C#

bool condition = true;  
  
if (condition)  
{  
 Console.WriteLine("The variable is set to true.");  
}  
else  
{  
 Console.WriteLine("The variable is set to false.");  
}

CONSOLA:

Class HolaMundo{

Static void Main{

System.Console.writeline (“Hola Mundo”);

console.readkey();

}

}

En program.cs se encuentra definido el namespace, la clase program y el método: static void Main()

insertar una **CLASE**:

Definir las variables:

private int A\_numero1;

El método y las variables no pueden tener el mismo nombre porque genera error.

Para definir el método se debe especificar los eventos GET y SET.

class calculo

{

private double \_numero1;

private double \_numero2;

private double \_numero3;

public int numero1 {

get { return A\_numero1; }

set { A\_numero1 = value; }

}

public int suma

{

get { return A\_numero1 + A\_numero2 + A\_numero3; }

}

En el formulario:

public partial class Form1 : Form

{

calculo DOL = new calculo();

En el botón:

private void btnConvertir\_Click(object sender, EventArgs e)

{

DOL.numero1 = Convert.ToInt16(txtDolares1.Text);

DOL.numero2 = Convert.ToInt16(txtDolares2.Text);

DOL.numero3 = Convert.ToInt16(txtDolares3.Text);

lblResultado = Convert.ToString(DOL.suma);

HACER CÁLCULO EN EL CONSTRUCTOR:

namespace AgenciaViajes

{

class Viaje

{

private int \_cantidad;

private int \_valorU;

private Boolean \_Promo;

//private int valor;

public int cantidad

{

get { return \_cantidad; }

set { \_cantidad = value; }

}

public int valorU {

get { return \_valorU ;}

set { \_valorU =value ;}

}

public Boolean Promo {

set { \_Promo =value;}

}

public int valorTotal {

get{int valor=(\_cantidad\*\_valorU);

if(\_Promo==true){

valor=valor-valor\*20/100;

}

return valor;

}

}

}

}

PASAR DE UN FORMULARIO A OTRO:

Form2 frm = new Form2();

this.Hide();

frm.Show();

CONEXIÓN A UNA BASE DE DATOS:

using System.Data.SqlClient; ‘insertar al principio del formulario.

private void btnConectar\_Click(object sender, EventArgs e)

{

string Connstring = "Database=clinica; DataSource=localhost;UserId=root;password=111";

MySqlConnection myConexion=new MySqlConnection(Connstring);

MySqlConnection Conection = new MySqlConnection(Connstring);

MySqlCommand Command = Conection.CreateCommand();

MySqlDataReader reader;

Command.CommandText = "Select \* from tbEmpleados";

Conection.Open();

reader = Command.ExecuteReader();

while (reader.Read())

{

comboBox1.Items .Add (reader ["NomEmpleado"].ToString ());

comboBox2.Items.Add(reader["apeEmpleado"].ToString());

comboBox3 .Items .Add (reader["dirEmpleado"].ToString ());

}

}

LLENAR UN DATAGRID CON INFORMACIÓN DE UNA TABLA:

private void btnCargar\_Click(object sender, EventArgs e)

{

server = "localhost";

database = "clinica";

uid = "root";

password = "111";

string connectionString;

connectionString = "server="+server +";"+"DataBase="+database +";"+"uid="+uid +";"+"PASSWORD="+password +";";

Connection = new MySqlConnection(connectionString);

if (this.openConnection() == true) {

MySqlDataAdapter adp = new MySqlDataAdapter("Select \* from tbEmpleados", Connection);

DataSet DS = new DataSet();

adp.Fill(DS);

dataGridView1.DataSource = DS.Tables[0];

this.CloseConnection();

}

private bool openConnection()

{

try {

Connection.Open();

return true;

}catch(MySqlException ex){

switch (ex.Number ){

case 0:

MessageBox.Show ("No hay conexion con el servidor");

break;

case 1:

MessageBox.Show ("Invalidado el usuario, intente de nuevo");

break;

default:

MessageBox.Show (ex.Message );

break ;

}

return false;

}

}

private bool CloseConnection() {

try {

Connection.Close();

return true;

}catch (MySqlException ex){

MessageBox.Show(ex.Message );

return false;

}

}

ABRIR IMAGEN:

private void btnAbrir\_Imagen\_Click(object sender, EventArgs e)

