Introducció a ASP.NET

Jordi Sánchez Cano

PID_00173510

UOC
Universitat Oberta de Catalunya
www.uoc.edu

Índex

Introducció

ASP.NET és un dels entorns de treball (*frameworks*) més estesos per a la programació d'aplicacions web. Combinat amb els entorns de treball de Microsoft, ASP.NET permet crear llocs web sofisticats d'una manera visual i ràpida. També inclou components AJAX que permeten crear pàgines més dinàmiques i interactives.

En aquest mòdul es fa una introducció a la plataforma .NET i a l'entorn de treball ASP.NET utilitzant el llenguatge de programació Visual C#. Inicialment s'expliquen alguns aspectes principals de l'entorn de treball integrat (IDE) Visual Web Developer Express, eina gratuïta de Microsoft per al desenvolupament d'aplicacions web. Posteriorment, es descriuen els passos per a crear llocs web interactius que responguin a diferents accions de l'usuari. Finalment, s'expliquen els controls ASP.NET AJAX i com utilitzar-los a tall d'exemple.

IDE

IDE és la sigla d'*integrated development environment*.

Els exemples i el codi d'aquest mòdul parteixen del .NET Framework 3.5 i s'han de fer amb la versió 2008 o superiors de Visual Studio o Visual Web Developer.

Objectius

Amb l'estudi d'aquest mòdul, assolireu els objectius següents:

1. Mostrar l'ús de tecnologia AJAX en un dels entorns de treball més utilitzats avui en dia: .NET

2. Conèixer què és ASP.NET i la seva implementació AJAX.

3. Familiaritzar-se amb l'entorn de treball integrat Visual Web Developer.

4. Crear llocs web i dissenyar pàgines ASP.NET interactives que responguin a les diferents accions dels usuaris.

5. Comprendre el cicle de vida d'una pàgina ASP.NET entre diferents tipus de peticions.

6. Estudiar com s'han d'afegir funcionalitats AJAX en una pàgina ASP.NET. i fer alguns exemples.

1. Introducció a ASP.NET

A continuació, s'expliquen alguns conceptes bàsics sobre el sistema ASP.NET.

1.1. L'entorn de treball .NET

L'entorn de treball[1] .NET és un conjunt de tecnologies dissenyades per al desenvolupament i l'execució de diferents tipus d'aplicacions com són portals web, aplicacions d'escriptori i serveis de Windows.

(1)En anglès, *framework*.

Aquesta plataforma es pot dividir en tres components principals:

1) Common Language Runtime (CLR): és el nucli i motor de la plataforma. Permet executar aplicacions escrites per a aquesta plataforma i proveeix d'un entorn d'execució amb un conjunt de serveis com ara la seguretat entre processos, la gestió de memòria, l'administració d'excepcions, etc. El CLR[1] està basat en una màquina virtual que interpreta un codi intermedi anomenat MSIL[2]. Qualsevol aplicació .NET es compila en aquest codi i aconsegueix així independència del maquinari i del sistema operatiu.

(2)*MSIL* és la sigla de *Microsoft intermediate language*.

MSIL

MSIL és el codi generat en compilar tota aplicació .NET independentment del llenguatge utilitzat. Es pot comparar al Bytecode de Java.

Màquina virtual

Una màquina virtual és un entorn d'execució de processos que ofereix una abstracció del maquinari i del sistema operatiu.

Un exemple molt conegut és la màquina virtual de Java, que permet l'execució d'aplicacions en plataformes heterogènies.

2) Llenguatges de programació: entre els quals s'inclou C# i Visual Basic. Hi ha una interfície comuna que permet la implementació de llenguatges per a la plataforma .NET. En l'actualitat hi ha implementacions per a diversos llenguatges com Cobol, Perl, + + C, Delphi, Fortran i J#.

3) Biblioteca de classes base: conjunt de classes que ofereixen suport per a operacions bàsiques com encriptació de dades, *sockets*, manipulació d'arxius, gestió de finestres, etc.

En l'actualitat, Microsoft només ha implementat versions del .NET Framework per a la seva família de sistemes operatius. Hi ha una implementació lliure per als sistemes Linux i MAC anomenada Projecte Mono.

1.2. ASP.NET

ASP.NET és una tecnologia basada en el .NET Framework per a crear aplicacions web interactives. Ofereix un model de programació molt similar al de les aplicacions d'escriptori mitjançant les pàgines ASP.NET.

Les pàgines ASP.NET estan compostes per una part visual que s'executa en el costat del client i una part lògica que s'executa en el costat del servidor per a respondre a les diferents accions executades pels usuaris.

La part visual combina HTML estàtic amb controls que són processats pel servidor. El resultat són interfícies complexes que són transformades a tecnologies suportades pels navegadors en sol·licitar una pàgina. El codi executat en el costat del servidor pot ser qualsevol admès pel .NET Framework, com C# o Visual Basic. Aquest permet respondre a les accions dels usuaris entre les diferents peticions.

Controls

Els controls són elements visuals o funcionals que formen part d'una interfície gràfica com ara botons, quadres de text, etc.

1.3. ASP.NET AJAX

ASP.NET AJAX va ser inclòs en el .NET Framework 3.5 juntament amb Visual Studio 2008.

Escriure aplicacions AJAX sense l'ajuda d'un entorn de treball requereix coneixements profunds de JavaScript, DOM[3], XML, XSLT i l'objecte XMLHttpRequest. D'altra banda, s'han de tenir en compte les diferents implementacions del DOM entre navegadors. ASP.NET AJAX és una col·lecció de tecnologies del costat client i servidor que permeten als desenvolupadors web crear d'una manera senzilla aplicacions web riques i interactives sense que siguin necessaris coneixements profunds de les tecnologies emprades per AJAX.

(3) *DOM* és la sigla de *document object model.*

XMLHttpRequest

XMLHttpRequest és un objecte accessible des de JavaScript per a fer peticions HTTP asíncrones. Les peticions asíncrones permeten obtenir dades del servidor sense interferir amb la pàgina activa. Podeu veure en detall l'objecte XMLHttpRequest en el mòdul "AJAX" d'aquesta assignatura.

ASP.NET AJAX inclou els components següents:

1) ASP.NET AJAX Library: és una biblioteca JavaScript del costat client que ofereix un conjunt de funcionalitats que faciliten el desenvolupament web compatible amb els navegadors més populars. Ofereix una capa amb objectes i funcions JavaScript per a interactuar amb el DOM, fer peticions asíncrones, actualitzar parts d'una pàgina, etc.

2) ASP.NET AJAX Control Toolkit: aquest paquet conté una extensa col·lecció de components del costat del servidor que proveeixen de funcionalitats AJAX.

2. Primers passos amb ASP.NET

En aquest apartat, s'expliquen alguns aspectes principals de l'entorn de treball integrat (IDE) Visual Web Developer i com es pot crear un lloc web.

2.1. Introducció

Visual Web Developer és un IDE dissenyat per a desenvolupar aplicacions web en la plataforma .NET. És una versió reduïda del paquet Visual Studio i disposa d'una versió gratuïta, encara que més limitada, anomenada Visual Web Developer Express.

Visual Studio IDE

Un IDE (*integrated development environment*, 'entorn de desenvolupament integrat') és un programa per al desenvolupament d'aplicacions que ofereix un conjunt d'eines per a facilitar aquesta tasca. Visual Studio és un IDE creat per Microsoft per al desenvolupament d'aplicacions per a Windows. Es pot utilitzar per a dissenyar tot tipus d'aplicacions: web, aplicacions d'escriptori, etc.

