CONTROLES DE USUARIO

Introducción

Además de utilizar controles de servidor Web en las páginas Web ASP.NET, puede crear sus propios controles personalizados reutilizables con las mismas técnicas que para crear páginas Web ASP.NET. Estos controles se denominan controles de usuario.

Un control de usuario es un tipo de control compuesto que funciona de forma similar a la de una página Web ASP.NET: se pueden agregar controles de servidor Web y marcado a un control de usuario, así como definir propiedades y métodos para el control. A continuación, puede incrustarlos en páginas Web ASP.NET, donde actúan como una unidad.

Controles de Usuario vs Controles personalizados

En ocasiones, es posible que necesite cierta funcionalidad en un control que no está incluida en los controles de servidor Web ASP.NET integrados. En estos casos, puede crear sus propios controles. Dispone de dos opciones. Puede crear:

- **Controles de usuario**. Los controles de usuario son contenedores en los que puede colocar controles de formato y de servidor Web. A continuación puede tratar el control de usuario como una unidad y definir propiedades y métodos para el mismo.
- **Controles personalizados**. Un control personalizado es una clase escrita por un desarrollador que se deriva de Control o *WebControl*.

Los controles de usuario son mucho más fáciles de crear que los controles personalizados, ya que es posible reutilizar los ya existentes. Esto permite crear con facilidad controles con elementos de interfaz de usuario complejos.

Estructura de los controles de usuario

Un control de usuario Web ASP.NET es similar a una página Web ASP.NET completa (archivo .aspx) e incluye una página de interfaz de usuario y código. El proceso de creación del control de usuario es muy similar al proceso de creación de una página ASP.NET, sólo que al final se agregan el formato y los controles secundarios necesarios. Al igual que una página, un control

de usuario puede incluir el código necesario para manipular su contenido e incluso realizar tareas como el enlace de datos.

Un control de usuario se diferencia de una página Web ASP.NET en los siguientes aspectos:

- La extensión de nombre de archivo para el control de usuario es .ascx.
- En lugar de una directiva @ Page, el control de usuario contiene una directiva @ Control que define la configuración y otras propiedades.
- Los controles de usuario no se pueden ejecutar como archivos independientes. En su lugar, debe agregarlos a las páginas ASP.NET, como haría con cualquier otro control.
- El control de usuario no contiene elementos html, body o form. Estos elementos deben estar en la página de alojamiento.

En un control de usuario puede utilizar los mismos elementos HTML (excepto html, body y form) y controles Web que en una página Web ASP.NET. Por ejemplo, si está creando un control de usuario para utilizar una barra de herramientas, puede colocar una serie de controles de servidor Web Button en el control y crear controladores de eventos para los botones.

En el ejemplo siguiente se muestra un control de usuario que implementa un control de número en el que los usuarios pueden hacer clic en los botones arriba y abajo para mostrar varias opciones de un cuadro de texto.

```
<% @ Control Language="C#" ClassName="UserControl1" %>
<script runat="server">
    protected int currentColorIndex;
    protected String[] colors = {"Red", "Blue", "Green", "Yellow"};
    protected void Page_Load(object sender, EventArgs e)
    {
        if (IsPostBack)
        {
            currentColorIndex =
                Int16.Parse(ViewState["currentColorIndex"].ToString());
        }
        else
        {
            currentColorIndex = 0;
            DisplayColor();
        }
    }

    protected void DisplayColor()
    {
        textColor.Text = colors[currentColorIndex];
        ViewState["currentColorIndex"] = currentColorIndex.ToString();
    }

    protected void buttonUp_Click(object sender, EventArgs e)
    {
        if(currentColorIndex == 0)
        {
```

```
            currentColorIndex = colors.Length - 1;
        }
        else
        {
            currentColorIndex -= 1;
        }
        DisplayColor();
    }

    protected void buttonDown_Click(object sender, EventArgs e)
    {
        if(currentColorIndex == (colors.Length - 1))
        {
            currentColorIndex = 0;
        }
        else
        {
            currentColorIndex += 1;
        }
        DisplayColor();
    }
</script>
<asp:TextBox ID="textColor" runat="server"
    ReadOnly="True" />
<asp:Button Font-Bold="True" ID="buttonUp" runat="server"
    Text="^" OnClick="buttonUp_Click" />
<asp:Button Font-Bold="True" ID="buttonDown" runat="server"
    Text="v" OnClick="buttonDown_Click" />
```

Tenga en cuenta que el control de usuario es muy similar a una página ASP.NET, ya que contiene varios controles (un control TextBox y dos controles Button) y código que controla los eventos Click de los botones y el evento Load de la página. Sin embargo, el control no contiene ningún formato, excepto para los controles, y en lugar de una directiva @ Page contiene una directiva @ Control.

