|  |
| --- |
| **Programación .NET (Intermedio)**  **Desarrollo de Aplicaciones Web con Visual Studio 2008** |
| **Laboratorios** |
|  |
|  |
|  |
|  |
|  |
|  |

**Índice**

[**Módulo I: Introducción a Aplicaciones Web**](#h.gjdgxs)

[Ejercicio 1](#h.30j0zll)

[Ejercicio 2](#h.1fob9te)

[**Módulo II: Aplicaciones Web**](#h.3znysh7)

[Ejercicio 1](#h.2et92p0)

[**Módulo III: Validación**](#h.tyjcwt)

[Ejercicio 1](#h.3dy6vkm)

[**Módulo IV: Seguimiento en ASP.NET**](#h.1t3h5sf)

[Ejercicio 1](#h.4d34og8)

[**Módulo V: Controles de Usuario**](#h.2s8eyo1)

[Ejercicio 1](#h.17dp8vu)

[Ejercicio 2](#h.3rdcrjn)

[**Módulo VI: Manejo de Estado**](#h.26in1rg)

[Ejercicio 1](#h.lnxbz9)

[Ejercicio 2](#h.35nkun2)

[Ejercicio 3](#h.1ksv4uv)

[**Módulo VII: Acceso a Datos**](#h.44sinio)

[Ejercicio 1](#h.2jxsxqh)

[Ejercicio 2](#h.z337ya)

[Ejercicio 3](#h.3j2qqm3)

[Ejercicio 4](#h.1y810tw)

[Ejercicio 5](#h.4i7ojhp)

[Ejercicio 6](#h.2xcytpi)

[**Módulo VIII: Datos en Formato XML**](#h.1ci93xb)

[Ejercicio 1](#h.3whwml4)

[Ejercicio 2](#h.2bn6wsx)

[**Módulo IX: AJAX**](#h.qsh70q)

[Ejercicio 1](#h.3as4poj)

[Ejercicio 2](#h.1pxezwc)

[**Módulo X: Servicios Web XML**](#h.49x2ik5)

[Ejercicio 1](#h.2p2csry)

[Ejercicio 2](#h.147n2zr)

[**Módulo XI: Configurar, Optimizar e Implementación de Aplicaciones ASP.NET**](#h.3o7alnk)

[Ejercicio 1](#h.23ckvvd)

[Ejercicio 2](#h.ihv636)

[Ejercicio 3](#h.32hioqz)

[Ejercicio 4](#h.1hmsyys)

[**Módulo XII: Seguridad**](#h.41mghml)

[Ejercicio 1](#h.2grqrue)

# Módulo I: Introducción a Aplicaciones Web

## Ejercicio 1

En este ejercicio se trabajará con un proyecto Web, incluyendo controles de servidor.

**Pasos:**

1. Abrir Visual Studio 2008.
2. En el menú **File**, hacer click en **New**, y luego en **Project**.
3. Aparecerá el cuadro de diálogo **New Project**.
4. Para elegir el tipo de proyecto, expandir **Visual C#** y hacer click en **Web**, seleccionar la plantilla **ASP.NET Web Application**.
5. Especificar un nombre y una ubicación para el proyecto y presionar **OK**. Visual Studio creará el proyecto.
6. Sobre el nombre del proyecto presione botón derecho – Propiedades. Verá que puede trabajar con propiedades de proyecto desde esta ventana
7. Sobre la página Default.aspx creada automáticamente, diseñe el siguiente formulario:

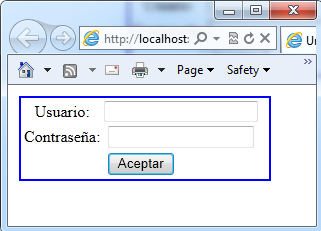


Tabla Html: Puede crearse desde el Menu: *Layout – Insert Table*

Rows: 1

Columns: 1

Celda: Width: 100% (utilice style)

Height: 400px

Text-align: Center

Vertical-align: Middle

Tabla Html (Recuadro datos. Colocar dentro de la tabla anterior). Puede crearse desde el Menu: *Layout – Insert Table*

Rows: 3

Columns: 2

Width: 200px

En la solapa *Borders:*

Size: 2

Color = Blue

Usuario (colocar en las 2 celdas superiores de esta última tabla):

Label: Name: lblUsuario

Text: Usuario:

TextBox: Name: txtUsuario

Text: “”

MaxLenght: 10

Contraseña (colocar en las 2 celdas del medio de esta última tabla):

Label: Name: lblContraseña

Text: Contraseña:

TextBox: Name: txtContraseña

Text: “”

MaxLenght: 10

TextMode: Password

Aceptar (colocar en las 2 celdas inferiores de esta última tabla):

Celda: ColSpan=2

Align: center

Button: Name: btnAceptar

Text: “Aceptar”

1. Ejecute e ingrese información sobre los textbox. Presione el botón Aceptar. ¿Qué pasó con los valores ingresados en los textboxs? ¿Por qué?
2. Sobre el Internet Explorer, presione botón derecho y seleccione la opción **View Source** (ver Código Fuente). Verifique la existencia del **ViewState**. Verifique también el código de identificación de los controles

**Solución**

**7)**

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Untitled Page</title>

<style type="text/css">

.style1

{

width: 100%;

height: 100%;

text-align: center;

vertical-align: middle;

}

.style2

{

width: 200px;

border: 2px solid #0000FF;

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td>

<table class="style2">

<tr>

<td>

<asp:Label ID="lblUsuario" runat="server" Text="Usuario:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtUsuario" runat="server" MaxLength="10"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="lblContraseña" runat="server" Text="Contraseña:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtContraseña" runat="server" MaxLength="10" TextMode="Password"></asp:TextBox>

</td>

</tr>

<tr>

<td align="center" colspan="2">

<asp:Button ID="btnAceptar" runat="server" Text="Aceptar" />

</td>

<td>

&nbsp;

</td>

</tr>

</table>

</td>

</tr>

</table>

</div>

</form>

</body>

</html>

**8)** Los textboxs no pierden los valores ingresados (si cuando esta marcado de tipo Password), por que el estado de los controles se mantiene gracias al uso de ViewState.

**9)** Los controles de tipo <asp:NombreControl se convirtieron en <input type…> y existe un <input type=”hidden” llamado \_\_ViewState que es el control que mantiene el estado.

## Ejercicio 2

En este ejercicio trabajará con Sitios web, controles y master pages.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#**. Presionar Ok.
4. Creando el sitio de esta manera no es necesaria la instalación del IIS. No genera un proyecto con lo cual no tiene acceso a las propiedades del proyecto, deberá configurarlas usando el web.config.
5. Crear una master page: Seleccionar **Add New Item** con botón derecho sobre el nombre del proyecto en la Solution Explorer. Seleccionar **Master Page**, en el nombre ingresar CursoBecas.Master, seleccionar el lenguaje C# y dejar seleccionado “Place code in Separate File”.
6. Elegir la solapa Diseño. Aparece un **ContentPlaceHolder**.
7. Diseñar la master page de manera que presente el siguiente formato (trabajar por fuera del ContentPlaceHolder):

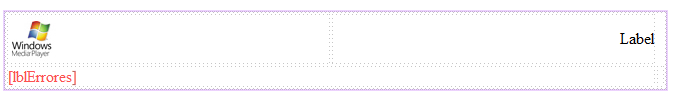


Tabla Html (Superior). Puede crearse desde el Menu: *Layout – Insert Table*

Rows: 2

Columns: 2

Width: 100%

Sin borde

Cada celda: Width: 50%

Celda Derecha, justificar a derecha

Imagen (colocar en la primera celda a la izquierda de la tabla creada en el punto

anterior):

Image: ImageURL: Alguna imagen

(Buscar alguna imagen, incorporarla al proyecto, preferentemente en una carpeta llamada Imágenes – Crear esta carpeta).

Fecha (colocar en la primera celda de la derecha de la tabla creada en el punto anterior):

Label: Name: lblFecha

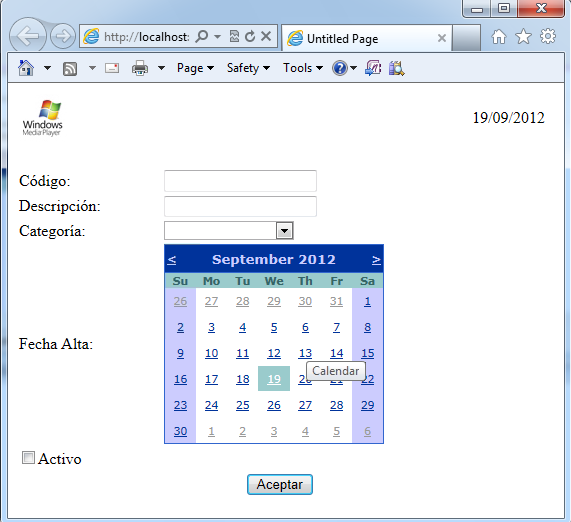
Errores (colocar en la segunda celda, colspan=2)

Label: Name: lblErrores

ForeColor: red

Text=””

1. En el evento **Load** de la master page escribir un código para mostrar la fecha en formato dia / mes / año en el Label llamada lblFecha
2. Crear una nueva página que utilice la master page creada en el punto anterior.
3. Diseñarla de manera que quede similar a:



1. En el evento **Clic** del Botón Aceptar escribir un código que permita poner un mensaje de error en el **Label** lblErrores que se encuentra en la master page cuando el textbox del código está vacío.

**Solución**

**Master Page**

**7)**

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Pruebas Sitio Web</title>

<asp:ContentPlaceHolder id="head" runat="server">

</asp:ContentPlaceHolder>

<style type="text/css">

.style1

{

width: 100%;

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td align="left" width="50%">

<asp:Image ID="Image1" runat="server"

ImageUrl="~/Imagenes/wmpnss\_color48.jpg" />

</td>

<td align="right" width="50%">

<asp:Label ID="lblFecha" runat="server" Text="Label"></asp:Label>

</td>

</tr>

<tr>

<td colspan="2">

<asp:Label ID="lblErrores" runat="server" ForeColor="#FF5050"></asp:Label>

</td>

<td>

&nbsp;</td>

</tr>

</table>

<asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">

</asp:ContentPlaceHolder>

</div>

</form>

</body>

</html>

**8)**

protected void Page\_Load(object sender, EventArgs e)

{

lblFecha.Text=DateTime.Today.ToString("dd/MM/yyyy");

}

**Página**

**10)**

<%@ Page Language="C#" MasterPageFile="~/CursoBecas.master" AutoEventWireup="true" CodeFile="Default2.aspx.cs" Inherits="Default2" Title="Untitled Page" %>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

<table class="style1">

<tr>

<td>

<asp:Label ID="Label1" runat="server" Text="Código:"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label2" runat="server" Text="Descripción:"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label3" runat="server" Text="Categoría:"></asp:Label>

</td>

<td>

<asp:DropDownList ID="DropDownList1" runat="server" Width="130px">

</asp:DropDownList>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label4" runat="server" Text="Fecha Alta: "></asp:Label>

</td>

<td>

<asp:Calendar ID="Calendar1" runat="server"

BackColor="White" BorderColor="#3366CC"

BorderWidth="1px" CellPadding="1"

DayNameFormat="Shortest" Font-Names="Verdana"

Font-Size="8pt"

ForeColor="#003399" Height="200px" Width="220px">

<SelectedDayStyle BackColor="#009999"

Font-Bold="True" ForeColor="#CCFF99" />

<SelectorStyle BackColor="#99CCCC"

ForeColor="#336666" />

<WeekendDayStyle BackColor="#CCCCFF" />

<TodayDayStyle BackColor="#99CCCC"

ForeColor="White" />

<OtherMonthDayStyle ForeColor="#999999" />

<NextPrevStyle Font-Size="8pt"

ForeColor="#CCCCFF" />

<DayHeaderStyle BackColor="#99CCCC"

ForeColor="#336666" Height="1px" />

<TitleStyle BackColor="#003399"

BorderColor="#3366CC" BorderWidth="1px"

Font-Bold="True" Font-Size="10pt"

ForeColor="#CCCCFF" Height="25px" />

</asp:Calendar>

</td>

</tr>

<tr>

<td>

<asp:CheckBox ID="CheckBox1" runat="server"

Text="Activo" />

</td>

<td>&nbsp;</td>

</tr>

<tr>

<td align="center" colspan="2">

<asp:Button ID="Button1" runat="server" Text="Aceptar" />

</td>

<td>&nbsp;</td>

</tr>

</table>

</asp:Content>

**11)**

protected void Button1\_Click(object sender, EventArgs e)

{

if (txtCodigo.Text == "")

{

Label lblMensajes = (Label)Page.Master.FindControl("lblErrores");

lblMensajes.Text = "Debe Ingresar el Código";

}

}

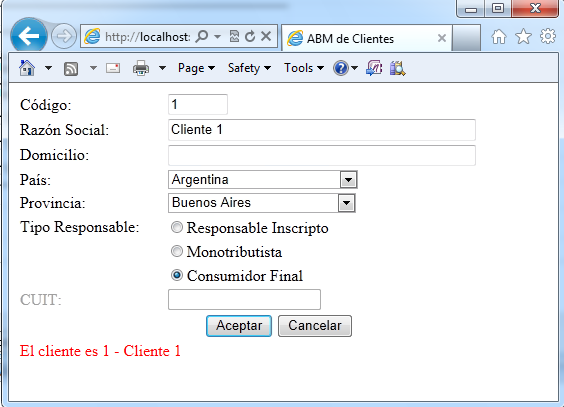
# Módulo II: Aplicaciones Web

## Ejercicio 1

En este ejercicio se trabajará con un sitio Web, incluyendo controles de servidor y código.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#**. Presionar Ok.
4. Diseñe el siguiente formulario para armar un ABM de Clientes (tenga en cuenta de completar las propiedades Id, Text, MaxLength, y cualquier otra que corresponda a cada control)



1. En el evento **Load** del formulario cargue algunos países en la lista desplegable de Países.
2. Escriba código para poder cargar la lista de provincias correspondientes a cada país según que país se haya seleccionado, al momento de cambiar la selección. Arme las provincias preguntando por el país y cargando las provincias manualmente (como todavía no se vio acceso a datos escriba el código cargando todo manualmente).
3. Las opciones para los tipos de responsables deben formar un solo grupo. Escriba un evento (solo uno) de manera que si se seleccionan las opciones Responsable Inscripto o Monotributista se habiliten las opciones de CUIT, y si se selecciona Consumidor final se deshabiliten estas opciones.
4. Escriba el evento Clic del botón Aceptar de manera que llene la leyenda que se ve en rojo, mostrando el código y la razón social ingresados

**Solución**

**4)**

<head runat="server">

<title>ABM de Clientes</title>

<style type="text/css">

.style1

{

width: 100%;

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td>

<asp:Label ID="Label1" runat="server"

Text="Código:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtCodigo" runat="server"

MaxLength="5" Width="54px"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label2" runat="server" Text="Razón

Social:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtRazon" runat="server"

MaxLength="50" Width="302px"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label3" runat="server"

Text="Domicilio:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtDomicilio" runat="server"

MaxLength="50" Width="302px"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label4" runat="server"

Text="País:"></asp:Label>

</td>

<td>

<asp:DropDownList ID="ddlPaises" runat="server"

Width="190px" AutoPostBack="True"

onselectedindexchanged="ddlPaises\_SelectedIndexChanged">

</asp:DropDownList>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label5" runat="server"

Text="Provincia:"></asp:Label>

</td>

<td>

<asp:DropDownList ID="ddlProvincias"

runat="server" Width="188px">

</asp:DropDownList>

</td>

</tr>

<tr>

<td>Tipo Responsable:</td>

<td>

<asp:RadioButton ID="rdbTipoResp" runat="server"

AutoPostBack="True"

Text="Responsable Inscripto"

oncheckedchanged="rdbTipoResp\_CheckedChanged"

GroupName="TipoResp" />

</td>

</tr>

<tr>

<td>

&nbsp;</td>

<td>

<asp:RadioButton ID="rdbMontotrib" runat="server"

AutoPostBack="True"

oncheckedchanged="rdbTipoResp\_CheckedChanged"

Text="Monotributista" GroupName="TipoResp" />

</td>

</tr>

<tr>

<td>

&nbsp;</td>

<td>

<asp:RadioButton ID="rdbConsFinal" runat="server"

AutoPostBack="True"

oncheckedchanged="rdbTipoResp\_CheckedChanged"

Text="Consumidor Final" GroupName="TipoResp" />

</td>

</tr>

<tr>

<td>

<asp:Label ID="lblCUIT" runat="server"

Text="CUIT:"></asp:Label></td>

<td>

<asp:TextBox ID="txtCUIT" runat="server"

MaxLength="13"></asp:TextBox>

</td>

</tr>

<tr>

<td style="text-align:center" colspan="2">

<asp:Button ID="btnAceptar" runat="server"

Text="Aceptar" onclick="btnAceptar\_Click" />

<asp:Button ID="btnCancelar" runat="server"

Text="Cancelar" />

</td>

<td>&nbsp;</td>

</tr>

<tr>

<td style="text-align:left" colspan="2">

<asp:Label ID="lblMensajes" runat="server"

ForeColor="Red"></asp:Label>

</td>

<td>&nbsp;</td>

</tr>

</table>

</div>

</form>

</body>

</html>

**5)**

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

ddlPaises.Items.Add("Argentina");

ddlPaises.Items.Add("Brasil");

ddlPaises.Items.Add("Uruguay");

}

}

**6)**

protected void ddlPaises\_SelectedIndexChanged(object sender, EventArgs e)

{

ddlProvincias.Items.Clear();

if (ddlPaises.SelectedItem.Text == "Argentina")

{

ddlProvincias.Items.Add("Buenos Aires");

ddlProvincias.Items.Add("Córdoba");

ddlProvincias.Items.Add("Mendoza");

}

else if (ddlPaises.SelectedItem.Text == "Brasil")

{

ddlProvincias.Items.Add("San Pablo");

ddlProvincias.Items.Add("Rio Grande");

ddlProvincias.Items.Add("Bahia");

}

else if (ddlPaises.SelectedItem.Text == "Uruguay")

{

ddlProvincias.Items.Add("Salto");

ddlProvincias.Items.Add("Maldonado");

ddlProvincias.Items.Add("Colonia");

}

}

**7)**

protected void rdbTipoResp\_CheckedChanged(object sender, EventArgs e)

{

if (rdbConsFinal.Checked)

{

lblCUIT.Enabled = false;

txtCUIT.Enabled = false;

}

Else

{

lblCUIT.Enabled = true;

txtCUIT.Enabled = true;

}

}

**8)**

protected void btnAceptar\_Click(object sender, EventArgs e)

{

lblMensajes.Text = "El cliente es " + txtCodigo.Text + " - " + txtRazon.Text;

}

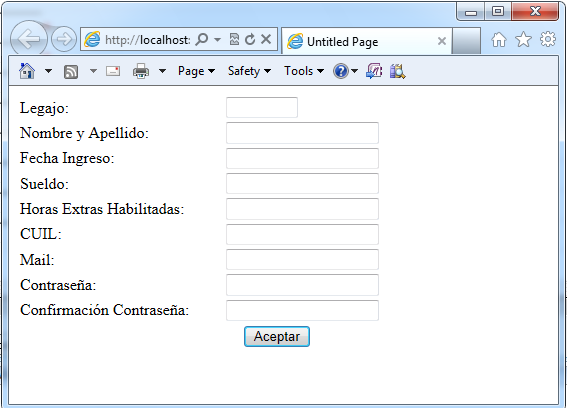
# Módulo III: Validación

## Ejercicio 1

En este ejercicio se trabajará con un sitio Web, incluyendo controles de servidor y validando el ingreso de los datos.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. Diseñe el siguiente formulario para armar un ABM de Empleados (tenga en cuenta de completar las propiedades Id, Text, MaxLength, y cualquier otra que corresponda a cada control)



1. Legajo: Validar que sea obligatorio y numérico
2. Nombre y Apellido: Validar que sea obligatorio
3. Fecha de Ingreso: Validar que sea una fecha válida y anterior a hoy u hoy.
4. Sueldo: Validar que sea numérico mayor a cero y obligatorio
5. Horas Extras: Validar que sea numérico y los valores a ingresar estén entre 0 y 50
6. CUIL: Validar que tenga formato CUIL
7. Mail: Validar que tenga formato Mail
8. Contraseña: Es obligatoria, con una cantidad mínima de 5 caracteres y debe ser igual a la confirmación
9. Todas las validaciones deben mostrar un asterisco cerca del control que validan y el mensaje en un MessageBox
10. El botón aceptar debe mostrar un texto que contenga el legajo y nombre y apellido ingresados. Repetir la validación del lado del servidor.

**Solución**

**4 – 12)**

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Untitled Page</title>

<style type="text/css">

.style1

{

width: 100%;

}

.style2

{

width: 202px;

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td class="style2">

<asp:Label ID="Label1" runat="server"

Text="Legajo:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtCodigo" runat="server"

MaxLength="5" Width="66px">

</asp:TextBox>

<asp:RequiredFieldValidator

ID="RequiredFieldValidator1" runat="server"

ErrorMessage="Debe ingresar el legajo"

ControlToValidate="txtCodigo">\*

</asp:RequiredFieldValidator>

<asp:CompareValidator ID="CompareValidator4"

runat="server"

ErrorMessage="El legajo debe ser numérico"

ControlToValidate="txtCodigo"

Display="Dynamic" Operator="DataTypeCheck"

Type="Integer">\*

</asp:CompareValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label2" runat="server"

Text="Nombre y Apellido:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtNombre" runat="server"

MaxLength="50"></asp:TextBox>

<asp:RequiredFieldValidator

ID="RequiredFieldValidator2" runat="server"

ErrorMessage="El nombre/apellido es obligatorio"

ControlToValidate="txtNombre">\*

</asp:RequiredFieldValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label3" runat="server"

Text="Fecha Ingreso:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtFhIng" runat="server"

MaxLength="10"></asp:TextBox>

<asp:CompareValidator ID="CompareValidator1"

runat="server" ErrorMessage="La fecha de ingreso no es una fecha válida o debe ser anterior a hoy"

ControlToValidate="txtFhIng"

Operator="LessThanEqual" Type="Date">\*

</asp:CompareValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label4" runat="server"

Text="Sueldo:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtSueldo"

runat="server"></asp:TextBox>

<asp:RequiredFieldValidator

ID="RequiredFieldValidator4" runat="server"

ControlToValidate="txtSueldo"

ErrorMessage="El sueldo es obligatorio">\*

</asp:RequiredFieldValidator>

<asp:CompareValidator ID="CompareValidator2"

runat="server"

ErrorMessage="El sueldo debe ser mayor a cero"

ControlToValidate="txtSueldo" Display="Dynamic"

Operator="GreaterThan" Type="Integer"

ValueToCompare="0">\*

</asp:CompareValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label5" runat="server"

Text="Horas Extras Habilitadas:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtHsExtras"

runat="server"></asp:TextBox>

<asp:RangeValidator ID="RangeValidator1"

runat="server"

ErrorMessage="La horas extras deben estar entre 0 y 50" ControlToValidate="txtHsExtras"

MaximumValue="50" MinimumValue="0"

Type="Integer">\*</asp:RangeValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label6" runat="server"

Text="CUIL:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtCUIL" runat="server"

MaxLength="13"></asp:TextBox>

<asp:RegularExpressionValidator

ID="RegularExpressionValidator1" runat="server"

ErrorMessage="El formato del CUIL es incorrecto" ControlToValidate="txtCUIL"

ValidationExpression="\d{2}-\d{8}-\d{1}">\*

</asp:RegularExpressionValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label7" runat="server"

Text="Mail:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtMail"

runat="server"></asp:TextBox>

<asp:RegularExpressionValidator

ID="RegularExpressionValidator2" runat="server"

ErrorMessage="El formato del mail es incorrecto" ControlToValidate="txtMail" ValidationExpression="\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*">

\*</asp:RegularExpressionValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label9" runat="server"

Text="Contraseña:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtContraseña" runat="server"

MaxLength="10" TextMode="Password">

</asp:TextBox>

<asp:RequiredFieldValidator

ID="RequiredFieldValidator3" runat="server"

ErrorMessage="Debe ingresar la contraseña"

ControlToValidate="txtContraseña">\*

</asp:RequiredFieldValidator>

<asp:CustomValidator ID="CustomValidator1"

runat="server"

ClientValidationFunction="CantCaracteres"

ControlToValidate="txtContraseña"

Display="Dynamic"

ErrorMessage="La contraseña tiene que tener al menos 5 caracteres"

onservervalidate="CustomValidator1\_ServerValidate">\*

</asp:CustomValidator>

</td>

</tr>

<tr>

<td class="style2">

<asp:Label ID="Label10" runat="server"

Text="Confirmación Contraseña:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtConfirmacion" runat="server"

TextMode="Password"></asp:TextBox>

<asp:CompareValidator ID="CompareValidator3"

runat="server"

ErrorMessage="La contraseña y su confirmación no son iguales" ControlToCompare="txtContraseña"

ControlToValidate="txtConfirmacion">\*

</asp:CompareValidator>

</td>

</tr>

<tr>

<td style="text-align:center" colspan="2">

<asp:Button ID="Button1"

runat="server" Text="Aceptar"

onclick="Button1\_Click" />

</td>

<td>&nbsp;</td>

</tr>

<tr>

<td colspan="2">

<asp:Label ID="lblMensaje" runat="server"

ForeColor="Red"></asp:Label>

</td>

<td>&nbsp;</td>

</tr>

</table>

</div>

<asp:ValidationSummary ID="ValidationSummary1" runat="server"

ShowMessageBox="True" ShowSummary="False" />

</form>

<script type="text/javascript" >

function CantCaracteres(source, args) {

if (args.Value.length <= 5)

args.IsValid = false;

else

args.IsValid = true;

}

</script>

</body>

</html>

**14)**

protected void Page\_Load(object sender, EventArgs e)

{

CompareValidator1.ValueToCompare = DateTime.Today.ToShortDateString();

}

protected void CustomValidator1\_ServerValidate(object source, ServerValidateEventArgs args)

{

if (args.Value.Length <= 5)

args.IsValid = false;

else

args.IsValid = true;

}

protected void Button1\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

lblMensaje.Text = "Legajo: " + txtCodigo.Text + " - " + txtNombre.Text;

}

# Módulo IV: Seguimiento en ASP.NET

## Ejercicio 1

En este ejercicio se trabajará con el sitio Web creado en el módulo 2 para probar las técnicas de seguimiento.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008.
2. Abrir la aplicación desarrollada en el módulo II
3. En la página Default.aspx agregue el atributo **Trace** en el **@ Page**
4. Ejecute y verifique la información aportada por Trace
5. Incorpore un método **Write**, cuya categoría sea Load y la leyenda diga que pasa por ahí solo la primera vez. Escribirlo dentro del evento **Load** del formulario, dentro del If (!IsPostBack)
6. Incorpore un método **Warn** sin categoría y que la leyenda diga que por ahí se pasa siempre. Escribirlo al final del evento **Load** del formulario, fuera del if mencionado en el punto anterior.
7. Probar y verificar cuando aparecen las leyendas
8. Borre el atributo Trace del @ Page de la página e incorpore el Trace en el Web.Config. Las opciones deberán ser que solo se vea localmente y que el resultado no aparezca en la página
9. Ejecute y verifique la información de Trace utilizando la página **Trace.axd**

**Solución**

**5 y 6)**

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

Trace.Write("Load", "Solo lo muestra la primera vez");

ddlPaises.Items.Add("Argentina");

ddlPaises.Items.Add("Brasil");

ddlPaises.Items.Add("Uruguay");

}

Trace.Warn("", "Lo muestra siempre");

}

**8)**

<trace enabled="true" localOnly="true" pageOutput="false" />

# Módulo V: Controles de Usuario

## Ejercicio 1

En este ejercicio se trabajará con el sitio Web creado un control de usuario.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. Agregue a la aplicación un control de usuario, presionando botón derecho sobre el nombre del proyecto en el Explorador de Soluciones, seleccionar **Add Item** – **Web User Control**. Llame al control **NumBox.ascx**.
5. Para armar el control agregar un **textbox** (llamarlo txtNumero), un **RequiredFieldValidator** (que valide que el textbox sea obligatorio) y un **CompareValidator** (que valide que el dato ingresado en el textbox sea numérico)
6. Cree una propiedad llamada Text que exponga la propiedad Text del textbox
7. Cree una propiedad llamada MensajeErrorObligatorio que exponga la propiedad ErrorMessage del RequiredFieldValidator
8. Cree una propiedad llamada MensajeErrorNumerico que exponga la propiedad ErrorMessage del CompareValidator
9. Compilar

**Solución**

**5)**

<%@ Control Language="C#" AutoEventWireup="true" CodeFile="NumBox.ascx.cs" Inherits="NumBox" %>

<asp:TextBox ID="txtNumero" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator1"

runat="server" ControlToValidate="txtNumero"

ErrorMessage="El dato es obligatorio">\*

</asp:RequiredFieldValidator>

<asp:CompareValidator ID="CompareValidator1" runat="server"

ControlToValidate="txtNumero"

ErrorMessage="El dato debe ser numérico"

Operator="DataTypeCheck" Type="Integer">\*

</asp:CompareValidator>

**6-8)**

public partial class NumBox : System.Web.UI.UserControl

{

public string Text

{

get {return txtNumero.Text; }

set { txtNumero.Text = value; }

}

public string MensajeErrorObligatorio

{

get {return RequiredFieldValidator1.ErrorMessage; }

set { RequiredFieldValidator1.ErrorMessage = value; }

}

public string MensajeErrorNumerico

{

get { return CompareValidator1.ErrorMessage; }

set { CompareValidator1.ErrorMessage = value; }

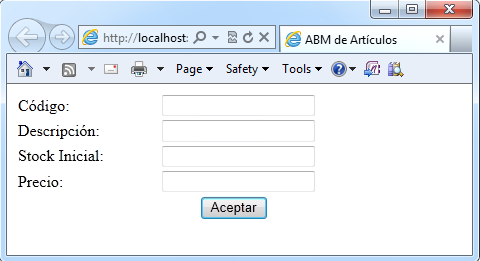
}

}

## Ejercicio 2

En este ejercicio se creará una página web que utilice el control creado en el ejercicio anterior

1. En la misma aplicación que el ejercicio anterior abrimos la página Default.aspx creado automáticamente en el sitio.
2. Diseñe la página similar a la imagen para armar un ABM de Artículos. Utilice el control NumBox para los campos Código, Stock y Precio.



**Solución**

<%@ Register src="NumBox.ascx" tagname="NumBox" tagprefix="uc1" %>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>ABM de Artículos</title>

<style type="text/css">

.style1

{

width: 100%;

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td>

<asp:Label ID="Label1" runat="server"

Text="Código:"></asp:Label>

</td>

<td>

<uc1:NumBox ID="NumBox1" runat="server" />

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label2" runat="server"

Text="Descripción:"></asp:Label>

</td>

<td>

<asp:TextBox ID="TextBox1"

runat="server"></asp:TextBox>

</td>

</tr>

<tr>

<td>Stock Inicial:</td>

<td>

<uc1:NumBox ID="NumBox2" runat="server" />

</td>

</tr>

<tr>

<td>Precio:</td>

<td>

<uc1:NumBox ID="NumBox3" runat="server" />

</td>

</tr>

<tr>

<td style="text-align:center" colspan="2">

<asp:Button ID="Button1" runat="server"

Text="Aceptar" />

</td>

<td>&nbsp;</td>

</tr>

</table>

</div>

</form>

</body>

</html>

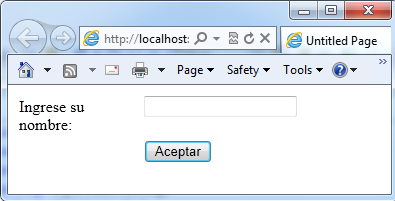
# Módulo VI: Manejo de Estado

## Ejercicio 1

En este ejercicio se trabajará usando cookies.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. Agregar una nueva página llamada Nombre.aspx. Diseñar esta página de manera que quede similar a la imagen.



1. Controle que el nombre a ingresar sea obligatorio usando un control de validación.
2. En el evento clic del botón Aceptar deberá crear una cookie llamada Nombre y agregarle como valor el dato ingresado en el textbox. La validez de la cookie deberá ser de 5 minutos.
3. En la página Default.aspx ingrese un label llamado lblBienvenido. En el evento Load de esta página verifique si la cookie “Nombre” existe. Si no existe direccione la llamada a la página Nombre.aspx de manera de permitir ingresar el dato y crear la cookie. Si existe escriba “Bienvenido/a xxxx” en el label lblBienvenido, siendo xxxx el dato tomado de la cookie.
4. Para probar el funcionamiento verifique que la página Default.aspx sea la página a ejecutar.
5. Una vez probado puede intentar una nueva prueba pasados los 5 minutos.

**Solución**

**4)**

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title>Ingresar Nombre</title>

<style type="text/css">

.style1

{

width: 100%;

}

.style2

{

width: 121px;

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td class="style2">

<asp:Label ID="Label1" runat="server"

Text="Ingrese su nombre:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtNombre"

runat="server"></asp:TextBox>

<asp:RequiredFieldValidator

ID="RequiredFieldValidator1" runat="server"

ControlToValidate="txtNombre"

ErrorMessage="Debe ingresar el nombre">

</asp:RequiredFieldValidator>

</td>

</tr>

<tr>

<td class="style2">

&nbsp;</td>

<td>

<asp:Button ID="btnAceptar" runat="server"

onclick="btnAceptar\_Click" Text="Aceptar" />

</td>

</tr>

</table>

</div>

</form>

</body>

</html>

**5)**

protected void btnAceptar\_Click(object sender, EventArgs e)

{

HttpCookie ck = new HttpCookie("Nombre");

ck.Value = txtNombre.Text;

ck.Expires = DateTime.Now.AddMinutes(5);

Response.Cookies.Add(ck);

Response.Redirect("Default.aspx");

}

**6)**

<body>

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

<div>

<asp:Label ID="lblBienvenido" runat="server"

Text="Bienvenido "></asp:Label>

</div>

</form>

</body>

protected void Page\_Load(object sender, EventArgs e)

{

HttpCookie ck = Request.Cookies["Nombre"];

if (ck == null)

Response.Redirect("Nombre.aspx");

Else

lblBienvenido.Text = "Bienvenido/a " + ck.Value.ToString();

}

## Ejercicio 2

En este ejercicio se creará una página web que utilice manejo de estado del lado del servidor

**Pasos**

1. Sobre el mismo ejemplo del Ejercicio 1, agregar un archivo **global.asax**.
2. En el evento **Application\_Start** de dicho archivo crear una variable de aplicación llamada Visitantes e inicializarla en cero.
3. En la página Default.aspx agregar código para sumar 1 a la variable “Visitantes” cada vez que se ingresa a la página. Mostrar el resultado en un nuevo Label.
4. En la página Nombre.aspx luego de crear la cookie (hecho en el ejercicio anterior), crear una variable de sesión llamada “Hora” y asociarle la fecha y hora actual.
5. Modificar el código de Default.aspx de manera que al mostrar el texto Bienvenido/a xxxx incluya también la fecha y hora del momento en que se registró. Tener en cuenta que las sesiones pueden caducar.

**Solución**

**2)**

void Application\_Start(object sender, EventArgs e)

{

Application["Visitantes"] = 0;

}

**3-5)**

**Nombre.aspx**

protected void btnAceptar\_Click(object sender, EventArgs e)

{

HttpCookie ck = new HttpCookie("Nombre");

ck.Value = txtNombre.Text;

ck.Expires = DateTime.Now.AddMinutes(5);

Response.Cookies.Add(ck);

Session["Hora"] = DateTime.Now;

Response.Redirect("Default.aspx");

}

**Default.aspx**

protected void Page\_Load(object sender, EventArgs e)

{

HttpCookie ck = Request.Cookies["Nombre"];

if (ck == null)

Response.Redirect("Nombre.aspx");

Else

{

if (Session["Hora"] == null)

Session["Hora"] = DateTime.Now;

lblBienvenido.Text = "Bienvenido/a " + ck.Value.ToString()

+ ". Conectado a las " + Session["Hora"].ToString();

}

if (!Page.IsPostBack)

Application["Visitantes"] = (int)Application["Visitantes"] + 1;

lblVisitantes.Text = "Cantidad de Visitantes: " +

Application["Visitantes"].ToString();

}

## Ejercicio 3

En este ejercicio configurará el estado para que sea guardado en una base de datos y luego en un proceso de memoria independiente.

**Pasos**

1. En la línea de comando de .net ejecutar el archivo **aspnet\_regsql.exe** de manera de crear la base de datos **AspState** para guardar el estado. Cree esta base con los permisos del usuario con el que está trabajando y configure la creación de manera que las tablas se creen dentro de esta base de datos.
2. Sobre la misma aplicación creada en el ejercicio 1 configure en el **web.config** para trabajar guardando el estado en base de dato. También configure que el timeout de las sesiones sea de 10 minutos y que el sessionid se guarde en cookie o en la URL dependiendo de las posibilidades del navegador.
3. Ejecutar la aplicación
4. Verificar que en la tabla **ASPStateTempSessions** haya al menos un registro que identifique su sesión actual
5. Modificar ahora el web.config de manera que el estado se guarde en un proceso independiente.
6. Arrancar el servicio necesario para trabajar con procesos de memoria independientes.
7. Probar la aplicación

**Solución**

**1)**

aspnet\_regsql.exe -S NombreServidor -E -ssadd -sstype p

**2)**

<sessionState mode="SQLServer" timeout="10" cookieless="UseDeviceProfile" sqlConnectionString ="Data Source=NombreServidor;Integrated Security=True"/>

**5)**

<sessionState mode="StateServer" stateConnectionString="tcpip=127.0.0.1:42424"/>

**6) ASP.NET State Service**

# Módulo VII: Acceso a Datos

## Ejercicio 1

En este ejercicio se trabajará usando ADO.NET en modo desconectado.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. Cree una entrada en el **web.config** que le permita guardar el string de conexión a la base de datos **AdventureWorks**. Puede utilizar un **SQLDataSource** y después eliminarlo de la página
5. Cree una clase llamada **Class1**. Dentro de ella crear un nuevo método llamado **TraerPaises**, el cual devolverá en un DataSet la tabla completa llamada Person.CountryRegion, tabla de países de la base de datos AdventureWorks.
6. Utilizando la página Default.aspx creada automáticamente, agregar un **ObjectDataSource** y **GridView**.
7. Configurar el **ObjectDataSource** usando el smart tag. Elegir la clase creada en el punto 5 en la primera solapa y luego, en la segunda solapa, el método creado dentro de esa clase. Confirmar.
8. Relacione el GridView con el ObjectDataSource usando el smar tag y configurando la opción “Choose Data Source”
9. Configurar la paginación para que muestre 15 registros por página. Configurar también el ordenado por columna
10. Ejecutar

**Solución**

**4)**

<connectionStrings>

<add name="AdvWorks" connectionString="Data Source=.;Initial Catalog=AdventureWorks;Integrated Security=True"

providerName="System.Data.SqlClient" />

</connectionStrings>

**5)**

using System.Data.SqlClient;

public class Class1

{

public DataTable TraerPaises()

{

using (DataTable dt = new DataTable())

{

using (SqlConnection cn = new SqlConnection(ConfigurationManager.ConnectionStrings["AdvWorks"].ConnectionString))

{

using (SqlDataAdapter da = new SqlDataAdapter("Select \* from Person.CountryRegion", cn))

{

da.Fill(dt);

}

}

return dt;

}

}

}

**6 - 9)**

<body>

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

<div>

<asp:ObjectDataSource ID="ObjectDataSource1" runat="server"

SelectMethod="TraerPaises" TypeName="Class1">

</asp:ObjectDataSource>

</div>

<asp:GridView ID="GridView1" runat="server" CellPadding="4"

DataSourceID="ObjectDataSource1" ForeColor="#333333"

GridLines="None" Width="289px" AllowPaging="True"

AllowSorting="True" PageSize="15">

<RowStyle BackColor="#FFFBD6" ForeColor="#333333" />

<FooterStyle BackColor="#990000" Font-Bold="True"

ForeColor="White" />

<PagerStyle BackColor="#FFCC66" ForeColor="#333333"

HorizontalAlign="Center" />

<SelectedRowStyle BackColor="#FFCC66" Font-Bold="True"

ForeColor="Navy" />

<HeaderStyle BackColor="#990000" Font-Bold="True"

ForeColor="White" />

<AlternatingRowStyle BackColor="White" />

</asp:GridView>

</form>

</body>

## Ejercicio 2

En este ejercicio se trabajará usando ADO.NET en modo desconectado.

**Pasos**

1. Sobre el ejercicio anterior configure el GridView de manera que solo muestre los campos CountryRegionCode y Name.
2. El primer campo debería tener como encabezado la palabra Código, ocupar solo el 20% del tamaño de la grilla y mantener la posibilidad de ordenar por columna
3. El segundo campo debería tener como encabezado Nombre, ocupar el 80% y también tener la posibilidad de ser ordenado por columna

**Solución**

<body>

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

<div>

<asp:ObjectDataSource ID="ObjectDataSource1" runat="server"

SelectMethod="TraerPaises" TypeName="Class1">

</asp:ObjectDataSource>

</div>

<asp:GridView ID="GridView1" runat="server" CellPadding="4"

DataSourceID="ObjectDataSource1" ForeColor="#333333"

GridLines="None" Width="289px" AllowPaging="True"

AllowSorting="True" AutoGenerateColumns="False"

PageSize="15">

<RowStyle BackColor="#FFFBD6" ForeColor="#333333" />

<Columns>

<asp:BoundField DataField="CountryRegionCode"

HeaderText="Código" SortExpression="CountryRegionCode">

<ItemStyle Width="20%" />

</asp:BoundField>

<asp:BoundField DataField="Name" HeaderText="Nombre"

SortExpression="Name">

<ItemStyle Width="80%" />

</asp:BoundField>

</Columns>

<FooterStyle BackColor="#990000" Font-Bold="True"

ForeColor="White" />

<PagerStyle BackColor="#FFCC66" ForeColor="#333333"

HorizontalAlign="Center" />

<SelectedRowStyle BackColor="#FFCC66" Font-Bold="True"

ForeColor="Navy" />

<HeaderStyle BackColor="#990000" Font-Bold="True"

ForeColor="White" />

<AlternatingRowStyle BackColor="White" />

</asp:GridView>

</form>

</body>

## Ejercicio 3

En este ejercicio se trabajará usando ADO.NET en modo conectado.

**Pasos**

1. Sobre el ejercicio anterior agregar una nueva página. Ingresar un control DropDownList
2. Crear una conexión para comunicarse con la base de datos AdventureWorks. Utilice el string de conexión ingresado en el web.config en el ejercicio 1
3. Crear un comando que permita leer de la tabla Production.Product solo los productos cuyo nombre comience con “a”.
4. Cargar el DropDownList con la información levantada de la tabla de productos utilizando un DataReader
5. Ejecutar

**Solución**

using System.Data.SqlClient;

protected void Page\_Load(object sender, EventArgs e)

{

using (SqlConnection cn = new SqlConnection(ConfigurationManager.ConnectionStrings["AdvWorks"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand("Select \* from Production.Product where name like 'a%'",cn))

{

cn.Open();

using (SqlDataReader dr = cmd.ExecuteReader())

{

while (dr.Read())

{

DropDownList1.Items.Add(dr["Name"].ToString());

}

}

}

}

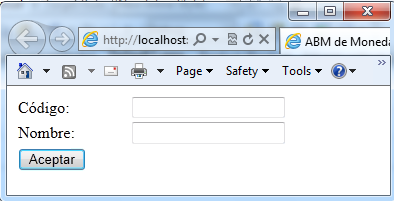
}

## Ejercicio 4

En este ejercicio se trabajará usando ADO.NET para grabar información en una tabla

**Pasos**

1. Sobre el ejercicio anterior agregar una nueva página.
2. Diseñe la página de manera que quede similar a la imagen. Configure al primer textbox para que solo pueda aceptar 3 caracteres, y el segundo solo 50 caracteres.



1. En el evento clic del botón Aceptar escriba código para grabar los datos ingresados en la tabla Sales.Currency de la base de datos AdventureWorks.
2. Configure primero la conexión, luego cree un comando y utilice parámetros para pasar la información a la base de datos.
3. Controlar la posibilidad de errores durante la grabación. En el caso de que se produzca un error mostrar el mensaje en un label
4. Una vez grabado los datos dejar la pantalla preparada para un nuevo ingreso
5. Probar
6. Agregar una nueva página y crear un ejemplo similar usando un procedimiento almacenado.

**Solución**

**2)**

<head runat="server">

<title>ABM de Monedas</title>

<style type="text/css">

.style1

{

width: 100%;

}

</style>

</head>

<body>

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

<div>

<table class="style1">

<tr>

<td>

<asp:Label ID="Label1" runat="server"

Text="Código:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtCodigo" runat="server"

MaxLength="3"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label2" runat="server"

Text="Nombre:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtNombre" runat="server"

MaxLength="50"></asp:TextBox>

</td>

</tr>

<tr>

<td>

<asp:Button ID="Button1" runat="server"

onclick="Button1\_Click" Text="Aceptar" />

</td>

<td>&nbsp;</td>

</tr>

<tr>

<td>

<asp:Label ID="lblErrores" runat="server"

ForeColor="Red"></asp:Label>

</td>

<td>&nbsp;</td>

</tr>

</table>

</div>

</form>

</body>

**3-6)**

using System.Data.SqlClient;

protected void Button1\_Click(object sender, EventArgs e)

{

try

{

using (SqlConnection cn = new SqlConnection(ConfigurationManager.ConnectionStrings["AdvWorks"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand("Insert into Sales.Currency (CurrencyCode, Name) Values (@Cod, @Nombre)", cn))

{

cmd.Parameters.Add(new SqlParameter("@Cod", SqlDbType.VarChar, 3));

cmd.Parameters.Add(new SqlParameter("@Nombre", SqlDbType.VarChar, 50));

cmd.Parameters["@Cod"].Value = txtCodigo.Text; cmd.Parameters["@Nombre"].Value = txtNombre.Text;

cn.Open();

cmd.ExecuteNonQuery();

txtCodigo.Text = "";

txtNombre.Text = "";

txtCodigo.Focus();

}

}

}

catch (SqlException ex)

{

lblErrores.Text = ex.ToString();

}

catch (Exception ex)

{

lblErrores.Text = ex.ToString();

}

}

**8)**

**Procedimiento Almacenado**

CREATE PROCEDURE GrabarMonedas @Cod varchar(3), @Nombre varchar(50)

AS

Insert into Sales.Currency (CurrencyCode, Name) Values (@Cod, @Nombre)

**Código**

using System.Data.SqlClient;

protected void Button1\_Click(object sender, EventArgs e)

{

try

{

using (SqlConnection cn = new SqlConnection(ConfigurationManager.ConnectionStrings["AdvWorks"].ConnectionString))

{

using (SqlCommand cmd = new SqlCommand("GrabarMonedas", cn))

{

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add(new SqlParameter("@Cod", SqlDbType.VarChar, 3));

cmd.Parameters.Add(new SqlParameter("@Nombre", SqlDbType.VarChar, 50));

cmd.Parameters["@Cod"].Value = txtCodigo.Text;

cmd.Parameters["@Nombre"].Value = txtNombre.Text;

cn.Open();

cmd.ExecuteNonQuery();

txtCodigo.Text = "";

txtNombre.Text = "";

txtCodigo.Focus();

}

}

}

catch (SqlException ex)

{

lblErrores.Text = ex.ToString();

}

catch (Exception ex)

{

lblErrores.Text = ex.ToString();

}

}

## Ejercicio 5

En este ejercicio se trabajará usando LINQ

**Pasos**

1. Sobre el ejercicio anterior agregar una nueva página. Agregar un GridView.
2. Agregar un **LINQ to SQL Classes**. Incorporar las tablas Person.CountryRegion y Person.StateProvince de la base de datos AdventureWorks
3. Escribir una sentencia usando LINQ de manera de mostrar el código y nombre de los países y el código y nombre de las provincias asociados a ellos. Solo deben aparecer los países que tienen provincias ingresadas.
4. Mostrar el resultado en el GridView ingresado
5. Ejecutar

**Solución**

using System.Data.SqlClient;

protected void Page\_Load(object sender, EventArgs e)

{

DataClassesDataContext dc = new DataClassesDataContext();

var q = from c in dc.CountryRegions

join s in dc.StateProvinces

on c.CountryRegionCode equals s.CountryRegionCode

select new { c.CountryRegionCode, c.Name,

s.StateProvinceCode, NombreProvincia=s.Name };

GridView1.DataSource=q;

GridView1.DataBind();

}

## Ejercicio 6

En este ejercicio se trabajará usando Entity FrameWork

**Pasos**

1. Sobre el ejercicio anterior agregar una nueva página. Agregar un GridView.
2. Agregar un **ADO.NET Entity Data Model**. Seguir el Wizard (asistente) e incorporar las tablas Person.CountryRegion y Person.StateProvince de la base de datos AdventureWorks
3. Escribir la misma sentencia del ejemplo anterior y mostrar el resultado en el GridView (la sentencia tiene una sintaxis diferente si usamos Entity Framework)
4. Ejecutar

**Solución**

using System.Data.SqlClient;

protected void Page\_Load(object sender, EventArgs e)

{

AdventureWorksModel.AdventureWorksEntities edm = new AdventureWorksModel.AdventureWorksEntities();

var q = from c in edm.CountryRegion

from s in c.StateProvince

select new

{

c.CountryRegionCode, c.Name,

CodigoProvincia = s.StateProvinceCode,

NombreProvincia = s.Name

};

GridView1.DataSource = q;

GridView1.DataBind();

}

# Módulo VIII: Datos en Formato XML

## Ejercicio 1

En este ejercicio se trabajará usando datos en XML.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. En la página Default.aspx creada automáticamente, agregar un Button.
5. En el evento clic del button, crear una DataTable que contenga los datos de la tabla Person.CountryRegion de la base de datos AdventureWorks. Al crear la tabla asignarle un nombre.
6. Generar 3 archivos en formato XML, uno sin esquema, otro con esquema y el último en formato DiffGramm
7. Ejecutar
8. Revisar los archivos generados

**Solución**

using System.Data.SqlClient;

protected void Button1\_Click(object sender, EventArgs e)

{

using (DataTable dt = new DataTable())

{

using (SqlConnection cn = new SqlConnection(ConfigurationManager.ConnectionStrings["AdvWorks"].ConnectionString))

{

using (SqlDataAdapter da = new SqlDataAdapter("Select \* from Person.CountryRegion", cn))

{

da.Fill(dt);

}

}

dt.TableName = "Paises";

dt.WriteXml(Server.MapPath("Paises.xml"), XmlWriteMode.IgnoreSchema);

dt.WriteXml(Server.MapPath("PaisesConEsquema.xml"), XmlWriteMode.WriteSchema);

dt.WriteXml(Server.MapPath("PaisesDG.xml"), XmlWriteMode.DiffGram);

}

}

## Ejercicio 2

En este ejercicio se trabajará usando datos en XML.

**Pasos**

1. Sobre la misma página del ejercicio anterior agregar un nuevo Button.
2. En el evento clic del segundo button crear un DataSet que contenga dos tablas. Una basada en la tabla Person.CountryRegion y la segunda en la tabla Person.StateProvince.
3. Generar una relación entre ambas tablas (**DataRelation**) usando el campo CountryRegionCode presente en ambas tablas
4. Generar dos archivos XML, el primero generando el anidado de las tablas y el segundo sin anidar
5. Ejecutar
6. Revisar los archivos generados, verificar las diferencias entre ellos

**Solución**

protected void Button2\_Click(object sender, EventArgs e)

{

using (DataSet ds = new DataSet())

{

using (SqlConnection cn = new SqlConnection(ConfigurationManager.ConnectionStrings["AdvWorks"].ConnectionString))

{

using (SqlDataAdapter da = new SqlDataAdapter("Select \* from Person.CountryRegion", cn))

{

da.Fill(ds, "Paises");

}

using (SqlDataAdapter da = new SqlDataAdapter("Select \* from Person.StateProvince", cn))

{

da.Fill(ds, "Provincias");

}

}

DataRelation dr = new DataRelation("Pais-Provincia",

ds.Tables["Paises"].Columns["CountryRegionCode"],

ds.Tables["Provincias"].Columns["CountryRegionCode"]);

ds.Relations.Add(dr);

dr.Nested = true;

ds.WriteXml(Server.MapPath("PaisesConRelacion.xml"), XmlWriteMode.WriteSchema);

dr.Nested = false;

ds.WriteXml(Server.MapPath("PaisesSinRelacion.xml"), XmlWriteMode.WriteSchema);

}

}

# Módulo IX: AJAX

## Ejercicio 1

En este ejercicio se trabajará usando AJAX para crear el manejo de dos listas desplegables asociadas entre ellas.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. En la página Default.aspx creada automáticamente, agregar un **ScriptManager**.
5. Agregar también un **UpdatePanel**.
6. Dentro del UpdatePanel agregar dos DropDownList. El primero contendrá los datos de los países, el segundo los datos de las provincias. Modificar para el primero la propiedad **AutoPostBack** a **true**.
7. En este ejercicio se utilizaran las tablas Person.CountryRegion y Person.StateProvince de la base de datos AdventureWorks. El acceso a los datos debe hacerlo usando **LINQ**
8. Agregar a la aplicación un **LINQ to SQL Classes** con las tablas mencionadas en el punto anterior.
9. En el evento Load de la página escriba código para llenar el DropDownList de países.
10. En el evento **SelectedIndexChanged** del DropDownList de países escriba código para mostrar las provincias asociadas al país seleccionado
11. Probar. Tener en cuenta que no todos los países tienen provincias ingresadas, algunos países que si tienen son: Australia, Canadá y United State. Si selecciona alguno de los demás la lista de provincias aparecerá vacía.

**Solución**

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

DataClassesDataContext dc = new DataClassesDataContext();

var q = from c in dc.CountryRegions select c;

ddlPaises.DataSource = q;

ddlPaises.DataTextField = "Name";

ddlPaises.DataValueField = "CountryRegionCode";

ddlPaises.DataBind();

}

}

protected void ddlPaises\_SelectedIndexChanged(object sender, EventArgs e)

{

DataClassesDataContext dc = new DataClassesDataContext();

var q = from s in dc.StateProvinces

where s.CountryRegionCode == ddlPaises.SelectedItem.Value

select s;

ddlProvincias.DataSource = q;

ddlProvincias.DataTextField = "Name";

ddlProvincias.DataValueField = "StateProvinceCode";

ddlProvincias.DataBind();

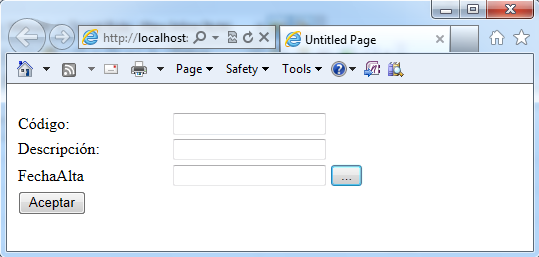
}

## Ejercicio 2

En este ejercicio se trabajará usando el AJAX Toolkit.

**Pasos**

1. Descargar el AJAX Toolkit de la página <http://ajaxcontroltoolkit.codeplex.com/>. Descomprimir el archivo descargado.
2. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
3. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
4. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
5. En la **ToolBox** crear una nueva solapa llamada AJAX Toolkit. Presionar botón derecho y seleccionar la opción **Elegir Elementos.** Hacer clic en el botón **Examinar** y navegar hasta encontrar el archivo dll extraído en el paso 1, presionar Aceptar
6. Diseñar la página Default.aspx creada automáticamente de manera que sea similar a la imagen (Agregar como primer paso un **ScriptManager**)



1. Tanto el código como la descripción deben ser obligatorios. Agregar controles de validación para verificar que se ingresen datos.
2. Al textbox del Código agregarle un Extender de tipo **FilteredTextBox**. Configurarlo para que solo puedan ingresarse números.
3. Al textbox de la Fecha de Alta agregarle un Extender de tipo **Calendar**. Configurarlo de manera que el formato de la fecha sea dd/mm/aaaa y que el calendario se muestre al presionar el botón que se encuentra a su lado.
4. A los controles de validación ingresados agregarles un Extender de tipo **ValidatorCallout**.
5. Ejecutar.

**Solución**

<head runat="server">

<title>Untitled Page</title>

<style type="text/css">

.style1

{

width: 100%;

}

</style>

</head>

<body>

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

<div>

<asp:ScriptManager ID="ScriptManager1" runat="server">

</asp:ScriptManager>

<br />

<table class="style1">

<tr>

<td>

<asp:Label ID="Label1" runat="server"

Text="Código:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtCodigo"

runat="server"></asp:TextBox>

<cc1:FilteredTextBoxExtender

ID="txtCodigo\_FilteredTextBoxExtender"

runat="server" Enabled="True"

FilterType="Numbers"

TargetControlID="txtCodigo">

</cc1:FilteredTextBoxExtender>

<asp:RequiredFieldValidator

ID="RequiredFieldValidator1" runat="server"

ControlToValidate="txtCodigo"

ErrorMessage="El código es obligatorio">\*

</asp:RequiredFieldValidator>

<cc1:ValidatorCalloutExtender

ID="RequiredFieldValidator1\_ValidatorCalloutExtender"

runat="server" Enabled="True"

TargetControlID="RequiredFieldValidator1">

</cc1:ValidatorCalloutExtender>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label2" runat="server"

Text="Descripción:"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtDescripcion"

runat="server"></asp:TextBox>

<asp:RequiredFieldValidator

ID="RequiredFieldValidator2" runat="server"

ControlToValidate="txtDescripcion"

ErrorMessage="La descripción es obligatoria">\*

</asp:RequiredFieldValidator>

<cc1:ValidatorCalloutExtender

ID="RequiredFieldValidator2\_ValidatorCalloutExtender"

runat="server" Enabled="True"

TargetControlID="RequiredFieldValidator2">

</cc1:ValidatorCalloutExtender>

</td>

</tr>

<tr>

<td>

<asp:Label ID="Label3" runat="server"

Text="FechaAlta"></asp:Label>

</td>

<td>

<asp:TextBox ID="txtFecha"

runat="server"></asp:TextBox>

<cc1:CalendarExtender

ID="txtFecha\_CalendarExtender" runat="server"

DaysModeTitleFormat="dd/MM/yyyy" Enabled="True"

Format="dd/MM/yyyy" PopupButtonID="btnAyuda"

TargetControlID="txtFecha"

TodaysDateFormat="dd/MM/yyyy">

</cc1:CalendarExtender>

<asp:Button ID="btnAyuda" runat="server"

Text="..." />

</td>

</tr>

<tr>

<td>

<asp:Button ID="btnAceptar" runat="server"

Text="Aceptar" />

</td>

<td>&nbsp;</td>

</tr>

</table>

</div>

</form>

</body>

# Módulo X: Servicios Web XML

## Ejercicio 1

En este ejercicio creará un servicio web XML. Se utilizará el Entity Framework para comunicarse con la base de datos.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Service**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. Crear un nuevo método llamado TraerProductos que devuelva todos los campos de la tabla Production.Product de la base de datos AdventureWorks. El tipo de dato de la devolución será una lista genérica **List<AdventureWorksModel.Product>**, siendo AdventureWorksModel el nombre del **Entity Data Model** creado en el siguiente punto
5. Este método debe comunicarse con la base de datos usando **Entity FrameWork**. Para ello agregue al servicio web un **ADO.NET Entity Data Model** e incluya la tabla mencionada en el punto anterior.
6. Compilar
7. Ejecutar. Probar el método en la página de ayuda.

**Solución**

[WebMethod()]

public List<AdventureWorksModel.Product> TraerProductos()

{

AdventureWorksModel.AdventureWorksEntities edm = new AdventureWorksModel.AdventureWorksEntities();

var q = from p in edm.Product select p;

List<AdventureWorksModel.Product> lista = new List<AdventureWorksModel.Product>();

foreach (var item in q)

{

lista.Add((AdventureWorksModel.Product)item);

}

return lista;

}

## Ejercicio 2

En este ejercicio creará un cliente para el servicio web XML creado en el ejercicio anterior.

**Pasos**

1. Crear un sitio web dentro de la misma solución del servicio creado en el ejercicio anterior. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site.** Marcarlo como proyecto de inicio.
3. Crear una **Referencia Web**. En el Explorador de Soluciones, botón derecho sobre el nombre del sitio web, **Add Web Reference**.
4. En la ventana de diálogo de Agregar Referencia Web, seleccionar **Web Service in this Solution**. Esto encontrará el servicio creado en el ejercicio anterior. Seleccionarlo y presionar **Add Reference**. Si quiere puede modificar el namespace localhost que aparece en forma predeterminada.
5. En la página Default.aspx creada automáticamente, poner un GridView
6. En el evento Load de la página escribir el código para invocar al método TraerProductos del servicio. Atajar errores.
7. Mostrar el resultado en el Gridview.
8. Ejecutar

**Solución**

protected void Page\_Load(object sender, EventArgs e)

{

try

{

localhost.Service obj = new localhost.Service();

GridView1.DataSource = obj.TraerProductos();

GridView1.DataBind();

}

catch (System.Net.WebException ex)

{

Label1.Text = ex.ToString();

}

catch (System.Web.Services.Protocols.SoapException ex)

{

Label1.Text = ex.ToString();

}

}

# Módulo XI: Configurar, Optimizar e Implementación de Aplicaciones ASP.NET

## Ejercicio 1

En este ejercicio trabajará con Cache.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. En la página Default.aspx creada automáticamente, agregar un Button.
5. En el evento Clic del Button crear un objeto Cache llamado CachePrueba, como valor asignarle la fecha y hora actual, no generar dependencia, la vida del objeto será de 30 minutos exactos y la prioridad deberá ser alta.
6. Luego crear un segundo objeto llamado CachePrueba2 cuyo valor sea “Texto de Prueba”, con una dependencia a un archivo que se encuentra en el directorio raíz de la aplicación llamado Archivo.xml. La vida del objeto es relativa al uso, si no se la accede en 15 minutos debería eliminarse y su prioridad deberá ser normal.

**Solución**

using System.Web.Caching;

protected void Button1\_Click(object sender, EventArgs e)

{

Cache.Add("CachePrueba", DateTime.Now, null,

DateTime.Now.AddMinutes(30), Cache.NoSlidingExpiration,

CacheItemPriority.High, null);

Cache.Add("CachePrueba2", "Texto de Prueba", new

CacheDependency(Server.MapPath("Archivo.xml")),

Cache.NoAbsoluteExpiration, TimeSpan.FromMinutes(15),

CacheItemPriority.Normal, null);

}

## Ejercicio 2

En este ejercicio trabajará con output cache.

**Pasos**

1. Dentro del mismo sitio del ejercicio 1 crear una nueva página.
2. Agregar un Label.
3. En el evento Load de la página asignar la fecha y hora actual a la propiedad Text del Label
4. Ejecutar. Si presiona varias veces F5 verá que la hora va cambiando
5. Agregarle a la página una directiva **output cache** de manera que almacene la imagen durante 30 segundos.
6. Ejecutar. La hora no debería cambiar durante 30 segundos.

**Solución**

**3)**

protected void Page\_Load(object sender, EventArgs e)

{

Label1.Text = DateTime.Now.ToString();

}

**5)**

<%@ OutputCache Duration="30" VaryByParam="None" %>

## Ejercicio 3

En este ejercicio trabajará con el archivo de configuración.

**Pasos**

1. Dentro del mismo sitio del ejercicio 1 crear una nueva página.
2. Agregar un Button.
3. En el evento clic del Button escribir un código que genere una división por cero. No escribir código de atajado de errores. La idea es generar un código que provoque un error no atajado.
4. En el evento Load de la página crear una variable de sesión llamada “Pagina” cuyo valor sea el nombre de la página en la cual está trabajando.
5. Crear otra página llamada Errores.aspx. Incluir un Label cuya propiedad Text esté vacía y un Button que diga “Volver”
6. En el evento Load de la página de Errores asignar el mensaje “Se produjo un error” a la propiedad Text del Label
7. En el evento clic del button Volver redireccionar a la página cuyo nombre se encuentra en la variable de sesión. Tener en cuenta que las sesiones pueden caducar.
8. En el **web.config** configurar la sección **customErrors** de manera de redireccionar los errores a la página Errores.aspx
9. Ejecutar seleccionando **Start without Debugging** desde el menú **Debug** (si ejecuta normalmente los errores son atajados por el IDE, con lo cual no podemos probar esta funcionalidad)
10. Crear dentro del **web.config** una **variable de configuración** llamada MensajeError. Como valor ponerle “La página generó una excepción”
11. Modificar el código de la página Errores.aspx de manera que muestre el mensaje leyéndolo del web.config y no el texto fijo que pusimos en el punto 6.
12. Puede utilizar la herramienta **ASP.NET Configuration** (**Administración de Sitios Web)** para generar las modificaciones del web.config. La misma se encuentra en el menú **WebSite**.

**Solución**

**3)**

protected void Button1\_Click(object sender, EventArgs e)

{

int a = 10;

int b = 0;

int c = a / b;

}

**4)**

protected void Page\_Load(object sender, EventArgs e)

{

Session["Pagina"] = "Default.aspx";

}

**6, 7) Errores.aspx**

public partial class Errores : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

lblMensaje.Text = "Se produjo un error";

}

protected void Button1\_Click(object sender, EventArgs e)

{

if (Session["Pagina"] != null)

{

Response.Redirect(Session["Pagina"].ToString());

}

Else

Response.Redirect("Default.aspx");

}

}

**8)**

<customErrors mode="On" defaultRedirect="~/Errores.aspx" />

**10)**

<appSettings>

<add key="MensajeError" value="La página generó una excepción" />

</appSettings>

**11)**

protected void Page\_Load(object sender, EventArgs e)

{

lblMensaje.Text =

ConfigurationManager.AppSettings["MensajeError"].ToString();

}

## Ejercicio 4

En este ejercicio se implementará el sitio web creado en los ejercicios anteriores de las dos formas existentes para sitios web.

**Pasos**

1. Crear dos directorios virtuales de manera de probar las dos opciones de implementación que tenemos disponibles en .Net
2. Para crear un directorio virtual puede crear las carpetas dentro de la ruta **\inetpub\wwwroot** y después convertir estas carpetas en directorios virtuales o puede ir directamente al IIS y seguir el Wizard (Asistente) de creación de sitios web (Aplicaciones en IIS 7.0).
3. Las carpetas podrían llamarse ProduccionCopy y ProduccionPublish
4. Para entrar al IIS puede hacerlo haciendo botón derecho sobre el icono de **Mi PC** y seleccionar la opción **Manage (Administrar)**. En la solapa **Services and Aplication (Servicios y Aplicaciones)** encontraremos al **Internet Information Service (IIS)**. Si este no figura en por qué no está instalado.
5. También puede entrar desde el **Panel de control**, **Herramientas administrativas** y luego seleccionar **Servicios de Internet Information Server**.
6. Si creó las carpetas en la ruta definida en el punto 2 aparecerá la carpeta en la lista de **Default Web Site**, botón derecho sobre la carpeta **Propiedades**, en la solapa general presionar el botón **Create** (si se está trabajando con el IIS 7.0 deberá hacer botón derecho **Convert to Application**).
7. Si decide usar el Wizard (Asistente) sobre la opción **Default Web Site**, botón derecho **Add New WebSite (Agregar Nuevo Sitio)**, si está trabajando con IIS 7.0 **Add Aplication (Agregar Aplicación)**. Luego siga las instrucciones del asistente.
8. Primero se usará la opción **Copy Web Site** para implementar el sitio. Esta opción se encuentra dentro del menú **WebSite**.
9. Dentro del cuadro de diálogo del **Copy Web Site** presionar el botón **Connect**, seleccionar la carpeta llamada ProduccionCopy.
10. Seleccionar sobre el cuadro de la izquierda todos los archivos que quiere copiar. Copiarlos usando los botones con flechitas que se encuentra en el medio de la ventana.
11. Ejecute ahora desde producción. Dentro del Internet Explorer la dirección deberá ser <http://localhost/ProduccionCopy/default.aspx> (puede poner el nombre de la máquina en vez de localhost)
12. Dentro del Visual Studio 2008 generar alguna modificación sobre alguna de las páginas existentes o agregar una nueva página. Verifique que en el cuadro de diálogo del Copy Web Site se reflejan estas modificaciones.
13. Ahora se usará la opción de **Publish**, para acceder a la misma puede presionar botón derecho sobre el nombre del proyecto en el Explorador de Soluciones y seleccionar la opción **Publish**.
14. En el cuadro del diálogo redireccione la carpeta a donde copia y apunte a la carpeta ProduccionPublish creada en el punto 3
15. No modifique el resto de las opciones y confirme
16. Verifique que archivos se copiaron a la carpeta ProduccionPublish. Verifique las dll que fueron creadas durante el proceso.
17. Para probar desde el Internet Explorer, <http://localhost/ProduccionPublish/default.aspx>
18. Si vuelve a publicar podrá verificar que el nombre de las dll creadas cambia con cada publicación

# Módulo XII: Seguridad

## Ejercicio 1

En este ejercicio trabajará con seguridad de formularios.

**Pasos**

1. Abrir Microsoft Visual Studio .Net 2008. Seleccionar **Create** *–* **New Site**.
2. Dentro de la ventana de Create elegir **ASP.NET Web Site**.
3. En **Location** elegir “**File System**”. Seleccionar una carpeta y agregarle el nombre del sitio. Como lenguaje seleccione **C#.** Presionar Ok.
4. En la página Default.aspx agregar un Label.
5. En el evento Load de esta página escribir código para mostrar en ese Label el usuario logueado usando la propiedad **User**. Marcar que se quiere navegar esta página al ejecutar el sitio.
6. Agregar una nueva página llamada Autenticacion.aspx. En la misma armar un pedido de usuario y contraseña.
7. En el clic del botón Aceptar verificar si el usuario y contraseña ingresados por el usuarios son correcto (validar contra valores predefinidos en el código). Si son correctos redireccionar la llamada a la página a la cual se estaba intentando entrar (en este caso a la página Default.aspx, pero generar el código para que el nombre de la página sea genérico y se cree la cookie de control de autenticación). Si son incorrectos mostrar un mensaje de error.
8. Configurar el **web.config** para habilitar la seguridad de formularios. En el caso de no estar autenticado redireccionar a la página Autenticacion.aspx
9. Configurar el **web.config** para no permitir el acceso de usuarios anónimos
10. Ejecutar

**Solución**

**5)**

protected void Page\_Load(object sender, EventArgs e)

{

lblUsuario.Text = User.Identity.Name;

}

**6)**

<body>

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

<div>

<table>

<tr>

<td>Usuario:</td>

<td>

<asp:TextBox ID="txtUsuario" runat="server" />

</td>

<td>

<asp:RequiredFieldValidator ID="RequiredFieldValidator1"

ControlToValidate="txtUsuario" Display="Dynamic"

ErrorMessage="Debe ingresar el usuario." runat="server" />

</td>

</tr>

<tr>

<td>Contraseña:</td>

<td>

<asp:TextBox ID="txtCont" TextMode="Password" runat="server"

/>

</td>

<td>

<asp:RequiredFieldValidator ID="RequiredFieldValidator2"

ControlToValidate="txtCont" runat="server"

ErrorMessage="Debe ingresar la contraseña." />

</td>

</tr>

</table>

<asp:Button ID="btnAceptar" Text="Aceptar" runat="server" OnClick="btnAceptar\_Click" />

<p>

<asp:Label ID="Msg" runat="server" ForeColor="Red"></asp:Label>

</div>

</form>

</body>

**7)**

<authentication mode="Forms">

<forms name="autenticacion" loginUrl="Autenticacion.aspx"/>

</authentication>

**8)**

<authorization>

<deny users="?"/>

</authorization>