{

OpenFileDialog Dialog = new OpenFileDialog();

DialogResult Result;

Result= Dialog.ShowDialog ();

if (Result == DialogResult.OK) {

pictureBox1.Image = Image.FromFile(Dialog.FileName);

}

}

PROGRESS BAR:

private void btnBarra\_Click(object sender, EventArgs e)

{

int inicio = 0;

int fin = 200000;

progressBar1.Minimum = inicio;

progressBar1.Maximum = fin;

for (int i = inicio; i < fin; i++)

this.progressBar1.Value = i;

}

LOG IN USUARIO Y PASSWORD (CON ACCESO A BASE DE DATOS):

private void btnACEPTAR\_Click(object sender, EventArgs e)

{

string connstring = "Database=clinica;Datasource=localhost;UserId=root;Password=111";

MySqlConnection myConexion = new MySqlConnection(connstring);

myConexion.Open();

string query = "SELECT COUNT(\*) As Cant FROM usuarios where usuario=?usuario AND password=?pass";

MySqlCommand myComando = new MySqlCommand(query, myConexion);

myComando.Parameters.Add("?usuario", textBoxUsuario.Text);

myComando.Parameters.Add("?pass", textBoxPass.Text);

int cant = System.Convert.ToInt32(myComando.ExecuteScalar());

if (cant > 0)

{

Form3 a = new Form3();

a.ShowDialog();

this.Hide();

}

else {

Form4 b = new Form4();

b.ShowDialog();

this.Hide();

}

}

TIMER QUE ACTUALIZA UN PROGRESSBAR:

private void timer1\_Tick(object sender, EventArgs e)

{

progressBar1.Value++;

if (progressBar1.Value == 100) {

timer1.Stop();

Dispose();

}

if (R1.Checked) {

label1.Text =" "+R1.Text;

}else

{label1.Text =" "+R2.Text ;}

}

CONEXIÓN CON SQL SERVER:

Insertar un objeto ADO.NET Entity Data Model (Es una clase que manejará la conexión con la Base de Datos). Llamado ADO2

En el using del nuevo elemento insertar:

using System.Data.SqlClient;

namespace WindowsFormsApp1

{

using System;

using System.Data.Entity;

using System.Linq;

using System.Data.SqlClient;

using System.Collections.Generic;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

public class ADO2 {

string cadena = "Data Source=CARLOS-PC;Initial Catalog=model;Integrated Security=true";

public SqlConnection MyConnexion = new SqlConnection();

public void conectar() {

MyConnexion.ConnectionString = cadena;

MessageBox.Show("Conectado"); //Requiere: using System.Windows.Forms;

}

public string abrir() {

try

{

MyConnexion.Open();

return "Conexion Abierta";

}

catch (Exception ex){

Console.WriteLine ("Error al abrir la BD "+ ex.Message);

return "Problema de conexion";

}

}

public void cerrar() {

MyConnexion.Close();

}

}

}

private void button1\_Click(object sender, EventArgs e)

{

string mensaje;

ADO2 conexion = new ADO2();

conexion.conectar();

mensaje = conexion.abrir();

conexion.cerrar();

label2.Text = mensaje;

}

MOSTRAR UN CUADRO DE DIÁLOGO DE MENSAJE:

using System.Windows.Forms;

MessageBox.Show("Error en La instrucción insertar");

private void button2\_Click(object sender, EventArgs e)

{

conexion.conectar();

conexion.abrir();

DataTable dt = new DataTable();

dt = conexion.abrirTabla();

dgv.DataSource = dt;

conexion.cerrar();

}

//en el ADO2.cs

public DataTable abrirTabla(){

dt = new DataTable();

da = new SqlDataAdapter("Select \* from usuarios",MyConnexion);

da.Fill(dt);

return dt;

}

TIPOS DE DATOS C#

|  |  |  |
| --- | --- | --- |
| **Tipo** | **Intervalo** | **Tamaño** |
| [sbyte](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/sbyte) | De -128 a 127 | Entero de 8 bits con signo |
| [byte](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/byte) | De 0 a 255 | Entero de 8 bits sin signo |
| [char](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/char) | U+0000 a U+ffff | Carácter Unicode de 16 bits |
| [short](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/short) | De -32 768 a 32 767 | Entero de 16 bits con signo |
| [ushort](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/ushort) | De 0 a 65.535 | Entero de 16 bits sin signo |
| [int](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/int) | De -2.147.483.648 a 2.147.483.647 | Entero de 32 bits con signo |
| [uint](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/uint) | De 0 a 4.294.967.295 | Entero de 32 bits sin signo |
| [long](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/long) | De -9.223.372.036.854.775.808 a 9.223.372.036.854.775.807 | Entero de 64 bits con signo |
| [ulong](https://docs.microsoft.com/es-es/dotnet/csharp/language-reference/keywords/ulong) | De 0 a 18.446.744.073.709.551.615 | Entero de 64 bits sin signo |