Crear controles de usuario

Los controles de usuario ASP.NET se crean casi de la misma forma en la que se diseñan las páginas Web ASP.NET. Se pueden usar los mismos elementos y controles HTML en un control de usuario que en una página ASP.NET estándar. Sin embargo, el control de usuario no tiene elementos html, body ni form; además, la extensión de nombre de archivo debe ser .ascx.

Para crear un control de usuario ASP.NET:

1. Abra el proyecto de sitio Web al que desee agregar controles de usuario. Si aún no tiene ningún proyecto de sitio Web, puede crear uno.
2. En el menú Sitio Web, haga clic en Agregar nuevo elemento.
   Aparecerá el cuadro de diálogo Agregar nuevo elemento.

3. En la opción Plantillas instaladas de Visual Studio del cuadro de diálogo Agregar nuevo elemento, haga clic en Control de usuario Web.
4. En el cuadro Nombre, escriba un nombre para el control.
   De forma predeterminada, la extensión de nombre de archivo .ascx se anexa al nombre de control que escriba.
5. En la lista Lenguaje, seleccione el lenguaje de programación que desee utilizar.
6. Opcionalmente, si desea mantener algún código del control de usuario en un archivo independiente, active la casilla Colocar el código en un archivo independiente.
7. Haga clic en Agregar.

   Se crea el nuevo control de usuario ASP.NET y, a continuación, se abre en el diseñador. El código de formato para este nuevo control es similar al de una página Web ASP.NET, salvo que contiene una directiva @ Control en vez de una directiva @ Page; además, el control no tiene los elementos html, body ni form.

   Agregue los códigos de formato y los controles al nuevo control de usuario, así como el código de las tareas que realizará el control de usuario, como controlar los eventos del control o la lectura de los datos desde un origen de datos.

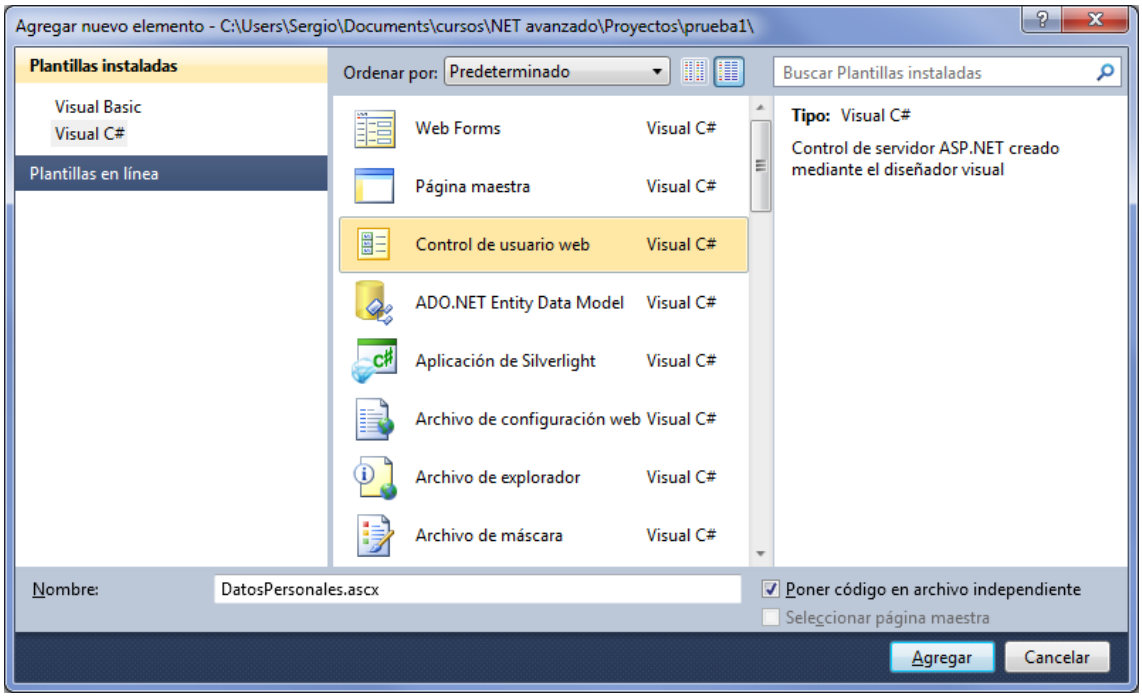

Ilustración 1: Crear un control de usuario

Incluir controles de usuario

Mediante el diseñador

Para agregar un control de usuario ASP.NET a una página Web se sigue un proceso parecido al de agregar otros controles de servidor. Sin embargo, se debe asegurar de seguir el siguiente procedimiento para que todos los elementos necesarios se agreguen a la página.

