

**UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL GENERAL PACHECO**

**TÉCNICO SUPERIOR EN PROGRAMACIÓN
PROGRAMACIÓN III**

UNIDAD 5:

***Variables sesión y aplicación – Cookies -
GridView***

TSSI TAMARA HERRERA

Contenidos de la unidad

1. Variables de tipo Session y Application

- Declaración de variable
- Leer variables
- Borrar variable

2. Cookies

- Declaración de cookies
- Leer cookies
- Borrar cookies

3. GridView

- Configuración manual de los ítems de una grilla
- Edición de datos dentro de una grilla
- Borrar datos dentro de una grilla
- Selección de datos dentro de una grilla

Introducción

En esta unidad aprenderemos a utilizar variables de tipo sesión y aplicación. También conoceremos cómo funcionan las cookies y cómo podemos configurar un GridView. Para el GridView utilizaremos la base de datos Librería.

¿Cómo guardar información de manera global dentro de nuestro proyecto?

Existen dos tipos de variables globales, las variables de tipo sesión o las variables de tipo aplicación. Para guardar información de un usuario particular podemos utilizar las variables sesión y para guardar información común a todos los usuarios podemos utilizar las variables aplicación. Al ser global, se va a poder acceder a ella desde cualquier formulario web.

Variables de sesión

Estas variables duran lo que el usuario se mantiene en el navegador, si cierra el navegador la variable desaparece.

Variables de aplicación

Estas permanecen a pesar de que el usuario cierre el navegador ya que se mantienen mientras el servidor se esté ejecutando.

¿Cómo declarar estas variables?

Las variables de aplicación o de sesión funcionan de la misma manera. Para declarar estas variables, solo tengo que colocar un nombre dentro de los corchetes e igualarlos a un valor.

Ejemplo de declaración:

```
Session ["Cantidad"] = 5;  
Session ["Nombre"] = "Pepe";  
Session ["Tabla"] = (DataTable)x;
```

Como pueden observar, puedo guardar cualquier tipo de valor, String, Int, DataTable, etc. Para declarar variables de tipo aplicación sería lo mismo:

```
Application ["Cantidad"] = 5;  
Application ["Nombre"] = "Pepe";  
Application ["Tabla"] = (DataTable)x;
```

¿Cómo puedo consultar por el contenido de una variable Session o Application?

Para consultar simplemente tenemos que recordar el nombre que le dimos al crearla. En el siguiente ejemplo consulto por lo que hay en la sesión que se llama Nombre y lo convierto a String para poder guardarlo en una variable String.

```
string aux2 = Session["Nombre"].ToString();
```

Fijarse que es importante que recordemos el tipo de datos que está asociado a esa variable. Si guardamos un int, deberemos castearlo a int para tratarlo como tal, lo mismo para los otros tipos de datos.

Ejemplos:

- *Convierto a entero lo que había dentro de la variable Session, le sumo dos y lo vuelvo a guardar.*

```
Session["Cantidad"] = Convert.ToInt32(Session["Cantidad"]) + 2;
```

- *Guardo en una variable DataTable lo que había dentro de la sesión.*

```
DataTable dt = (DataTable)Application["Tabla"];
```

¿Cómo puedo borrar una variable Session o Application?

Sencillo, simplemente deberemos igual esa variable cookie a null para borrarla.

```
Application["cant"] == null;
```

Cookies

Las cookies proporcionan un medio para almacenar información específica del usuario en las aplicaciones Web. Por ejemplo, cuando un usuario visita un sitio, las cookies pueden emplearse para almacenar las preferencias del usuario u otra información. Cuando el usuario visite el sitio Web de nuevo, la aplicación podrá recuperar la información que almacenó previamente.

Una cookie es un pequeño fragmento de texto que acompaña a las solicitudes y a las páginas mientras éstas se transmiten del servidor Web al explorador y viceversa. La cookie contiene información que la aplicación Web puede leer cada vez que el usuario visita el sitio.

Por ejemplo, si un usuario solicita una página de un sitio y la aplicación no solo envía una página, sino también una cookie que contiene la fecha y la hora, cuando el explorador del usuario obtenga la página, también obtendrá la cookie, que se almacenará en una carpeta en el disco duro del usuario.

Cualquiera de las alternativas colocadas a continuación funcionan de igual manera, ustedes decidirán cuál les parece más conveniente.

¿Cómo crear una cookie?

Alternativa 1:

```
HttpCookie ck = new HttpCookie("NombreUsuario", txtUsuario.Text);  
//ck.Expires = DateTime.Now.AddDays(1);  
ck.Expires = DateTime.Now.AddSeconds(15);  
this.Response.Cookies.Add(ck);
```

En este ejemplo, creo un tipo de dato `HttpCookie` y en el constructor envío el *nombre* de la cookie y su *valor*. En este caso se va a crear una cookie llamada “*NombreUsuario*” y su valor será lo que ingrese el usuario en el *TextBox*. A las cookies es necesario asignarles un tiempo de duración, pueden ser :segundos, minutos, días, etc.

El último paso sería agregar esa cookie al servidor. Fijarse que a diferencia de las variables `Session` y `Application`, en las cookies solo se le puede guardar un tipo de dato: `String`.

Alternativa 2:

```
Response.Cookies["NombreUsuario"].Value = txtUsuario.Text;  
Response.Cookies["NombreUsuario"].Expires = DateTime.Today.AddDays(1);
```

Otra forma de crear una cookie, es crearla directamente en el servidor. Fijarse que en la primera línea la creo, asignándole un nombre y valor. En la segunda línea asigno el tiempo de duración.

¿Cómo consultar por una cookie ya creada?

Alternativa 1:

```
HttpCookie ck;  
ck = this.Request.Cookies["usuario"];  
String aux = ck.Value;
```

Mediante un *Request* al servidor, obtengo la cookie y la igalo al tipo de dato `HttpCookie` para así obtener el valor.

Alternativa 2:

```
String s = Request.Cookies["NombreUsuario"].Value;
```

Obtengo directamente el valor de la cookie, realizando un `Request` al servidor.