**WEB SERVICE (C#)**

Brandan Jones

<https://www.youtube.com/channel/UCYZCPdpz0ClWyGJWd1dZsug>

WEB SERVICES GRATIS:

<http://www.webservicex.net:500/Prices?ticker=BASMNYARA189N>

<http://www.plantplaces.com/perl/mobile/viewplantsjson.pl?Combined_Name=Oak>

Nuevo proyecto: C# -> Service Application WCF

public class TemperatureConverter : System.Web.Services.WebService

{

[WebMethod]

public double FarenheitToCelsius(double f)

{

return (f-32)\*5/9;

}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

APLICACIÓN WEB CON JSON

Nuevo proyecto: C# -> Service Application WCF

public class CommonToGenus : System.Web.Services.WebService

{

[WebMethod]

public string ConvertCommonToGenus(string common)

{

string returnValue = "No genus found";

using (var webClient = new WebClient())

{

string rawData= webClient.DownloadString("http://www.plantplaces.com/perl/mobile/viewplantsjson.pl?Combined\_Name="+common);

}

JsonConvert.DeserializeObject<>

return returnValue;

}

}

->Instalar paquete nuget -> Newtonsoft.json

<http://jsonviewer.stack.hu/>

En el navegador -> view-source:<http://www.plantplaces.com/perl/mobile/viewplantsjson.pl?Combined_Name=Oak>

->Seleccionar y Copiar el contenido

->Ingresar en <http://jsonviewer.stack.hu> y pegarlo. En la pestaña VIEWER se ve la estructura JSON.

**MENU AYUDA C#: ALT + ENTER**

-> Aplicacion web ASP.NET

LLenar el título de la aplicación:

<title></title>

Entre las etiquetas <div></div>

Insertar una tabla y entre las etiquetas <td></td> insertar Label y TextBox

Insertar un Botón (Button1):

protected void Button1\_Click(object sender, EventArgs e)

{

Response.Write("<script>alert('Bienvenido "+TextBox1.Text+"')</script>");

}

<https://andrestrianalopez.wordpress.com/2012/06/16/realizando-una-aplicacin-web-con-conexin-a-base-de-datos-sql-server/>

APLICACIÓN WEB C# CON BASE DE DATOS

agregar un Folder de ASP.NET “**App\_Data**

Con las siguientes instrucciones se configura un ConnectionString global en el archivo Web.config.

En el folder: “App\_Data”, deberemos adicionar un Nuevo Item de tipo “Web Form” que llamaremos “Default.aspx. No deja ingresar en la carpeta App\_Data. Pero si afuera de ella.

En el folder: “App\_Data”, deberemos adicionar un Nuevo Item C# -> Data -> Base de datos SQL Server.

En el menú Ver -> Explorador de Servidores se crea una base de datos y una tabla.

abriremos el archivo “**Web.config**”

En primer lugar presionaremos Doble Click sobre el botón Guardar e incluiremos el siguiente código:

a) Se adicionaran los siguientes Espacios de Nombres:

using System.Data; //Utilizado para realizar Operaciones DML con la Base de Datos SQL Server

using System.Data.SqlClient; //Utilizado para realizar Operaciones DML con la Base de Datos SQL Server

using System.Configuration; //Utilizada para extraer la cadena de conexion de nuestro archivo Web.config y poder usarla en nuestra aplicación

b) Crearemos la variable sqlConn de tipo SqlConnection y a ésta le asignaremos la cadena de conexión que creamos en el archivo Web.config. Esto lo hacemos a través de la instrucción (ConfigurationManager.ConnectionStrings["CF"].ConnectionString) donde “CF” es el nombre que le dimos a la conexión. La instrucción quedaría de la siguiente manera:

var sqlConn = newSqlConnection(ConfigurationManager.ConnectionStrings["CF"].ConnectionString);