1. Abra la página Web a la que desee agregar el control de usuario ASP.NET.
2. Cambie a la vista Diseño.
3. En el Explorador de soluciones, seleccione el archivo del control de usuario y arrástrelo a la página.

El control de usuario ASP.NET se agrega a ella. Además, el diseñador crea la directiva @Register, que se necesita para que la página reconozca el control de usuario. Ya puede trabajar con las propiedades y métodos públicos del control.

Manualmente

1. En la página Web ASP.NET contenedora, cree una directiva @ Register que incluya lo siguiente:
   - Un atributo TagPrefix, que permite asociar un prefijo al control de usuario. Este prefijo se incluirá en la etiqueta de apertura del elemento del control de usuario.
   - Un atributo TagName, que permite asociar un nombre al control de usuario. Este nombre se incluirá en la etiqueta de apertura del elemento del control de usuario.
   - Un atributo Src, que permite definir la ruta de acceso virtual al archivo del control de usuario que se va a incluir.
     Nota:
     El valor del atributo Src puede ser una ruta de acceso relativa o absoluta al archivo de código fuente del control de usuario partiendo del directorio raíz de la aplicación. Para mayor flexibilidad, se recomienda utilizar una ruta de acceso relativa. El carácter tilde (~) representa el directorio raíz de la aplicación. Los controles de usuario no pueden situarse en el directorio App_Code.
2. En el cuerpo de la página Web, declare el elemento de control de usuario dentro del elemento form.

3. Si el control de usuario expone propiedades públicas, también puede establecerlas mediante declaración.

Ejemplo

En el ejemplo siguiente se muestra una página Web ASP.NET que contiene un control de usuario. Éste se encuentra en el archivo Spinner.ascx de la carpeta Controls. En la página, el control se registra para que utilice el prefijo uc y el nombre de etiqueta Spinner. Las propiedades MinValue y MaxValue del control de usuario se establecen mediante declaración.

```
<%@ Page Language="C#" %>
<%@ Register TagPrefix="uc" TagName="Spinner"
    Src="~\Controls\Spinner.ascx" %>
<html>
<body>
<form runat="server">
    <uc:Spinner id="Spinner1"
        runat="server"
        MinValue="1"
        MaxValue="10" />
</form>
</body>
```

Convertir páginas de formularios Web Forms en controles de usuario

Si ha desarrollado una página Web ASP.NET y desea tener acceso a su funcionalidad en toda la aplicación, puede realizar algunos cambios menores en la página para convertirla en un control de usuario.

Para convertir una página Web ASP.NET de un solo archivo en un control de usuario

1. Cambie el nombre del control para que la extensión del nombre de archivo sea .ascx.
2. Quite los elementos html, body y form de la página.
3. Cambie la directiva @ Page por una directiva @ Control.
4. Quite todos los atributos de la directiva @ Control, excepto Language, AutoEventWireup (si existe), CodeFile e Inherits.
5. Incluya un atributo className en la directiva @ Control. Esto permite agregar el control de usuario a una página con establecimiento inflexible de tipos.

Para convertir una página Web ASP.NET de código subyacente en un control de usuario

1. Cambie el nombre del archivo .aspx para que la extensión del nombre de archivo sea .ascx.
2. Cambie el nombre del archivo de código subyacente para que tenga la extensión de nombre de archivo .ascx.cs.
3. Abra el archivo de código subyacente y cambie la clase de la que hereda de Page a UserControl.
4. En el archivo .aspx, haga lo siguiente:
    a) Quite los elementos html, body y form de la página.
    b) Cambie la directiva @ Page por una directiva @ Control.
    c) Quite todos los atributos de la directiva @ Control, excepto Language, AutoEventWireup (si existe), CodeFile e Inherits.
    d) En la directiva @ Control, cambie el atributo CodeFile para que señale al archivo de código subyacente cuyo nombre acaba de cambiar.
5. Incluya un atributo className en la directiva @ Control. Esto permite agregar el control de usuario a una página con establecimiento inflexible de tipos.