En aquest punt es crearà una petita aplicació web per a familiaritzar-se amb qualsevol d'aquests IDE i amb la programació de pàgines ASP.NET. També s'utilitzarà el dissenyador WYSIWYG que incorpora Visual Studio, que permetrà crear formularis web d'una manera ràpida i visual.

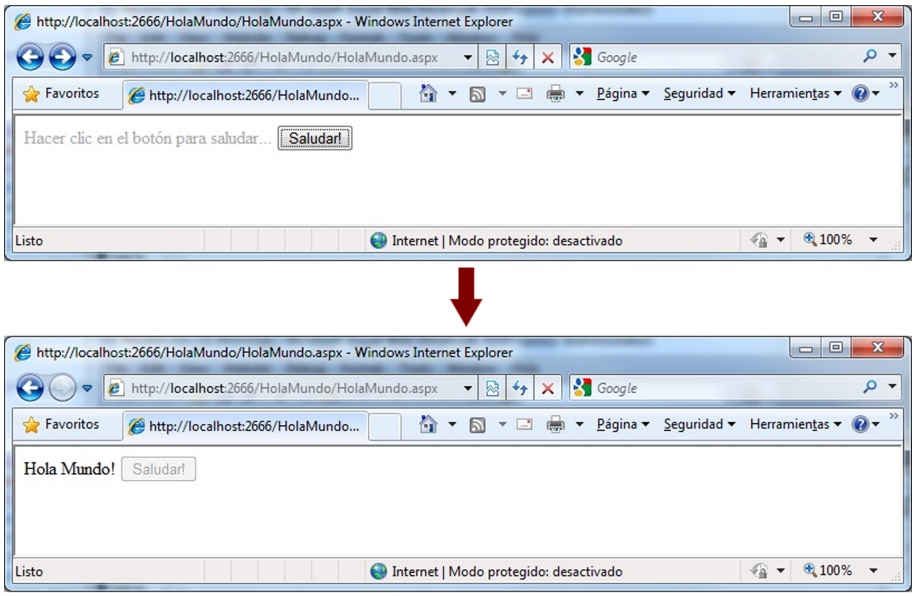
WYSIWYG

WYSIWYG és la sigla de *what you see is what you get*. Es podria traduir com 'el que veus és el que obtens', que fa referència a la característica d'un editor de poder mostrar en temps de disseny el resultat visual d'un codi d'una interfície gràfica.

Els llocs web que es desenvoluparan durant aquest mòdul s'executaran sota Visual Studio, que incorpora un servidor de desenvolupament que allotjarà les nostres aplicacions.

L'exemple consta d'un formulari amb un text i un botó. En fer clic sobre el botó, es canviarà el text per "Hola Mundo!", com es mostra en la figura 1.

Figura 1. Resultat final de l'exemple



2.2. Creació del nou lloc web

Un lloc web és un model de projecte que utilitza tot el contingut d'un directori que conté els arxius d'una aplicació web: pàgines ASP.NET, arxius de configuració, codi de servidor, etc.

Per a crear un lloc web es poden seguir els passos següents:

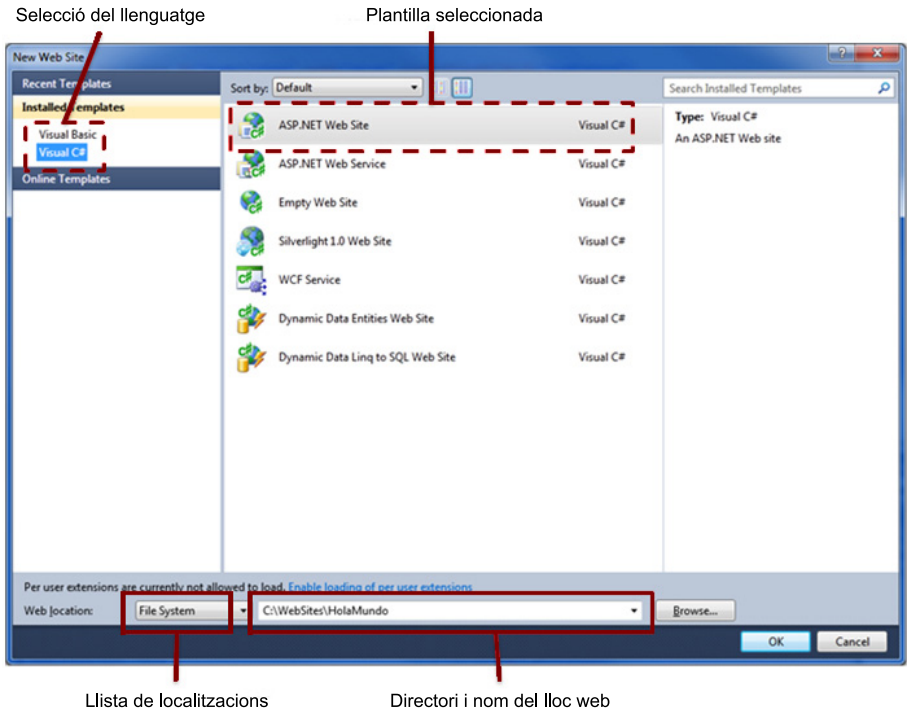
1) Per començar, s'ha d'obrir l'IDE Visual Studio o Visual Web Developer.

2) El nou lloc web es pot crear de dues maneres: la primera és fent clic en l'enllaç *New Web Site*... en la pàgina inicial de l'IDE. La segona és accedint al menú superior i fent clic a File > New Web Site. En qualsevol dels dos casos, es mostrarà un quadre de diàleg com el de la figura 2.

Creació del lloc web

Els exemples es poden fer tant amb Visual Studio com amb Visual Web Developer 2008 o posterior. El llenguatge utilitzat durant el mòdul és Visual C#.

Figura 2. Creació d'un nou lloc web



3) Seleccionar la plantilla ASP.NET Web Site.

Plantilles

Les plantilles són formes predefinides d'elements d'un projecte o de projectes per si mateix. Permeten crear projectes o parts d'un projecte d'una manera automàtica i predefinida a partir d'unes bases: fitxers, codi, configuració, etc.

Les plantilles mostrades dependran de les característiques seleccionades durant la instal·lació de Visual Studio o Visual Web Developer.

4) Seleccionar el llenguatge Visual C# de la llista dels llenguatges instal·lats.

5) La part inferior mostra un conjunt de controls per a seleccionar la localització en què es trobaran els fitxers del lloc web. La llista de localitzacions permet elegir entre File System, HTTP o FTP. File System permet disposar del lloc web en el sistema local de fitxers. Si se selecciona HTTP, l'IDE crearà un directori virtual nou en l'IIS en l'URL indicat. FTP permet desenvolupar el lloc web des d'un lloc remot accessible mitjançant el protocol FTP. Per a aquest projecte se seleccionarà File System i s'establirà la ruta volguda, en què l'últim directori de la ruta serà `HolaMundo`, que serà el nom del lloc web.

