ASP.NE

Tutorial ASP.NET: Controles de servidor y eventos

Sin saberlo ya hemos estado trabajando con controles de servidor y usando sus eventos en los capítulos anteriores. Los controles de servidor son identificados con la etiqueta runat="server" y por medio de su ID nos permite manipularlos y programarlos en tiempo de ejecución mediante código C Sharp.

Fernando Giardina enero 6 2011

Muchos de los controles de servidor son similares a controles HTML como los cuadros de texto o botones. Pero hay controles de servidor muchos más complejos que brindan numerosas funcionalidades: grillas, calendarios, vista de árbol, menúes y hasta controles de conectividad.

¿Qué son los eventos?

Básicamente un evento responde a una acción que sucede en relación al objeto. Lo más usual son los eventos que suceden en la interfaz de usuario. Por ejemplo, ya hemos visto un control `BOTON` y su evento `OnClick`. El evento `OnClick` se produce al presionar el botón y se ejecuta el código que ingresamos para ese evento.

Agregar controles de servidor en tiempos de diseño:

Agregue controles a una página Web ASP.NET de la misma manera que agrega cualquier elemento HTML. Puede utilizar un diseñador visual y agregar un control del cuadro de herramientas, o bien, puede escribir el elemento que representa el control.

```
<asp:Button ID="btnAceptar" runat="server" Text="Acepta
 onclick="btnAceptar_Click"/>
```

Los controles de servidor Web se declaran con una etiqueta XML que hace referencia al espacio de nombres `asp`. El inicio y fin del tag está determinado por `<asp />`

- `runat="server"`: Nos indica que se trata de un control de servidor.
- `onclick="btnAceptar_Click"`: btnAceptar contiene el código que se va a ejecutar cuando se produzca el evento `onclick`.

Agregar controles de servidor en tiempos de ejecución:

(Mediante programación)

Muchas veces es necesario crear un control en tiempo de ejecución en lugar de hacerlo en tiempo de diseño. Supongamos que dinámicamente debemos crear varios Combos pero no sabemos la cantidad hasta que la página vaya recibiendo información. En este caso es necesario crearlo mediante código indicando la cantidad de veces que necesitamos crearlo.

Ejemplo de creación de un TextBox por programación.

```
TextBox txt = new TextBox();
txt.ID = "txtNombre";
txt.Text = "MDW";
```

Ahora este mismo textbox lo vamos a incluir en un Panel.

```
Panel pnl = new Panel();
pnl.ID = "Panel";
pnl.Controls.Add(txt);
```

Cuando ejecutemos nuestra página veremos que en pantalla se crea un cuadro de texto con el contenido “MDW” dentro de un control PANEL.

En la siguiente rutina veremos cómo agregar dinámicamente controles de servidor en tiempo de ejecución. Creamos un archivo llamado `Controles.aspx` y copiamos el siguiente código.

```
<%@ Page Language="C#" AutoEventWireup="true" Co
<html xmlns="http://www.w3.org/1999/xhtml"> <head runat=
<title></title> </head>
<body> <form id="form1" runat="server"> <div>
<asp:Label ID="lblCantidad" runat="server" Text="Ingrese
onclick="btnProceso_Click" /> </div>
</form> </body>
</html>
```

Ahora creamos otro archivo llamado Controles.aspx.cs y copiamos el siguiente código.

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

[Guia ASP.NET Por Maestros del web]
public partial class Controles : System.Web.UI.Page {
}
protected void Page_Load(object sender, EventArgs e) {
}
protected void btnProceso_Click(object sender, EventArg
Table tbl = new Table(); try
{
int cantidad = Convert.ToInt32(txtCantidad.Text.Trim())
Label lbl; for (int i = 0; i < cantidad; i++) {
lbl = new Label(); lbl.ID = "IdLabel" + i.ToString(); l
AgregarLabel(tbl, lbl); Page.Controls.Add(tbl);
}
} catch (Exception ex) {
Label lbl = new Label(); lbl.ID = "error"; lbl.Text = e
}
private void AgregarLabel(Table t,Label l) {
}
}
TableCell tc = new TableCell(); TableRow tr = new Table
tc.Controls.Add(l); tr.Cells.Add(tc); t.Rows.Add(tr);
```

Acabamos de ver como generamos controles de servidor por programación, ahora vamos a

ver como trabajamos con eventos en forma dinámica.

```
<%@ Page Language="C#" AutoEventWireup="true" Co
<html xmlns="http://www.w3.org/1999/xhtml"> <head runat=
<title></title> </head>
<body> <form id="form1" runat="server"> <div>
<asp:Label ID="lblCantidad" runat="server" Text="Ingrese
</div>
</form> </body>
</html>
```

Aquí tenemos el mismo formulario a diferencia que el botón no tiene el evento OnClick. Ahora lo vamos a hacer en tiempo de ejecución. Esto es equivalente a `onclick="btnProceso_Click"` pero por programación.`btnProceso.Click += new EventHandler(btnProceso_Click);`

Este código lo vamos a agregar en el evento Page_Load que se ejecuta cada vez que se carga la página.

```
protected void Page_Load(object sender, EventArgs e) {
btnProceso.Click += new EventHandler(btnProceso_Click);}
```

Se agrega un concepto nuevo en el evento `Page_Load` “`IsPostBack`”. Se usa para saber si ya se ha ejecutado el método Page_Load. Como en este caso sólo necesitamos que se declare el sólo una vez preguntamos si es la primera vez que se ejecuta mediante `if(!IsPostBack)`.

Las veces posteriores que se cargue la página no entrarán en ese `IF`. El signo `!` es el negado de la función, es como preguntar “¿NO ES POSTBACK?“

Capítulos del Tutorial

En el próximo capítulo que se publica el martes veremos: **Controles de Usuarios Reutilizables.**

Fernando Giardina

En 1997 comienzo a desarrollar en lenguaje C y Oracle programando IVRs, automatización de procesos y sistemas de CRM. Continuo con JAVA, Perl y PHP hasta que en el año 2002 me inicio en la tecnología .NET desarrollando aplicaciones web, desktop y mobile.

http://www.metalstudio.com.ar

Otros artículos que podrían interesarte

Google Wing hace ver los drones de Amazon como juguetes

Derechos digitales en América Latina, los gobiernos te están observando (y espiando)

¿Odias el internet lento? Lucha por la Neutralidad de la Red

Fedora Conference 2014: software libre para la administración pública, cervezas y un paseo por Praga

Cartas Natales

Gracias muy buen post, me ha sido muy util!

Tutorial ASP.NET: Controles de servidor y eventos

[…] Capítulo 6: Controles de servidor y eventos […]

Tutorial ASP.NET: Controles de usuarios reutilizables

[…] Capítulo 6: Controles de servidor y eventos […]

Tutorial ASP.NET: Utilización de estilos en ASP.NET (CSS)

[…] Capítulo 6: Controles de servidor y eventos […]

Tutorial ASP.NET: Crear una página de login, autenticación y seguridad

[…] Capítulo 6: Controles de servidor y eventos […]

Tutorial ASP.NET: Utilización de Master Pages

[…] Capítulo 6: Controles de servidor y eventos […]

hector

Hola antes que nada muchas gracias por estos manuales que estan muy buenos, con ellos he estado aprendiendo bastante..