

**UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL GENERAL PACHECO**

**TÉCNICO SUPERIOR EN PROGRAMACIÓN
PROGRAMACIÓN III**

UNIDAD 1:

Introducción al entorno y sus controles

TSSI TAMARA HERRERA

UNIDAD 1 -

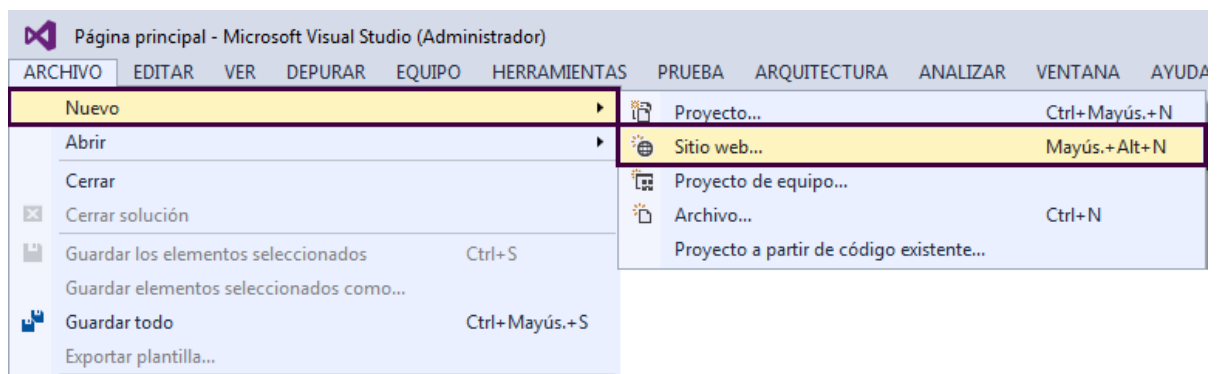
1. ¿Cómo crear un sitio web?
2. ¿Cómo incorporar formularios Web Forms a un sitio web?
3. Aplicación 1: Introducción a controles
4. Aplicación 2: Cargar código HTML a un Label
5. Aplicación 3:PostBack
 - DropDownList
 - CheckBoxList
6. Aplicación 4: AutoPostBack
7. Aplicación 5: Response - Request
8. Aplicación 6: Server.Transfer

Introducción

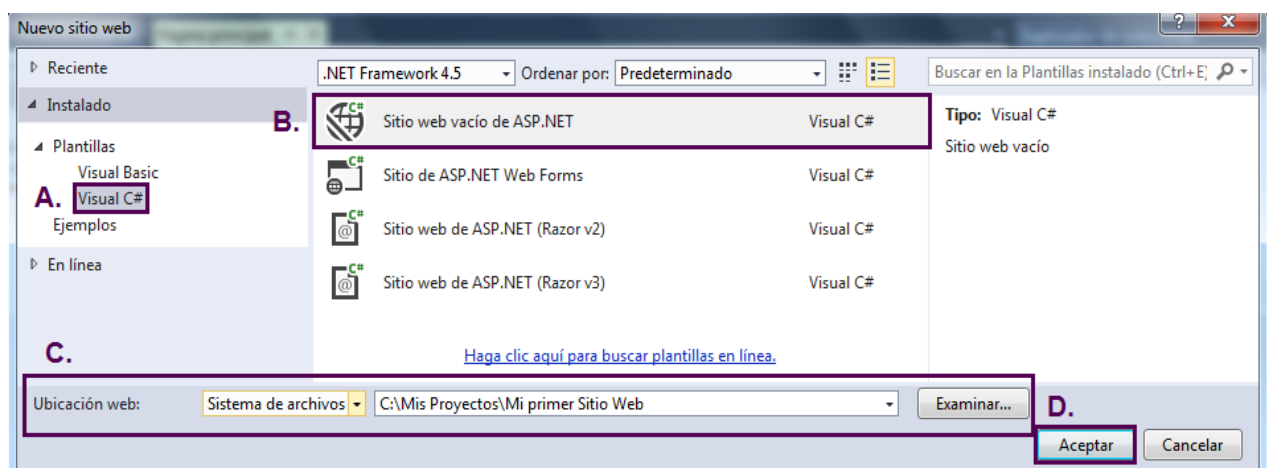
Nuestro objetivo en esta unidad es aprender a realizar aplicaciones web, comprender su estructura, utilizar controles básicos y conocer las diferentes formas mediante las cuales podemos traspasar información a través de diferentes formularios. Así que empecemos, primero vamos a conocer como se crea un sitio web.

¿Cómo crear un sitio web?

1. Para crear un sitio web debemos ir a: **Archivo -> Nuevo -> Sitio Web**

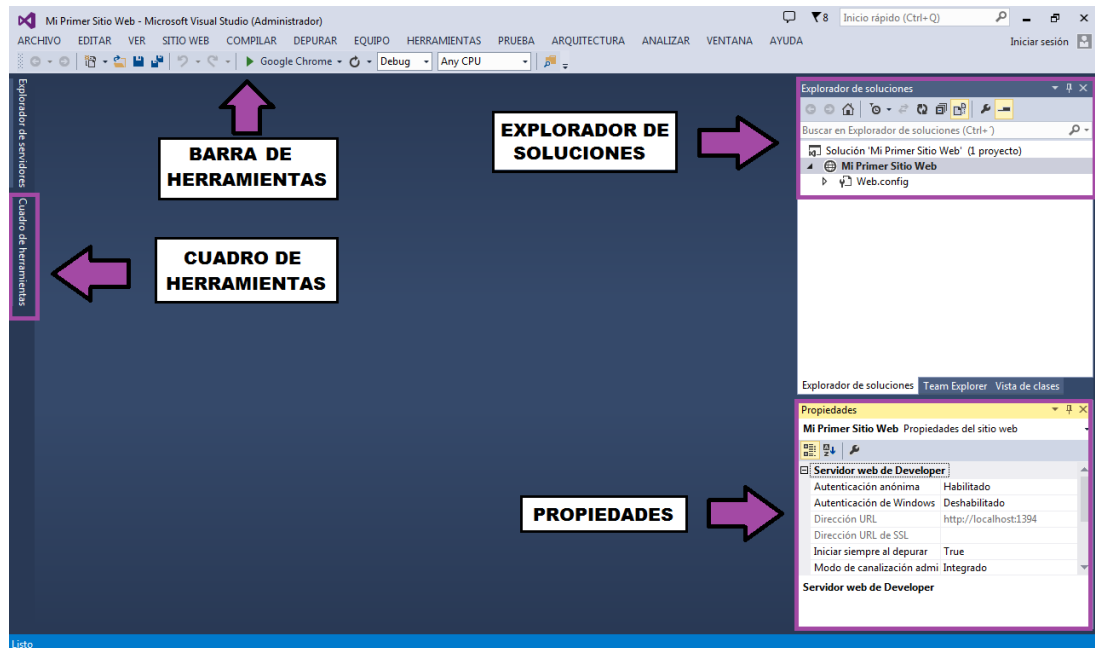


2. Nos aparecerá el siguiente cuadro:

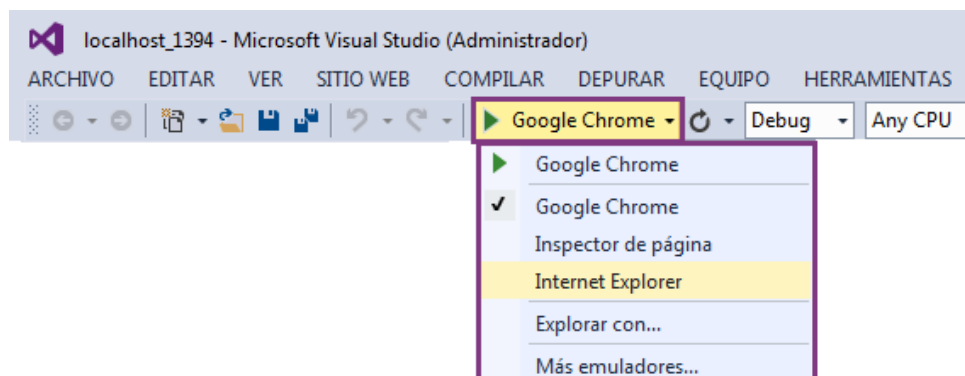


- A. Elegiremos el idioma con el cual queremos trabajar. En este caso **Visual C#**
- B. Seleccionaremos la opción: **“Sitio web vacío de ASP.NET”**.

- C. Deberemos elegir en que directorio guardaremos nuestra página web. En este caso, estoy generando una carpeta llamada Mi primer Sitio Web, que va a contener los archivos de nuestra página, y va a estar ubicada en el siguiente directorio: C:\Mis Proyectos
- D. Clic en Aceptar.
3. Luego de crear el sitio se podrá visualizar lo siguiente.



- ✓ **Barra de herramientas:** Es el menú que se encuentra en la parte superior de nuestro entorno. Desde allí podremos determinar qué elementos queremos visualizar, cambiar configuraciones y demás. Algo muy importante, aquí se encuentra la opción para ejecutar nuestra página web y verla desde el navegador.





Presionando este botón, se compilará y cargará el formulario en el navegador. Podemos elegir: Mozilla, Internet Explorer, Google Chrome, etc.

- ✓ **Explorador de soluciones:** Proporciona una vista organizada del proyecto y sus archivos. Permite acceder de forma rápida a cualquier elemento de nuestro proyecto.

Para acceder en el caso de que no esté visible:

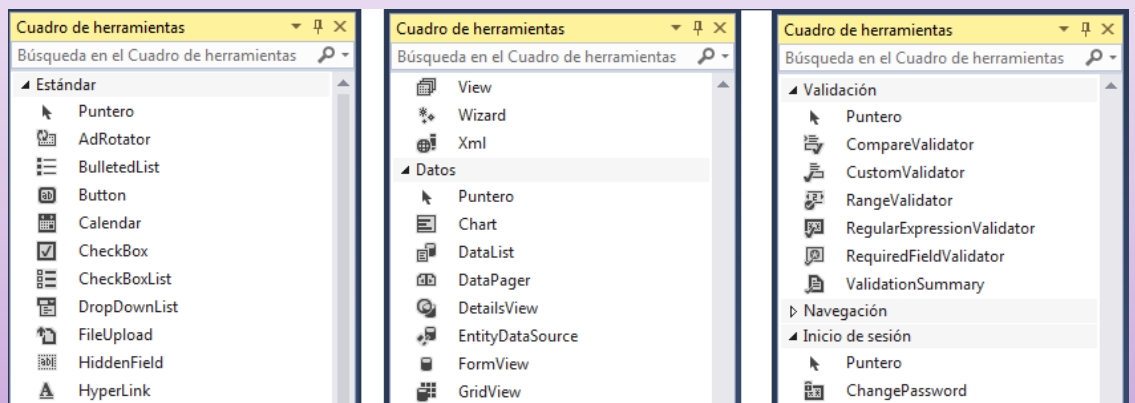
Barra de herramientas -> Ver -> Explorador de soluciones

- ✓ **Cuadro de herramientas:** Aquí tenemos todos los controles que podemos agregar a nuestros formularios Web Forms. Solo podremos ver los controles que posee el cuadro de herramientas si estamos situados sobre un formulario.

Para acceder en el caso de que no esté visible:

Barra de herramientas -> Ver -> Cuadro de herramientas

Imagen de algunos controles que se encuentran en el cuadro de herramientas:



- ✓ **Propiedades:** Visualizaremos las diferentes propiedades de los elementos seleccionados ya sea formularios o controles.

Para acceder en el caso de que no esté visible:

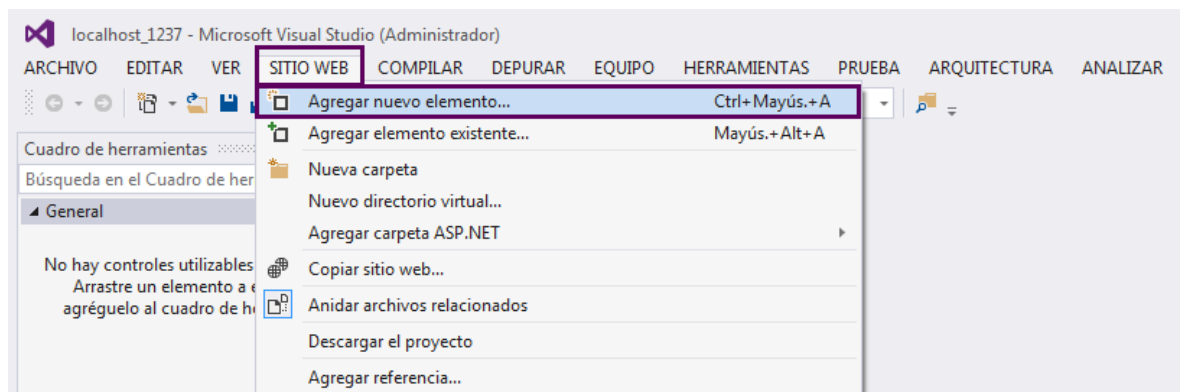
Segundo clic sobre un elemento -> elegir la opción ver propiedades

¿Cómo crear formularios Web Forms?

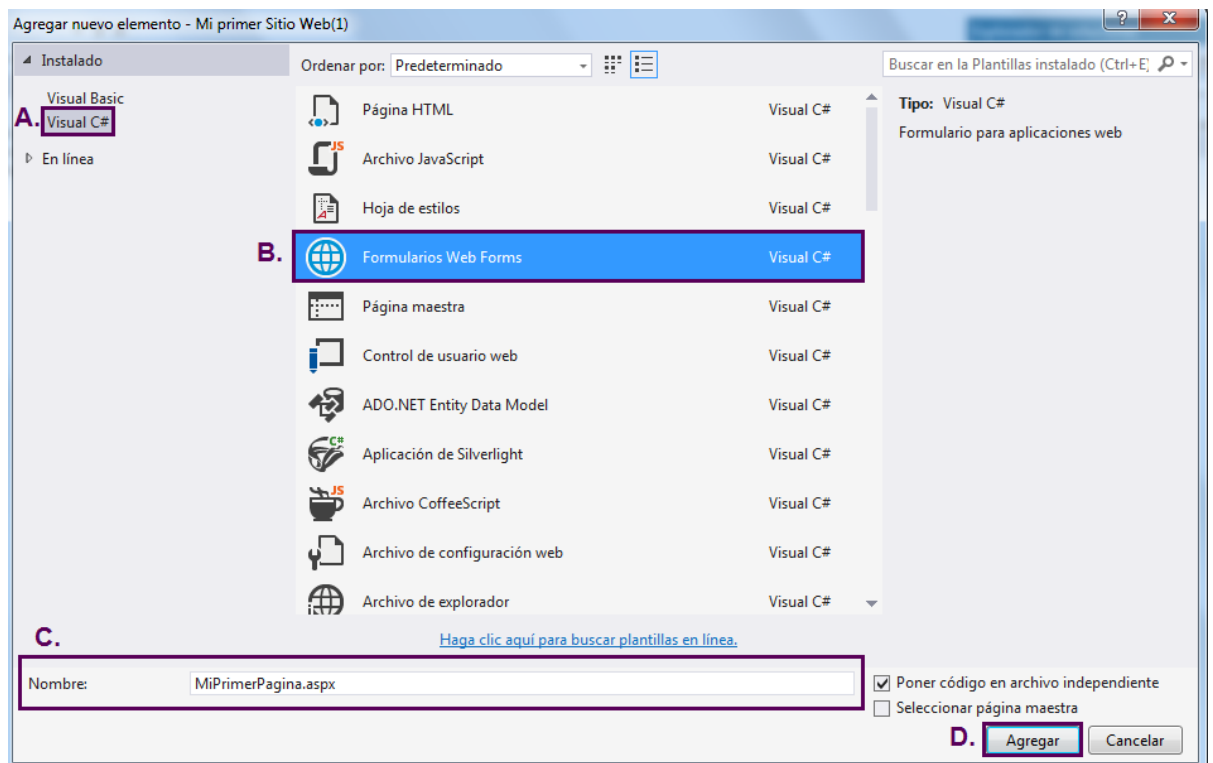
Los formularios Web Forms son plantillas en blanco donde podremos incorporar controles. Estos formularios son los que visualizaremos desde el explorador. Ahora, vamos a incorporar un formulario al sitio web que recién creamos.

1) Sobre el sitio web ya creado.

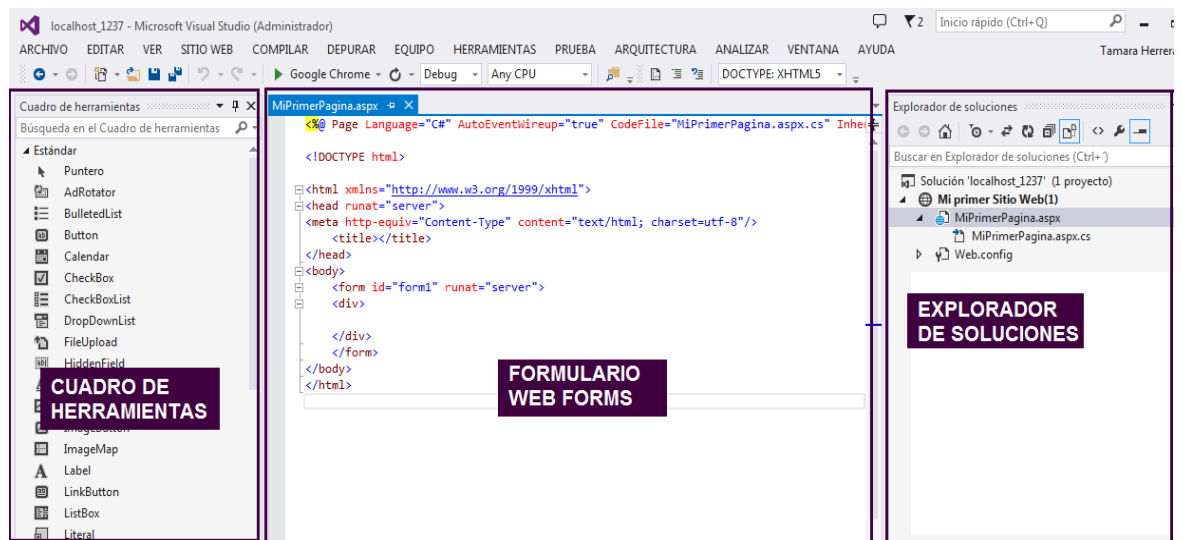
Ir a la **Barra de herramientas** -> **Sitio Web** -> **Agregar nuevo elemento**



2) Luego aparecerá el siguiente cuadro:



- A. Seleccionar idioma: **Visual C#**
 - B. Seleccionar: **Formularios Web Forms**.
 - C. Elegir un nombre, en este caso: **MiPrimerPagina.aspx**
 - D. Clic en **Agregar**
- 3) Una vez que agreguemos el formulario, lo podemos ver disponible en el explorador de soluciones.



- Si no está visible el formulario Web Forms, lo que debe hacer es ir al *Explorador de soluciones* -> *Doble clic sobre el formulario*.
- Recuerde que para ver el cuadro de herramientas, en el caso de que no esté visible debe ir a: *Barra de herramientas* → *Ver* → *Cuadro de herramientas*.
- En cuanto al cuadro de herramientas, solo va a poder observar los controles que este contiene, si está situado sobre la vista diseño o código de diseño del formulario, de lo contrario estará vacío.

En el explorador de soluciones puede ver el formulario Web Forms, con dos tipos de extensiones:

- **MiPrimerPagina.aspx** -> Si hace clic sobre este, va a poder trabajar con el diseño de la página Web.
- **MiPrimerPagina.aspx.cs** -> Si hace clic sobre este, va a poder trabajar con el código Visual C#.

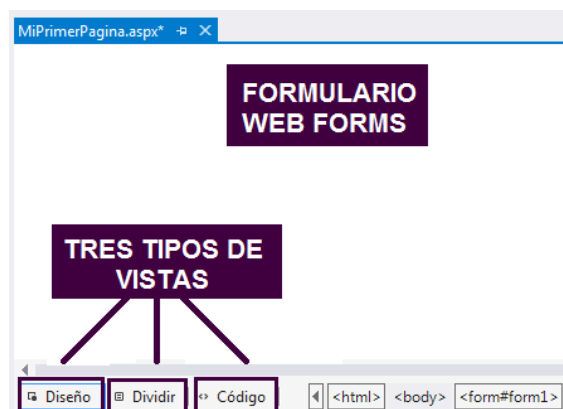
Elementos que componen un formulario Web Forms

Un formulario Web Forms está compuesto por tres partes:

- La parte gráfica
- El código de la parte gráfica
- El código C# que es desembocado por un control dibujado en la parte gráfica.

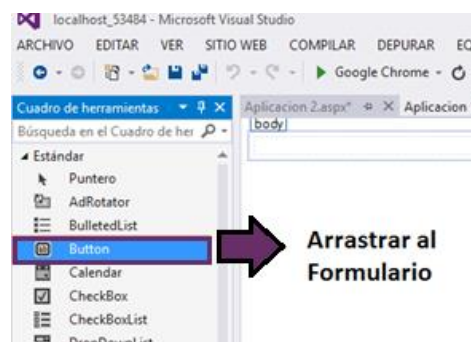
Dentro de lo que es el formulario, vamos a tener *tres tipos de vistas* distintas:

- *La vista diseño:* podemos ver de forma gráfica los controles que incorporemos
- *La vista código:* podemos ver el código HTML o ASP.NET de los controles incorporados al formulario.
- *La vista dividir:* se puede ver de manera simultánea, el dibujo del control y su código de creación.



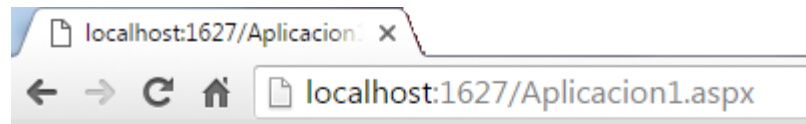
¿Cómo incorporar controles a nuestro formulario?

Sencillo, para incorporar controles solo deberemos arrastrarlos del cuadro de herramientas.



Aplicación 1: Button, TextBox y Labels

El usuario podrá colocar su nombre y al dar clic en aceptar, se le dará un mensaje de bienvenida.



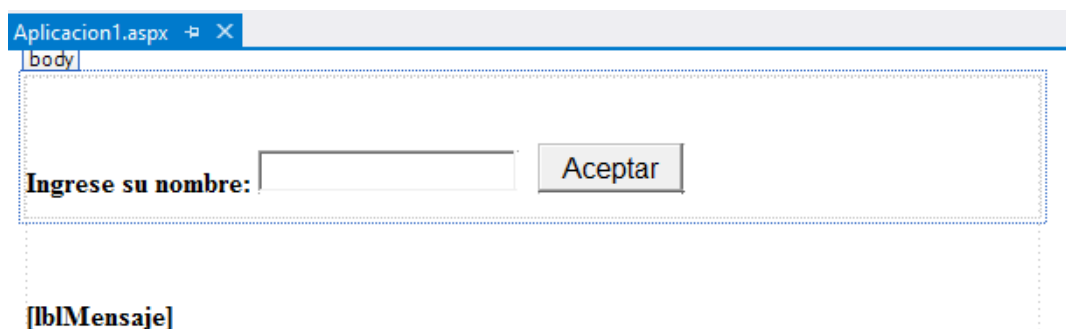
Ingrese su nombre:

Bienvenido: Pepe

Pasos a seguir:

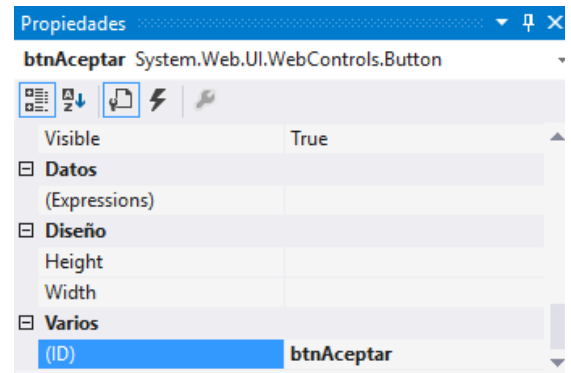
1. Primero vamos a crear un sitio web en la carpeta: **C:\ Unidad 1**
2. Luego incorporamos a ese proyecto un formulario: **Aplicación1.aspx**
3. Ahora le agregamos en la **vista diseño** de ese formulario los siguientes elementos: las palabras “Ingrese su nombre:”, un TextBox, un Button y un Label.

Imagen de la vista diseño



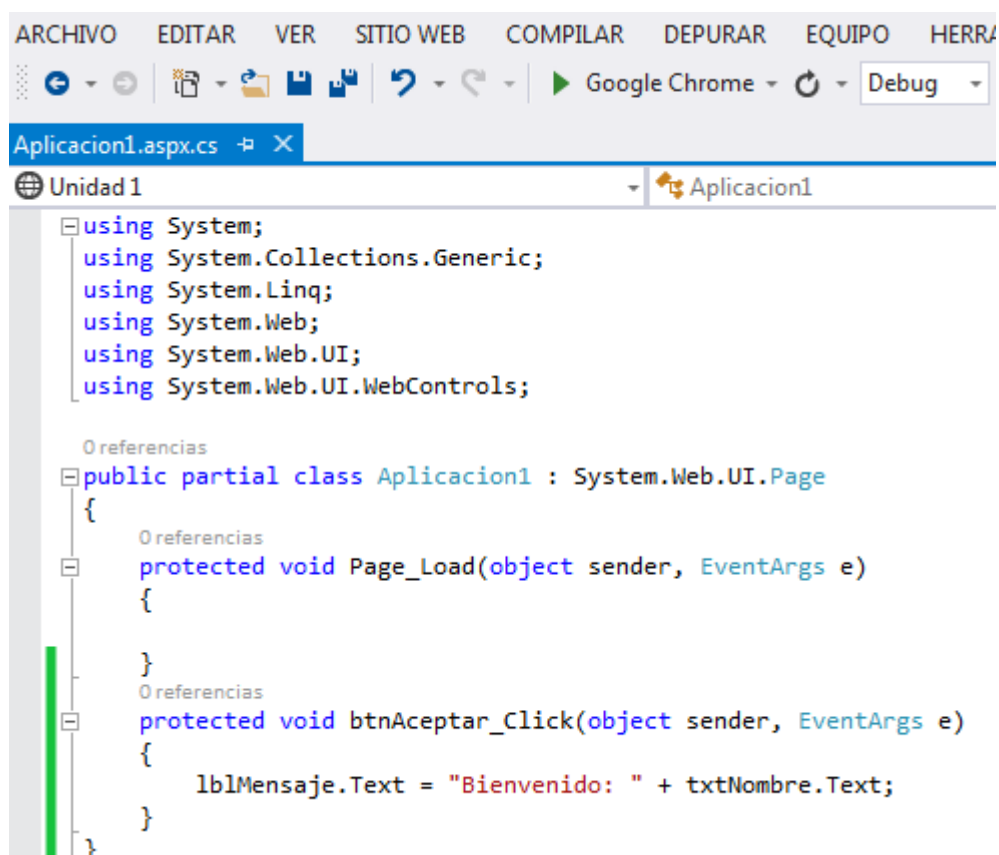
4. Cambiar las propiedades de los controles incorporados. Debemos realizar segundo clic sobre el control – Propiedades.

