CREACIÓN DE CRUD CON LLAMADOS JQUERY & JSON

1. Agregamos una nueva página de contenido al Site.Master llamado Bodegas2 y lo movemos a la carpeta correspondiente.
2. Colocamos un título y arrastramos la tabla bodegas para que nos cree el GridView, luego cambiamos los nombres tal y como muestra este código:

<h1>Bodegas</h1>

<asp:GridView ID="bodegasGridView" runat="server" AllowPaging="True" AllowSorting="True" AutoGenerateColumns="False" CellPadding="4" DataKeyNames="IDBodega" DataSourceID="bodegasSqlDataSource" EmptyDataText="No hay registros de datos para mostrar." ForeColor="#333333" GridLines="None">

<AlternatingRowStyle BackColor="White" />

<Columns>

<asp:CommandField ShowDeleteButton="True" ShowEditButton="True" />

<asp:BoundField DataField="IDBodega" HeaderText="ID Bodega" ReadOnly="True" SortExpression="IDBodega" />

<asp:BoundField DataField="Descripcion" HeaderText="Descripción" SortExpression="Descripcion" />

</Columns>

<EditRowStyle BackColor="#2461BF" />

<FooterStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />

<HeaderStyle BackColor="#507CD1" Font-Bold="True" ForeColor="White" />

<PagerStyle BackColor="#2461BF" ForeColor="White" HorizontalAlign="Center" />

<RowStyle BackColor="#EFF3FB" />

<SelectedRowStyle BackColor="#D1DDF1" Font-Bold="True" ForeColor="#333333" />

<SortedAscendingCellStyle BackColor="#F5F7FB" />

<SortedAscendingHeaderStyle BackColor="#6D95E1" />

<SortedDescendingCellStyle BackColor="#E9EBEF" />

<SortedDescendingHeaderStyle BackColor="#4870BE" />

</asp:GridView>

<asp:SqlDataSource ID="bodegasSqlDataSource" runat="server" ConnectionString="<%$ ConnectionStrings:DefaultConnection %>" DeleteCommand="DELETE FROM [Bodega] WHERE [IDBodega] = @IDBodega" InsertCommand="INSERT INTO [Bodega] ([Descripcion]) VALUES (@Descripcion)" ProviderName="<%$ ConnectionStrings:DefaultConnection.ProviderName %>" SelectCommand="SELECT [IDBodega], [Descripcion] FROM [Bodega]" UpdateCommand="UPDATE [Bodega] SET [Descripcion] = @Descripcion WHERE [IDBodega] = @IDBodega">

<DeleteParameters>

<asp:Parameter Name="IDBodega" Type="Int32" />

</DeleteParameters>

<InsertParameters>

<asp:Parameter Name="Descripcion" Type="String" />

</InsertParameters>

<UpdateParameters>

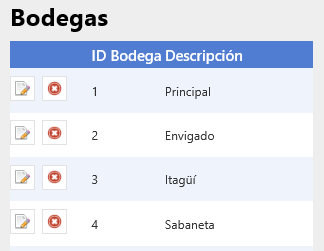
<asp:Parameter Name="Descripcion" Type="String" />

<asp:Parameter Name="IDBodega" Type="Int32" />

</UpdateParameters>

</asp:SqlDataSource>

1. Descargamos unos íconos para editar, borrar, guardar y cancelar, pequeños, puede ser de 16 x 16. Y colocamos los respectivos íconos al GridView, luego colocamos las operaciones como un template field.
2. Probemos lo que hemos hecho hasta el momento:



1. Cambiar:

<asp:TemplateField ShowHeader="False">

<EditItemTemplate>

<asp:ImageButton ID="ImageButton1" runat="server" CausesValidation="True" CommandName="Update" ImageUrl="~/Images/Icons/save.png" Text="Actualizar" />

&nbsp;<asp:ImageButton ID="ImageButton2" runat="server" CausesValidation="False" CommandName="Cancel" ImageUrl="~/Images/Icons/cancel.png" Text="Cancelar" />