c) Luego creamos la variable strSQL donde ingresaremos la sentencia SQL a ejecutar, en nuestro caso, la sentencia INSERT INTO. Dentro de la sentencia vincularemos los Textbox para insertar la información que el usuario ingrese, de la siguiente manera:

var strSQL = "INSERT INTO tbl\_Personas VALUES (‘"+txtCedula.Text+"’,’"+txtNombre.Text+"’,’"+txtApellidos.Text+"’,’"+txtCorreo.Text+"’)";

d) Seguido a esto se crea la variable cmd de tipo SqlCommand y como argumentos le pasaremos la sentencia a ejecutar (strSQL) y la conexión a utilizar (sqlConn).

var cmd = new SqlCommand(strSQL, sqlConn);

e) Abrimos la conexion: sqlConn.Open();

f) Ejecutamos el comando a través del método ExecuteNonQuery: cmd.ExecuteNonQuery();

g) Cerramos la conexión: sqlConn.Close();

h) El código completo quedaría así:



Con esto ya tenemos listo le código para el botón Insertar.

Luego se creará el código para **Consultar** registros al invocar el evento Click del botón Consultar una vez hallamos digitado la Cedula, para esto presionamos Doble Click sobre el botón Consultar e incluiremos el siguiente código:

a) Crearemos la variable sqlConn de tipo SqlConnection y a ésta le asignaremos la cadena de conexión que creamos en el archivo Web.config. Esto lo hacemos a través de la instrucción (ConfigurationManager.ConnectionStrings["CF"].ConnectionString) donde “CF” es el nombre que le dimos a la conexión. La instrucción quedaría de la siguiente manera:

var sqlConn = newSqlConnection(ConfigurationManager.ConnectionStrings["CF"].ConnectionString);

b) Luego creamos la variable strSQL donde ingresaremos la sentencia SQL a ejecutar, en nuestro caso, la sentencia SELECT \* FROM. Dentro de la sentencia vincularemos El Textbox la Cedula (txtCedula) para enviárselo como parámetro a la instrucción SQL, de la siguiente manera:

var strSQL = "SELECT \* FROM tbl\_Personas WHERE Cedula = ‘" + txtCedula.Text + "’";

c) Seguido a esto se crea la variable cmd de tipo SqlCommand y como argumentos le pasaremos la sentencia a ejecutar (strSQL) y la conexión a utilizar (sqlConn).

var cmd = new SqlCommand(strSQL, sqlConn);

d) Se crea una variable llamada “ds” de tipo DataSet la cual utilizaremos para almacenar el resultado de nuestra consulta.

var ds = new DataSet();

e) Se crea la variable llamada “da” de tipo SqlDataAdapter la cual se encargará de llenar el DataSet que creamos anteriormente y le enviamos como Argumento el comando a ejecutar.

var da = new SqlDataAdapter(cmd);

f) Abrimos la conexion: sqlConn.Open();

g) A partir de SqlDataAdapter creado “da” hacemos uso de su propiedad Fill para llenar el DataSet con el resultado de la ejecución del comando “cmd”. A esta instrucción le enviamos 2 argumentos (el DataSet a cargar y el nombre de la Tabla que se utilizará)

da.Fill(ds,”tblUsuarios”);

h) Cerramos la conexión: sqlConn.Close();

i) Establecemos una variable de tipo DataTable “dt”, de esta manera podremos extraer la información contenida en el DataSet. Para esto, a través de la propiedad Rows, indicamos la fila y la columna donde se encuentra la información de deseamos extraer. Ejemplo: En la tabla tbl\_Personas, el primer campo es la Cedula, para acceder a esta podemos decirle que de la Fila [0] extraiga la columna [0]

var dt=ds.tables[0];

txtcedula.text=dt.Rows[0][0].ToString();

En el WebForm:

protected void Button2\_Click(object sender, EventArgs e)

{

var sqlConn = new SqlConnection(ConfigurationManager.ConnectionStrings["BDmodel"].ConnectionString);

//var sqlConn = new SqlConnection("Data Source=CARLOS-PC;Initial Catalog=model;Integrated Security=True");

var strSQL = "INSERT INTO usuarios(usuario,password,correo) VALUES('"+TextBox1.Text+"','"+TextBox2.Text+"','"+TextBox3.Text+"')";

