



ASP.NET 3.5

IDE VISUAL STUDIO .NET

CLIPSA



cipsa.net

DISTRIBUIDO POR:

**CENTRO DE INFORMÁTICA PROFESIONAL
S.L.**

C/ URGELL, 100
08011 BARCELONA
TFNO: 93 426 50 87

C/ RAFAELA YBARRA, 10
48014 BILBAO
TFNO: 94 448 31 33

www.cipsa.net

**RESERVADOS TODOS LOS DERECHOS. QUEDA PROHIBIDO
TODO TIPO DE REPRODUCCIÓN TOTAL O PARCIAL DE
ESTE MANUAL, SIN PREVIO CONSENTIMIENTO POR EL
ESCRITOR DEL EDITOR**

CLIPSA

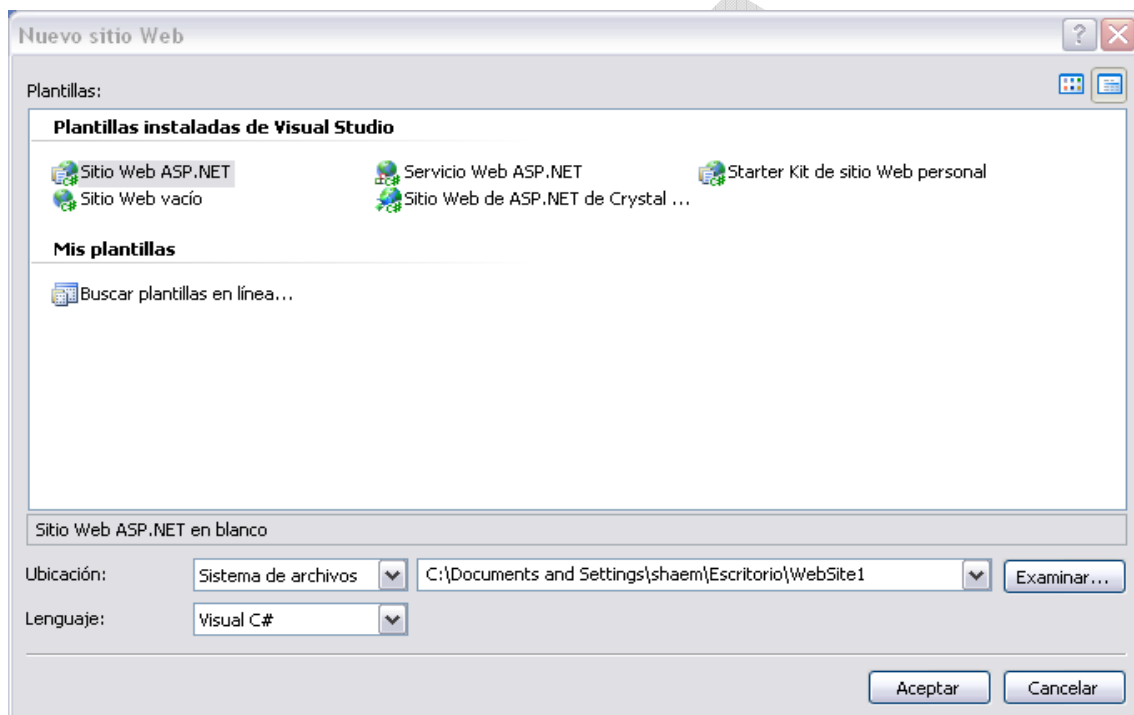
Introducción al IDE Visual Studio

El Visual Studio es un IDE que permite tanto el desarrollo de aplicaciones de Windows convencionales como aplicaciones Web. La principal diferencia entre ambos tipos es que las aplicaciones Web requieren un servidor Web para su funcionamiento desde exploradores.

CREACION DE UN SITIO WEB

Abrimos el Visual Studio:

Archivo → Nuevo → Sitio Web.



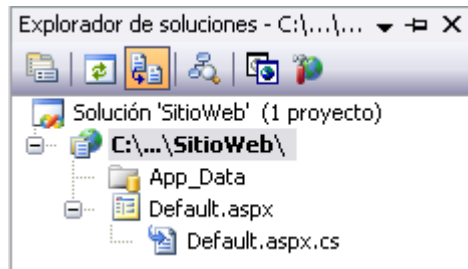
Al crear una nueva aplicación en Visual Studio podemos elegir entre tres opciones de almacenamiento que condicionan la ejecución de la aplicación:

- **Sistema de Archivos:** El Visual Studio almacenará la aplicación Web en la carpeta local del equipo indicada. No requiere tener instalado ningún servicio Web y la ejecución de la aplicación Web se realizará exclusivamente desde el servidor Web del Visual Studio. (*Cassini*)
- **HTTP:** El Visual Studio almacena directamente la aplicación en el IIS instalado en la misma máquina. La aplicación puede ejecutarse mediante un navegador Web independientemente del Visual Studio.
- **FTP:** El Visual Studio almacena la aplicación Web en un Servidor Web remoto empleando el protocolo FTP. La aplicación se ejecutará desde el Servidor Web correspondiente.

También es posible seleccionar el lenguaje de programación entre Visual Basic, C#, y J#.

ESTRUCTURA DE FICHEROS DE UNA APLICACIÓN WEB ASP.NET

Una aplicación ASP.NET consta de multitud de elementos y ficheros. Al crear una nueva aplicación Web, Visual Studio crea una carpeta donde se almacenan todos los archivos.



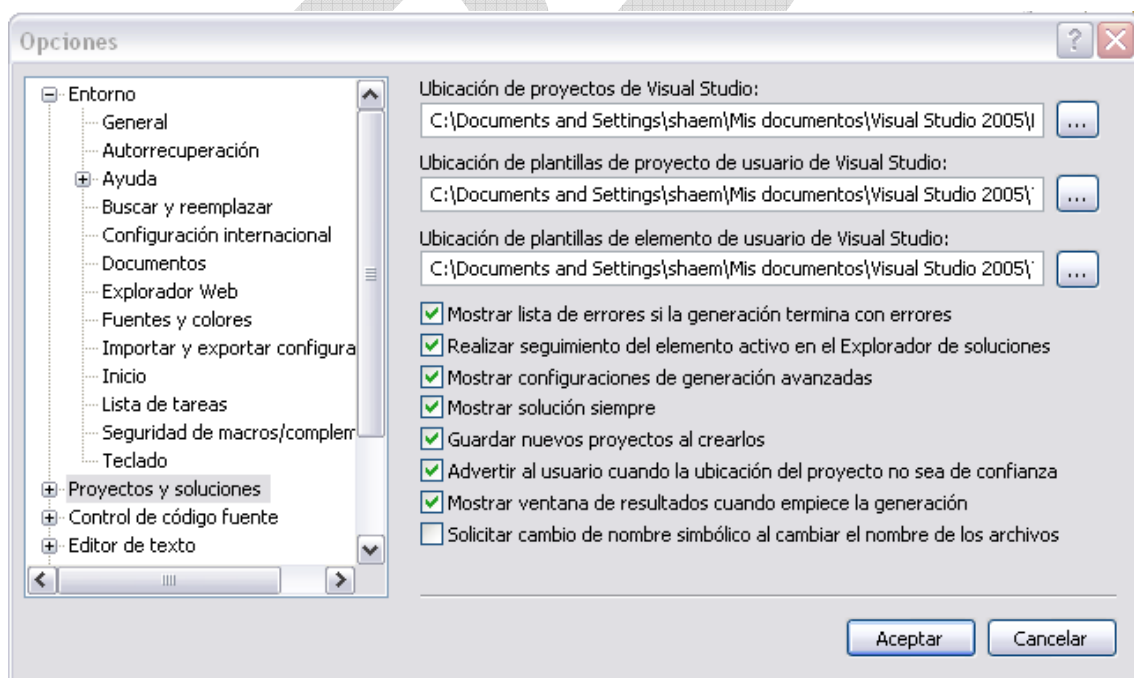
Todos los archivos que componen una aplicación Web componen una solución que se muestra en la ventana del Explorador de Soluciones.

Las páginas Web siempre se encuentran en la solución, no dentro de proyectos.

Las soluciones pueden contener proyectos adicionales con servicios Web, bibliotecas de clases, bibliotecas de Controles Web, Aplicaciones de Windows... etc.