</EditItemTemplate>

<ItemTemplate>

<asp:ImageButton ID="ImageButton1" runat="server" CausesValidation="False" CommandName="Edit" ImageUrl="~/Images/Icons/edit.png" Text="Editar" />

&nbsp;<asp:ImageButton ID="ImageButton2" runat="server" CausesValidation="False" CommandName="Delete" ImageUrl="~/Images/Icons/delete.png" Text="Eliminar" />

</ItemTemplate>

</asp:TemplateField>

Por:

<asp:TemplateField ShowHeader="False">

<EditItemTemplate>

<asp:ImageButton

ID="ImageButton1"

runat="server"

CausesValidation="True"

CommandName="Update"

ImageUrl="~/Images/Icons/save.png"

ToolTip="Guarda los cambios"

Text="Actualizar" />

<asp:ImageButton ID="ImageButton2"

runat="server"

CausesValidation="False"

CommandName="Cancel"

ImageUrl="~/Images/Icons/cancel.png"

ToolTip="Cancela los cambios"

Text="Cancelar" />

</EditItemTemplate>

<ItemTemplate>

<asp:ImageButton

ID="ImageButton1"

runat="server"

CausesValidation="False"

CommandName="Edit"

ImageUrl="~/Images/Icons/edit.png"

ToolTip="Edita el registro"

Text="Editar" />

<asp:ImageButton

ID="ImageButton2"

runat="server"

ToolTip="Borrar registro"

CausesValidation="False"

AlternateText='<%# Eval("IDBodega") %>'

OnClientClick="javascript:return confirmarBorrado(this.alt);"

CommandName="Delete"

ImageUrl="~/Images/Icons/delete.png"

Text="Eliminar" />

</ItemTemplate>

</asp:TemplateField>

1. Agregue el siguiente Script:

<script>

function confirmarBorrado(ID) {

alert(ID);

$("<div></div>").html("¿Realmente desea borrar el registro?").

dialog({

title: "Mensaje", modal: true, buttons: [

{

text: "Si",

click: function () {

alert("Yes");

$(this).dialog("close");

}

},

{

text: "No",

click: function () {

alert("No");

$(this).dialog("close");

}

}

]

}

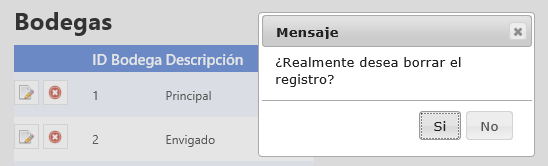
);

return false;

}

</script>

1. Probemos lo que llevamos hasta el momento:



1. Reemplacemos los alert por el verdadero llamado a la función que valida y borra la bodega. Reemplace el script por:

function confirmarBorrado(ID) {

$("<div></div>").html("¿Realmente desea borrar el registro?").

dialog({

title: "Mensaje", modal: true, buttons: [

{

text: "Si",

click: function () {

BorrarRegistro(ID);

$(this).dialog("close");

}

},

{

text: "No",

click: function () {

$(this).dialog("close");

}

}

]

}

);

return false;

}

1. Agreguemos la función BorrarRegistro:

function BorrarRegistro(ID) {

var pageUrl = '<%= ResolveUrl("~/Mantenimientos/Bodegas2.aspx/BorrarRegistro") %>';

var parameter = { "ID": ID }

$.ajax({

type: 'POST',

url: pageUrl,

data: JSON.stringify(parameter),

contentType: 'application/json; charset=utf-8',

dataType: 'json',

success: function (data) {

$("<div></div>").html(data.d).

dialog({

title: "Confirmación", modal: true, buttons: [

{

text: "Ok",

click: function () {

$(this).dialog("close");

}

}

]

}

);

},

error: function (data, success, error) {

$("<div></div>").html("No se puede borrar bodega, tiene registros relacionados").

dialog({

title: "Error", modal: true, buttons: [

{

text: "Ok",

click: function () {

$(this).dialog("close");

}

}

]

}

);

}

});

}

1. Creamos el método DeleteBodega en el CADBodega
2. Agreguemos el Web Method BorrarRegistro:

[WebMethod]

public static string BorrarRegistro(int ID)

{

CADBodega.DeleteBodega(ID);

return "Bodega borrada";

}

1. Probemos lo que llevamos hasta el momento:



1. Ahora agreguemos un botón de nuevo, después del título de bodegas:

<asp:Button ID="nuevoButton" runat="server" Text="Nuevo Registro" CssClass="nuevoButton" />

1. Y creemos el div con el contenido de nueva bodega:

<div id="dialog-nuevo" title="Nuevo registro">

<p>

Descripción:

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

</p>

</div>

1. Adicionemos este código al script para que llame el dialogo (antes de </script>):

$(document).ready(function () {

$("#dialog-nuevo").hide();

$(".nuevoButton").click(function () {

<% descripcionTextBox.Text = string.Empty; %>

$("#dialog-nuevo").dialog({

resizable: false,

height: 200,

width: 350,

modal: true,

buttons: {

"Agregar": function () {

alert("Yes");

$(this).dialog("close");

},

"Cancelar": function () {

$(this).dialog("close");

}

}

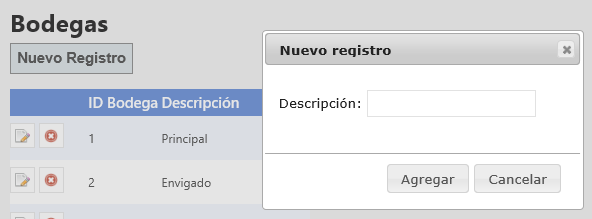
});

return false;

});

});

1. Probemos lo que llevamos hasta el momento:



1. Cambien el alert(“Yes”) por el verdadero llamado la función que agrega la bodega:

Cambie:

alert("Yes");

Por:

InsertBodega();

1. Y agreguemos la función InsertBodega(), antes del $(document):

function InsertBodega() {

var pageUrl = '<%= ResolveUrl("~/Mantenimientos/Bodegas2.aspx/InsertBodega") %>';

var descripcion = $("#<%= descripcionTextBox.ClientID %>").val();

var parameter = { "descripcion": descripcion, }

$.ajax({

type: 'POST',

url: pageUrl,

data: JSON.stringify(parameter),

contentType: 'application/json; charset=utf-8',

dataType: 'json',

success: function (data) {

$("<div></div>").html(data.d).

dialog({

title: "Confirmación", modal: true, buttons: [

{

text: "Ok",

click: function () {

$(this).dialog("close");

}

}

]

}

);

},

error: function (data, success, error) {

$("<div></div>").html("Error: " + error).

dialog({

title: "Error", modal: true, buttons: [

{

text: "Ok",

click: function () {

$(this).dialog("close");

}

}

]

}

);

}

});

}

1. Y creemos el Web Method InsertBodega():

[WebMethod]

public static string InsertBodega(string descripcion)

{

if (descripcion == string.Empty)

{

return "Bebe ingresar una descripción";

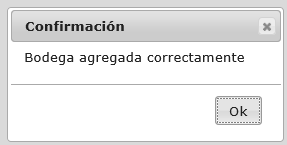
}

CADBodega.InsertBodega(descripcion);

return "Bodega agregada correctamente";

}

1. Probemos lo que hemos logrado!!!





1. Para que la página haga una recarga automática, agregue estas líneas antes del último asp:content

<div id="hide">

<asp:Button ID="fakeButton" CssClass="fakeButton" runat="server" Text="Fake" OnClick="fakeButton\_Click"/>

</div>

1. Y en el evento fakeButton\_Click coloque está línea:

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

{

Response.Redirect("~/Mantenimientos/Bodegas2.aspx");

}

1. Ocultemos esta división, justo después donde ocultamos del dialog-nuevo:

$("#dialog-nuevo").hide();

$("#hide").hide();

1. Agreguemos un llamando al fakeButton justo antes de cerrar los dialofos de confirmación de agregar y eliminar bodega:

title: "Confirmación", modal: true, buttons: [

{

text: "Ok",

click: function () {

$(".fakeButton").click();

$(this).dialog("close");

}

}

]

1. De esta manera ya no tenemos el problema de refrescado de pantalla!!!