hola*, *hoola*, *hooola*, *hooola*, etcétera.

?

- El signo de interrogación indica que el carácter al que sigue puede aparecer como mucho una vez. Por ejemplo, "ob?scuro" casa con *oscuro* y *obscuro*.

*

- El asterisco indica que el carácter que lo precede puede aparecer cero, una, o más veces. Por ejemplo, "0*42" casa con *42*, *042*, *0042*, *00042*, etcétera.

Agrupación

- Los paréntesis pueden usarse para definir el ámbito y precedencia de los demás operadores. Por ejemplo, "(p|m)adre" es lo mismo que "padre|madre", y "(des)?amor" casa con *amor* y con *desamor*.

Nombre: DE 8-25 caracteres. **ValidationExpression**="\w{8,25}"

Telefono: **ValidationExpression**="\d{3}[\.\]?\d{2}[\.\]?\d{2}[\.\]?\d{2}"

Email: **ValidationExpression**="\S+@\S+\.\S{2,3}"

Ejemplo:

El nombre del usuario debe contener entre 3 y 20 caracteres.

Los apellidos del usuario deben contener entre 2 y 50 caracteres.

El correo electrónico debe tener el carácter @ y el . con el dominio de 2 ó 3 caracteres.

RegularExpressionValidator.aspx
--

```
<form id="form1" runat="server">
  <div>
    <b>Nombre:</b><asp:TextBox ID="txtnombre" runat="server"></asp:TextBox>
    <asp:RequiredFieldValidator ID="rfvnombre" runat="server"
      ErrorMessage="Debe introducir el nombre"
      ControlToValidate="txtnombre"
      Display="Dynamic"></asp:RequiredFieldValidator>
    <asp:CompareValidator ID="CompareValidator1" runat="server"
```

```

        ControlToValidate="txtnombre" ErrorMessage="Debe ser una cadena de
caracteres"
        Operator="DataTypeCheck"></asp:CompareValidator>
<asp:RegularExpressionValidator ID="RegularExpressionValidator1"
runat="server"
ErrorMessage="Debe ser una cadena de caracteres entre 3 y 20
caracteres."
ControlToValidate="txtnombre" ValidationExpression="\w{3,20}">
</asp:RegularExpressionValidator>
<br />
<b>Apellidos: </b><asp:TextBox ID="txtapellidos"
runat="server"></asp:TextBox>
<asp:RequiredFieldValidator ID="rfvapellidos" runat="server"
ErrorMessage="Debe introducir los apellidos"
ControlToValidate="txtapellidos"
Display="Dynamic"></asp:RequiredFieldValidator>
<asp:RegularExpressionValidator ID="RegularExpressionValidator2"
runat="server"
ErrorMessage="Debe ser una cadena de caracteres entre 2 y 50
caracteres."
ControlToValidate="txtapellidos" Display="Dynamic"
ValidationExpression="\w{2,50}">
</asp:RegularExpressionValidator>
<br />
<b>Edad:</b><asp:CompareValidator ID="CompareValidator2" runat="server"
ControlToValidate="txttfno" ErrorMessage="Debe ser numerico"
Operator="DataTypeCheck" Type="Integer"></asp:CompareValidator>
<asp:TextBox ID="txttfno" runat="server"></asp:TextBox>
<asp:CompareValidator ID="cvedad" runat="server"
ErrorMessage="Debe ser mayor de edad" ControlToValidate="txttfno"
Display="Dynamic" Operator="GreaterThanEqual"
ValueToCompare="18"></asp:CompareValidator>
<asp:RangeValidator ID="RangeValidator1" runat="server"
ControlToValidate="txttfno" Display="Dynamic"
ErrorMessage="La edad debe estar comprendida entre 18 años y 65
años."
MaximumValue="65" MinimumValue="18"></asp:RangeValidator><br />
<b>Correo: </b><asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
<asp:RegularExpressionValidator ID="RegularExpressionValidator3"
runat="server"
ControlToValidate="TextBox1" Display="Dynamic"
ErrorMessage="Debe ser un correo electrónico"
ValidationExpression="\S+@\S+\.\S{2,3}">
</asp:RegularExpressionValidator>

</div>
<asp:Button ID="Button1" runat="server" onclick="Button1_Click"
Text="Aceptar" />
</form>

```

CustomValidator

Si los controles estándar de validación no son suficientes, es posible especificar controles personalizados de validación. Esto se realizará llamando a una función definida por el usuario que será la que realice la validación. Esta función puede estar definida en el cliente o en el servidor. Este control de validación ofrece las siguientes propiedades destacadas:

ClientValidationFunction: en esta propiedad indicaremos el nombre de la función del lado del cliente que realizará la validación, en este caso la función será una función definida en un lenguaje JavaScript o VBScript.