Control	Propiedades
TextBox	ID: txtNombre
Button	ID: btnAceptar
Label	ID: lblMensaje Text: dejar en blanco



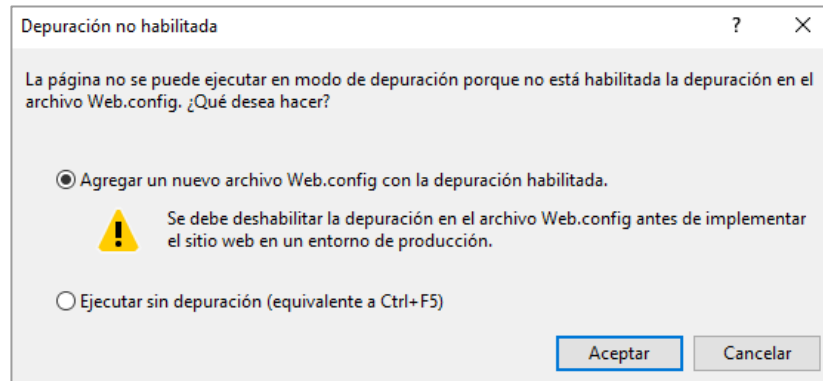
5. Realizar doble clic sobre el botón. del evento btnAceptar_Click agregar la siguiente línea:

```
lblMensaje.Text = "Bienvenido: " + txtNombre.Text;
```



Nota: Fijarse que la extensión .CS es destinada a que programemos código Visual C#.

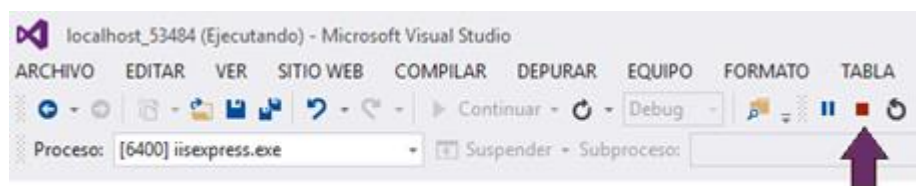
6. Compilar, presionando el botón  de la barra de herramientas. Si aparece el siguiente mensaje, clic en aceptar.



Nota: El archivo *Web.config* va a contener las configuraciones de nuestro proyecto. Contiene información que controla la carga de módulos, configuraciones de seguridad, configuraciones del estado de la sesión, opciones de compilación y el lenguaje de la aplicación.

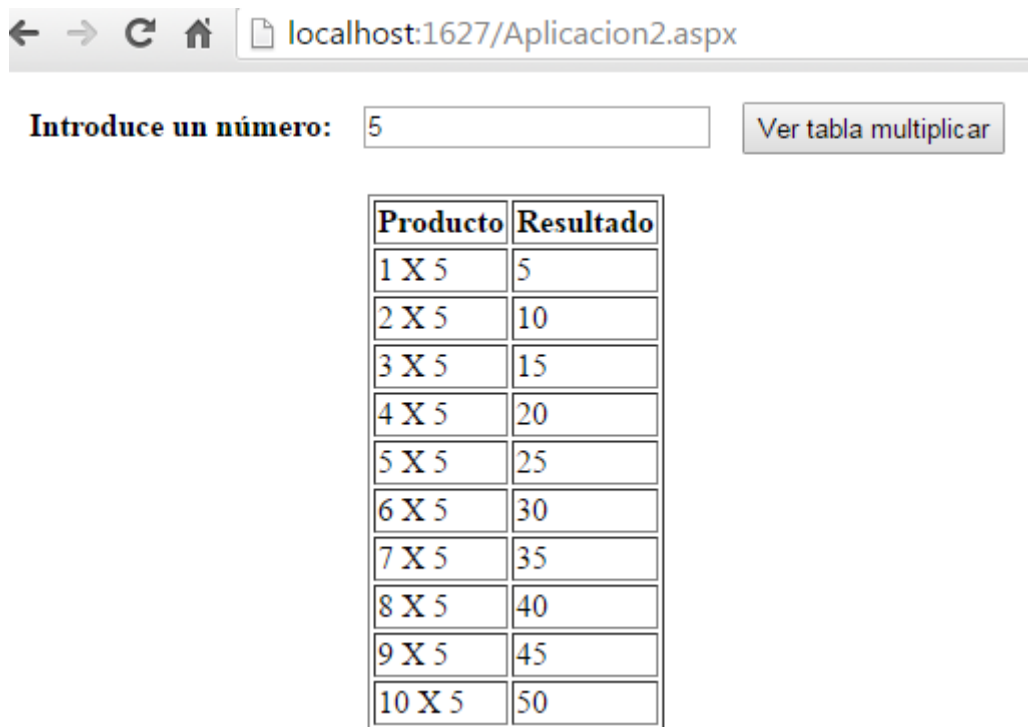
7. Probar la aplicación.

8. Por último, debemos detener la depuración.



Aplicación 2: Cargar código HTML a un Label

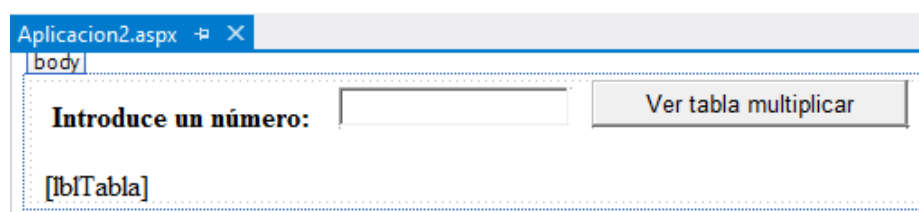
El usuario podrá colocar un número sobre el TextBox y al dar clic en Ver Tabla Multiplicar, se le mostrará la tabla de multiplicar correspondiente al número escrito.



Producto	Resultado
1 X 5	5
2 X 5	10
3 X 5	15
4 X 5	20
5 X 5	25
6 X 5	30
7 X 5	35
8 X 5	40
9 X 5	45
10 X 5	50

Pasos a seguir:

1. Crear otro formulario llamado: **Aplicación2.aspx**
2. La vista diseño tiene que quedar de la siguiente manera:



3. Incorporar las palabras: “Introduce un número” y los siguientes controles:

Control	Propiedades
TextBox	ID: txtNumero
Button	ID: btnMultiplicar
Label	ID: lblTabla Text: dejar en blanco

4. En el siguiente paso, vamos a crear una tabla y cargarla en un Label, pero para eso primero tenemos que saber ¿Cómo se crea una tabla en HTML?

Una tabla se crea con la siguiente etiqueta HTML

```
<table>
    <tr>
        <td></td>
        <td></td>
    </tr>
</table>
```

Etiquetas de apertura

Table: Comienzo de código de Tabla

Tr: Filas.

Td: Columnas.

Th: Columnas encabezado.

Etiquetas de cierre

Finalizan con /

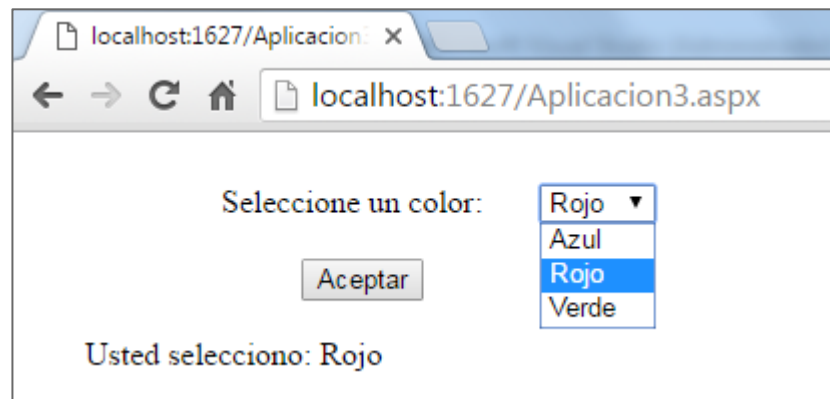
5. Hacer doble clic sobre el Button, y en el evento btnMultiplicar_Click escribir:

```
protected void btnMultiplicar_Click(object sender, EventArgs e)
{
    int numero = int.Parse(txtNumero.Text);
    String tabla = "<table border='1'>";
    tabla += "<tr><th>Producto</th><th>Resultado</th></tr>";
    for (int i = 1; i <= 10; i++)
    {
        tabla += "<tr>";
        tabla += "<td>" + i + " X " + numero + "</td>";
        tabla += "<td>" + i * numero + "</td>";
        tabla += "</tr>";
    }
    tabla += "</table>";
    lblTabla.Text = tabla;
}
```

Nota: Fijarse que estoy cargando sobre un Label, un string compuesto con código HTML. Para trabajar con números es necesario que se los convierta primero, puede utilizarse `int.Parse()` o `Convert.ToInt32()`

Aplicación 3: El postBack

El usuario seleccionará un color de la lista desplegable y al dar clic en el botón aceptar se le mostrará cual fue el color seleccionado.



Pasos a seguir:

1. Crear un formulario llamado: **Aplicación3.aspx**
2. Agregar al diseño las palabras “Seleccione un color” y “Usted seleccionó” y los siguientes controles:

Control	Propiedades
DropDownList	ID: ddlColores
Button	ID: btnAceptar
Label	ID: lblMensaje Text: dejar en blanco

El Label se encuentra al lado de las palabras “Usted seleccionó”

3. Hacer doble clic sobre el botón. Luego escribir el código que aparece a continuación en el evento btnAceptar_Click y en el evento Page_Load. Lo que estoy haciendo en el Page_Load es agregar ítems al DropDownList.

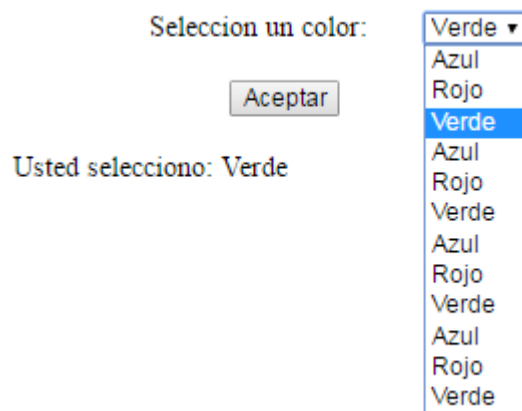
```

0 referencias
public partial class Aplicacion3 : System.Web.UI.Page
{
    0 referencias
    protected void Page_Load(object sender, EventArgs e)
    {
        ddlColores.Items.Add("Azul");
        ddlColores.Items.Add("Rojo");
        ddlColores.Items.Add("Verde");
    }
    0 referencias
    protected void btnAceptar_Click(object sender, EventArgs e)
    {
        lblMensaje.Text = ddlColores.SelectedItem.ToString();
    }
}

```

¿Qué pasa si damos clic en el botón aceptar varias veces?

Vamos a ver que cada vez que realicemos un clic sobre el botón, se volverán a cargar los ítems del DropDownList. Esto sucede porque cada vez que se produce un evento, se recarga la página y se vuelve a llamar al evento Page_Load que agregaba los ítems.



El funcionamiento es el siguiente: una máquina cliente produce un evento que llama al servidor, el servidor analiza el pedido y devuelve una nueva página web, generando lo que se conoce como **postback**. Por cada postback que se produce, se vuelve a generar una nueva página web a partir de la última creada.

¿Cómo hacemos para que algo que agreguemos al Page Load solo se ejecute una vez?

Debemos preguntarle a la variable **IsPostBack**. Si es falsa, es la primera vez que se carga la página, si es verdadera es de la segunda vez en adelante.

```
protected void Page_Load(object sender, EventArgs e)
{
    if(IsPostBack==false)
    {
        ddlColores.Items.Add("Azul");
        ddlColores.Items.Add("Rojo");
        ddlColores.Items.Add("Verde");
    }
}
```

Acerca del DropDownList

En el ejemplo anterior para agregar un ítem, hacíamos lo siguiente.

```
ddlColores.Items.Add("Azul");
ddlColores.Items.Add("Rojo");
ddlColores.Items.Add("Verde");
```

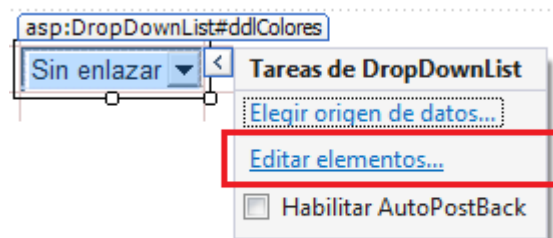
Lo negativo de esta primera forma, es que no podemos colocar un valúe a cada elemento incorporado. Otra Forma que podemos utilizar para agregar un ítem es la siguiente:

```
ListItem itemAzul = new ListItem();
itemAzul.Text = "Azul";
itemAzul.Value = "1";
ListItem itemRojo = new ListItem();
itemRojo.Text = "Rojo";
itemRojo.Value = "2";
ListItem itemVerde = new ListItem();
itemVerde.Text = "Verde";
itemVerde.Value = "3";
ddlColores.Items.Add(itemAzul);
ddlColores.Items.Add(itemRojo);
ddlColores.Items.Add(itemVerde);
```

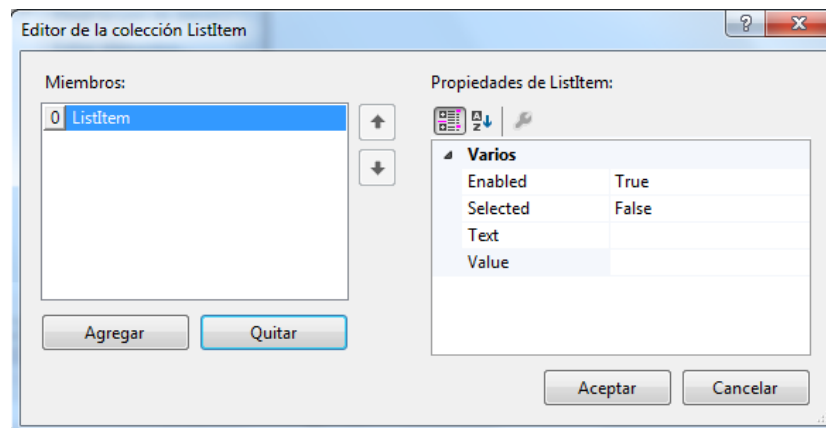

Y si no queremos crear un ListItem por cada ítem, podemos utilizar una colección de ListItem:

```
ListItemCollection coleccion = new ListItemCollection();
coleccion.Add(new ListItem("Azul", "1"));
coleccion.Add(new ListItem("Rojo", "2"));
coleccion.Add(new ListItem("Verde", "3"));
ddlColores.DataSource = coleccion;
ddlColores.DataBind();
```

Por último, también tenemos otra forma de incorporar elementos pero a través del diseño.



Se abrirá el siguiente cuadro en donde debo agregar los ListItem que necesite.



¿Cómo sabemos el Text del elemento del DropDownList que selecciono el usuario?

```
ddlColores.SelectedItem.ToString();
```

¿Cómo saber cuál es el value de ese elemento?

```
lblMensaje.Text = ddlColores.SelectedValue;
```

Acerca del CheckBoxList

¿Cómo incorporar elementos al CheckBoxList?

Para incorporar elementos, lo podemos hacer igual que el DropDownList. En el evento Page_Load haríamos lo siguiente:

```
if(!IsPostBack)
{
    chkFrutas.Items.Add("Manzanas");
    chkFrutas.Items.Add("Peras");
    chkFrutas.Items.Add("Bananas");
}
```

¿Cómo recorrer los elementos y saber cuál está seleccionado?

Cada elemento que compone el CheckBoxList, es un ListItem. Por este motivo podemos recorrerlo con un foreach, consultando si cada elemento está seleccionado.

```
string aux=" ";
foreach(ListItem s in chkFrutas.Items)
{
    if(s.Selected)
    {
        //ESTA SELECCIONADO
        aux += " " + s;
    }
}
Label1.Text = aux;
```

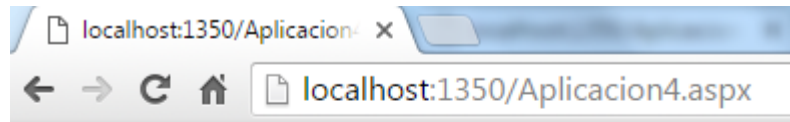
¿Puedo recorrerlo con un for?

Si! Podemos recorrerlo con un for, consultando si cada uno de sus elementos está seleccionado.

```
string aux = "";
for (int i = 0; i < chkFrutas.Items.Count; i++)
{
    if (chkFrutas.Items[i].Selected)
    {
        aux += " " + chkFrutas.Items[i].Text;
    }
}
Label1.Text = aux;
```

Aplicación 4: El AutoPostBack

El usuario seleccionará un color de la lista desplegable y automáticamente al seleccionar el color se cargará en el Label. A diferencia del ejercicio anterior, no tendremos el botón aceptar.



Seleccione un color: Verde ▼

Usted seleccionó: Verde

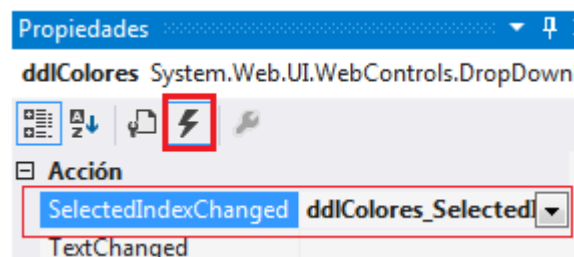
Pasos a seguir:

1. Crear un formulario llamado: **Aplicación4.aspx**
2. Agregar a la vista diseño del formulario los siguientes controles:

Control	Propiedades
DropDownList	ID: ddlColores
Label	ID: lblMensaje Text: dejar en blanco

Además agregarle las palabras “Seleccione un color” y “Usted seleccionó”

3. Dentro de las propiedades del DropDownList, buscar el evento SelectedIndexChanged y realizar doble clic.



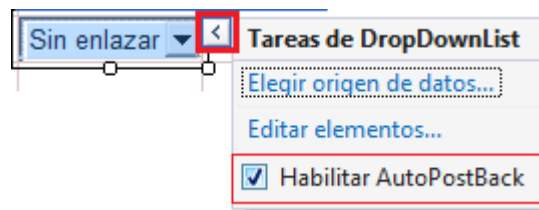
4. Sobre el evento SelectedIndexChanged escribir lo siguiente:

```
protected void ddlColores_SelectedIndexChanged(object sender, EventArgs e)
{
    lblMensaje.Text = ddlColores.SelectedItem.ToString();
}
```

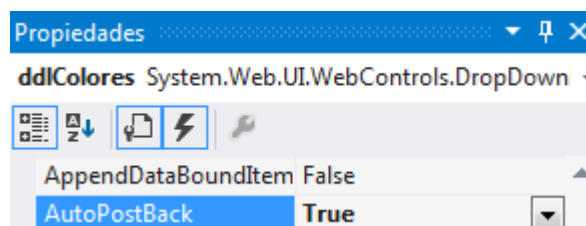
5. Sobre el evento Page_Load escribir lo siguiente:

```
protected void Page_Load(object sender, EventArgs e)
{
    if(IsPostBack==false)
    {
        ddlColores.Items.Add("Azul");
        ddlColores.Items.Add("Rojo");
        ddlColores.Items.Add("Verde");
    }
}
```

6. Lo último que tenemos que hacer para que nuestro programa funcione es habilitar el AutoPostBack en el DropDownList



También podemos habilitarlo desde las propiedades del control, fijarse que se puede habilitar la propiedad AutoPostBack en true o false.

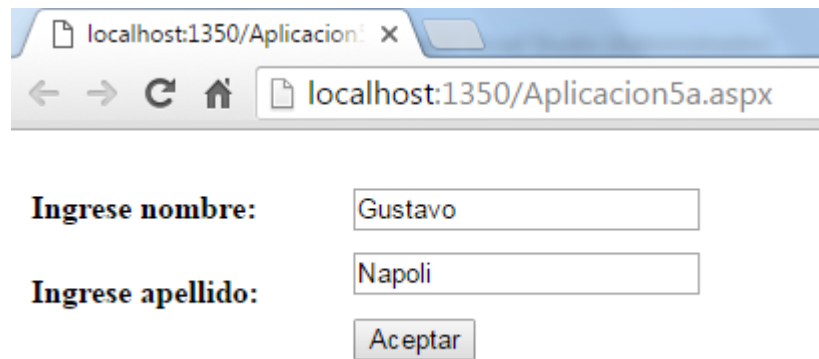


Nota: El AutoPostBack, permite que el control al cual se lo habilitemos genere un PostBack al modificarse el contenido del mismo.

7. Solo nos queda probar la aplicación.

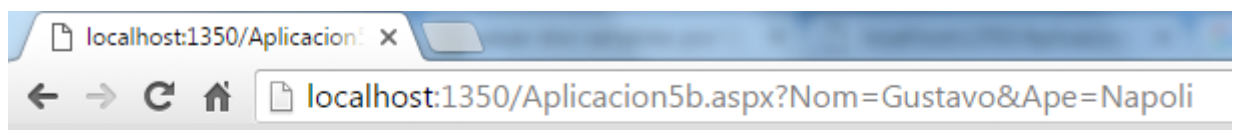
Aplicación 5: Request / Response

El usuario completará su nombre y apellido en el formulario Aplicacion5a.aspx. Luego al dar clic en Aceptar, esa información se mostrará en otro formulario: Aplicacion5b.aspx.



Ingrese nombre:

Ingrese apellido:



Bienvenido: Gustavo Napoli

Pasos a seguir:

1. Crear el formulario llamado: **Aplicación5a.aspx**
2. A ese formulario agregarle los siguientes controles:

Control	Propiedades
TextBox	ID: txtNombre
TextBox	ID: txtApellido
Button	ID: btnAceptar

3. Crear otro formulario llamado: **Aplicación5b.aspx**
4. A ese formulario agregarle los siguientes controles:

Control	Propiedades
Label	ID: lblMensaje TEXT: dejar vacio

5. Hacer doble clic sobre el botón que se encuentra en **Aplicación5a.aspx** y escribir el siguiente código

```
protected void btnAceptar_Click(object sender, EventArgs e)
{
    Response.Redirect("Aplicacion5b.aspx?Nom=" + txtNombre.Text
        + "&Ape=" + txtApellido.Text);
}
```

Funcionamiento del Response.Redirect

Dentro del Response.Redirect() lo que tenemos que escribir es una página a donde queremos redireccionar por ejemplo podríamos haber colocado solo lo siguiente:

```
Response.Redirect("Aplicacion5b.aspx");
```

Si hubiéramos colocado la instrucción anterior, solo se redireccionaría a esa página pero lo que yo quiero es que además de redireccionar, que se envíen parámetros. Para eso tendría que hacer lo siguiente:

```
Response.Redirect("Aplicacion5b.aspx?Nom=Pepe&Ape=Lopez");
```

- ✓ Al finalizar el nombre del formulario coloco un signo de pregunta.
- ✓ Elijo un nombre para cada parámetro. En este caso es Nom y Ape.
- ✓ Igualo los parámetros a los valores variables.
- ✓ Fijarse que puedo enviar n cantidad de parámetros, solo tengo que colocar &
- ✓ Desde la otra página voy a preguntar por Nom y por Ape para obtener "Pepe" y "Lopez".

Nota: A través del Response estamos enviando información al servidor.

6. Ahora vamos a programar sobre **Aplicación5b.aspx**. Lo que voy a hacer es obtener los parámetros que fueron enviados desde la URL. En el evento Page_Load vamos a escribir la siguiente línea:

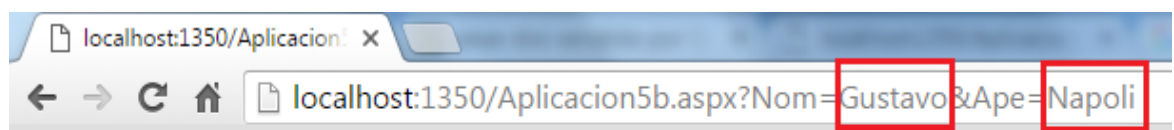
```
protected void Page_Load(object sender, EventArgs e)
{
    string Nombre;
    string Apellido;
    Nombre = Request.QueryString["Nom"];
    Apellido = Request.QueryString["Ape"];
    lblMensaje.Text = "Bienvenido: " + Nombre + " " + Apellido;
}
```

A través del Request estoy obteniendo información del servidor. Request.QueryString["Nom"], lo que hace es obtener de la URL, el dato relacionado a la variable Nom.

7. Probemos el código.

¿Cuál es el aspecto negativo?

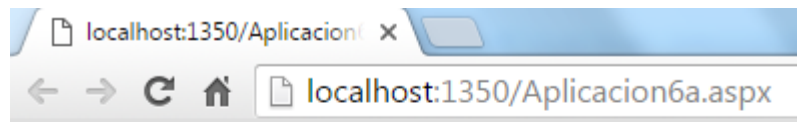
Que en la URL vamos a estar viendo los contenidos enviados



Bienvenido: Gustavo Napoli

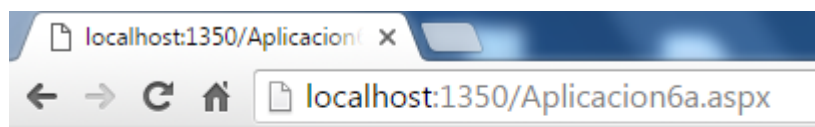
Aplicación 6: Server

A continuación vamos a hacer lo mismo que el ejercicio anterior, vamos a enviar parámetros pero sin que se vean en la URL.



Ingrese nombre:

Ingrese apellido:



Bienvenido: Gabriel Iglesias

Pasos a seguir:

1. Crear el formulario llamado: **Aplicación6a.aspx**
2. A ese formulario agregarle los siguientes controles:

Control	Propiedades
TextBox	ID: txtNombre
TextBox	ID: txtApellido
Button	ID: btnAceptar

3. Crear otro formulario llamado: **Aplicación6b.aspx**
4. A ese formulario agregarle los siguientes controles:

Control	Propiedades
Label	ID: lblMensaje TEXT: dejar vacio

5. Dentro del formulario Aplicacion6a.aspx hacer doble clic sobre el botón y programar lo siguiente:

```
protected void btnAceptar_Click(object sender, EventArgs e)
{
    Server.Transfer("Aplicacion6b.aspx");
}
```

Nota: Lo que estoy haciendo es transferir los controles de la página Aplicacion6a.aspx a la página Aplicacion6b.aspx

6. Dentro del formulario Aplicación6b.aspx, en el evento Page_Load programar lo siguiente:

```
protected void Page_Load(object sender, EventArgs e)
{
    string Nombre;
    string Apellido;

    Nombre = Request["txtNombre"].ToString();

    Apellido = ((TextBox)PreviousPage.FindControl("txtApellido")).Text;

    lblMensaje.Text = "Bienvenido: " + Nombre + " " + Apellido;
}
```

Fijarse que hay dos maneras de obtener el control de la página anterior:

❖ **Request["txtNombre"].ToString()**

De esta manera solo podemos obtener la propiedad Text

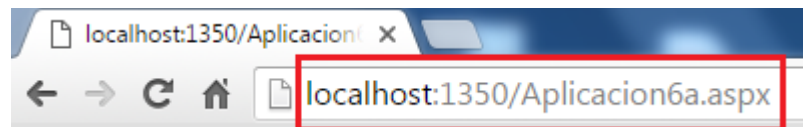
❖ ((TextBox)PreviousPage.FindControl("txtApellido")).Text

De esta forma lo que estoy haciendo es encontrar el control de la página anterior, castearlo a TextBox, y accediendo a cualquier propiedad de ese elemento.

7. Probemos la aplicación

¿Cuál es el aspecto negativo de esta forma de enviar información?

Fíjense que aunque lo haya redireccionado a Aplicacion6b.aspx en la URL va a seguir apareciendo como Aplicación6a.aspx



Bienvenido: Gabriel Iglesias