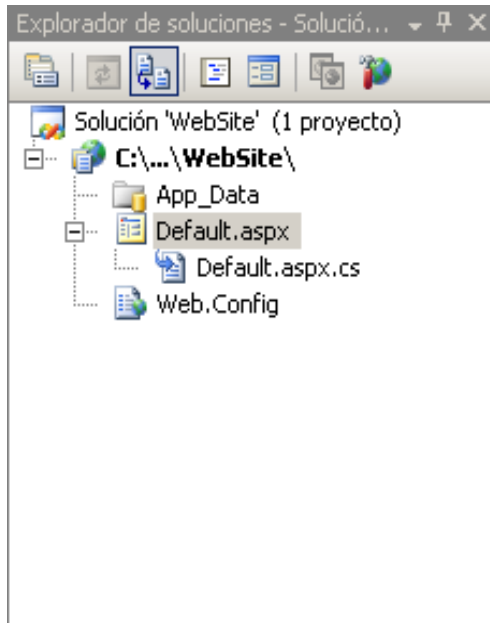
El almacenamiento mediante sistema de archivos, se realiza en la carpeta con el nombre dado al crear la aplicación Web. Adicionalmente, Visual Studio crea dos archivos más: un archivo de soluciones (.sln), y un archivo de preferencias (.suo), ambos con el mismo nombre del sitio Web pero almacenados en la carpeta “*Ubicación de proyectos de Visual Studio*”.

Herramientas → Opciones..



Cuando se crea una nueva aplicación Web, Visual Studio crea por defecto la página Default.aspx acompañada de su fichero de código Default.aspx.cs en caso de haber seleccionado Visual C# como lenguaje de codificación. En caso de seleccionarse Visual Basic, el archivo se denomina Default.aspx.vb.

Cuando se crea una nueva aplicación Web en Visual Studio, se crea por defecto una serie de archivos y carpetas iniciales:



Nodo Solución: Representa a la aplicación Web con todas las páginas, recursos, y proyectos adicionales que requiera.

Nodo Sitio: Representa a la aplicación Web propiamente dicha.

App_Data: Carpeta especial para conexiones a bases de datos.

Default.aspx: La página de inicio de la aplicación Web.

Default.aspx.cs: El fichero con la clase y el código ejecutable asociados a la página Default.aspx y sus componentes.

Web.Config: Fichero de configuración de la aplicación Web con parámetros y valores codificados en valores XML.

La página inicial: Default.aspx es la que se invocará en primer lugar cuando se ejecuta la aplicación para probarla desde el Visual Studio. Para cambiar la página de inicio basta seleccionar cualquier otra página, hacer clic con el botón derecho y seleccionar la opción “Establecer como página de inicio” en el menú contextual.

CARGA DE UNA APLICACION WEB

Para abrir una aplicación Web ya existente:

Archivos → Abrir → Sitio web



Al abrir una aplicación Web podemos seleccionar “Sistema de Archivos” para abrir la aplicación almacenada en una carpeta local de nuestro equipo.

La opción IISLocal permite cargar la solución desde el directorio virtual del Internet Information Service instalado en el propio equipo.

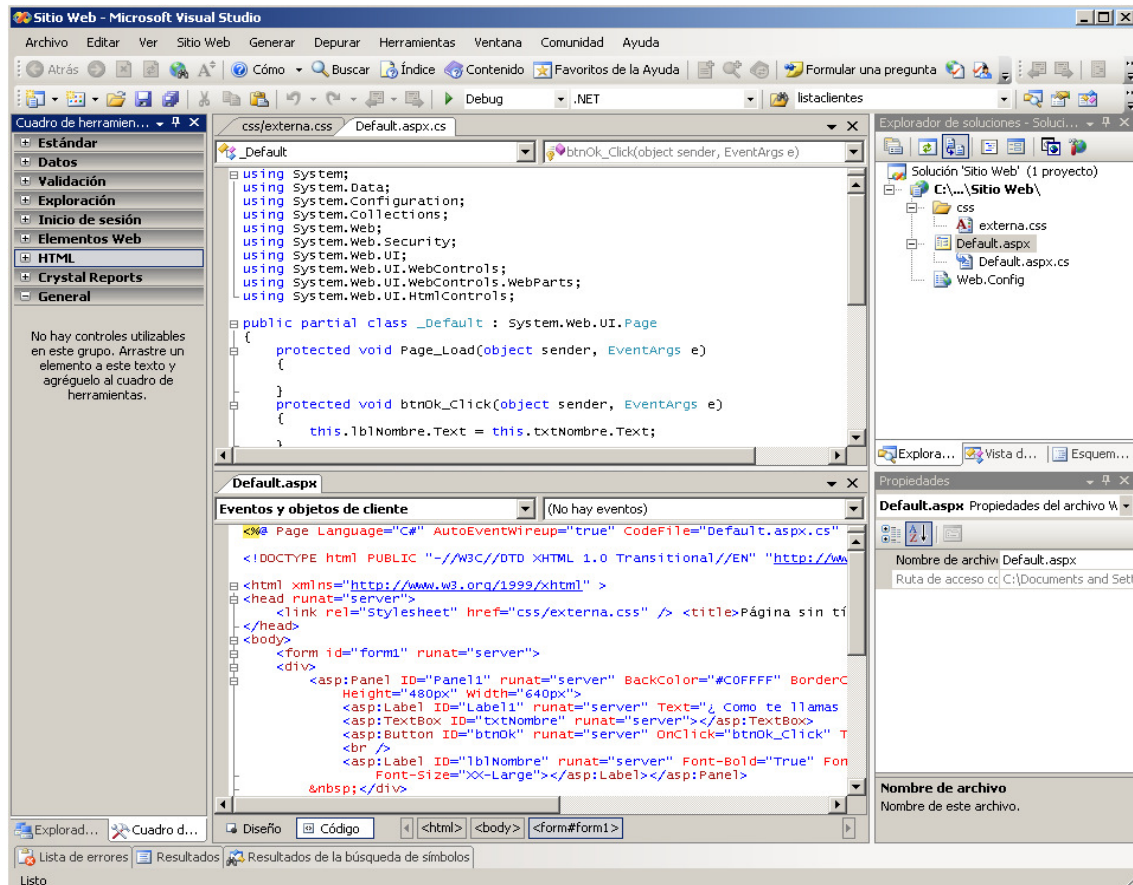
La opción Sitio FTP permite cargar la solución vía FTP desde un servidor Web remoto.

La opción Sitio Web permite cargar la solución vía HTTP desde un servidor Web remoto configurado con extensiones de servidor de FrontPage.

Una diferencia importante al cargar un proyecto Web con respecto a un proyecto convencional de Windows, es que en la aplicación Web se selecciona una carpeta, mientras que en las aplicaciones de Windows se selecciona el archivo de solución (.sln).

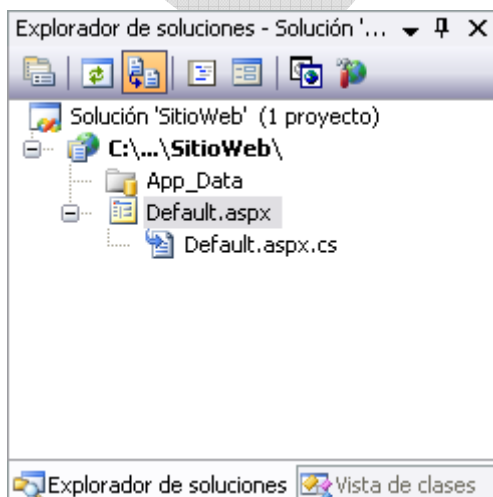
Visión General del IDE


El entorno de desarrollo del Visual Studio está compuesto por un conjunto de ventanas que conforman el entorno de trabajo:




EXPLORADOR DE SOLUCIONES

La ventana Explorador de Soluciones permite ver la estructura completa de una aplicación Web de forma jerárquica. En la barra de herramientas superior se pueden destacar los siguientes componentes:



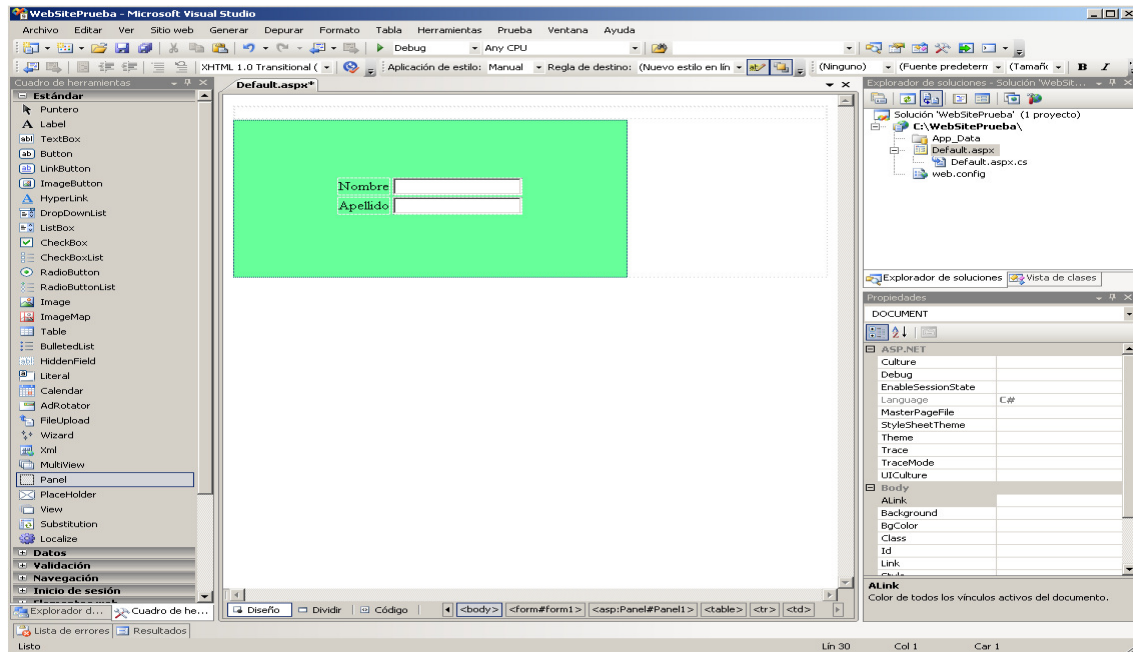
 **Diseño / Código:** Conmutan entre la vista de diseño y de código de una página:

 **Copiar sitio Web:** Permite “subir” la aplicación Web a un servidor Web local o remoto.

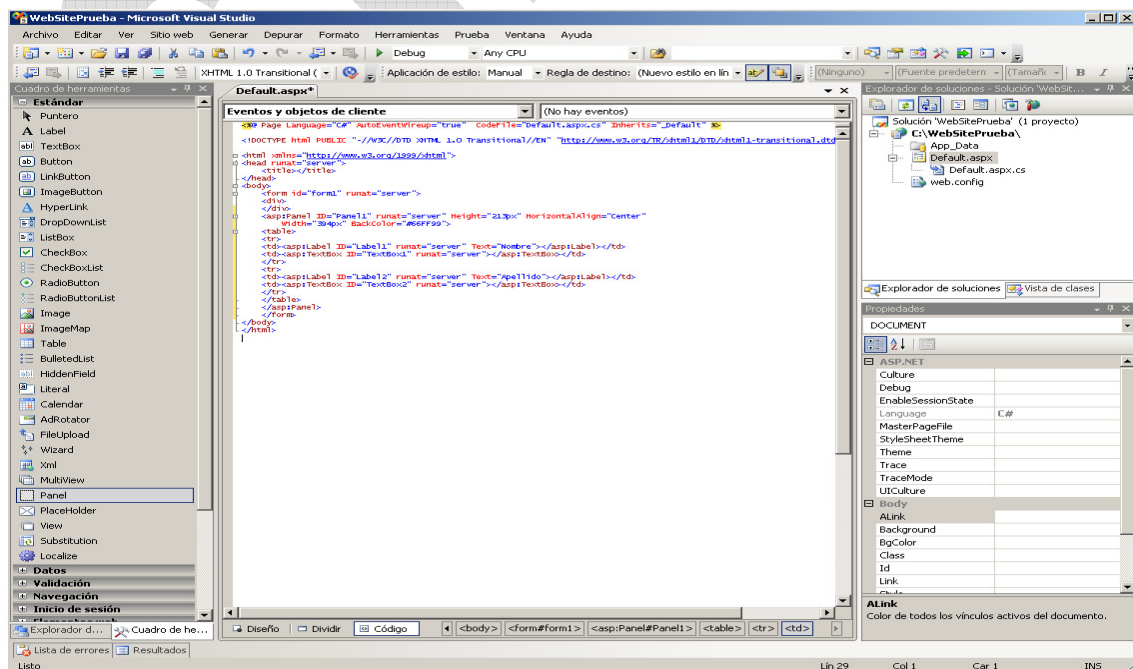
 **Configurar sitio Web:** Permite configurar la aplicación Web mediante una página web autogenerada.

VISTAS DE DISEÑO Y CODIGO

Al trabajar sobre una página ASPX existen dos vistas. Si seleccionamos el fichero de diseño de página (fichero con extensión .aspx) se despliega entonces la ventana de diseño. Esta ventana permite crear y modificar el diseño visual de la página que estamos creando y ofrece tres posibles vistas accesibles mediante las pestañas; “Diseño”, “Código” y “Dividir”.

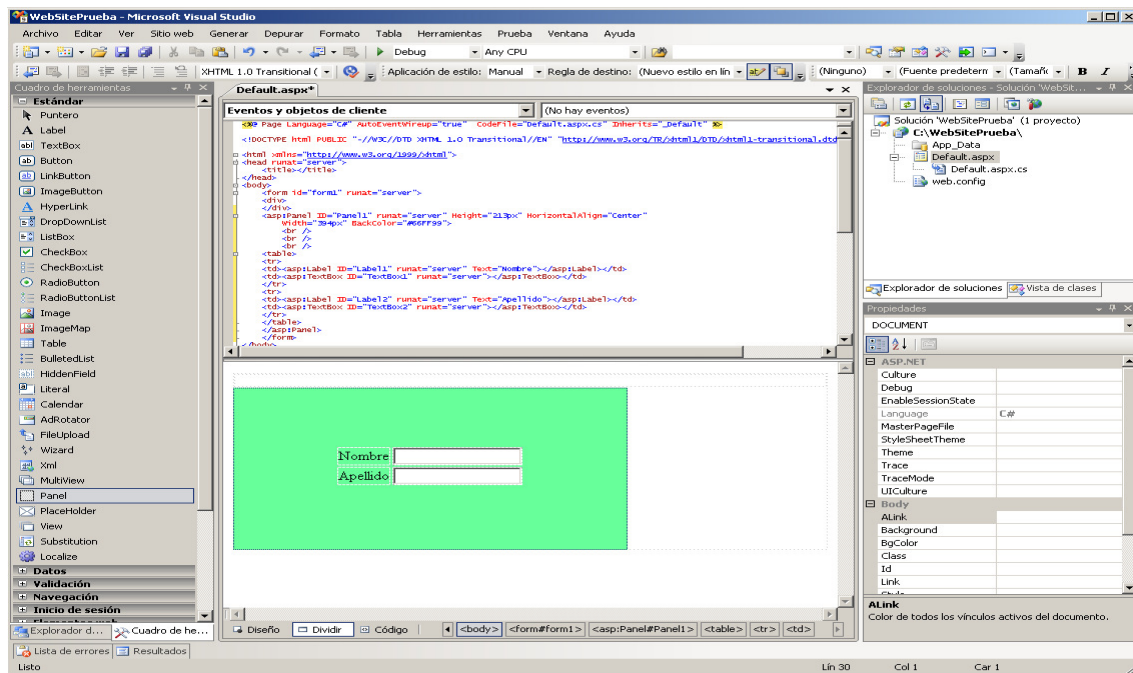


La pestaña de diseño muestra una representación de la página Web en la que se pueden añadir elementos para construir la página visualmente, de igual manera que si fuera un formulario de Windows. La pestaña de código muestra el código HTML de la página, de manera que se puede modificar directamente.

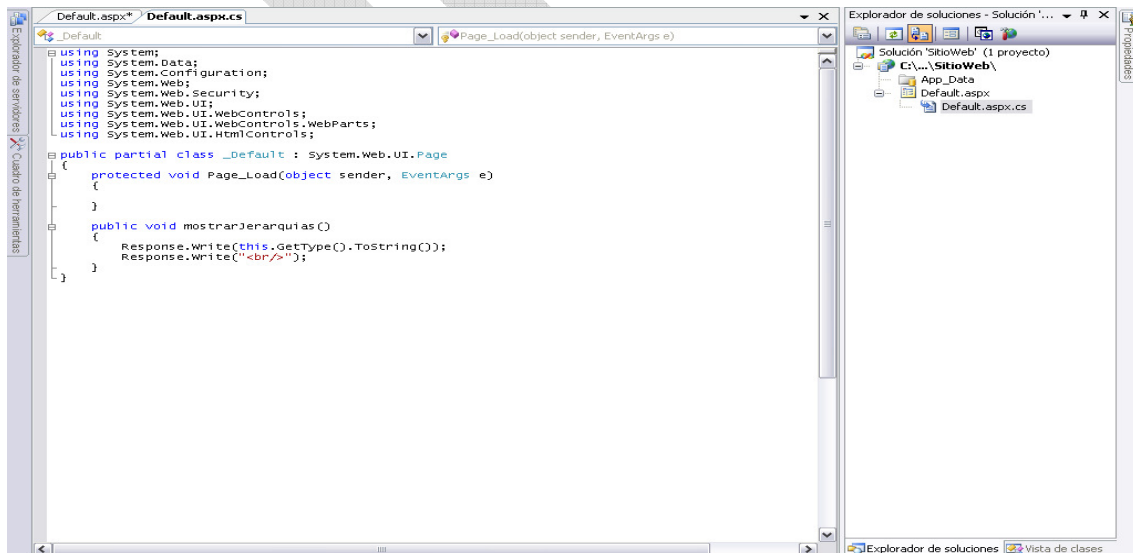


Introducción IDE Visual Studio

Adicionalmente, se puede observar una tercera pestaña llamada “dividir” que permite precisamente dividir el área de trabajo central del IDE en dos ventanas que permiten ver el diseño y el código de diseño al mismo tiempo.



En Visual Studio se puede codificar ‘en línea’ o con “código en el lado”, de manera que cada página lleva asociado un fichero con el código de sus funciones y eventos. El fichero de código de una página (extensión .cs, o .vb) se muestra subordinado en el Explorador de Soluciones. Al seleccionarlo, se muestra su código en una ventana de código.



Cada una de las páginas ASP de una aplicación son clases derivadas de la clase padre System.Web.UI.Page. Esta clase es la clase principal de todas las páginas ASP, también llamadas formularios Web.

VENTANA CUADRO DE HERRAMIENTAS

El cuadro de herramientas nos muestra todos los controles que podemos situar en una página Web de la misma manera que lo haríamos con un formulario. Dichos controles se encuentran subdivididos en distintas categorías. Las más comunes son:



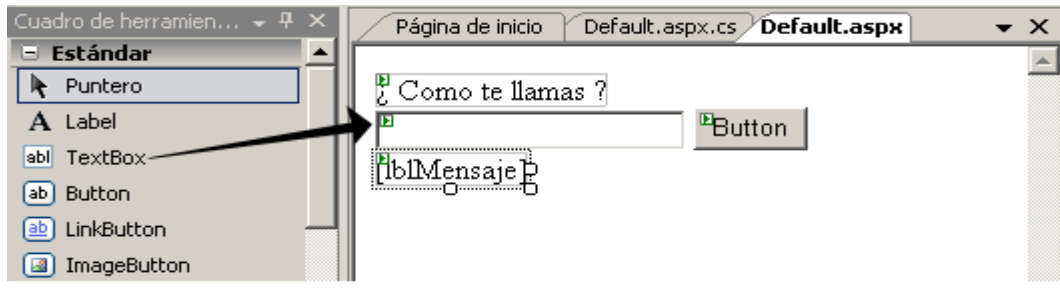
Estandar: Los controles en esta categoría son controles propios de ASP.NET que se ejecutan exclusivamente en el servidor y generan HTML al ser enviados al navegador cliente

HTML: Los controles típicos de HTML. Pueden configurarse para ejecutarse en el navegador del cliente, o en el servidor añadiendo atributo runat="Server"

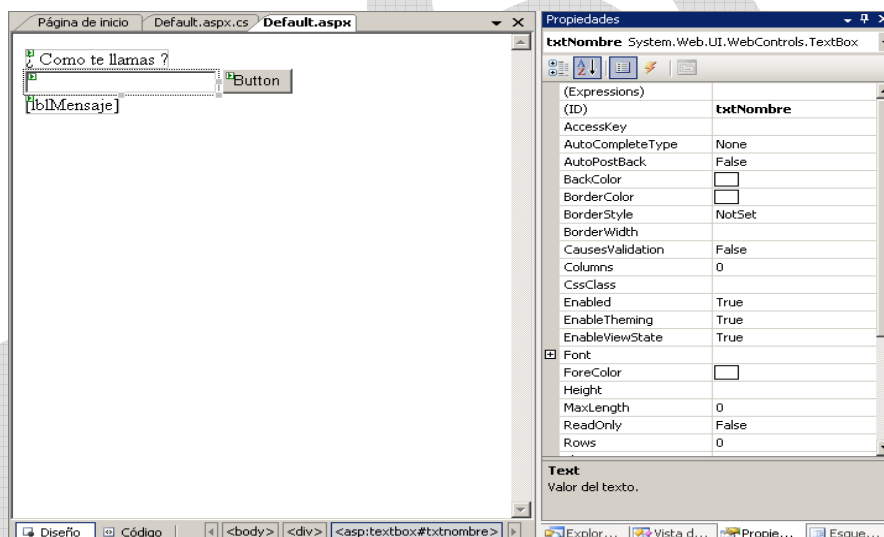
Edición de una página ASP.NET

AÑADIR CONTROLES

Para añadir controles a la página ASP se seleccionan en el cuadro de herramientas y se colocan en la vista diseño de la página Web arrastrando y soltando.



Cada control Web tiene sus propias propiedades y eventos que pueden modificarse en la Ventana de Propiedades al seleccionarlos.



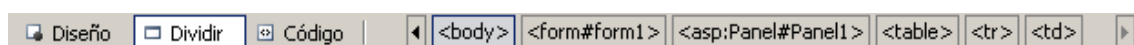
Propiedades del control TextBox al seleccionarlo en la ventana de diseño

La propiedad más importante de cualquier control es ID. Esta propiedad es común a todos los controles e indica el nombre de la instancia que referencia al control en el código ejecutable de la página.

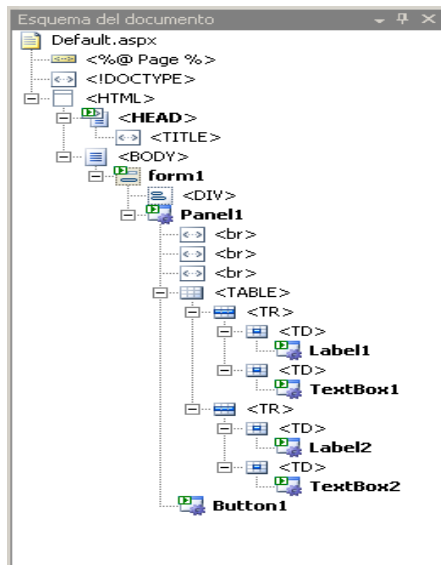
Para modificar las propiedades de un control es necesario seleccionarlo.

La selección de un control de la página puede hacerse de tres maneras:

- 1.- Seleccionándolo con el ratón en la vista diseño
- 2.- Seleccionando su etiqueta en la vista de código de diseño.
- 3.- Seleccionando en la barra de etiquetas inferior:



4.- Seleccionando el elemento en la vista jerárquica del explorador de documentos:



La ventana de Esquema de documento presenta una visión jerárquica de los controles dentro de una página ASP.

Si esta ventana no aparece debe seleccionarse la opción:

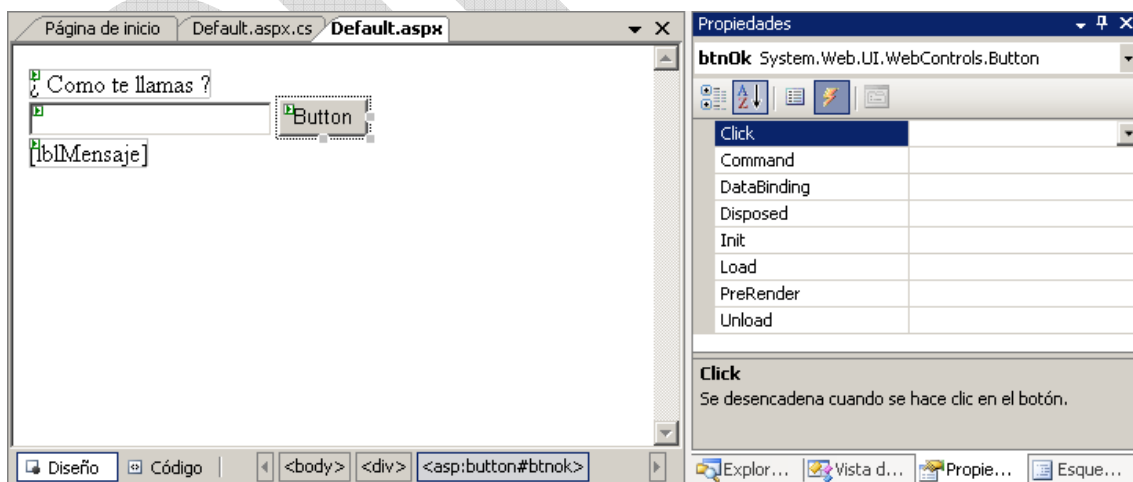
Ver→Otras Ventanas→Esquema de Documento

Mediante esta ventana se pueden seleccionar más cómodamente todos los controles de una página Web.

MANEJO DE EVENTOS

Los controles Web poseen una serie de eventos que se ejecutan cuando el usuario lleva a cabo determinadas acciones sobre el control. Estos eventos pueden programarse asociando un determinado comportamiento.

Los eventos de un control son visibles en la Ventana de Propiedades si pulsamos el icono del rayo (cuatro icono de la barra de herramientas superior en la ventana de Propiedades).



Eventos de un control Boton de ASP.NET

Si hacemos doble-click en el cuadro del evento de click, se creará automáticamente el método `btnOk_Click(..)`. Este método será llamado al producirse el evento click del control `btnOn`.

```
protected void btnok_Click(object sender, EventArgs e)
```

Introducción IDE Visual Studio

Los eventos se programan empleando delegados. Un delegado es un objeto que referencia a un método para que sea invocado cuando se produce el evento. Estos métodos de respuesta poseen por norma general dos parámetros de entrada:

- **Parámetro sender:** Es la instancia del control originario del evento.
- **Parámetro e:** Es una instancia de la clase EventArgs que contiene diferentes informaciones adicionales sobre las características del evento en cuestión.

El código contenido en ese nuevo método se ejecutará cada vez que se pulse el botón en la página. Si queremos que el texto escrito en la caja de texto de la página (cuya nombre ID es txtNombre), se muestre en la etiqueta (lblMensaje) cada vez que se pulse el botón deberíamos añadir el siguiente código:

```
protected void btnOk_Click(object sender, EventArgs e)
{
    lblMensaje.Text = txtNombre.Text;
}
```

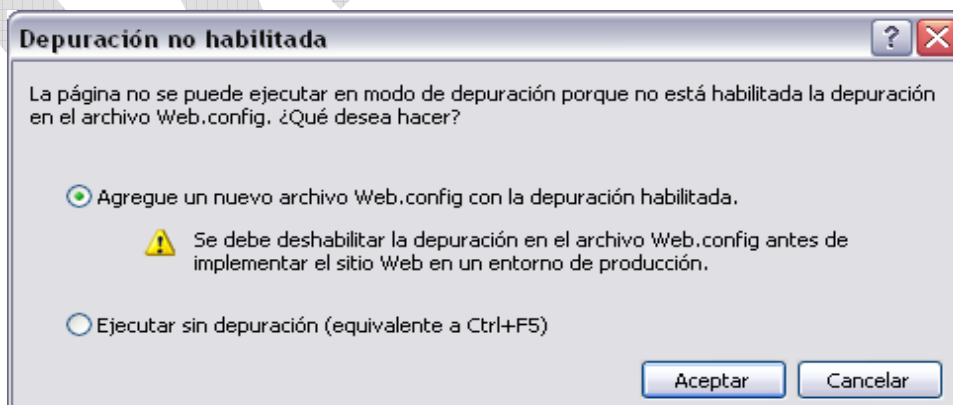
Además de los eventos de los controles Web, también existen eventos relativos a la propia página. El evento principal es Page_Load que se ejecuta cuando la página es solicitada por un navegador Web. El Visual Studio inserta el método de atención a este evento en el código de las páginas ASP por defecto.

EJECUCION DE LA PAGINA

La ejecución de la aplicación Web puede hacerse con depuración o sin depuración. La opción de depuración permite que ASP.NET muestre mensajes de error con información detallada en caso de error.

Depurar → Inicio Depuración

La primera vez que ejecutemos la aplicación Web, Visual Studio nos advertirá de la necesidad de alterar el archivo Web.Config para añadir la opción de depuración habilitada:

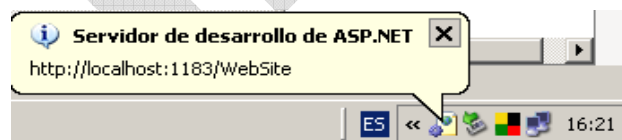


El archivo Web.Config es un fichero en formato XML que contiene parámetros de configuración de la aplicación Web:

```
<?xml version="1.0"?>
<configuration>
  <appSettings/>
  <connectionStrings/>
  <system.web>
    <!--
      Establezca debug="true" en la compilación para insertar símbolos
      de depuración en la página compilada. Dado que este
      proceso afecta al rendimiento, debe establecer este valor como true
      durante la depuración.
    -->
    <compilation debug="true"/>
    <!--
      La sección <authentication> permite configurar
      el modo de autenticación de seguridad utilizado por
      ASP.NET para identificar a un usuario entrante.
    -->
    <authentication mode="windows"/>
    <!--
      La sección <customErrors> permite configurar
      las acciones que se deben llevar a cabo/cuando un error no controlado tiene lugar
      durante la ejecución de una solicitud. Específicamente,
      permite a los desarrolladores configurar páginas de error html
      que se mostrarán en lugar de un seguimiento de pila de errores.
    -->
    <customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
      <error statusCode="403" redirect="NoAccess.htm" />
      <error statusCode="404" redirect="FileNotFound.htm" />
    </customErrors>
  </system.web>
</configuration>
```

El elemento `<compilation debug="true" />` insertado dentro de la categoría `<configuration>` activa la ejecución de la aplicación Web en modo de depuración o “debug”. Esta opción permite mostrar información detallada en el navegador cliente en caso de error. Esto es muy útil en la fase de desarrollo para depurar errores, sin embargo; debe recordarse deshabilitarlo cuando la aplicación Web vaya a ponerse en funcionamiento definitivamente

Para ejecutar la aplicación Web, el Visual Studio crea un servidor HTTP virtual y abre el navegador Web que tengamos instalado invocando la página de inicio de la aplicación Web en dicho servidor.



El servidor Web virtual del Visual Studio cuando se inicia una aplicación ASP.NET

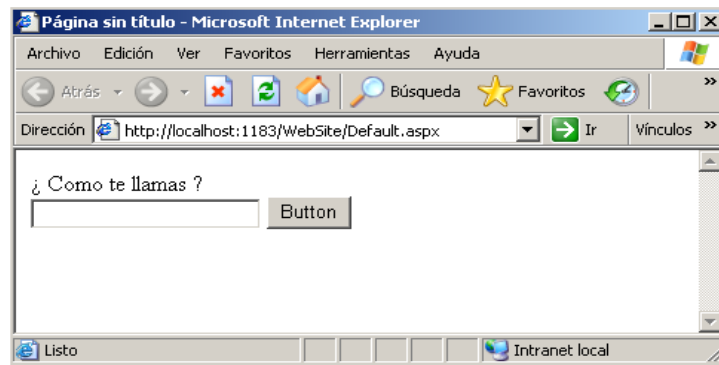
El servidor HTTP virtual se sitúa siempre en la máquina en la que se ejecuta el Visual Studio, con un número de puerto cambiante para no interferir con otros Servidores HTTP en funcionamiento. En el ejemplo el puerto es el 1183.

El navegador Web es invocado entonces por el Visual Studio para solicitar la página de inicio en la dirección:

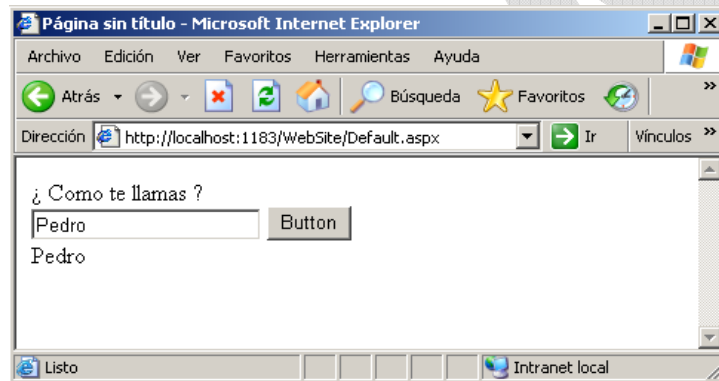
`http://localhost:1183/website/Default.aspx`

Introducción IDE Visual Studio

El resultado mostrado por el navegador web debería ser semejante al mostrado:



Si escribimos el nombre “Pedro” dentro de la caja de texto, y pulsamos el botón, la página se recarga mostrando el nombre justo debajo; tal y como indicamos en el código del evento.



Depuración de ASP.NET

La depuración es el proceso por el que se detectan y eliminan los errores presentes en una página ASP.NET. Estos errores pueden diferenciarse en tres tipos:

Errores de sintaxis:

Un error de sintaxis es un error en la estructura de una línea de código que impide que el ordenador la entienda y la pueda ejecutar. Las causas pueden ser un punto y coma olvidado al final de una línea, una llave o corchete de más o de menos, una palabra clave como if, do, else... etc; mal escrita. Los errores de sintaxis son detectados automáticamente por el Visual Studio y aparecen subrayados en color rojo.

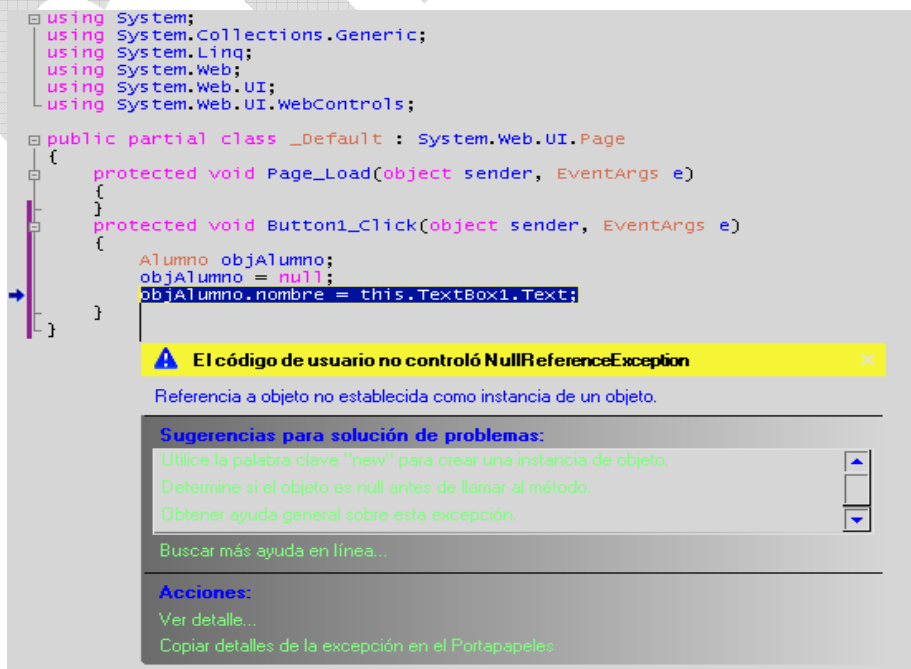
Errores de ejecución:

Un error de ejecución es aquel que se produce al solicitar una página y ejecutarse su código. Los errores de ejecución se producen cuando una línea de código produce una *excepción* al ejecutarse. La causa más habitual suele ser la invocación de un método o propiedad de una instancia no inicializada. La forma en que se presentan estos errores depende del modo en que se ejecute la aplicación Web.

Si estamos ejecutando la aplicación en el Visual Studio mediante la opción:

Depurar → Iniciar Depuración...

El error se muestra directamente en el código indicando la línea y la página donde se ha originado y un mensaje descriptivo de la excepción.



Introducción IDE Visual Studio

Si la aplicación se está ejecutando en un servidor Web real, o en el Visual Studio seleccionando la opción:

Depurar → Iniciar sin depurar...

Se muestra el mensaje de error en el mismo navegador Web:

Error de servidor en la aplicación '/WebSite1'.

Referencia a objeto no establecida como instancia de un objeto.

Descripción: Excepción no controlada al ejecutar la solicitud Web actual. Revise el seguimiento de la pila para obtener más información acerca del error y dónde se originó en el código.

Detalles de la excepción: System.NullReferenceException: Referencia a objeto no establecida como instancia de un objeto.

Error de código fuente:

```
Línea 15:         Alumno objAlumno;  
Línea 16:         objAlumno = null;  
Línea 17:         objAlumno.nombre = this.TextBox1.Text;  
Línea 18:     }  
Línea 19: }
```

Archivo de origen: c:\WebSite1\Default.aspx.cs **Línea:** 17

Seguimiento de la pila:

```
[NullReferenceException: Referencia a objeto no establecida como instancia de un objeto.]  
   Default.Button1_Click(Object sender, EventArgs e) in c:\WebSite1\Default.aspx.cs:17  
   System.Web.UI.WebControls.Button.OnClick(EventArgs e) +111  
   System.Web.UI.WebControls.Button.RaisePostBackEvent(String eventArgument) +110  
   System.Web.UI.WebControls.Button.System.Web.UI.IPostBackEventHandler.RaisePostBackEvent(String eventArgument) +10  
   System.Web.UI.Page.RaisePostBackEvent(IPostBackEventHandler sourceControl, String eventArgument) +13  
   System.Web.UI.Page.RaisePostBackEvent(NameValueCollection postData) +36  
   System.Web.UI.Page.ProcessRequestMain(Boolean includeStagesBeforeAsyncPoint, Boolean includeStagesAfterAsyncPoint) +1565
```

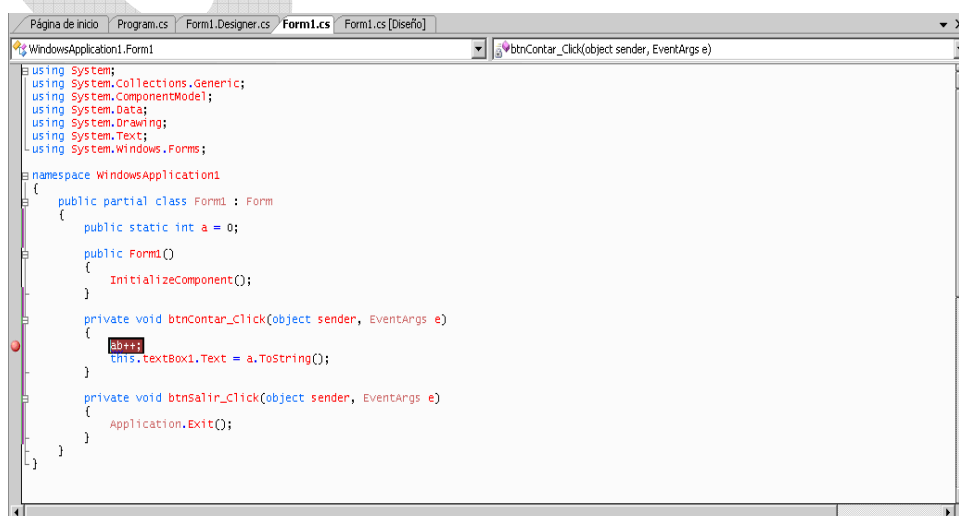
Información de versión: Versión de Microsoft .NET Framework: 2.0.50727.3053; Versión ASP.NET: 2.0.50727.3053

Errores lógicos:

Los errores lógicos se dan cuando el código se ejecuta correctamente sin producir ningún mensaje de error, pero no se comporta como es esperado.

Este tipo de errores se localizan situando **puntos de ruptura** en el código ejecutable de las páginas. Un punto de ruptura es una marca en una línea de código que hace que la ejecución se detenga al llegar a ella. Una vez detenida es posible comprobar el estado de las variables y/o continuar la ejecución paso a paso.

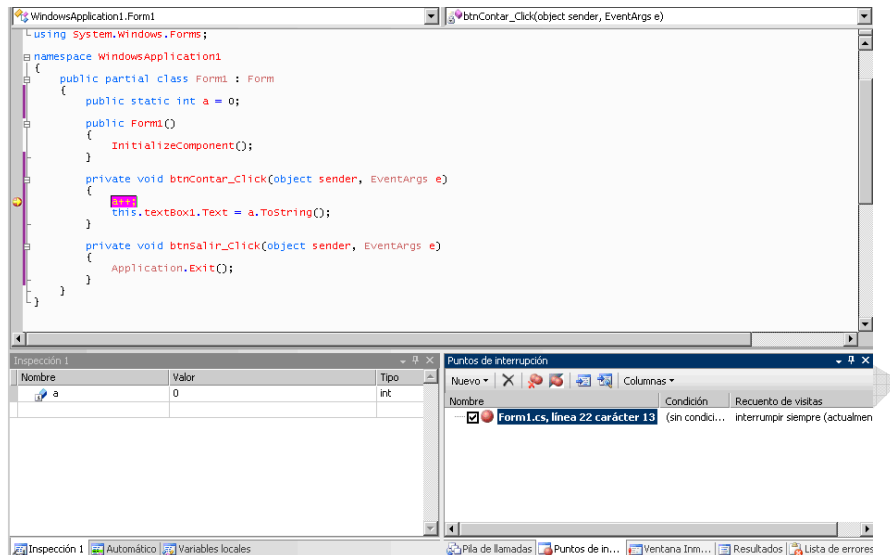
Para colocar un punto de ruptura se marcar la línea de código haciendo clic en su margen izquierdo. La línea se destaca entonces con un remarcado rojo por defecto.



Para que la ejecución se detenga al llegar a los puntos de ruptura es necesario seleccionar la opción:

Depurar → Iniciar depuración

Al llegar la ejecución a línea marcada como punto de ruptura se detendrá. La línea aparece entonces resaltada en amarillo.

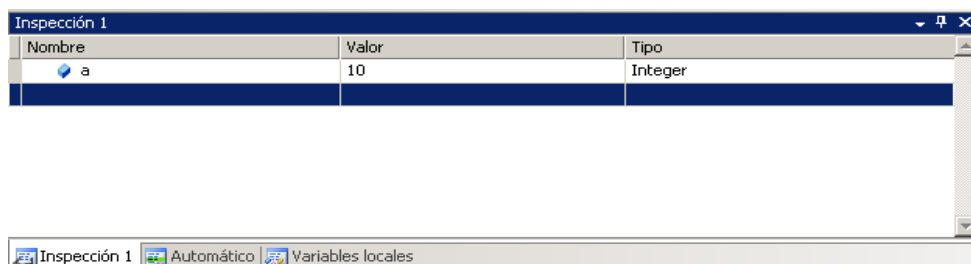


INSPECCION DE VARIABLES

La inspección permite examinar el valor de las variables de un programa en un determinado momento de la ejecución. Para ello es imprescindible disponer de un punto de ruptura que detenga la ejecución donde queramos. Una vez detenida las variables pueden inspeccionarse para conocer sus valores en ese momento.

Existen tres ventanas que informan del valor de las variables de la aplicación cuando ésta está detenida en un punto de ruptura.

- **Ventana variables locales:** Muestra el nombre, valor y tipo de las variables en el ámbito del procedimiento o función donde está detenida la ejecución.
- **Ventana Automático:** Muestra el tipo y valor de las variables globales de la aplicación.
- **Ventana Inspección :** Muestra el valor y el tipo de las variables indicadas por el usuario



TRACEAR LA EJECUCION

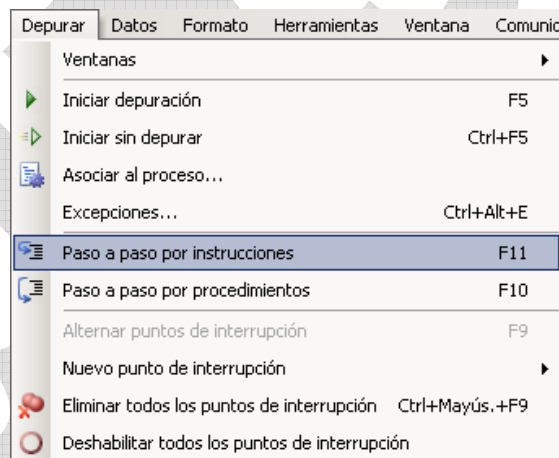
La acción de **tracear** consiste en ejecutar un programa instrucción a instrucción tras un punto de ruptura. Esto permite examinar la ejecución del código paso a paso comprobando la evaluación de condicionales, el número de veces que se ejecuta un bucle o los valores que van tomando las variables a lo largo del programa.. etc.

Para tracear el código una vez detenida la ejecución con un punto de ruptura, simplemente:

- **Depurar→Paso a paso por instrucciones (F11)**: Permite ejecutar el código instrucción a instrucción, entrando en el código de los procedimientos o funciones que sean llamados.
- **Depurar→Paso a paso por procedimientos (F10)**: Permite ejecutar el código instrucción a instrucción, pero saltando el código de los procedimientos o funciones llamados.

Una vez depurada la aplicación podemos deshabilitar todos los puntos de ruptura (no detienen la ejecución, pero se conservan para futuras depuraciones), ó eliminarlos definitivamente.

Los puntos de ruptura se almacenan junto con el código al guardar la solución, pero sólo tienen efecto cuando ésta se ejecuta desde el Visual Studio.



Publicación de una aplicación Web

La publicación consiste en colocar las páginas y resto de archivos de una aplicación Web en el directorio virtual correspondiente de un servidor Web, para que queden disponibles al público.

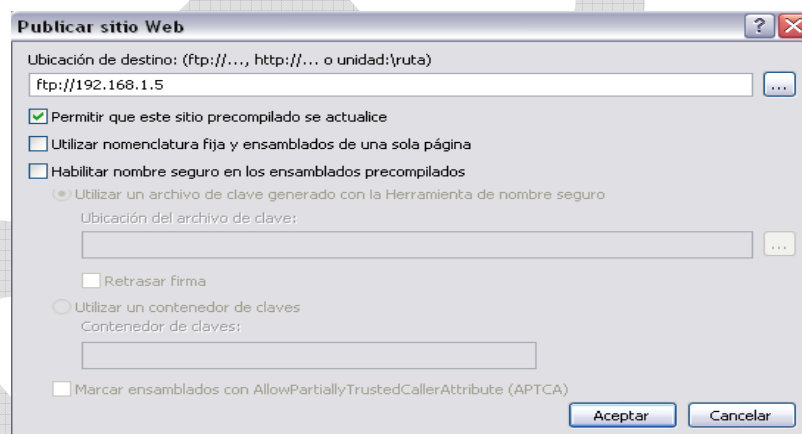
Antes de publicar la aplicación Web es importante recordar desactivar el modo de debug en el fichero de configuración *Web.Config*. Para ello basta modificar el valor del parámetro “*compilation debug*” a “*false*”.

Lo más normal es que el servidor HTTP donde debemos publicar nuestra aplicación Web esté en un equipo remoto con un servicio FTP asociado que permite “subir” los ficheros a la carpeta del sitio Web correspondiente. En tal caso necesitaremos conocer la dirección FTP, y el nombre y contraseña de una cuenta con permiso de acceso.

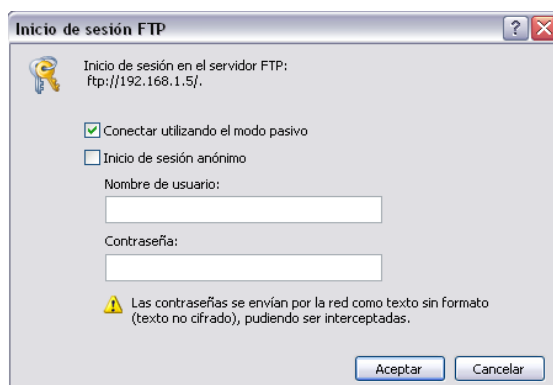
PUBLICAR SITIO WEB.

Para publicar nuestro sitio Web desde el Visual Studio 2005 existen dos métodos:

Generar → Publicar Sitio Web...



En ubicación de destino debemos introducir la dirección del servicio FTP al que deseamos conectarnos para la publicación.



Si la dirección es correcta, nos aparecerá la ventana de autenticación.

Si disponemos de una cuenta con permiso de acceso al servidor, debemos desmarcar la casilla “inicio de sesión anónimo” e introducir el nombre de usuario y contraseña correspondientes.

Introducción IDE Visual Studio

El Visual Studio compilará los ficheros de la Aplicación Web y los enviará vía FTP hasta el servidor remoto. La ventana de resultados deberá mostrar lo siguiente al terminar:

```
----- Operación Generar iniciada: proyecto: C:\...\Sitioweb\, configuración: Debug .NET -----
Precompilando el sitio web

Directorio de generación '/Sitioweb/'.
Precompilación completada
----- Publicación iniciada: proyecto: C:\...\Sitioweb\, configuración: Debug .NET -----
===== Generar: 1 correctos o actualizados, 0 incorrectos, 0 omitidos =====
===== Publicación: 1 procesados, 0 no procesados, 0 omitidos =====
```

El proceso de publicación de una aplicación Web compila la aplicación sustituyendo los ficheros de código fuente de cada una de las páginas por un ensamblado ya compilado. Esto libera al Servidor Web de compilar por sí mismo las páginas al ser solicitadas, acelerando así su ejecución. Por otro lado, tiene la ventaja de que el código fuente de cada página ya no queda almacenado en el servidor Web, sino únicamente su ensamblado.

Ej: Código de página Default.aspx sin publicar. En la etiqueta Page se hace referencia al fichero con el código ejecutable: Default.aspx.cs

```
<%@ page language="C#" autoeventwireup="true" inherits="_Default" CodeFile="Default.aspx.cs" %>

<!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">
  <link rel="stylesheet" href="css/externa.css" /> <title>Página sin título</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <asp:Panel ID="Panel1" runat="server" BackColor="#C0FFFF" BorderColor="Black"
BorderStyle="Solid"
Height="480px" width="640px">
        <asp:Label ID="Label1" runat="server" Text="¿ Como te llamas ?"></asp:Label>
        <asp:TextBox ID="txtNombre" runat="server"></asp:TextBox>
        <asp:Button ID="btnOk" runat="server" onClick="btnOk_Click" Text="Button" />
        <br />
        <asp:Label ID="lblNombre" runat="server" Font-Bold="True" Font-Names="Lucida Console"
Font-Size="XX-Large"></asp:Label></asp:Panel>
        &nbsp;</div>
    </form>
  </body>
</html>
```

Ej: Código de la página Default.aspx ya publicada. El fichero de código ha sido precompilado y sustituido por el ensamblado: "App_web_s0ek_emp.dll" almacenado en la carpeta "bin".

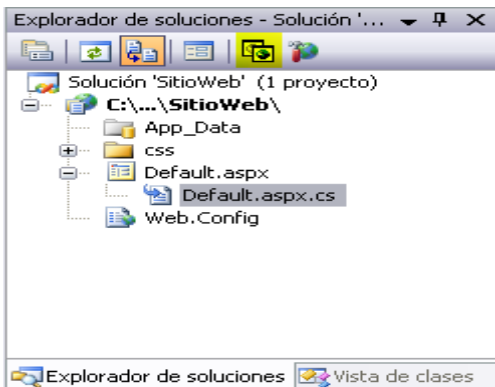
```
<%@ page language="C#" autoeventwireup="true" inherits="_Default, App_web_s0ek_mp1" %>

<!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">
  <link rel="stylesheet" href="css/externa.css" /> <title>Página sin título</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <asp:Panel ID="Panel1" runat="server" BackColor="#C0FFFF" BorderColor="Black"
BorderStyle="Solid"
Height="480px" width="640px">
        <asp:Label ID="Label1" runat="server" Text="¿ Como te llamas ?"></asp:Label>
        <asp:TextBox ID="txtNombre" runat="server"></asp:TextBox>
        <asp:Button ID="btnOk" runat="server" onClick="btnOk_Click" Text="Button" />
        <br />
        <asp:Label ID="lblNombre" runat="server" Font-Bold="True" Font-Names="Lucida Console"
Font-Size="XX-Large"></asp:Label></asp:Panel>
        &nbsp;</div>
    </form>
  </body>
</html>
```

COPIADO DE APLICACIÓN WEB.

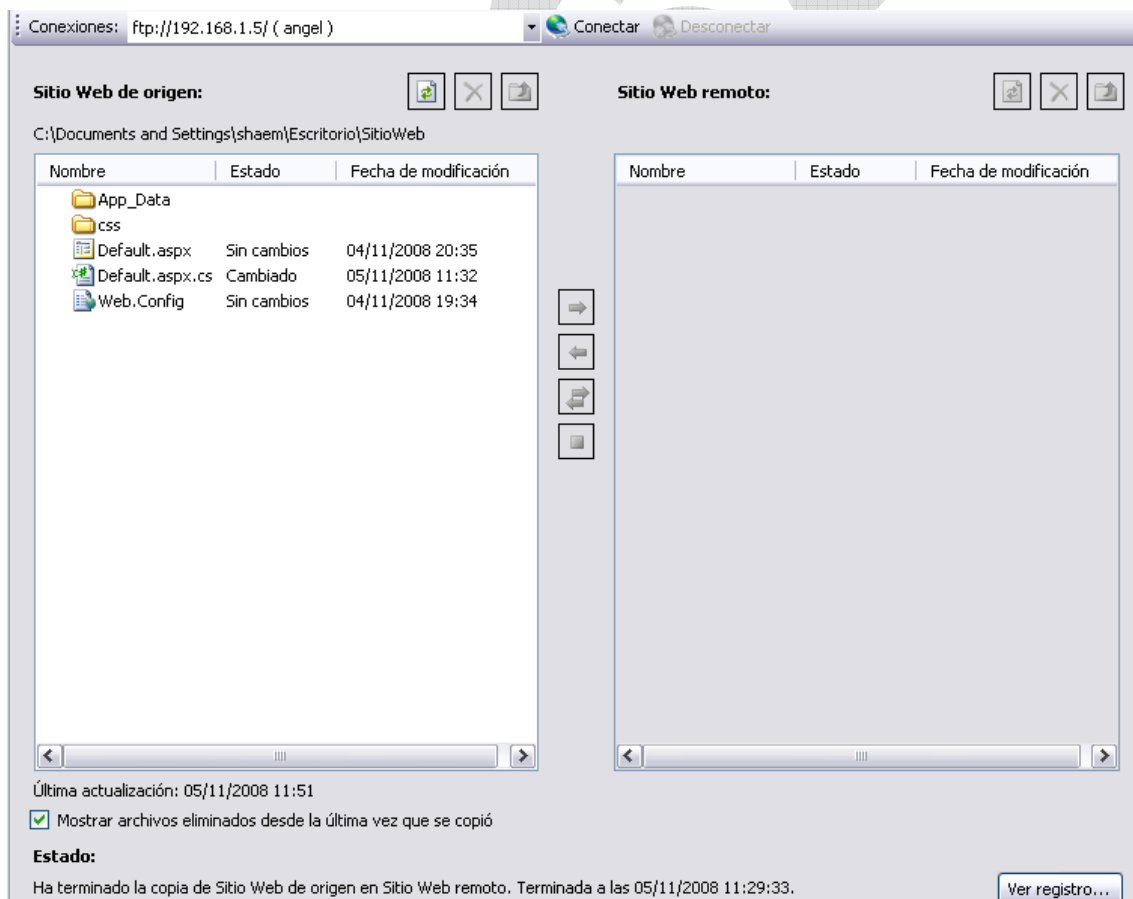
La opción de “Copiar Sitio Web” es similar a la publicación, pero nos permite seleccionar qué archivos en concreto deseamos enviar al servidor.



Si seleccionamos dentro de la ventana del Explorador de Soluciones el nodo de la aplicación Web, se muestran una barra de herramientas con 6 botones.

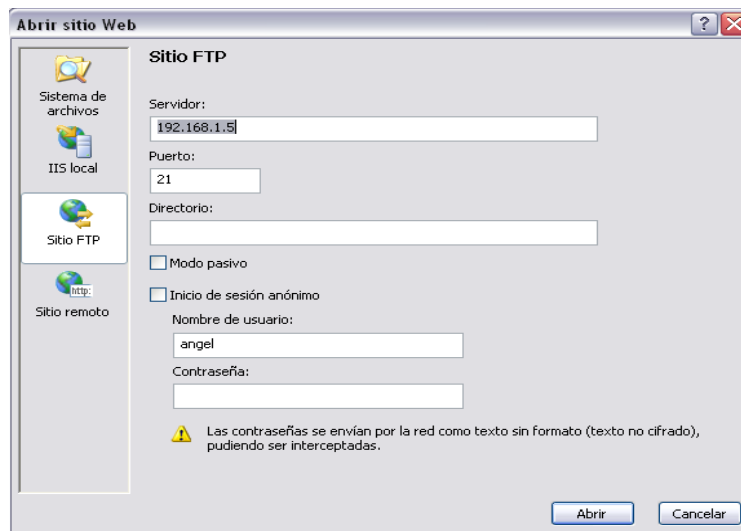
El segundo icono por la derecha en la barra de herramientas del Explorador de Soluciones permite acceder a la ventana “Copiar Web”

La ventana se divide en dos espacios. El espacio de la izquierda representa la estructura de ficheros de nuestra aplicación Web. El espacio de la derecha muestra el contenido del sitio Web remoto una vez nos conectamos.



Introducción IDE Visual Studio

Para establecer conexión con el sitio remoto basta pulsar el botón “Conectar” y se nos mostrará seguidamente la ventana “Abrir sitio Web”

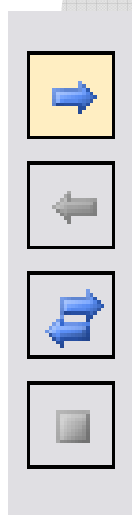


En esta ventana podemos seleccionar el tipo de conexión que vamos a establecer con el servidor remoto. En el caso de emplear el protocolo FTP, debemos indicar la dirección, el puerto de conexión (por defecto el 21), y la carpeta dentro del servidor FTP a la que deseamos conectar si es necesario.

El modo pasivo debe emplearse si nuestro equipo está detrás de un Proxy o empleamos cortafuegos que impidan conexiones entrantes.

Si poseemos una cuenta con permiso de acceso al servicio FTP debemos dejar desmarcada la casilla de verificación “Inicio de sesión anónimo” e introducir nuestro nombre de usuario y contraseña.

Una vez establecida la conexión podemos seleccionar los archivos que deseamos transferir al servidor remoto seleccionando los archivos de origen y empleando los botones de transferencia:



Las funciones de cada uno de los botones son las siguientes en orden:

- Copia un archivo del equipo local al remoto
- Copia un archivo del equipo remoto al local
- Sincroniza un archivo presente en ambos equipos. La sincronización consiste en actualizar el del equipo remoto si ha sido modificado desde la última vez que se transfirió
- Detener transferencia en curso.

Copiar no equivale a Publicar: En el caso de la copia; los archivos de la aplicación Web son copiados sin compilar; por lo que su código fuente queda expuesto en el servidor Web.