OnServerValidate: en esta propiedad debemos indicar el nombre de la función de servidor que se va a encargar de realizar la validación, esta función se ejecutará cuando se produzca el evento **ServerValidate**. Lo mejor es ofrecer las funciones de validación en el cliente y en el servidor, así nos aseguraremos que la validación siempre se realizará con independencia del tipo de navegador que ha cargado la página ASP.NET correspondiente.

En los dos casos la función de validación va a recibir **dos parámetros**:

uno de la clase **Object**, que representa al control de la clase CustomValidator, y otro objeto de la clase **ServerValidateEventArgs**, que se va a utilizar para indicar si la validación ha sido correcta o no. El objeto de clase **ServerValidateEventArgs** posee la propiedad **Value** para obtener el valor del control que se desea validar y la propiedad **IsValid**, en la que se indicará el resultado de la validación.

Ejemplo Validación en el cliente (JavaScript):

Ejemplo de un control que valida la introducción en un cuadro de texto de un valor par.

CustomValidatorCLIENTE.aspx

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

<!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>
    <script type="text/javascript" >

        function validaParCliente(source, arguments) {
            if (arguments.Value % 2 == 0)
                arguments.IsValid = true;
            else arguments.IsValid = false;
        }
    </script>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            Número par <asp:TextBox id="valor1" runat="server"></asp:TextBox>
            <asp:CustomValidator id="CustomValidator1" runat="server"
ErrorMessage="Debe introducirse un número par"
ControlToValidate="valor1" ClientValidationFunction="validaParCliente">
            </asp:CustomValidator><br />
            <asp:Button id="boton" runat="server" Text="Enviar datos" /><br />
            <asp:Label id="resultado" runat="server"></asp:Label>

        </div>
```

```

    </form>
</body>
</html>

```

Ejemplo Validación en el servidor (C#):

CustomValidatorSERVIDOR.aspx

```

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

<!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">
        <div>
            Número par <asp:TextBox id="txtnumero" runat="server"></asp:TextBox>
            <asp:CustomValidator id="CustomValidator1" runat="server"
ErrorMessage="Debe introducirse un número par"
ControlToValidate="txtnumero" onservervalidate="CustomValidator1_ServerValidate"
>
            </asp:CustomValidator><br />
            <asp:Button id="boton" runat="server" Text="Enviar datos" onclick="boton_Click"
/><br />
            <asp:Label id="lblresultado" runat="server"></asp:Label>
        </div>
    </form>
</body>

```

CustomValidatorSERVIDOR.aspx.cs

```

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

public partial class ValidacionServidor : System.Web.UI.Page
{
    protected void CustomValidator1_ServerValidate(object source,
ServerValidateEventArgs args)
    {
        long numero=Convert.ToInt64(args.Value);
        if ( numero % 2 == 0)
        {
            args.IsValid = true;
        }
        else
        {
            args.IsValid = false;
        }
    }
    protected void boton_Click(object sender, EventArgs e)
    {

```

```

        if (Page.IsValid)
        {
            lblresultado.Text = "El número par es: " + txtnumero.Text;
        }
    }
}

```

Otro Ejemplo:

Validación de introducción de ciertas cuentas de correo en un cuadro de texto.

CustomValidator.aspx

```

<form id="form1" runat="server">
    <div>
        <b>Correo: </b><asp:TextBox ID="txtcorreo" runat="server"></asp:TextBox>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator3"
runat="server"
            ControlToValidate="TextBox1" Display="Dynamic"
            ErrorMessage="Debe ser un correo electrónico"
ValidationExpression="\S+@\S+\.\S{2,3}"></asp:RegularExpressionValidator>
        <asp:CustomValidator ID="cvcorreos" runat="server"
            ControlToValidate="txtcorreo" Display="Dynamic"
            ErrorMessage="Debe ser un correo válido"
            onservervalidate="cvcorreos_ServerValidate">Debe
            ser uno de los válidos</asp:CustomValidator>
    </div>
    <asp:Button ID="Button1" runat="server" onclick="Button1_Click"
        Text="Aceptar" />
</form>

```

CustomValidator.aspx.cs

```

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

public partial class RegularExpression : System.Web.UI.Page
{
    protected void cvcorreos_ServerValidate(object source, ServerValidateEventArgs
args)
    {
        string correo = args.Value;
        if (correo == "marta@zabalburu.net" || correo == "ichueca@zabalburu.net"
||
        correo == "ibilbao@zabalburu.net" || correo == "fernando@zabalburu.net")
        {
            args.IsValid = true;
        }
        else
        {
            args.IsValid = false;
        }
    }
}

```

```
protected void Button1_Click(object sender, EventArgs e)
{
    if (Page.IsValid)
    {
        Response.Write("PRUEBA");
    }
}
}
```

Control ValidationSummary

Además de poder mostrar un mensaje de error en el lugar del control de validación con la propiedad **Text**, es posible especificar un resumen con todos los errores de validación de una página mediante el control **ValidationSummary**. Este control normalmente se emplea cuando hay muchos campos para validar.

Si se quiere emplear este control se debe especificar el mensaje a mostrar en la propiedad **Text** de los controles de validación. Hay que tener en cuenta que este mensaje de error, además de en el control **ValidationSummary**, aparecerá también en el control de validación a menos que se haya especificado también la propiedad **Text** (será ésta la que aparezca) o que se haya indicado **None** en la propiedad **Display** (no se mostrará nada en el control de validación).

```
<form id="form1" runat="server">
    <div>
        <b>Nombre:</b><asp:TextBox ID="txtnombre" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvnombre" runat="server"
            ErrorMessage="Debe introducir el nombre"
            ControlToValidate="txtnombre"
            Display="Dynamic">*</asp:RequiredFieldValidator>
        <asp:CompareValidator ID="CompareValidator1" runat="server"
            ControlToValidate="txtnombre" ErrorMessage="Debe ser una cadena de
            caracteres"
            Operator="DataTypeCheck">*</asp:CompareValidator>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator1"
            runat="server"
            ErrorMessage="El nombre debe ser una cadena de caracteres entre 3 y
            20 caracteres."
            ControlToValidate="txtnombre"
            ValidationExpression="\w{3,20}">*</asp:RegularExpressionValidator>
        <br />
        <b>Apellidos:</b><asp:TextBox ID="txtapellidos"
            runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="rfvapellidos" runat="server"
            ErrorMessage="Debe introducir los apellidos"
            ControlToValidate="txtapellidos"
            Display="Dynamic">*</asp:RequiredFieldValidator>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator2"
            runat="server"
            ErrorMessage="Los apellidos deben ser una cadena de caracteres entre
            2 y 50 caracteres."
            ControlToValidate="txtapellidos" Display="Dynamic"
            ValidationExpression="\w{2,50}">*</asp:RegularExpressionValidator>
        <br />
        <b>Edad:</b><asp:CompareValidator ID="CompareValidator2" runat="server"
            ControlToValidate="txttfno" ErrorMessage="Debe ser numerico"
```

```

        Operator="DataTypeCheck" Type="Integer"></asp:CompareValidator>
<asp:TextBox ID="txttfno" runat="server" ></asp:TextBox>
<asp:CompareValidator ID="cvedad" runat="server"
    ErrorMessage="Debe ser mayor de edad" ControlToValidate="txttfno"
    Display="Dynamic" Operator="GreaterThanEqual"
ValueToCompare="18">*</asp:CompareValidator>
<asp:RangeValidator ID="RangeValidator1" runat="server"
    ControlToValidate="txttfno" Display="Dynamic"
    ErrorMessage="La edad debe estar comprendiida entre 18 años y 65
años."
    MaximumValue="65" MinimumValue="18">*</asp:RangeValidator><br />
    <b>Correo: </b><asp:TextBox ID="TextBox1"
runat="server"></asp:TextBox>
<asp:RegularExpressionValidator ID="RegularExpressionValidator3"
runat="server"
    ControlToValidate="TextBox1" Display="Dynamic"
    ErrorMessage="Debe ser un correo electrónico"

    ValidationExpression="\S+@\S+\.\S{2,3}">*</asp:RegularExpressionVal
idator>
<asp:CustomValidator ID="cvcorreos" runat="server"
    ControlToValidate="TextBox1" Display="Dynamic"
    ErrorMessage="Debe ser uno de los correctos"
>*</asp:CustomValidator>
</div>
<asp:Button ID="Button1" runat="server"
    Text="Aceptar" />
<asp:ValidationSummary ID="ValidationSummary1" runat="server"
    HeaderText="Resumen de errores de validacion:" ShowMessageBox="True"
    ShowSummary="False" />
</form>

```

Ejemplo:

Nombre: An *

Apellidos: A *

Edad: 12 **

Correo: marta@zabalburu.net

Aceptar

Resumen de errores de validacion:

- El nombre debe ser una cadena de caracteres entre 3 y 20 caracteres.
- Los apellidos deben ser una cadena de caracteres entre 2 y 50 caracteres.
- Debe ser mayor de edad
- La edad debe estar comprendiida entre 18 años y 65 años.

Si ponemos las propiedades **ShowMessage** a true y **ShowSummary** a false, la salida será mostrada en un cuadro de mensaje, en vez de en el control de resumen:

