|  |
| --- |
| Intrducir esto para validaciones en web.config  <appSettings>  <add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />  </appSettings> |
| Agregar proyecto biblioteca de clases .NET Framework  DATOS NEGOCIO ENTIDADES |
| Agregar referencias en  DATOS Ensambaldos/System Configuration  Proyectos/Entidades  NEGOCIO Proyectos/ Datos, Entidades  SOLUCION Proyectos/Entidades, Negocio |
| En DATOS agregar una clase Conexión  Incluir  using System.Data.SqlClient;  using System.Configuration; |
| public class Conexion  {  protected SqlConnection cnn;  public Conexion()  {  cnn=new SqlConnection(ConfigurationManager.ConnectionStrings["NorthWindConnection"].ConnectionString);  }  public void abrir() {  if (cnn.State== System.Data.ConnectionState.Closed || cnn.State == System.Data.ConnectionState.Broken)  {  cnn.Open();  }    }  public void cerrar() {  if (cnn.State == System.Data.ConnectionState.Open)  {  cnn.Close();  }  } |
| Para obtener la conexión en el sqldatasource colocar como  Origen datos Microsoft sqlclient  nombre del servidor .\SQLEXPRESS  escoger la base de datos  copiar la cadena de conexión y en WEB:CONFIG añadir  <connectionStrings>  <add name="NorthWindConnection" providerName="System.Data.SqlClient" connectionString="Data Source=.\SQLExpress;Initial Catalog=Northwind;Integrated Security=True"/>  </connectionStrings> |
| En ENTIDADES crear una calse por cada entidad  Añadir en cada una  public class Pedido |
| En DATOS crear una clase por cada ENTIDAD  En cada clase hacerla public y añadir  using System.Data.SqlClient;  using ENTIDADES;  public class DAO\_Cliente:Conexion |
| En NEGOCIO crear una clase por cada ENTIDAD  Añadir en cada una  using ENTIDADES;  using DATOS;  public class DAL\_Empleado  {  DAO\_Empleado adoEmpleado = new DAO\_Empleado(); |
|  |
| Ne el fichero solución.aspx.cs  Añadir  using System.Web.UI.WebControls;  using ENTIDADES;  using NEGOCIO;  ESTO POR CADA DAL CREADO  {  DAL\_Cliente ncliente = new DAL\_Cliente();  DAL\_Empleado nEmpleado = new DAL\_Empleado();  DAL\_Producto nProducto = new DAL\_Producto();  DAL\_Pedido nPedido = new DAL\_Pedido();  DAL\_Lineas\_Pedidos nLineas\_Pedido = new DAL\_Lineas\_Pedidos(); |
|  |
|  |
|  |
| RELLENAR DE INICIO UNA DDL |
| protected void Page\_Load(object sender, EventArgs e)  {  if (!IsPostBack)  {  List<Editorial> editoriales = dEditorial.mostrarEditoriales();  foreach (var item in editoriales)  {  ddleditoriales.Items.Add(new ListItem(item.Pub\_id + "-" + item.Pub\_name, item.Pub\_id));  }  }  } |
| METODO DEL DAO |
| public List<Editorial> mostrarEditoriales()  {  List<Editorial> editoriales = new List<Editorial>();  SqlCommand cmd = new SqlCommand("Select pub\_id, pub\_name FROM publishers", cnn);  abrir();  SqlDataReader dr = cmd.ExecuteReader();  while (dr.Read())  {  editoriales.Add(new Editorial(dr[0].ToString(),dr[1].ToString()));  }  dr.Close();  cmd.Dispose();  cerrar();  return editoriales;  } |
| Rellenar una DDLIST en UN PASO |
| protected void ddlAlmacenes\_SelectedIndexChanged(object sender, EventArgs e)  {  //Obtengo el value del almacen selecionado  String idAlmacen = ddlAlmacenes.SelectedValue;  //Llamo al metodo de DAOSale para obtener las ventas  List<Sale> sales = Dsale.getSales(Convert.ToInt32(idAlmacen));    ddlVentas.Items.Clear();  //por cada linea de la lista guardo una linea de la ddl de ventas  foreach(Sale sale in sales)  {  ListItem item = new ListItem(sale.Ord\_num, sale.Ord\_num);  //Rellenao la lista con las ventas  ddlVentas.Items.Add(item);  }  } |
| EN VARIOS PASOS |
| protected void ddleditoriales\_SelectedIndexChanged(object sender, EventArgs e)  {  limpiar();  String idEditorial = ddleditoriales.SelectedValue;  rellenarListadoEmpleados(idEditorial);  } |
| private void rellenarListadoEmpleados(String pub\_id)  {  List<Empleado> empleados = dEmpleado.getEmpleados(pub\_id);  foreach (Empleado e in empleados)  {  ListItem liEmpleado = new ListItem(e.Fname, e.Emp\_id);  lstEmpleados.Items.Add(liEmpleado);  }  } |
| private void limpiar()  {  lstEmpleados.Items.Clear();  lstProfesiones.Items.Clear();  lblEmpleados.Visible = false;  } |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |