Conceptos.

Introducción

Hasta ahora, hemos aprendido a crear páginas web que utilizan el modelo postback. Con el postback, las páginas están perpetuamente reenviándose al servidor web y regenerándose.
Como desventaja del modelo postback tenemos que hay un parpadeo del contenido de la página cuando tiene que refrescarse. Además se tiene el problema que se reenvían todos los datos al servidor.
Recientemente, una nueva generación de aplicaciones web ha comenzado a aparecer que se comportan más como las aplicaciones de Windows que las tradicionales páginas web. Estas aplicaciones se refrescan en forma rápida y sin parpadeos. Entre los ejemplos notables incluyen el correo electrónico basado en web como Gmail y aplicaciones de la cartografía como Google Maps.
Esta nueva generación de aplicaciones web utiliza un conjunto de prácticas de diseño y tecnologías conocidas como Ajax.
Una de las características fundamentales de Ajax es la capacidad para actualizar parte de la página, mientras que deja el resto intacto.
AJAX son las siglas de Asynchronous JavaScript And XML. No es un lenguaje de programación sino un conjunto de tecnologías (HTML-JavaScript-CSS-DHTML-PHP/ASP.NET/JSP-XML) que nos permiten hacer páginas de internet más interactivas. La característica fundamental de AJAX es permitir actualizar parte de una página con información que se encuentra en el servidor sin tener que refrescar completamente la página. De modo similar podemos enviar información al servidor.
Veremos como ASP.Net nos esconde la complejidad de Ajax y nos permite una fácil transición entre aplicaciones web tradicionales y el nuevo modelo.

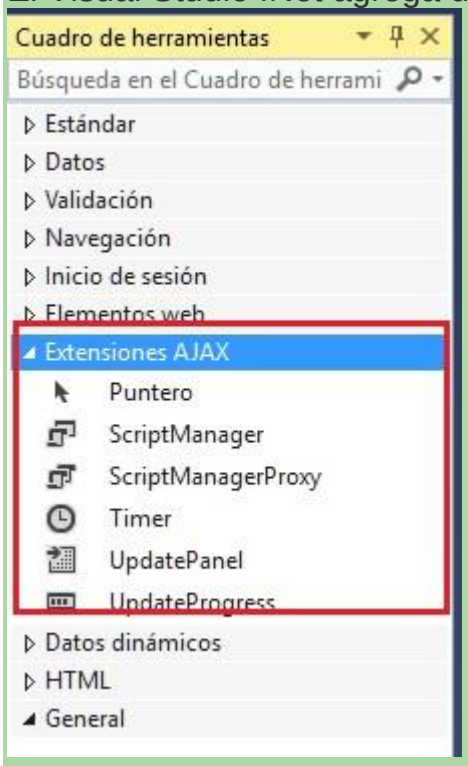
ASP.NET AJAX

Hay una variedad de formas de aplicar el Ajax en cualquier aplicación Web, incluyendo ASP.NET. Para ponerlo en práctica Ajax sin utilizar librerías, es necesario que se tenga una comprensión profunda de JavaScript, porque es el lenguaje JavaScript el que se ejecuta en el navegador, que solicita la nueva información del servidor web cuando sea necesario y la actualización de la página en consecuencia.
Si bien JavaScript no es terriblemente complejo, es muy difícil de programar correctamente, por dos razones:
- La aplicación de los principales detalles de JavaScript varía de navegador a navegador, que significa que necesita una enorme cantidad de experiencia para escribir una buena página web que se ejecuta igual de bien en todos los navegadores.
- JavaScript es un lenguaje muy laxo que tolera muchos pequeños errores. La captura de estos errores y la eliminación de ellos es un proceso tedioso. Aún peor, el error puede ser fatal en algunos navegadores y dañina en otros, lo que complica la depuración.
En este capítulo, usted NO va a usar JavaScript directamente. Por el contrario, utilizará un modelo de nivel superior llamado ASP.NET AJAX.
ASP.NET AJAX le da un conjunto de componentes del lado del servidor y los controles ASP.NET tradicionales que puede utilizar en el diseño de su página web. Estos componentes hacen automáticamente todos el código JavaScript que necesita para obtener el efecto deseado. El resultado es que puede crear una página con Ajax. Por

supuesto, usted no obtendrá el máximo control para personalizar hasta el último detalle de su página, pero usted recibirá una gran funcionalidad con un mínimo de esfuerzo.

Extensiones Ajax

El Visual Studio .Net agrega una pestaña que agrupa los controles referidos a Ajax:



Para poder utilizar ASP.NET AJAX es necesario un control de tipo ScriptManager. Este control es el cerebro de ASP.NET AJAX, Cuando disponemos un control de tipo ScriptManager en el formulario aparece un cuadro gris, pero cuando ejecutemos dicha página no aparecerá, es decir el control ScriptManager no genera etiquetas HTML.
El control ScriptManager genera todo el código JavaScript necesario para las llamadas asíncronas desde el navegador al servidor web.
Cada página que utiliza ASP.NET AJAX requiere solo una instancia de la clase ScriptManager, indistintamente la cantidad de regiones que vallamos a actualizar posteriormente.

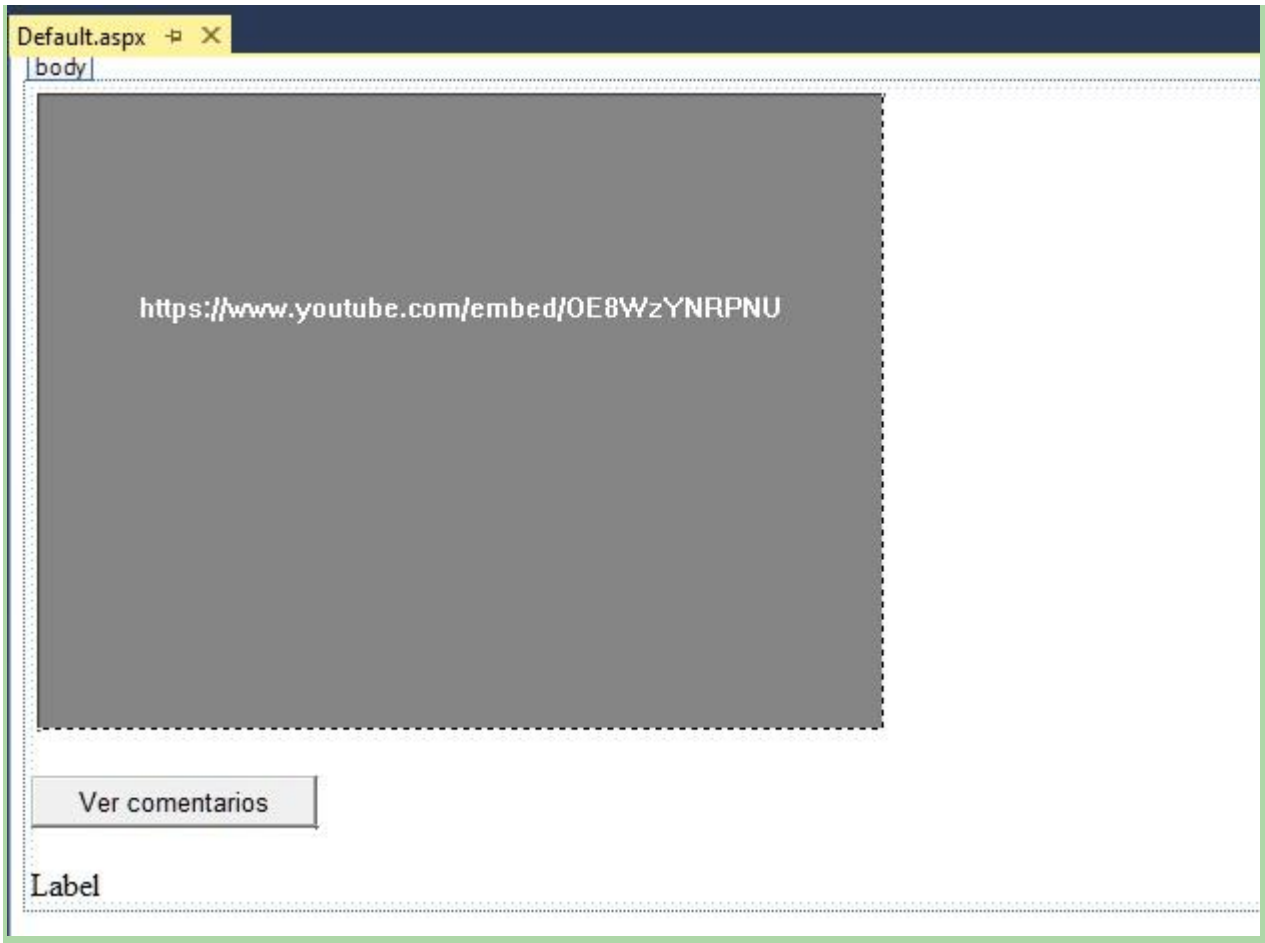
Refresco parcial de página.

Confeccionaremos un problema sin utilizar AJAX y luego lo resolveremos utilizando AJAX y veremos las ventajas que presenta.
El problema a resolver es mostrar un video del sitio de youtube.com, dispondremos un botón que cuando se presione recuperará los comentarios existentes para dicho video (para hacer el problema más corto evitaremos extraer los comentarios de una base de datos)
>

Crear un sitio web llamado ejercicio022 y agregar un Web Form llamado Default.aspx
Disponemos en vista de código las marcas HTML que nos provee el sitio youtube.com para insertar en nuestra página el video. Luego la página Default en vista de código queda:

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

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8"/>
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
      <iframe width="420" height="315"
src="https://www.youtube.com/embed/OE8WzYNRPNU"
frameborder="0" allowfullscreen></iframe>
    </div>
    </form>
</body>
</html>
```

Hemos dispuesto el código suministrado por youtube.com. Ahora en vista de Diseño agregamos un Button y una Label.:

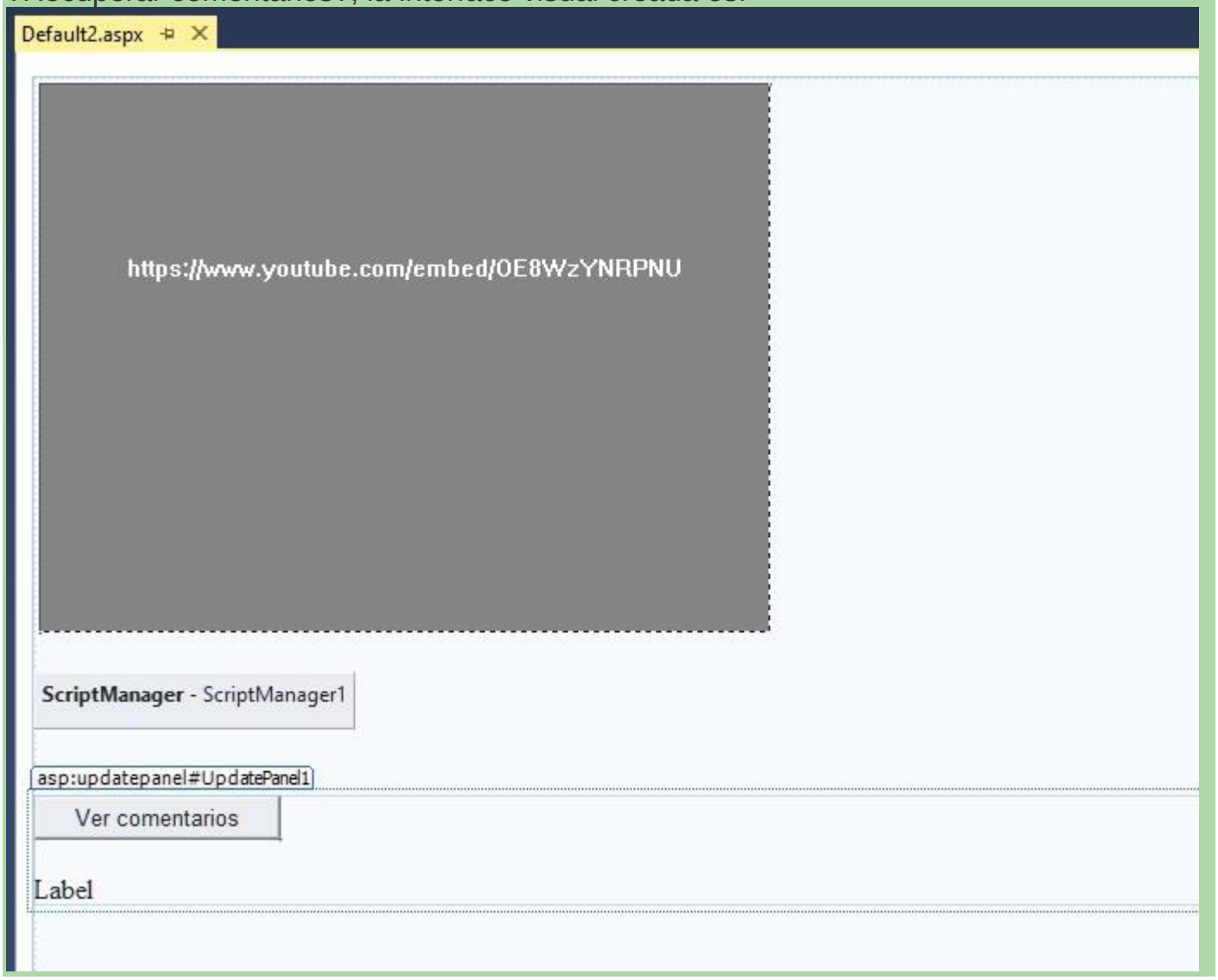


El código a implementar al presionar el botón es:

```
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 Page Load(object sender, EventArgs e)
    {

    }

    protected void Button1 Click(object sender,
EventArgs e)
    {
```

```
        Label1.Text = "1 - Muy buen video. <br> 2 -
Correcto. <br>3 - De lo mejor.";
    }
}
```

Es decir cargamos el contenido de la Label con los comentarios. Como podrán ver al ejecutar la aplicación si estamos viendo el video y presionamos el botón de ?Ver comentarios? el contenido de la página se recarga, es decir debemos comenzar de nuevo a ver el video (esta situación es poco agradable para el visitante del sitio) Ahora crearemos otra página y utilizaremos ASP.NET AJAX para no tener que refrescar toda la página sino la Label donde deben aparecer los comentarios. Creamos una segunda página (Default2.aspx), agregamos también el código suministrado por youtube.com.
Debemos primero insertar un objeto de la clase ScriptManager y un objeto de la clase UpdatePanel.
Todo lo contenido en el control UpdatePanel se refrescará sin tener que recargar la página por completo. Disponemos entonces dentro del UpdatePanel1 la Label y el botón para ?Recuperar comentarios?, la interface visual creada es:



AJAX con ASP.Net

Para el evento click del botón realizamos el mismo código hecho para la página anterior:

```
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)
    {

    }

    protected void Button1 Click(object sender,
EventArgs e)
    {
        Label1.Text = "1 - Muy buen video. <br> 2 -
Correcto. <br>3 - De lo mejor.";
    }
}
```

Si ejecutamos la página Default2.aspx veremos que la experiencia del usuario visitante será mejor. Probemos de iniciar el video y luego presionar el botón ?Ver comentarios?, como observaremos el video no se detiene y luego de un corto tiempo tenemos en la página los comentarios dispuestos en dicho video.

La idea básica es dividir su página web en una o más regiones distintas, cada una de las que se envuelve dentro de un UpdatePanel invisibles en tiempo de ejecución. Cuando un evento ocurre en un control que se ubicado dentro de un UpdatePanel, y este evento normalmente desencadena una página completa postback, el UpdatePanel intercepta el evento y realiza una llamada asincrónica. Aquí está un ejemplo de cómo esto sucede:

1. El usuario hace clic en un botón dentro de un UpdatePanel.
2. El UpdatePanel intercepta en el lado del cliente el clic. Ahora, ASP.NET AJAX realiza una llamada al servidor en lugar de una página completa postback.
3. En el servidor, se ejecuta su ciclo de vida en forma normal, con todos los eventos habituales.
4. ASP.NET AJAX recibe todo el HTML y se actualiza cada UpdatePanel y los cambios que no están dentro de un UpdatePanel se ignoran.