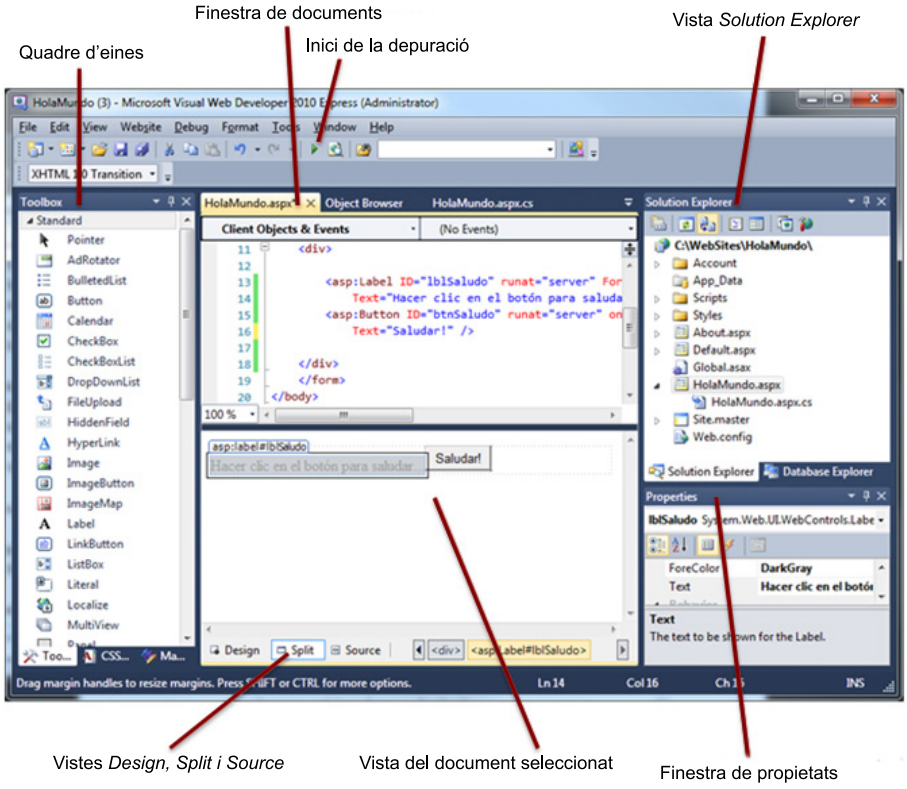
6) Fer clic en OK o Aceptar, segons l'idioma de l'IDE.

En fer els passos anteriors, es crearà el directori `HolaMundo` en el qual hi haurà tots els fitxers de l'aplicació. Durant el desenvolupament s'accedirà a cadascun dels fitxers del lloc web mitjançant l'explorador de solucions de l'IDE i la finestra de documents, indicats en la figura 3:

Portabilitat del lloc web

El lloc web pot ser portat a un altre equip copiant el directori. Des de Visual Studio o Visual Web Developer s'obrirà accedint-hi des del al menú File > Open Web Site.

Figura 3. Finestra principal de Visual Web Developer 2010 Express



Després de crear el lloc web, la finestra principal té un aspecte com es mostra en la figura 3, de la qual es poden destacar les parts de l'entorn següents:

1) Quadre d'eines: mostra els controls i elements HTML que poden ser arrossegats a una pàgina. Es mostren agrupats per categories.

2) Finestra de propietats: permet modificar les propietats d'alguns elements com, per exemple, controls, pàgines, elements HTML, etc.

3) Finestra de documents: mostra els documents oberts en finestres organitzades en fitxes. La finestra de documents d'una pàgina ASP.NET mostra en la part inferior tres fitxes per a seleccionar les vistes següents:

- ***Design***: la vista de disseny simula el resultat final de la pàgina en temps de desenvolupament a mesura que es fan modificacions. Sobre aquesta vista es poden arrossegar controls des del quadre d'eines i el codi es generarà de manera automàtica. També permet seleccionar elements d'una pàgina per a veure'n les propietats en la finestra de propietats.

- ***Source***: mostra el codi font de la pàgina.

- ***Split***: mostra una finestra dividida en dues seccions, l'una per a la vista *Design* i l'altra per a la vista *Source*.

4) Explorador de solucions: mostra tots els arxius i directoris pertanyents al lloc web. En fer doble clic sobre un fitxer en aquesta vista, aquest es mostrarà dins de la finestra de documents.

Parts de l'entorn

Les diferents vistes, i també el quadre d'eines o l'explorador de solucions, es poden distribuir en diferents parts de la finestra principal arrossegant-les amb el ratolí.

2.3. Pàgines web ASP.NET

Les pàgines web ASP.NET són la interfície d'usuari d'una aplicació web. Es componen d'una part visual i una part lògica:

Formularis web

Les pàgines ASP.NET també són anomenades *formularis web* (*web forms*) atesa la similitud de desenvolupament entre aquestes i els formularis de les aplicacions d'escriptori.

- **Part visual**: correspon a les marques i el codi executat en navegadors com HTML, codi JavaScript i CSS. També conté controls de servidor representats com etiquetes HTML. Aquests controls representen elements visuals o funcionals de la pàgina i es denominen *de servidor*, a causa que són processats i convertits en servidor abans d'enviar la pàgina al navegador.

  Processar una pàgina

  El codi generat en processar una pàgina pot estar adaptat al client que fa la petició. D'aquesta manera, les pàgines ASP.NET són compatibles amb diferents navegadors i dispositius mòbils.

- **Part lògica**: es compon de codi executat en el servidor que permet donar resposta a les accions efectuades en la pàgina. Aquest codi pot estar situat en el mateix arxiu com un bloc *script* incrustat, o en un fitxer separat amb extensió `.aspx.cs` o `.aspx.vb`, si s'usen els llenguatges Visual C# o Visual Basic. En cas que el codi estigui situat en un fitxer separat, aquest s'anomena *arxiu code-behind* o *arxiu de codi subjacent*.

Llenguatges

El llenguatge de la part de servidor pot ser qualsevol compatible amb *common language runtime* de l'entorn de treball.

A continuació, es crearà una pàgina nova al lloc web. Els passos que cal seguir són els següents:

1) Accedir al menú File > New File. Apareixerà una pantalla amb una llista de plantilles i s'haurà de seleccionar "Web Form". En la part inferior, en el quadre de text etiquetat com a "Name", s'escriurà el nom de la pàgina nova: `HolaMundo.aspx`.

2) Fer clic sobre OK. Es crearà el formulari web nou que apareixerà en la vista Explorador de solucions amb dos fitxers: `HolaMundo.aspx` i `HolaMundo.aspx.cs`.

La vista del document mostrarà el contingut de la pàgina `HolaMundo.aspx` que contindrà el codi següent:

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

<!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>

    </div>
    </form>
</body>
</html>
```

La pàgina creada té un aspecte molt semblant al d'una pàgina HTML corrent excepte per la directiva `<%@ Page %>` en la part superior. Aquesta indica algunes característiques de la pàgina, com ara el llenguatge de programació utilitzat en el costat del servidor o el fitxer en què hi ha el codi. Al cos hi ha el bloc `<form>`, dins del qual aniran col·locats tots els controls de servidor, que s'expliquen en el subapartat següent.

2.4. Controls

Els controls són components utilitzats per a construir interfícies gràfiques. La majoria són visuals i l'usuari hi pot interactuar: botons, quadres de text, calendaris, etc. En canvi, hi ha altres controls que fan altres tasques en la interfície: manipulació de dades, validació de camps de text, etc.

Dins d'una pàgina ASP.NET, podem trobar els tipus de controls següents:

- **Controls HTML**: són els controls simples HTML com taules, divisions, botons, etc.

- **Controls de servidor ASP.NET**: són els controls que s'executen en sol·licitar-se la pàgina en la qual estan allotjats. Hi ha una varietat homogènia a la dels controls bàsics HTML com taules, botons, llistes, combos, etc. En ser processats en el servidor, ofereixen els avantatges següents:
  - El programador té accés a les propietats i als mètodes del control com si es tractés d'un objecte sense haver de codificar en HTML.
  - En generar-se d'una manera dinàmica, aquests s'adapten al navegador que fa la petició. Aquesta característica ofereix més compatibilitat entre navegadors que poden tenir implementacions del DOM diferents.

- **Controls ASP.NET AJAX**: tenen tots els avantatges dels controls de servidor i afegeixen funcionalitats extres per a oferir més dinamisme a les pàgines.

- ***User Controls* i controls personalitzats**: els usuaris poden crear controls propis que poden ser afegits a la paleta de controls de Visual Studio.

Vegeu també

Vegeu més informació sobre el DOM en el mòdul "AJAX" d'aquesta assignatura.

Hi ha altres controls que afegeixen diferents funcionalitats a les pàgines, com la validació de camps a partir d'una expressió regular, controls d'autenticació i seguretat, etc.

Els passos següents mostren com s'afegeixen dos controls de servidor a la pàgina creada en el subapartat anterior:

1) Seleccionar la vista *Design* en la part inferior de la vista del document `HolaMundo.aspx`. Dins d'aquesta, cal seleccionar el requadre de l'element *div*.

2) Obrir el quadre d'eines i arrossegar un control Label cap a la superfície de disseny dins de la zona seleccionada en el punt anterior. Cal fer la mateixa acció amb el control Button.

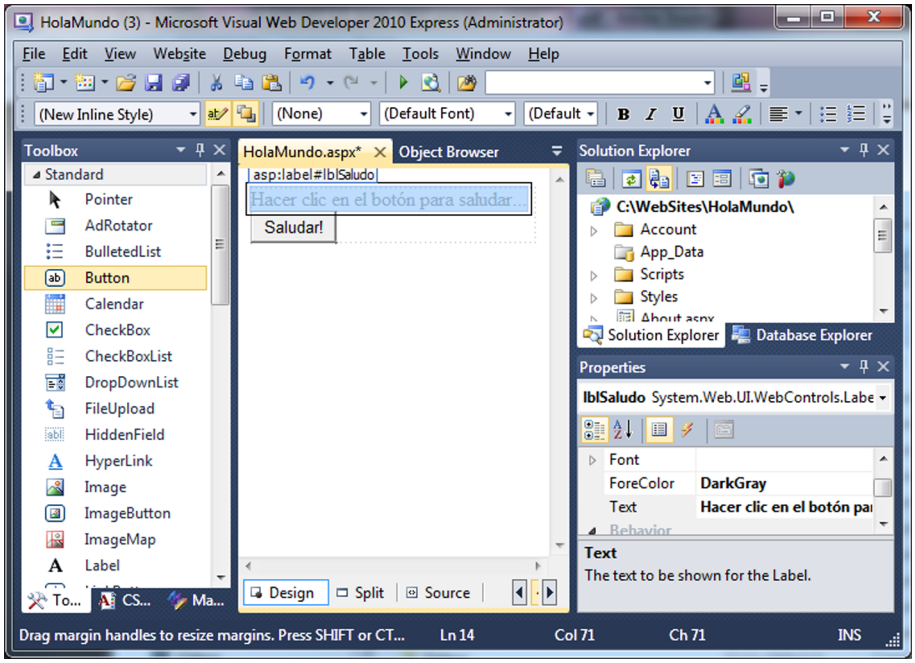
3) A continuació, seleccionar el botó i establir les propietats *ID* i *Text* amb els valors "btnSaludo" i "Saludar!", respectivament.

Prefixos en noms de variables

Per convenció, s'utilitza un prefix en els noms de variables per a identificar-ne ràpidament el tipus de dada. Per exemple, els prefixos *lbl* i *btn* permeten identificar les variables de tipus Label i Button, respectivament.

Una vegada afegits els controls, la pàgina tindrà un aspecte en vista de disseny similar al de la figura 4.

Figura 4. Vista de disseny i propietats del Label del formulari



Per a comprovar el codi generat es pot accedir a la vista *Source* de la finestra del document `HolaMundo.aspx`:

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

<!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>

        <asp:Label ID="lblSaludo" runat="server" ForeColor=" DarkGray"
            text="Hacer clic en el botón para saludar..."></asp:Label>
        <asp:Button ID="btnSaludo" runat="server" Text=" Saludar!" />

    </div>
    </form>
```

```
</body>
</html>
```

En l'interior de l'etiqueta `<form>` hi ha els dos controls de servidor arrossegats en vista de disseny: `<asp:Label>` i `<asp:Button>`. Tenen una sintaxi semblant a HTML però amb algunes particularitats: tots els controls de servidor tenen el prefix `asp:` i declaren un atribut anomenat `runat` amb el valor `server` perquè siguin processats pel servidor ASP.NET en sol·licitar la pàgina. Les propietats dels controls establertes d'una manera visual, es declaren com atributs. Per exemple, el control Label al qual s'ha assignat un text i un color, defineix aquestes propietats mitjançant els atributs *Text* i *ForeColor*.

Per a veure el resultat de la pàgina des d'un navegador, es pot executar l'aplicació de les maneres següents:

1) Clic en el menú Debug > start Debugging.

2) Clic en la icona Start Debugging[4], assenyalada en la figura 3 (inici de la depuració).

(4)

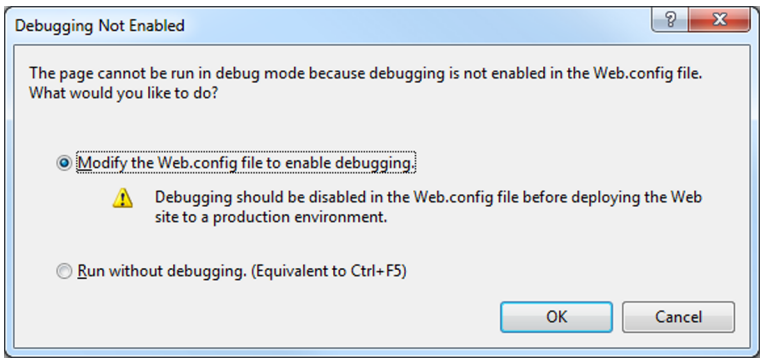
Si és la primera vegada que s'executa l'aplicació, l'IDE detectarà que no està configurada per a executar-se en mode de depuració i mostrarà un missatge com el que apareix en la figura 5.

Mode de depuració

El mode de depuració permet establir punts de ruptura en què l'execució s'aturarà. Els punts de ruptura (*breakpoints*) es poden definir per a línies de codi específiques o per a certes condicions (per exemple, que una variable canviï a un cert valor).

La detenció de l'execució permet observar variables, registres i pila de crides.

Figura 5. Activació de depuració per al lloc web



El diàleg ofereix dues opcions que permeten activar la depuració en el fitxer `web.config` o executar l'aplicació sense depuració. En aquest cas, se seleccionarà la primera opció i es farà clic sobre OK.

Fitxer web.config

`web.config` és el principal arxiu de configuració d'una aplicació ASP.NET en què es defineixen els mòduls que requereix l'aplicació, configuracions de seguretat i altres opcions.

Apache disposa d'un fitxer similar anomenat httpd.conf.

Cada vegada que s'iniciï l'aplicació, s'obrirà el navegador per defecte en l'URL següent:

```
http://localhost:puerto/HolaMundo/HolaMundo.aspx
```

Predeterminar la pàgina

Es pot predeterminar la pàgina `HolaMundo.aspx` perquè es mostri en accedir a l'URL `http://localhost:port`. Per a això, s'ha de fer clic sobre el botó dret del ratolí sobre la pàgina en la vista Explorador de solucions i seleccionar "Set as Start Page".

`localhost` fa referència a la màquina local en la qual s'executa el servei que allotja l'aplicació. El port indica l'adreça del *socket* que rep les peticions per a aquesta aplicació concreta. `HolaMundo` indica el lloc web i `HolaMundo.aspx` fa referència a la pàgina dins del lloc web.

Número de port

El número de port es genera de manera aleatòria cada vegada que s'executa l'aplicació.

La pàgina retornada pel servidor presenta algunes diferències després de ser processada. Això es pot comprovar visualitzant el codi font des del navegador. A continuació, es mostra part de la pàgina `HolaMundo.aspx` una vegada enviada al navegador:

```
...
<div>
    <span id="lblSaludo" style="color:DarkGray;">
        Hacer clic en el botón para saludar
    </span>
    <input type="submit" name="btnSaludo" value="Saludar!" id="btnSaludo" />
</div>
...
```

El control Label s'ha substituït per `<span>` i el color assignat en les propietats s'ha convertit en un atribut `style`. El text de la propietat *Text* s'ha introduït en l'interior de `<span>`. El control Button s'ha transformat en un `<input>` i el text del botó s'ha establert en l'atribut `value`.

La pàgina mostrada no ofereix cap interacció. En fer clic sobre el botó, s'envia una petició i es retorna la mateixa pàgina al navegador sense canvis. En el subapartat següent s'explica com es pot fer una acció determinada en clicar sobre el botó perquè el control Label canviï el text.

2.5. Establir un manipulador per a l'esdeveniment clic del botó

El model d'esdeveniments de les pàgines ASP.NET permet respondre a les diferents accions originades des del client. Per exemple, un clic originat des de la pàgina del client permet que s'executi cert codi en el servidor i rebre una altra pàgina de resposta.

Les pàgines ASP.NET i els controls de servidor poden generar esdeveniments, ja sigui en resposta a una acció de l'usuari o des d'un altre procés independent.

Processos independents

Un control Timer permet generar esdeveniments cada cert interval de temps sense la intervenció de l'usuari.

La majoria d'esdeveniments provoquen un POST de la pàgina en el qual s'envien les dades del formulari. El servidor rep la petició, processa els esdeveniments produïts i executa el codi associat dins de mètodes de classe, anomenats *manipuladors d'esdeveniments*. Fet això, s'envia la mateixa pàgina de tornada. Encara que dins del manipulador d'esdeveniment es pot readreçar cap a una altra pàgina diferent.

Alguns esdeveniments que tenen lloc amb freqüència com *onmouseover*, s'han de gestionar des del client mitjançant JavaScript. Si no es fes així, la pàgina es recarregaria contínuament, cosa que provocaria un efecte negatiu per a l'usuari i una sobrecàrrega per al servidor. Altres esdeveniments no generen una petició en el mateix instant, com pot ser un clic en una casella de selecció[5] o prémer una tecla en un quadre de text[6]. En aquest cas, l'esdeveniment s'emmagatzema en el client i s'envia en la petició següent.

(5)En anglès, *checkbox*.

(6)En anglès, *textbox*.

Petició automàtica

Es pot provocar una petició automàtica si es prem una tecla en un control Textbox i s'estableix la seva propietat *AutoPostback* a *true*.

L'objectiu de l'exemple és que el control Label, que inicialment es mostra en gris, canviï el seu text i color en fer clic al botó. Per a això, s'ha de definir un manipulador de l'esdeveniment clic en el qual es faran els canvis sobre el Label.

Hi ha dues maneres de definir un manipulador a l'esdeveniment clic del botó:

1) Fer doble clic sobre botó en la vista de disseny de la pàgina.

2) Seleccionar el botó en la vista de disseny i accedir a les seves propietats. En la finestra de propietats, fer clic sobre la icona de creació d'un manipulador d'esdeveniments[7]. Es mostrarà una llista amb dues columnes amb els esdeveniments disponibles. Situar el cursor sobre l'acció Click en la columna esquerra i fer doble clic en la columna dreta. Es crearà automàticament un mètode dins del fitxer `HolaMundo.aspx.cs` amb el nom `btnSaludo_Click` i es mostrarà el contingut del fitxer dins de la vista de document. Dins d'aquest mètode s'introduirà el següent codi C#:

Manipulador de l'esdeveniment

En fer doble clic sobre un control en la vista de disseny, es crea de manera automàtica el manipulador de l'esdeveniment que té per defecte. En el cas d'un Button, l'esdeveniment per defecte és Click.

(7)

```
public partial class HolaMundo : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void btnSaludo_Click(object sender, EventArgs e)
    {
        //S'estableix el color de lletra al control Label
        lblSaludo.ForeColor = System.Drawing.Color.Black;
        //S'estableix el text Hola, món! al control Label
        lblSaludo.Text = "Hola Mundo!";
        //Ja s'ha mostrat la salutació, es deshabilita el botó
```

```
        //perquè no es repeteixi aquesta acció
        btnSaludo.Enabled = false;
    }
}
```

Podem veure que .NET crea una classe per a la pàgina web que conté els controls de la pàgina i un conjunt de mètodes per a implementar els manipuladors dels esdeveniments. El nom del mètode associat a l'esdeveniment està compost per l'ID del botó que s'hagi establert en les propietats, més el tipus d'esdeveniment. En cridar-se al mètode, es faran les tres accions següents, una per cada instrucció:

1) La primera instrucció accedeix a la propietat *ForeColor* del control Label i estableix el color negre, que inicialment era gris.

2) La segona instrucció estableix en la propietat Text del Label el text "Hola Mundo!".

3) L'última deshabilita el botó perquè no faci cap acció més.

Com es pot comprovar, l'accés als controls es fa mitjançant membres de classe que tenen com a nom l'ID establert en les seves propietats. En aquest cas, la classe `HolaMundo` representa la pàgina, i els membres `lblSaludo` i `btnSaludo`, els controls Label i Button.

Herència

Tota pàgina ASP.NET hereta de System.Web.UI.Page.

En fer clic sobre el botó, el navegador farà una petició i hi inserirà la informació necessària perquè el servidor conegui els esdeveniments produïts. La petició es processa i es fa la crida al manipulador, que durà a terme els canvis en les propietats dels controls `lblSaludo` i `btnSaludo`. Finalment, es torna a processar la pàgina i es genera el nou codi HTML, que serà retornat al client amb els canvis nous.

2.6. Cicle de vida d'una pàgina

Quan el servidor envia una pàgina ja processada, destrueix totes les instàncies que la representaven: instàncies de petició, de pàgina, controls, etc. Si el mateix usuari torna a sol·licitar la mateixa pàgina, es trobarà amb les propietats inicials sense els canvis que havia pogut anar duent a terme. Això s'anomena **execució sense estat** i es deu al fet que la comunicació entre client i servidor ASP.NET es fa sota el protocol HTTP, que està orientat a transaccions sense mantenir la informació de peticions anteriors.

Com s'ha vist en l'exemple anterior, en fer clic sobre el botó, era possible accedir a les propietats de la pàgina i la resta de controls des del codi del servidor. La informació de la pàgina es manté entre els diferents esdeveniments que provoquen peticions d'anada i tornada. La manera com es fa és la següent: els controls són dins de l'etiqueta `<form>`. Quan es desencadena un esdeveniment, es crida al mètode `submit()` de `<form>` i tota la informació dels controls s'envia de tornada al servidor dins del cos d'una petició POST.

Postbacks

Les peticions d'anada i tornada són conegudes com a *postbacks*. Aquest terme fa referència a una petició HTTP POST que conté la informació de la pàgina retornada (*back*).

L'enumeració següent explica pas a pas el cicle de vida d'una pàgina:

1) L'usuari sol·licita una pàgina i el navegador fa una petició HTTP GET al servidor ASP.NET.

Vegeu també

Vegeu el protocol HTTP en el mòdul "Introducció a la programació web avançada" d'aquesta assignatura.

2) El servidor crea la instància de la pàgina i dels controls. Posteriorment, la processa i l'envia de tornada al navegador.

3) L'usuari modifica el valors dels controls en el navegador com, per exemple, quadres de text, llistes de selecció, botons d'opció, etc.

4) L'usuari fa una acció que desencadena un esdeveniment, com, per exemple, un clic en un botó.

5) El navegador fa una petició POST al servidor i envia tota la informació del formulari: valors dels quadres de text, estat dels botons d'opció, de les llistes de selecció, etc.

6) El servidor processa la petició i torna a crear una instància de la pàgina a partir dels valors obtinguts. D'aquesta manera, la informació del formulari serà accessible des del codi del servidor.

7) Una vegada creada la instància de la pàgina, es crida als mètodes associats als esdeveniments desencadenats en la pàgina. En aquest cas, es crida al mètode associat a l'esdeveniment clic del botó clicat.

8) El servidor processa la pàgina nova i l'envia de tornada al client.

3. Introducció a AJAX en ASP.NET

En aquest apartat es fa una introducció al desenvolupament amb AJAX en pàgines ASP.NET. Inicialment, s'explica la diferència entre peticions síncrones i asíncrones amb l'ús d'AJAX. Posteriorment, s'expliquen els controls ASP.NET AJAX més estesos.

Aquest apartat exposa tres exemples, dos per a comprendre la diferència entre peticions síncrones i asíncrones, i un últim exemple, que mostra com s'han d'utilitzar els controls AJAX que s'expliquen, inclòs un temporitzador per a generar esdeveniments asíncrons cada cert interval de temps.

3.1. Peticions síncrones i asíncrones

Hi ha dos tipus de peticions en ASP.NET:

1) Peticions síncrones. Són les peticions més comunes que es poden desencadenar en sol·licitar una pàgina, en fer clic sobre un enllaç, en desencadenar-se un esdeveniment en un control, etc. La resposta és una pàgina nova que reemplaça l'actual. L'usuari podrà apreciar que tota l'àrea de client del navegador s'actualitza.

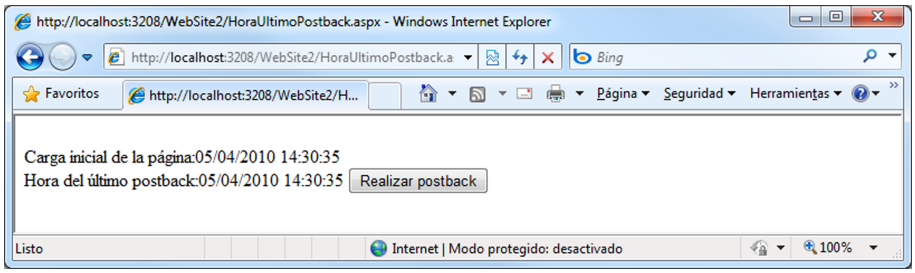
2) Peticions asíncrones. Són les peticions fetes mitjançant AJAX de manera asíncrona, és a dir, sense interferir en la interacció amb la pàgina. Algunes poden simplement enviar informació al servidor i d'altres fer una recàrrega parcial de la pàgina. L'usuari podrà apreciar que només canvia una part de la pàgina.

L'exemple següent mostra la diferència entre peticions síncrones i asíncrones amb ASP.NET. Consta d'una pàgina que mostra l'hora en què la pàgina s'ha carregat per primera vegada i l'hora de l'última vegada en què s'ha premut un botó de la pàgina. Les peticions d'anada i tornada es generaran mitjançant l'esdeveniment Click d'un botó. L'exemple té dues parts: la primera per a l'actualització total de la pàgina a causa de les peticions síncrones i la segona per a les actualitzacions parcials amb peticions asíncrones mitjançant AJAX.

3.1.1. Actualització total de la pàgina

El formulari tindrà un aspecte semblant al de la figura 6.

Figura 6. Data i hora de petició inicial i petició d'anada i tornada



Per començar, s'ha de crear un lloc web nou i una pàgina ASP.NET nova amb el nom `HoraUltimoPostback.aspx`. Una vegada creada, s'ha d'accedir a la vista de disseny i seguir els passos següents:

1) Arrossegar el control ScriptManager situat en la secció AJAX Extensions del quadre d'eines sobre la superfície de la pàgina. Aquest control habilitarà l'ús d'AJAX en la pàgina.

Vegeu també

Vegeu el control ScriptManager en el subapartat 3.3 d'aquest mòdul.

2) Situar el cursor després del control ScriptManager fent clic i prémer la tecla Enter per a generar un salt de línia. A continuació, escriure el text "Carga inicial de la página:".

3) Arrossegar un control Label a la dreta del text escrit. Establir la propietat ID en `lblCargaInicial`.

4) Situar el cursor després del control Label anterior i pressionar Enter. Introduir el text "Hora del último postback:".

5) Arrossegar un altre control Label a la dreta del text anterior i establir la seva propietat ID en `lblPostback`.

6) Arrossegar un control Button a continuació del control Label anterior. Establir en les propietats Text i ID els valors "Realizar postback" i `btnPostback`, respectivament.

Una vegada aquí, la interfície està preparada. Ara faltarà afegir un bloc de codi que permeti actualitzar els dos controls Label. En la vista Explorador de solucions, fer clic sobre la fletxa o el símbol + situat a l'esquerra del fitxer `HoraUltimoPostback.aspx`. Apareixerà el fitxer `HoraUltimoPostback.aspx.cs`, i fer-hi doble clic per a obrir-lo en la finestra de document.

El document `HoraUltimoPostback.aspx.cs` conté la classe `HoraUltimoPostack`, que inclou un mètode anomenat `Page_Load`. Inserir dins d'aquest mètode el codi següent:

```
protected void Page_Load(object sender, EventArgs e)
{
```

```
        lblPostback.Text = DateTime.Now.ToString();
        if (!IsPostBack)
        {
            lblCargaInicial.Text = DateTime.Now.ToString();
        }
    }
```

El mètode `Page_Load` s'executa en cada petició de la pàgina, ja en sigui la càrrega inicial o un reenviament originat per algun esdeveniment. Per tant, aquest mètode s'executarà en la càrrega inicial i també cada vegada que es faci clic sobre el botó del formulari. La primera línia estableix el text de l'hora en el control Label `lblPostback`. La condició següent avalua si la petició és un reenviament de pàgina o una càrrega inicial mitjançant la propietat `IsPostBack`. Si `IsPostBack` és *false* implicarà que la pàgina es carrega per primera vegada i també s'establirà l'hora en el Label `lblCargaInicial`. Si `IsPostBack` és *true*, es tractarà d'un reenviament de pàgina i s'ignorarà la instrucció.

Propietat IsPostBack

La propietat IsPostBack s'hereta de la classe System.Web.UI.Page.

Si s'executa l'exemple, es comprovarà que inicialment les dues etiquetes tenen la mateixa hora. Cada vegada que es faci clic sobre el botó "Realizar postback", la pàgina es reenviarà i només se n'actualitzarà la segona línia.

Tota petició feta en aquest exemple és síncrona i serveix per a veure la diferència entre una càrrega inicial i peticions d'anada i tornada d'una mateixa pàgina. En qualsevol cas, la pàgina obtinguda reemplaça la pàgina en curs del navegador.

3.1.2. Actualització parcial de la pàgina

En aquest segon apartat de l'exemple s'afegeix una recàrrega parcial de la pàgina `HoraUltimoPostback.aspx` mitjançant controls ASP.NET AJAX. Per continuar, cal obrir la vista de disseny de la pàgina i fer el següent:

1) Situar el cursor després del botó afegit en l'últim pas de l'exemple anterior i prémer Enter.

2) Localitzar el control UpdatePanel situat dins de la categoria AJAX Extensions del quadre d'eines. Arrossegar-lo dins de la vista de disseny, en la nova línia afegida en el pas anterior. El control és un contenidor i es mostrarà com un requadre en el qual es podran afegir més controls. UpdatePanel és un contenidor que ofereix funcionalitat AJAX en una pàgina i permet actualitzar-ne el contingut sense recarregar la pàgina completa.

Vegeu també

Vegeu el control UpdatePanel en el subapartat 3.3 d'aquest mòdul.

3) Situar el cursor dins del control UpdatePanel i escriure el text "Hora de la última actualización parcial:".

4) Arrossegar un control Label a continuació del text introduït en el pas anterior. Establir en la propietat ID el valor `lblRecargaParcial`. En la propietat Text, establir el text "Sin recarga parcial".

5) A continuació del Label anterior, cal afegir un botó. Establir en les propietats Text i ID els valors "Recarga parcial" i `btnRecargaParcial`, respectivament.

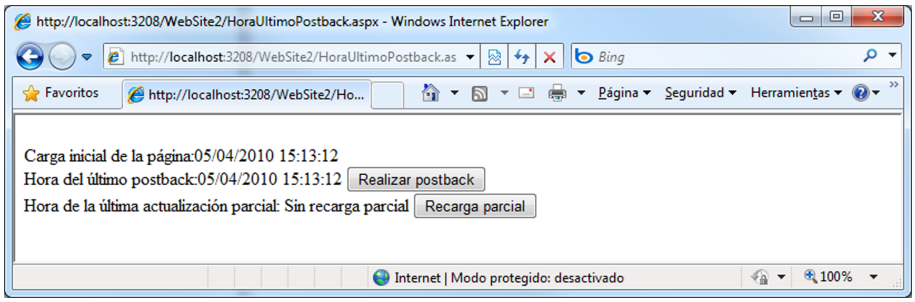
És important que els elements introduïts en els passos tercer, quart i cinquè siguin dins del control UpdatePanel. Finalment, es farà doble clic sobre el botó que s'acaba de crear per a afegir un manipulador d'esdeveniment. Dins del mètode associat, s'escriurà el codi següent:

```
protected void btnRecargaParcial_Click(object sender, EventArgs e)
{
    lblRecargaParcial.Text = DateTime.Now.ToString();
}
```

Es cridarà al mètode cada vegada que es faci clic sobre el botó situat dins del control UpdatePanel. Aquest actualitzarà de manera asíncrona el text del Label `lblRecargaParcial` amb la data i hora de la petició.

Si s'executa l'aplicació, apareixerà un formulari semblant al que es mostra a continuació:

Figura 7. Resultat de l'exemple



El botó nou afegit, situat dins del control UpdatePanel, igual que el botó anterior, fa una petició i envia la informació del formulari, amb la diferència que la petició és asíncrona. Això implica que només la part compresa pel control UpdatePanel serà actualitzada amb la resposta.

3.2. Controls AJAX en ASP.NET

A continuació, s'expliquen els controls de servidor ASP.NET AJAX més utilitzats.

- **ScriptManager**

Per a utilitzar AJAX en una pàgina ASP.NET cal afegir-hi el control ScriptManager una única vegada. Aquesta etiqueta gestiona els recursos *script* que han de ser enviats al client. L'etiqueta creada en inserir el control en una pàgina és la següent:

```
<asp:ScriptManager id="ScriptManager1" runat="server">
</asp:ScriptManager>
```

En la vista de disseny, el control es representa com un quadre gris. En sol·licitar la pàgina, el servidor processa el control i afegeix al document un conjunt d'enllaços cap a les biblioteques ASP.NET AJAX JavaScript.

Codi JavaScript

El codi JavaScript descarregat conté centenars de línies s'emmagatzema en la memòria cau del navegador perquè pugui ser utilitzat per múltiples pàgines del web. Si el navegador ho permet, es descarregaran amb compressió.

El resultat de tot això enviat al navegador del client serà semblant al codi que es mostra a continuació:

```
<script
    src="/WebSite2/WebResource.axd?d=ncfbOam8pYbSd23lXTki6g2&
    t=634046861371712000" type="text/javascript">
</script>
```

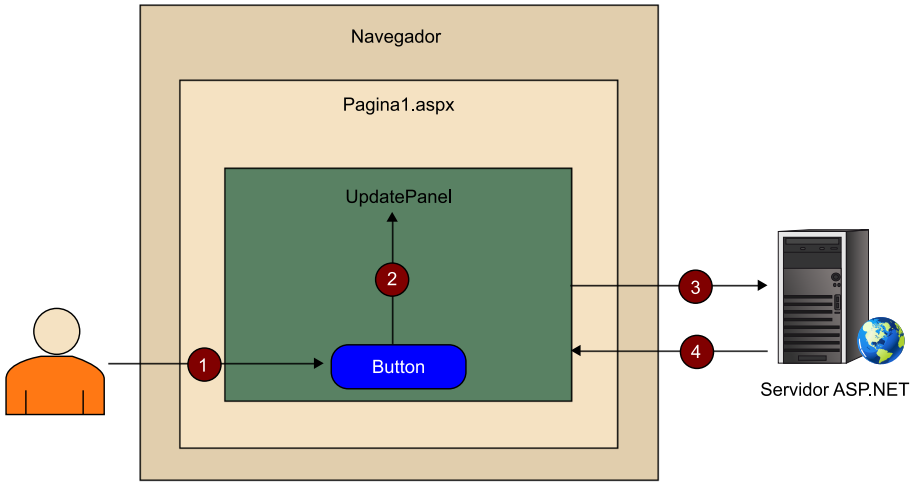
El bloc referencia codi JavaScript situat en un arxiu extern en un URL amb una codificació estranya. En realitat, fa referència a recursos JavaScript incrustats dins d'assembladors de l'entorn de treball.

- **UpdatePanel**. El control UpdatePanel és un contenidor que permet establir parts d'una pàgina que poden ser actualitzades de manera asíncrona. Aquesta acció es coneix com una *actualització parcial*. La figura 8 mostra el funcionament bàsic del control UpdatePanel. UpdatePanel actua de contenidor i disposa en el seu interior un conjunt de controls. Si un d'ells genera un esdeveniment, UpdatePanel l'intercepta i fa una petició asíncrona en lloc d'una petició normal. El servidor executarà la pàgina i tornarà al client el contingut nou del control UpdatePanel.

Exemple d'ús

En l'exemple anterior s'ha utilitzat el control UpdatePanel per a actualitzar una part de la pàgina.

Figura 8. Actualització parcial de pàgina amb el control UpdatePanel



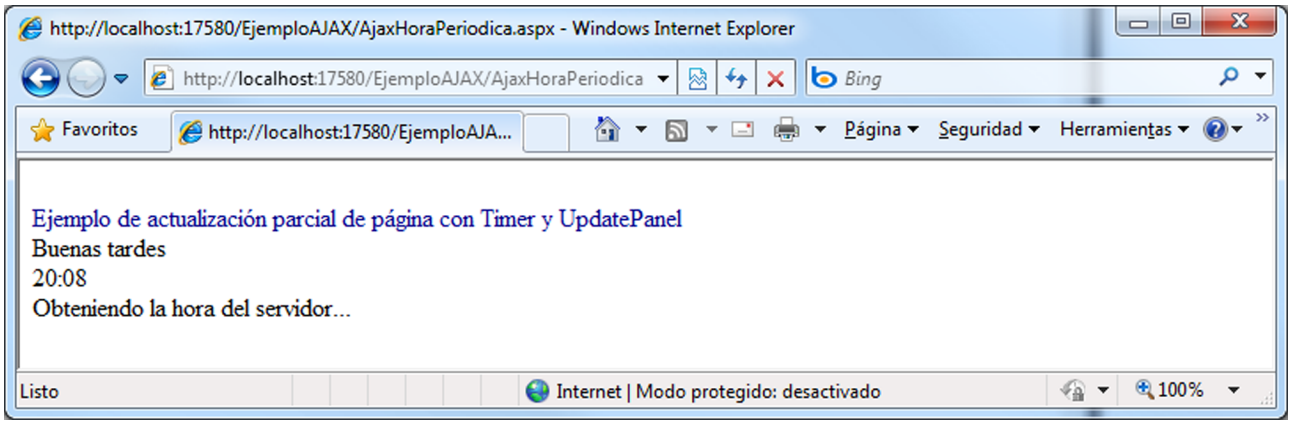
- **UpdateProgress**. Aquest control permet mostrar el progrés d'actualització del contingut d'un o més controls UpdatePanel. Representa un element `div` que es mostra si s'està fent una recàrrega parcial en un control UpdatePanel que tingui associat.

- **Timer**. Fa peticions de recàrrega cada un cert interval de temps indicat en la seva propietat *Interval*. Si s'inclou en un formulari, s'actualitzarà tota la pàgina. Si en canvi, s'inclou dins d'un UpdatePanel, provocarà que aquest s'actualitzi amb cada petició.

Exemple de controls AJAX en ASP.NET

L'exemple següent incorpora cadascun dels controls ASP.NET AJAX presentats en aquest punt: ScriptManager, UpdatePanel, UpdateProgress i Timer. Es tracta d'una pàgina que mostra l'hora i un text segons sigui matí, tard, nit o matinada. L'hora i el text s'obtenen del servidor de manera asíncrona i periòdica mitjançant un control Timer que fa peticions cada cinc segons.

Figura 9. Resultat de l'exemple



Per començar, cal crear un lloc web nou i afegir-hi una pàgina ASP.NET nova amb el nom `AjaxHoraPeriodica.aspx`. Caldrà obrir la vista de disseny i seguir els passos següents:

1) Seleccionar la divisió del formulari i arrossegar dins un control ScriptManager per a habilitar l'ús d'AJAX en la pàgina.

2) Situar el cursor després del control ScriptManager i prémer Enter.

3) Afegir un Label i establir en la propietat *Text* el text "Ejemplo de actualización parcial de página con Timer y UpdatePanel". En la propietat *ForeColor* establir "DarkBlue".

4) Situar el cursor després del control Label afegit en el pas anterior i prémer Intro. A continuació, afegir-hi un control UpdatePanel.

5) Situar el cursor dins del control UpdatePanel fent-hi clic. Els controls que s'afegeixen a continuació s'hauran de situar dins de l'àrea UpdatePanel.

6) Afegir-hi un control Label i establir `lblSaludo` en la propietat ID.

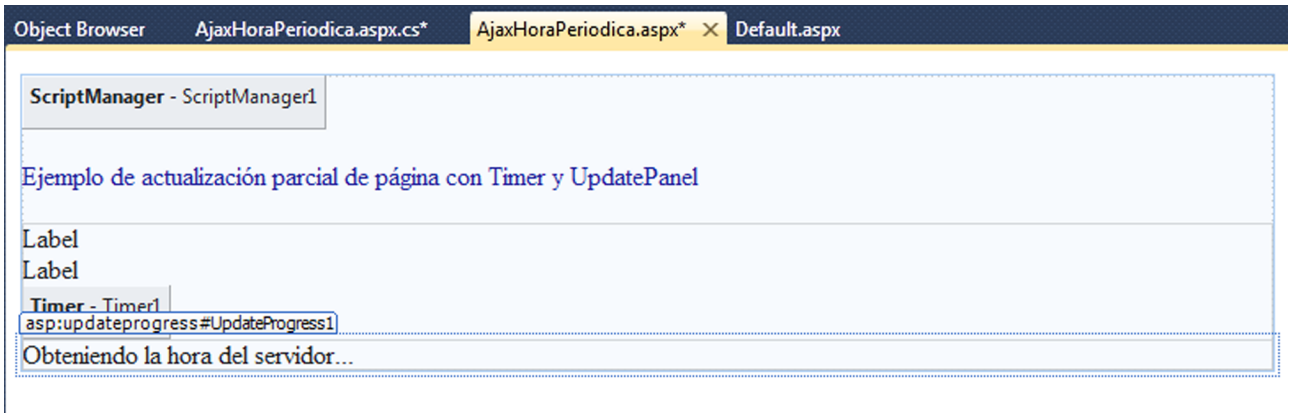
7) Establir el cursor després del Label afegit en el pas anterior i prémer Enter per a afegir-hi un salt de línia. Arrossegar un altre control Label amb l'ID `lblHora`.

8) Arrossegar un control Timer, que es troba situat en la categoria AJAX Extensions. Seleccionar el control per a establir la propietat Interval a 5.000 mil·lisegons.

9) Arrossegar un control UpdateProgress. Situar el cursor dins de la seva àrea i escriure el text "Obteniendo la hora del servidor...".

Una vegada fets aquests passos, la vista de disseny de la pàgina ha de tenir un aspecte semblant al de la figura 10:

Figura 10. Vista de disseny de la pàgina AjaxHoraPeriodica.aspx



Dins del fitxer `AjaxHoraPeriodica.aspx.cs`, cal definir el mètode `GetSaludo()` i afegir el codi dins de `Page_Load` com es mostra a continuació:

```
public partial class AjaxHoraPeriodica : System.Web.UI.Page
{
    private string GetSaludo()
    {
        int hora = DateTime.Now.Hour;
        string saludo = string.Empty;
        if (hora > 20 || hora < 6)
        {
            saludo = "Buenas noches";
        }
        else if (hora >= 6 && hora <= 12)
        {
            saludo = "Buenos días";
        }
        else
        {
            saludo = "Buenas tardes";
        }
        return saludo;
    }

    protected void Page_Load(object sender, EventArgs e)
    {
        lblSaludo.Text = GetSaludo();
        lblHora.Text = DateTime.Now.ToShortTimeString();
        //Per a la instrucció següent s'ha de declarar: using
        System.Threading;
        Thread.Sleep(3000);
    }
}
```

La instrucció `Thread.Sleep()` requereix referenciar la longitud del nom (*namespace*) `System.Threading`. Per a això, cal afegir, abans de la declaració de la classe `AjaxHoraPeriodica`, la línia següent:

```
using System.Threading;
```

En processar la pàgina, el servidor genera el codi JavaScript relacionat amb el control Timer, que fa peticions cada cinc segons. (Una freqüència d'actualització parcial de 5 segons és molt alta i pot perjudicar el rendiment del servidor, per la qual cosa no es recomana.)

Cada petició del Timer provoca la recàrrega parcial de la pàgina, i no, de la pàgina sencera. Això és perquè el control Timer es troba dins d'UpdatePanel. En cada petició, s'executa el mètode `Page_Load` que fa el següent:

1) S'estableix en el Label `lblSaludo` el text que s'ha de mostrar mitjançant el mètode `GetSaludo()`. `GetSaludo()` torna un text o un altre segons quina hora sigui.

2) En el Label `lblHora` s'estableix l'hora. El mètode `ToShortTimeString()` dóna el text de l'hora en format reduït, constituït per hores i minuts.

3) L'última instrucció fa que el procés es bloquegi durant tres segons. Això permetrà veure durant més temps el missatge "Obteniendo la hora del servidor..." establert en l'interior del contenidor UpdateProgress.

El control UpdateProgress permet mostrar-ne el contingut fins que es rep una resposta a partir de la petició asíncrona feta pel control Timer.

Figura 13. Resultat de l'exemple