Ejemplo

En el ejemplo siguiente se muestra una página Web ASP.NET de un solo archivo en su formato original y el control de usuario resultante después de convertir la página.

```
<%@ Page Language="C#" %>
<html>
<script runat=server>
void EnterBtn_Click(Object sender, EventArgs e)
{
    Label1.Text = "Hi " + Name.Text + " welcome to ASP.NET!";
}
</script>
<body>
<h3> <u>Web Forms Page</u> </h3>
<form>
Enter Name: <asp:textbox id="Name" runat=server/>
<asp:button Text="Enter" OnClick="EnterBtn_Click" runat=server/>
<br>
<br>
<asp:label id="Label1" runat=server/>
</form>
</body>
</html>
```

```
<%@ Control Language="C#" ClassName="SampleUserControl" %>
<h3> <u>User Control</u> </h3>
<script runat=server>
void EnterBtn_Click(Object Sender, EventArgs e)
{
    Label1.Text = "Hi " + Name.Text + " welcome to ASP.NET!";
}
</script>
Enter Name: <asp:textbox id="Name" runat=server/>
<asp:button Text="Enter" OnClick="EnterBtn_Click" runat=server/>
<br>
<br>
<asp:label id="Label1" runat=server/>
```

Crear instancias de controles de usuario ASP.NET mediante programación

La creación mediante programación de una instancia de un control de servidor en una página Web ASP.NET es muy similar a la creación de un control de usuario

Para crear una instancia de un control de usuario mediante programación

1. En el control de usuario, asegúrese de que la directiva @ Control contiene un atributo ClassName que asigna una clase al control de usuario.
   En el ejemplo siguiente se establece el atributo ClassName para realizar el establecimiento inflexible de tipos en un control de usuario.

   ```
   <%@ Control className="MyUserControl" %>
   ```

2. En la página en la que desea trabajar con el control de usuario, cree una referencia a dicho control con la directiva @ Reference.
   Cuando cree el control de usuario mediante programación, el tipo inflexible para dicho control estará disponible para la página Web ASP.NET sólo después de haber creado una referencia al mismo. Por ejemplo, el código siguiente crea una referencia a un control de usuario creado en el archivo MyUserControl.ascx.

   ```
   <%@ Reference Control="MyUserControl.ascx" %>
   ```

3. Cree una variable de instancia para el control de usuario, utilizando el nombre de clase del control. La clase será parte del espacio de nombres ASP.
   Por ejemplo, si desea crear una instancia del control de usuario declarada como clase Spinner, debe utilizar sintaxis como la siguiente:

```
Protected ASP.Spinner Spinner1;
```

4. Cree una instancia del control de usuario en el código llamando al método LoadControl.
5. Asigne los valores de las propiedades según sea necesario y, a continuación, agregue el control a la colección ControlCollection de un contenedor de la página, como un control PlaceHolder.

Ejemplo

En el ejemplo siguiente se muestra una página Web ASP.NET que carga un control de usuario mediante programación. La página incluye una directiva @ Reference para especificar el archivo del control. El método LoadControl lee el archivo y crea una instancia de éste en forma de control que se puede agregar a la página.

```
<%@ Page Language="C#" %>
<%@ Reference Control="~/Controls/Spinner.ascx" %>
<script runat="server">
private ASP.Spinner Spinner1;
protected void Page_Load(object sender, EventArgs e)
{
    Spinner1 = (ASP.Spinner)LoadControl("~/Controls/Spinner.ascx");
     // Set MaxValue first.
    Spinner1.MaxValue = 20;
    Spinner1.MinValue = 10;
    PlaceHolder1.Controls.Add(Spinner1);
}

protected void Button1_Click(object sender, EventArgs e)
{
    Label1.Text = Spinner1.CurrentNumber.ToString();
}
</script>

<html>
<head id="Head1" runat="server">
  <title>Load User Control Programmatically</title>
</head>
```

```
<body>
  <form id="form1" runat="server">
    <div>
      <asp:PlaceHolder runat=server ID="PlaceHolder1" />
      <br />
      <asp:Button ID="Button1" runat="server" Text="Button"
        OnClick="Button1_Click" />
      <br />
      <br />
      <asp:Label ID="Label1" runat="server" Text=""></asp:Label>
    </div>
  </form>
</body>
</html>
```

Crear controles de usuario ASP.NET con plantillas

Puede crear controles de usuario que implementen plantillas, que son una característica de ASP.NET que permite separar los datos del control de su presentación. Los controles con plantilla no proporcionan una interfaz de usuario. En lugar de ello, se escriben para que implementen un contenedor de nombres e incluyan una clase a cuyas propiedades y métodos puede tener acceso la página que aloja el control (o página host).