¿Cómo consultar si la cookie existe?

Fácil, le pregunto al servidor por el nombre de una cookie y si esta nula, significa no está siendo utilizada.

```
if(this.Request.Cookies["usuario"] == null)
{
    //Cookie inexistente
}
else
{
    //Cookie en el servidor
}
```

¿Cómo borrar una cookie?

Para borrarla, lo que tenemos que hacer es asignarle un tiempo de expiración con valor negativo.

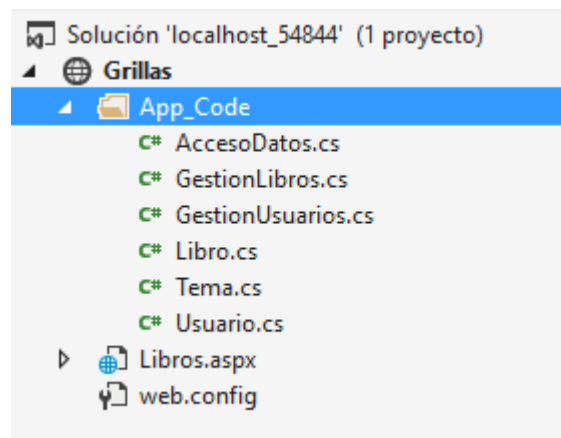
```
HttpCookie ck = new HttpCookie("NombreUsuario");
ck.Expires = DateTime.Now.AddDays(-1);
this.Response.Cookies.Add(ck);
```

Aquí lo que estoy haciendo es obtener la cookie del servidor, cambiarle el tiempo de expiración y nuevamente enviarla al servidor.

GridView

En la **unidad 2**, habíamos visto como mostrar información en un GridView. A partir de esta unidad vamos a realizar lo mismo pero utilizando **clases**. Recuerden que para incorporar una clase se debe hacer *segundo clic sobre el proyecto -> Agregar -> Agregar nuevo elemento -> Clase*

A continuación vamos a trabajar con el sitio web Grillas que contiene clases y se encuentra en el aula virtual.



En el formulario llamado *Libros.aspx*, hay dibujado una grilla llamada *grdLibros*. Esta grilla tiene configurado el formato automático a otoño.

Libros.aspx* [icon] X

Listado de libros:

Column0	Column1	Column2
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc
abc	abc	abc

¿Cómo configurar manualmente los ítems que quiero mostrar en un GridView?

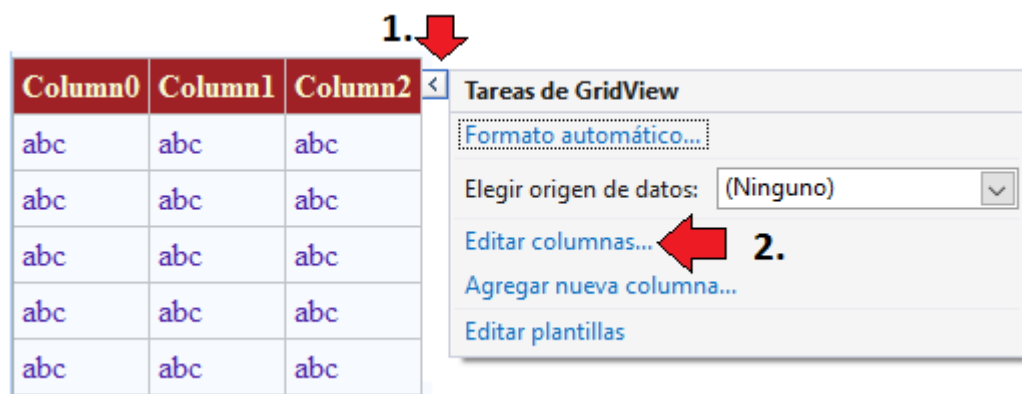
Nosotros hasta ahora, le volcamos información al GridView y el GridView automáticamente muestra toda esa información. Pero si quisiéramos nosotros podríamos configurar que ítems mostrar dentro de la grilla, sin utilizar el modo automático. Ahora vamos a ver como se realiza:

Paso 0: Desde el Load del formulario cargar la grilla con datos.

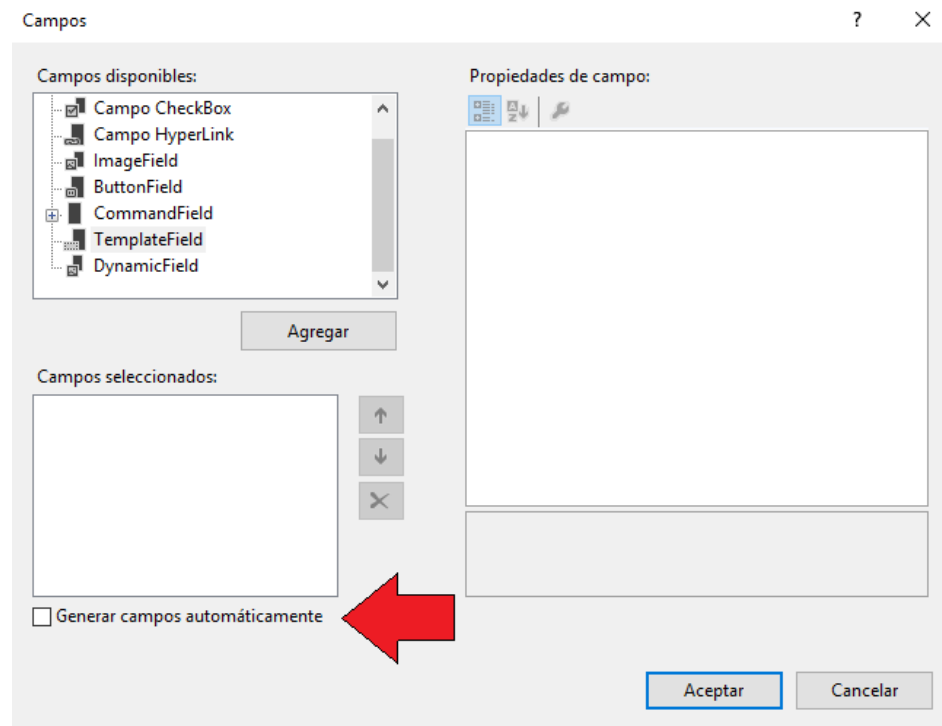
```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        cargarGridView();
    }
}

private void cargarGridView()
{
    GestionLibros glibros = new GestionLibros();
    grdLibros.DataSource = glibros.ObtenerTodosLosLibros();
    grdLibros.DataBind();
}
```

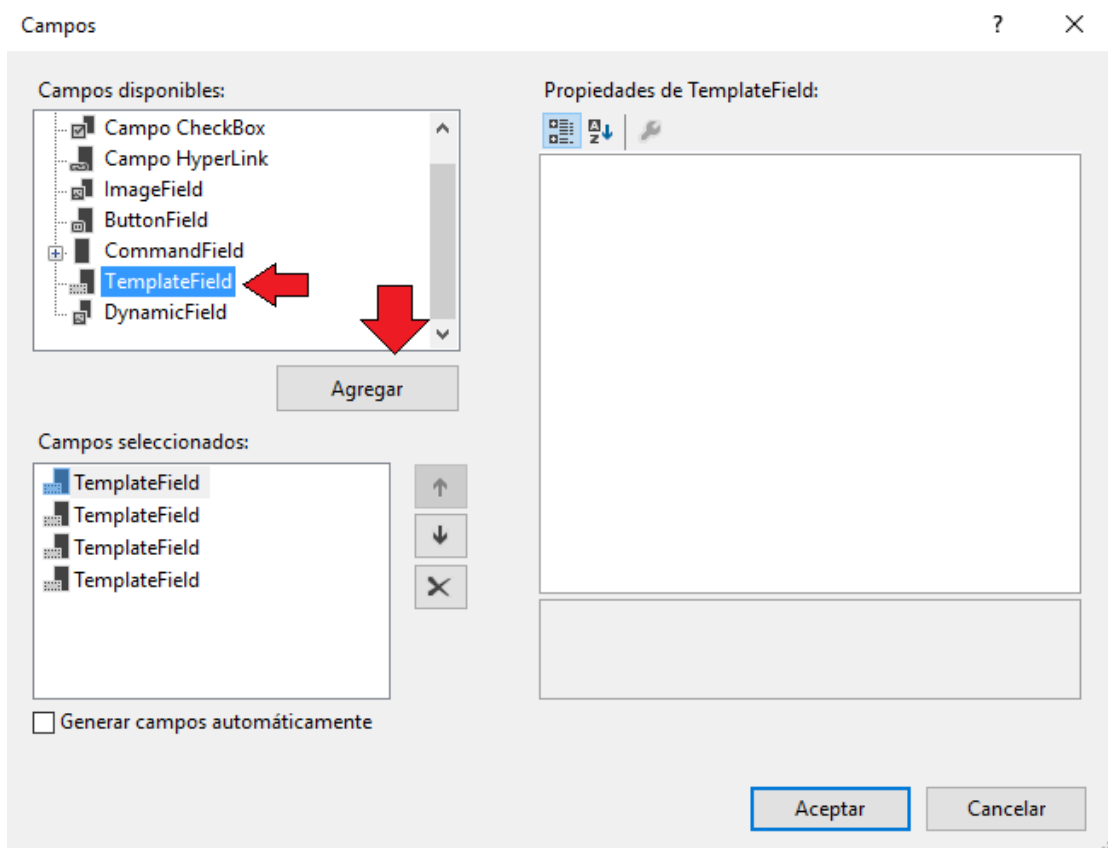
Paso 1: Ir al diseño de la grilla, editar las columnas



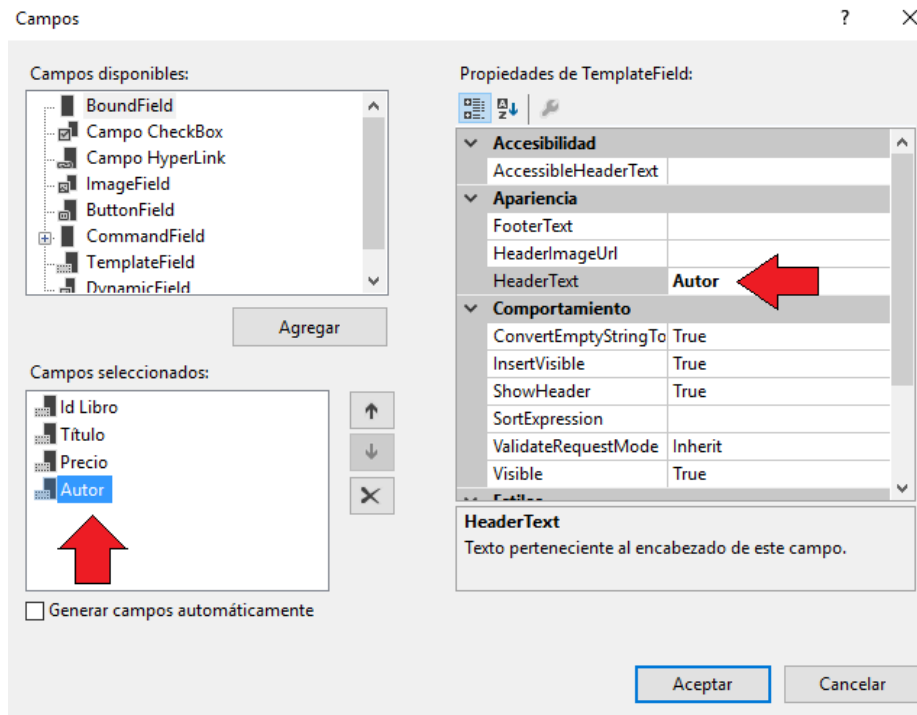
Paso 2: Destildar la opción para generar campos automáticamente.



Paso 3: Agregar cuatro campos de tipo TemplateField



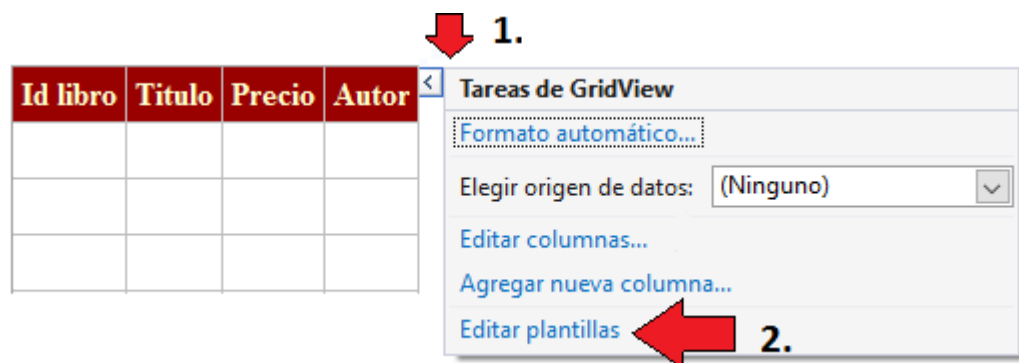
Paso 4: A cada TemplateField cambiarle la propiedad HeaderText. Les coloqué los siguientes nombres: Id Libro, Título, Precio y Autor.



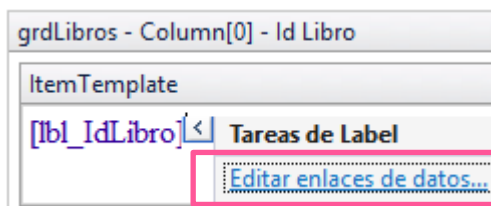
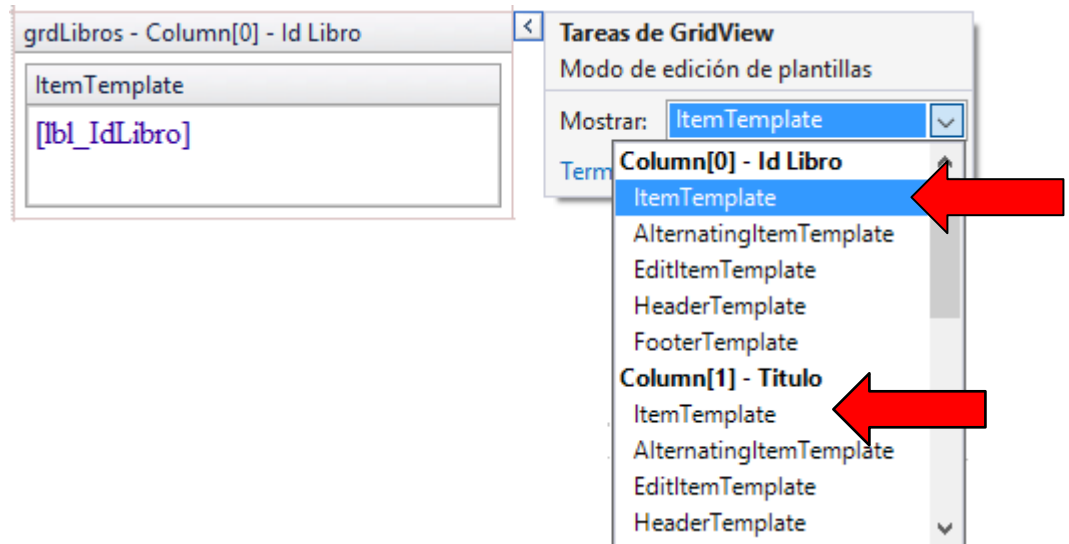
Paso 5: La grilla quedará vacía

Id libro	Título	Precio	Autor

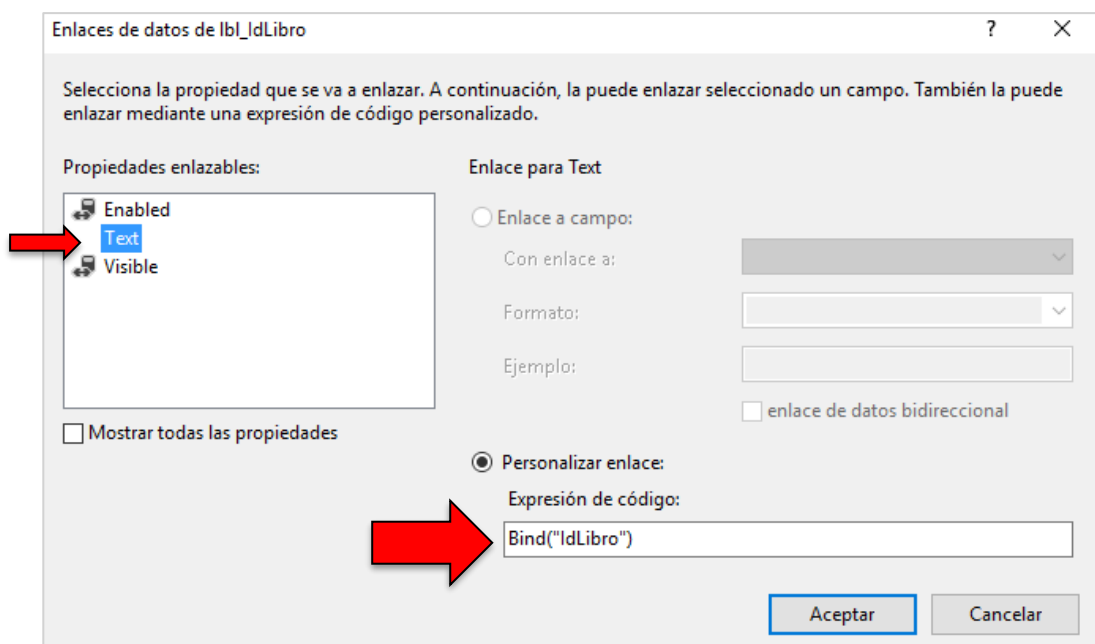
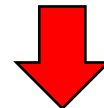
Paso 6: Ir a editar plantillas



Paso 7: En el modo de edición de plantillas, debemos colocar un Label en cada ítemTemplate de cada columna. Al cada Label le cambio el ID, por ejemplo al Label de Id Libro le coloco ID lbl_IdLibro



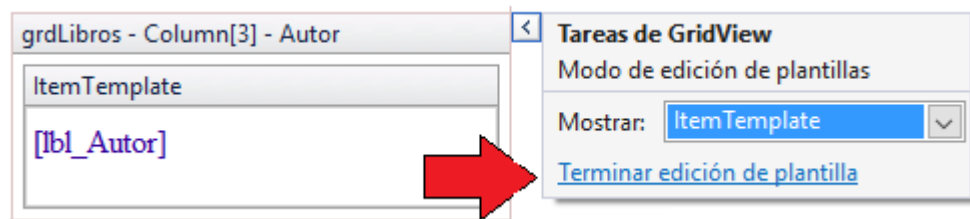
En cada Label, hacer clic sobre editar enlace de datos. Ahora solo nos queda enlazar cada uno de esos Labels a un campo de la BD.



Seleccionamos la propiedad **Text** de ese Label, y lo enlazamos a un campo de la base de datos a través del: **Bind(“NombreCampoTabla”)**

IMPORTANTE: Repetir este procedimiento con cada ítemTemplate de cada columna, es importante realizar el BIND para vincular el Texto del Label con la BD.

Paso 8: Al finalizar deberemos hacer clic en terminar edición de plantilla



El diseño de la grilla quedará de la siguiente manera:

Id Libro	Título	Precio	Autor
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound

Paso 9: Probar la aplicación



¿Cómo editar datos sobre la grilla?

Paso 0: Desde el Load de la página cargar los datos que va a tener esa grilla, a través del DataSource y DataBind.

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        cargarGridView();
    }
}

private void cargarGridView()
{
    GestionLibros glibros = new GestionLibros();
    grdLibros.DataSource = glibros.ObtenerTodosLosLibros();
    grdLibros.DataBind();
}
```

Paso 1: Realizar la configuración manual de cada ítem de la grilla (Repetir los pasos vistos anteriormente)

Paso 2: Además de configurar el ÍtemTemplate, deberemos configurar los EditItemTemplate. Para esto deberemos ir nuevamente a Editar plantillas.

Id Libro	Título	Precio	Autor
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound
DataBound	DataBound	DataBound	DataBound

Tareas de GridView

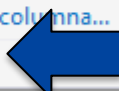
[Formato automático...](#)

Elegir origen de datos: (Ninguno)

[Editar columnas...](#)

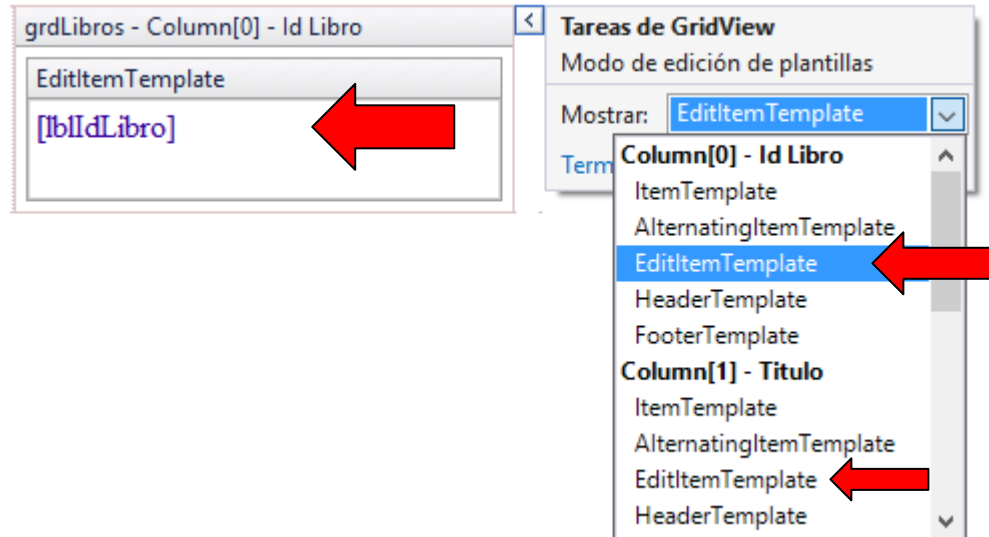
[Agregar nueva columna...](#)

[Editar plantillas](#)

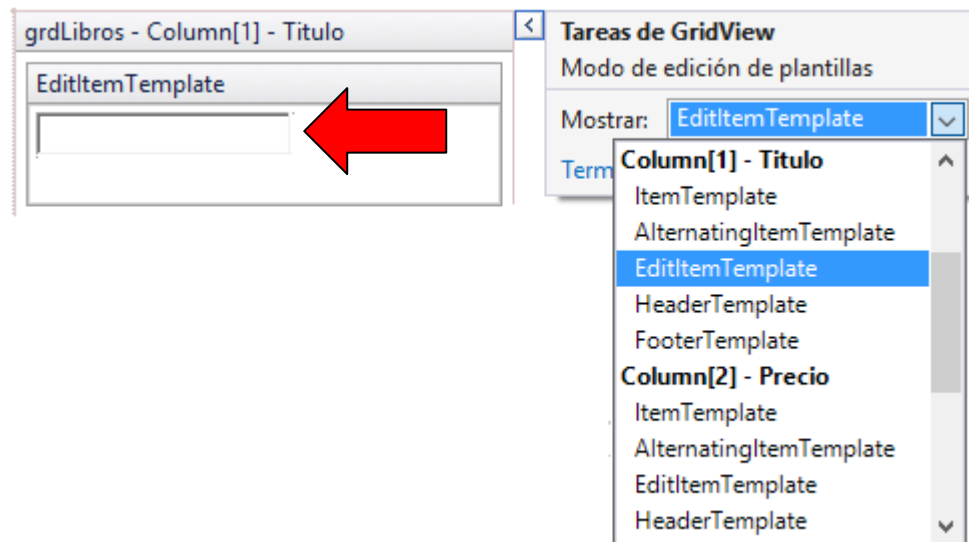


Paso 3: Dibujar los controles sobre los EditItemTemplate

Colocarle un Label al primer EditItemTemplate de Id Libro, así el usuario no puede modificarlo. Le coloque a ese Label ID=lblIdLibro



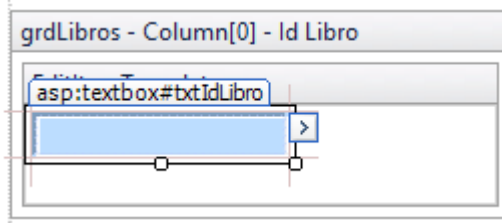
Colocarle un TextBox a los siguientes EditItemTemplate



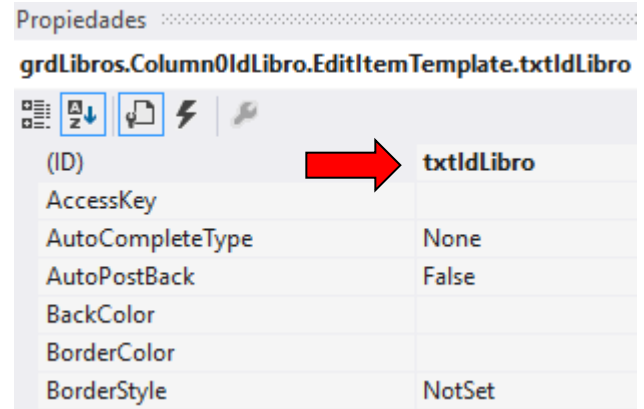
Cambiarle los ID a los TextBox

Para cambiarle el ID a los controles, se debe hacer clic sobre el control -> Propiedades

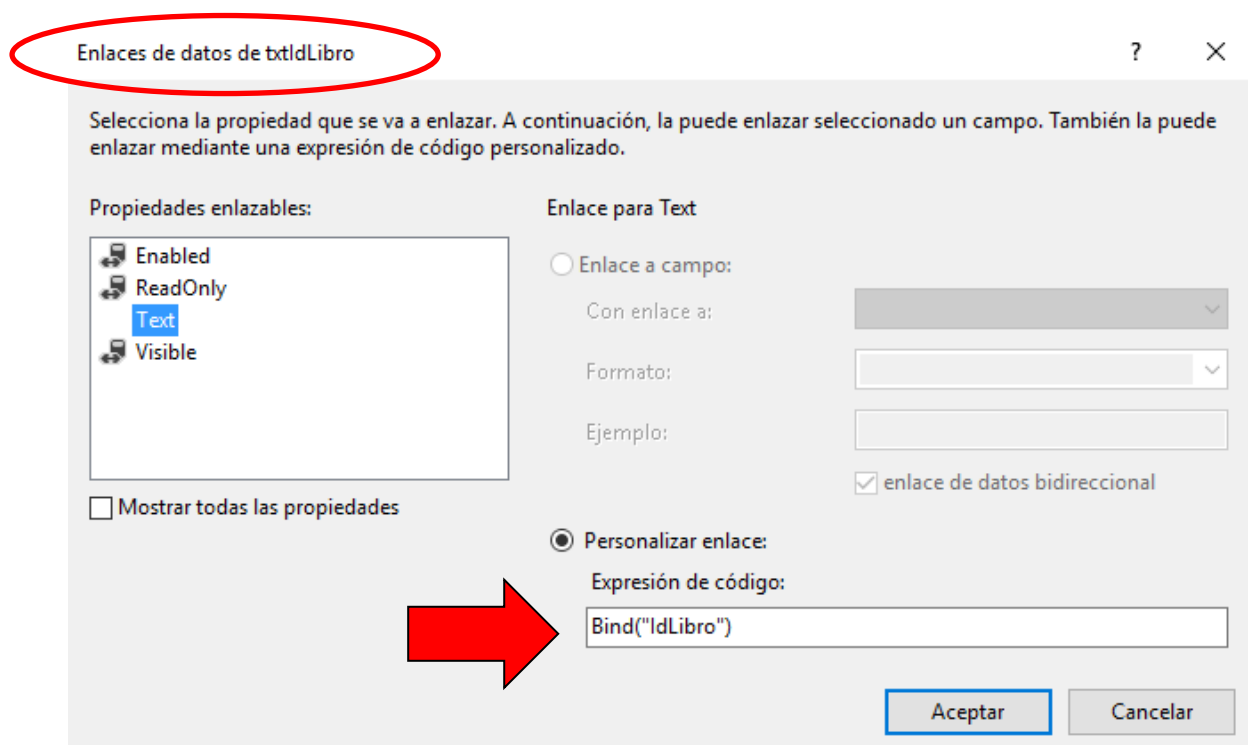
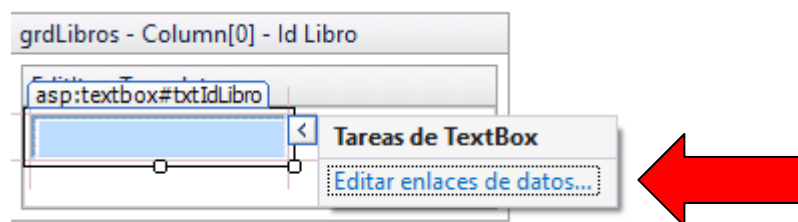
Clic sobre el TextBox



Cambio el ID

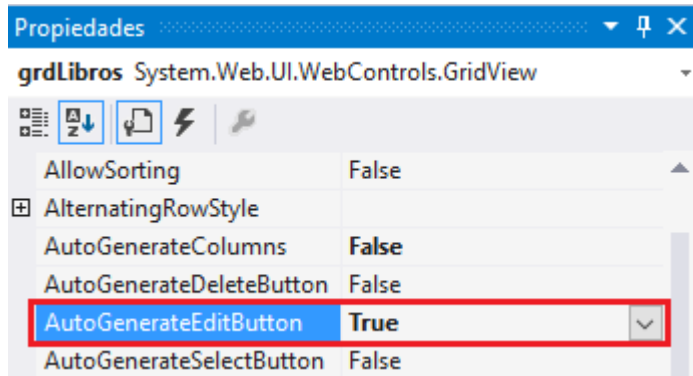


Paso 4: Enlazar esos controles a un campo de la base de datos



Paso 5: Sobre las propiedades de la grilla habilitar la generación automática del botón editar.

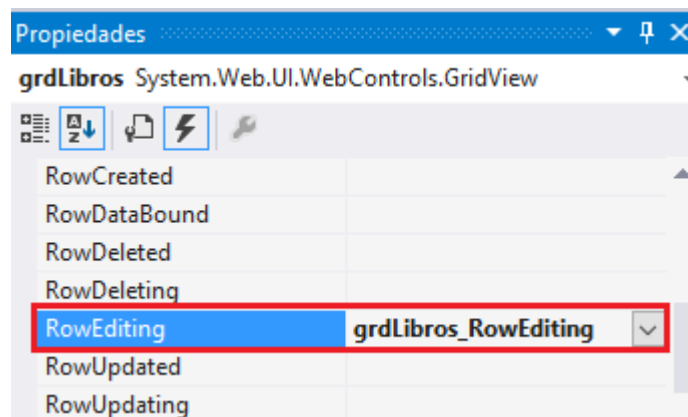
Habilitamos la propiedad



Diseño de la grilla:

	Id Libro	Titulo
Editar	DataBound	DataBound
Editar	DataBound	DataBound
Editar	DataBound	DataBound
Editar	DataBound	DataBound
Editar	DataBound	DataBound

Paso 5: Crear el evento RowEditing, haciendo doble clic sobre el mismo.

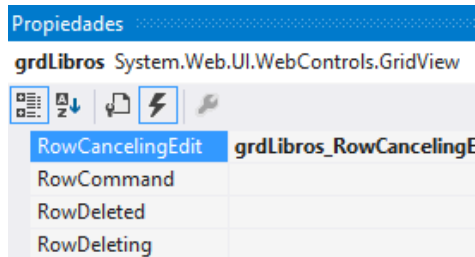


Paso 6: En el evento RowEditing colocar el siguiente código

```
protected void grdLibros_RowEditing(object sender,
                                   GridViewEditEventArgs e)
{
    grdLibros.EditIndex = e.NewEditIndex;
    cargarGridView();
}
```

*Estas líneas reconstruyen el GridView, luego que el usuario hizo clic en Editar.
La grilla se reconstruye con los TextBox del EditItemTemplate*

Paso 7: Crear el evento RowCancelingEdit e ingresar el siguiente código:



```
protected void grdLibros_RowCancelingEdit
{
    grdLibros.EditIndex = -1;
    cargarGridView();
}
```

Seteo el índice a -1 para que no haya ningún elemento seleccionado, vuelvo a cargar la grilla.

Paso 8: Crear el evento RowUpdating e ingresar el siguiente código:

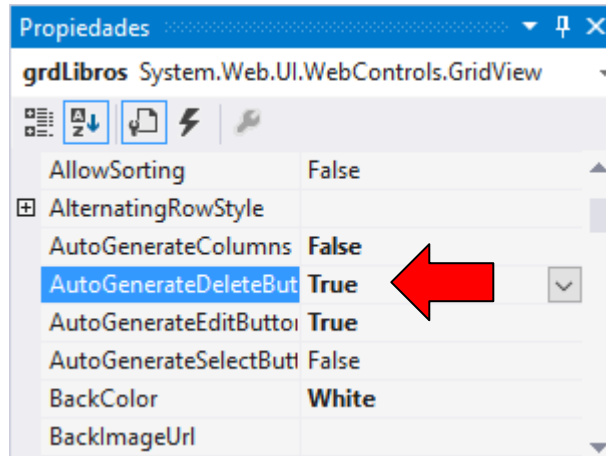
```
protected void grdLibros_RowUpdating(object sender, GridViewUpdateEventArgs e)
{
    Libro libro = new Libro();
    libro.Autor = ((TextBox)grdLibros.Rows[e.RowIndex].FindControl("txtAutor")).Text;
    libro.Titulo = ((TextBox)grdLibros.Rows[e.RowIndex].FindControl("txtTitulo")).Text;
    libro.Precio = Convert.ToDecimal(((TextBox)grdLibros.Rows[e.RowIndex].FindControl("txtPrecio")).Text);
    libro.IdLibro = Convert.ToInt32(((Label)grdLibros.Rows[e.RowIndex].FindControl("lblIdLibro")).Text);
    GestionLibros glib = new GestionLibros();
    glib.ActualizarLibro(libro);

    grdLibros.EditIndex = -1;
    cargarGridView();
}
```

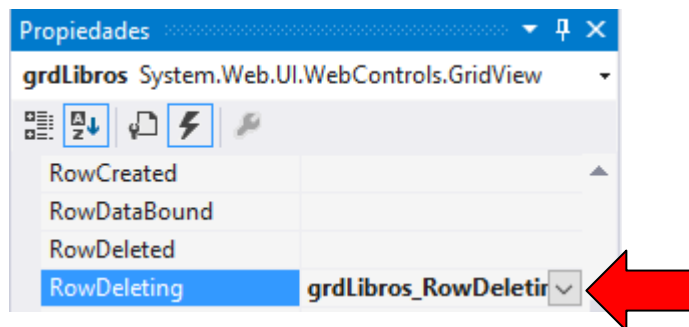
Aca lo que estoy haciendo es obtener el control de la fila que seleccionó el usuario, guardo todos esos datos para actualizarlos en la base de datos. A través del e de la función, se obtiene el índice de la fila seleccionada.

¿Cómo borrar datos sobre la grilla?

Paso 1: Agregar a lo realizado anteriormente el botón de eliminar



Paso 2: Construir el evento RowDeleting



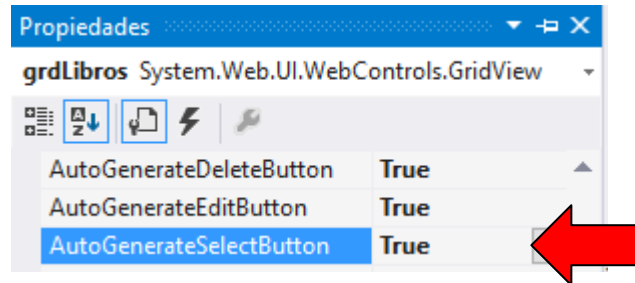
Colocar el siguiente código en el evento:

```
protected void grdLibros_RowDeleting(object sender, GridViewDeleteEventArgs e)
{
    Libro lib = new Libro();
    lib.IdLibro = Convert.ToInt32( ((Label)grdLibros.Rows[e.RowIndex].FindControl("lbl_IdLibro")).Text);
    GestionLibros glib = new GestionLibros();
    glib.EliminarLibro(lib);
    grdLibros.EditIndex = -1;
    cargarGridView();
}
```

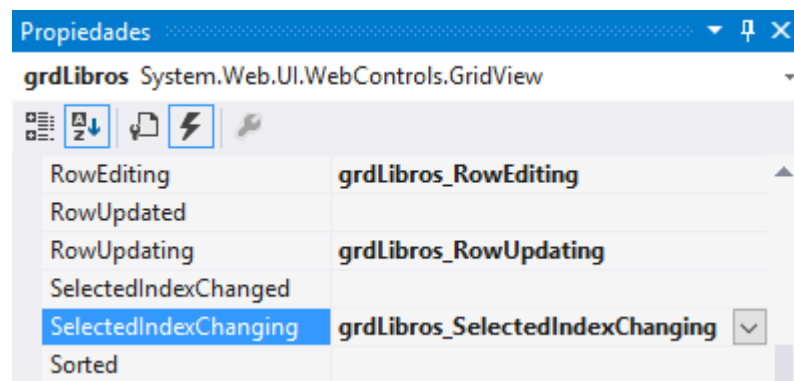
Mediante el FindControl busco el Label que contiene la información del ID del Libro a Eliminar. El Label que buscamos se llama ("lbl_IdLibro"), que es el nombre que del Label que configuramos el ItemTemplate.

¿Cómo seleccionar datos sobre la grilla?

Paso 1: Agregar a lo realizado anteriormente, el botón para seleccionar.



Paso 2: Programar el evento SelectedIndexChanged



Paso 3: Escribir lo siguiente en el evento SelectedIndexChanged

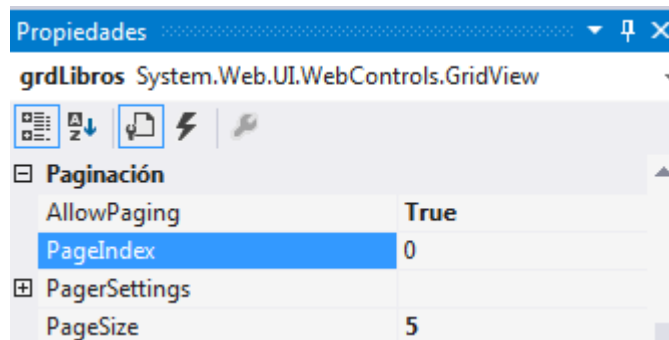
```
protected void grdLibros_SelectedIndexChanged(object sender, GridViewSelectEventArgs e)
{
    string s_IdLibro = ((Label)grdLibros.Rows[e.NewSelectedIndex].FindControl("lbl_IdLibro")).Text;
    string s_Titulo = ((Label)grdLibros.Rows[e.NewSelectedIndex].FindControl("lbl_Titulo")).Text;
    string s_Precio = ((Label)grdLibros.Rows[e.NewSelectedIndex].FindControl("lbl_Precio")).Text;
    string s_Autor = ((Label)grdLibros.Rows[e.NewSelectedIndex].FindControl("lbl_Autor")).Text;
}
```

Obtenemos el contenido de la fila que seleccionó el usuario, buscando los nombres de los Labels que dibujamos en el ItemTemplate.

¿Cómo paginar la grilla?

Paso 1: Para paginar la grilla deberemos habilitar las siguientes propiedades:

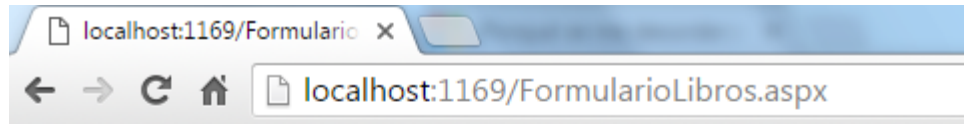
- ***AllowPaging***= *True*
- ***PageSize*** = *Cantidad de filas a mostrar por paginación*



Paso 2: También deberemos configurar el evento PageIndexChanged

```
protected void grdLibros_PageIndexChanged(object sender, GridViewPageEventArgs e)
{
    grdLibros.PageIndex = e.NewPageIndex;
    cargarGridView();
}
```

Luego de incorporar lo visto en esta unidad la grilla quedará de la siguiente manera:



Listado de libros:

	Id Libro	Titulo	Precio	Autor
Editar Eliminar Seleccionar	1	Libro1	100,0000	Autor1
Editar Eliminar Seleccionar	2	Libro2	800000,0000	Autor2
Editar Eliminar Seleccionar	3	Libro3	90,0000	Autor3
Editar Eliminar Seleccionar	4	Libro4	1260,0000	Autor4
Editar Eliminar Seleccionar	5	Libro5	120,0000	Autor5
1 2				

Observaciones:

- Tanto para la seleccionar como para eliminar en el GridView, se trabaja con los controles dibujados en el *ItemTemplate*.
- Para realizar la edición en el GridView se trabaja con los controles dibujados en el *EditItemTemplate*.