La interfaz de usuario correspondiente al control de usuario la proporciona un desarrollador de páginas en tiempo de diseño. El desarrollador crea las plantillas del tipo definido por el control de usuario y, a continuación, puede agregar a ellas controles y formato.

Para crear un control de usuario con plantillas:

1. En el archivo .ascx, agregue un control PlaceHolder de ASP.NET donde desee que aparezca la plantilla.
2. En el código del control de usuario, implemente una propiedad de tipo Itemplate.
3. Defina una clase de control de servidor que implemente la interfaz INamingContainer como contenedor, en la que se creará una instancia de la plantilla. Esta clase se denomina contenedor de nombres de la plantilla.
   Nota: El control se convierte básicamente en una clase anidada del control de usuario, aunque esto no es necesario.
4. Aplique el atributo TemplateContainerAttribute a la propiedad que implementa ITemplate y pase el tipo del contenedor de nombres de la plantilla como argumento al constructor de atributos.
5. En el método Init del control, repita los pasos siguientes una o varias veces:

- Cree una instancia de la clase del contenedor de nombres.
- Cree una instancia de la plantilla en el contenedor de nombres.
- Agregue la instancia del contenedor de nombres a la propiedad Controls del control de servidor PlaceHolder.
  Nota: Desde el punto de vista de la página que utiliza el control de usuario, la sintaxis del control de usuario con plantilla es idéntica a la de un control con plantilla personalizado.

Ejemplo

En el ejemplo siguiente se muestra un control de usuario con plantilla y una página que lo contiene. El control de usuario crea una plantilla que puede declararse en una página host como <MessageTemplate>. El control de plantilla también expone dos propiedades, Index y Message, a las que puede tener acceso la página host dentro de la plantilla.

En el primer ejemplo se muestra el control de usuario con plantilla. En el segundo ejemplo se muestra una página que contiene el control de usuario.

```
<%@ Control language="C#" ClassName="TemplatedUC" %>
<script runat=server>
private ITemplate messageTemplate = null;

[ TemplateContainer(typeof(MessageContainer)) ]
public ITemplate MessageTemplate {
    get
    {
        return messageTemplate;
    }
    set
    {
        messageTemplate = value;
    }
}

void Page_Init() {
    if (messageTemplate != null) {
        String[] fruits = {"apple", "orange", "banana", "pineapple" };
        for (int i=0; i<4; i++)
        {
            MessageContainer container = new MessageContainer(i, fruits[i]);
            messageTemplate.InstantiateIn(container);
            PlaceHolder1.Controls.Add(container);
        }
    }
```

```
}

public class MessageContainer: Control, INamingContainer {
    private int m_index;
    private String m_message;
    internal MessageContainer(int index, String message)
    {
        m_index = index;
        m_message = message;
    }
    public int Index {
        get
        {
            return m_index;
        }
    }
    public String Message
    {
        get
        {
            return m_message;
        }
    }
}
</script>
<asp:placeholder runat=server id="PlaceHolder1" />
```

```
<%@ Page Language="C#" %>
<%@ Register TagPrefix="uc" tagname="TemplateTest"
    Src="TemplatedUC.ascx" %>
<html>
<script runat=server>
    protected void Page_Load()
    {
        DataBind();
    }

</script>
<head>
<title>Templated User Control Test</title>
</head>
<body>
<h1>Testing Templated User Control</h1>
<form id="Form1" runat=server>
<uc:TemplateTest runat=server>
  <MessageTemplate>
    Index: <asp:Label runat="server" ID="Label1"
        Text='<%# Container.Index %>' />
    <br />
    Message: <asp:Label runat="server" ID="Label2"
        Text='<%# Container.Message %>' />
```

```
    <hr />
  </MessageTemplate>
</uc:TemplateTest>
</form>
</body>
</html>
```

Acceder a los controles incluidos en un control de usuario

Si tenemos un control de usuario llamado DatosPersonales1 que contiene un campo de texto llamado TextBoxNombre podemos acceder a todas las propiedades de la caja de texto mediante el método FindControl.

Por ejemplo, para obtener el valor:

```
((TextBox) DatosPersonales1.FindControl("TextBoxNombre")).Text ;
```

Pero cuando se crea un control de usuario lo que se busca es encapsular una cierta funcionalidad para ser reutilizado en otras partes de la aplicación o incluso en otras aplicaciones. Por lo que no sería necesario ni conocer el nombre de los controles incluidos en este.

Para ello se crean propiedades públicas para acceder a las características del control que se desee. Por ejemplo si queremos que se pueda leer y modificar el contenido de la caja de texto de nombre de un control de usuario DatosPersonales, haremos lo siguiente en DatosPersonales.ascx.cs

```
public partial class DatosPersonales : System.Web.UI.UserControl
{
    . . .

    public string Nombre
    {
        get
        {
            return(TextBoxNombre.Text);
        }
        set
        {
            TextBoxNombre.Text = value;
        }
    }
```

```
}
```

Con lo cual ya se podrá leer y modificar el nombre simplemente haciendo referéncia a esta propiedad.

Por ejemplo si en un aspx tenemos una instancia del control de usuario llamada DatosPersonales1

```
<uc1:DatosPersonales ID="DatosPersonales1" runat="server" />
```

podemos obtener el valor de la caja de texto del nombre con:

```
DatosPersonales1.Nombre
```

Eventos de controles de usuario

A la hora de crear controles de usuario solemos necesitar capturar un evento que se produce dentro del mismo, en la página padre. Además, normalmente, necesitaremos acceder a alguna de sus propiedades desde la página padre. En el ejemplo que os presento tenemos un control de usuario (un buscador) incrustado en una página que mostrará los resultados de la búsqueda.

Para ello el control de usuario posee un evento que se lanza una vez la búsqueda ha finalizado, además, los resultados de dicha búsqueda se almacenan en una propiedad del control de usuario que recupera la página para mostrarlos en un GridView.

Pasemos al código. El control de usuario, en su parte visual se compone únicamente de un TextBox (donde introducir el criterio de búsqueda) y un Button (que realiza la búsqueda).

La generación del evento se realiza desde el code-behind:

```
public event EventHandler AceptarClicked;

protected virtual void OnClick(object sender)
{
      if (this.AceptarClicked != null)
```

```
        {
                this.AceptarClicked(sender, new EventArgs());
        }
}
```

Este código declara el evento (cuyo nombre será AceptarClicked) e implementa el método que invoca el evento.

El código siguiente declara una propiedad pública dentro del control donde se almacenarán los resultados de la búsqueda.

```
Private List<string> resultados;
public List<string> Resultados
{
        get { return resultados; }
        set { resultados = value; }
}
```

En el evento del botón del control de usuario realizamos la llamada al evento público que será capturado por la página:

```
protected void btnBuscar_Click(object sender, EventArgs e)
{
        //Hardcode para simular una búsqueda
        this.resultados = new
        List<string>();
        if (String.IsNullOrEmpty(this.txtPalabraClave.Text))
        {
                this.resultados.Add("No se han encontrado resultados");
        }
        else
        {
                this.resultados.Add("Resultado1 para " +
                this.txtPalabraClave.Text);
```

```
            this.resultados.Add("Resultado2 para " +
            this.txtPalabraClave.Text);
      }
      //Se invoca el evento público una vez la función del control ha
      terminado
      OnClick(sender);
}
```

La página debe capturar el evento, esto se hace de igual forma que se captura el evento de un botón o cualquier control de .NET. Así, en la presentación de la página tendremos:

```
<uc1:Buscador ID="Buscador1"
runat="server" OnAceptarClicked="Buscador1_AceptarClicked"/>
```

Así el método de tratar el evento del control de usuario, dentro de la página padre será Buscador1_AceptarClicked y se codificará así:

```
protected
void Buscador1_AceptarClicked(object sender, EventArgs e)
{
      if (this.Buscador1.Resultados != null)
      {
            this.GridView1.DataSource = this.Buscador1.Resultados;
            this.GridView1.DataBind();
      }
}
```

Referencia global a controles de usuario

Si vamos a utilizar en muchos sitios del proyecto un mismo control de usuario este se puede incluir en el Web.config en vez de en cada .aspx que lo utilice. De esta forma nos evitamos tener que poner la etiqueta @ Resgister o @ Reference.

```
<system.web>
 <pages>
  <controls>
   <add src="~/Controles/DatosPersonales.ascx" tagName="DatosPersonales"
        tagPrefix="uc1" />
  </controls>
 </pages>
</system.web>
```