SqlCommand cmd = new SqlCommand(strSQL, sqlConn);

sqlConn.Open();

cmd.ExecuteNonQuery();

sqlConn.Close();

}

EN web.config:

<connectionStrings>

<add name="BDmodel" connectionString="Data Source=CARLOS-PC;Initial Catalog=model;Integrated Security=True"/>

</connectionStrings>

ConfigurationManager does not exist in the current context

1. Right-click on the *References / Dependencies*
2. Choose *Add Reference*
3. Find and add System.Configuration.

CREAR Y USAR UN SERVICIO WEB:

<https://msdn.microsoft.com/es-es/library/8wbhsy70(v=vs.100).aspx>

## Utilizar el servicio Web

Ahora que tiene un servicio Web, creará un sitio Web donde hará referencia al servicio que ha creado y lo utilizará. En el tutorial, creará un sitio Web independiente que tiene una página donde podrá iniciar los métodos del servicio Web que acaba de crear.

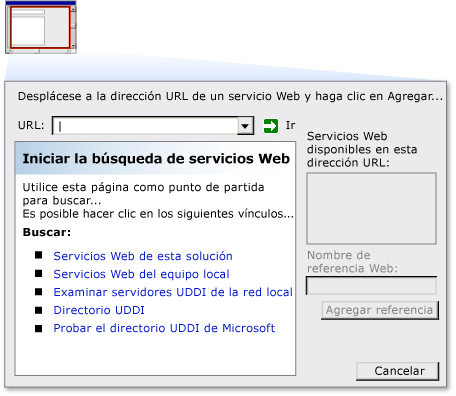
### Para crear un sitio Web para utilizar el servicio Web

1. En el menú **Archivo**, haga clic en **Nuevo sitio Web**.
2. En **Plantillas instaladas de Visual Studio**, haga clic en **Sitio web ASP.NET**.
3. Haga clic en **Examinar**.
4. Haga clic en **IIS local**.
5. Haga clic en **Sitio Web predeterminado**.
6. Haga clic en **Crear nueva aplicación Web**.
7. Visual Web Developer crea una nueva aplicación Web de IIS.
8. Escriba el nombre **TemperatureWeb**.
9. Haga clic en **Abrir**.
10. En la lista **Lenguaje**, haga clic en el lenguaje de programación con el que prefiera trabajar.
11. Haga clic en **Aceptar**.
12. Visual Web Developer crea un nuevo sitio Web de IIS local y una nueva página denominada Default.aspx.

## Agregar el servicio Web como un componente

El servicio Web es un componente al que puede hacer referencia en su aplicación. Por consiguiente, debe crear una referencia a él.

### Para crear una referencia al servicio Web

1. En el menú **Sitio web**, haga clic en **Agregarreferencia web**.
2. Aparecerá el cuadro de diálogo **Agregar referencia Web**, tal y como se muestra en la siguiente captura de pantalla.
3. Agregar referencia Web (Cuadro de diálogo)
4. 
5. En la lista **Dirección URL**, escriba la dirección URL siguiente para el servicio Web y, a continuación, haga clic en **Ir**:
6. **http://localhost/TemperatureWebService/Convert.asmx**
7. Cuando Visual Web Developer encuentra el servicio Web, aparece información sobre el servicio Web en el cuadro de diálogo **Agregar referencias Web**.

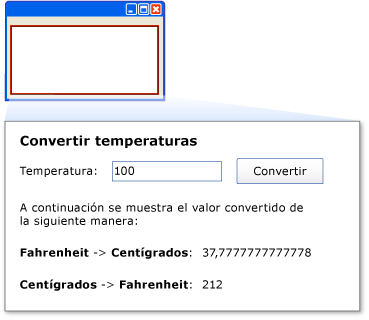
|  |
| --- |
| 1. Nota**Nota** |
| 1. Si no puede agregar una referencia a un servicio Web, podría deberse a que el servidor proxy no está configurado correctamente. En Microsoft Internet Explorer, en el menú **Herramientas**, haga clic en **Opciones de Internet**, haga clic en **Conexiones** y, a continuación, en **Configuración LAN**. Active la casilla **No usar servidor proxy para direcciones locales**. Además, establezca la dirección del servidor proxy en el nombre exacto del servidor proxy en lugar de permitir que Internet Explorer lo detecte. Para obtener más información, póngase en contacto con el administrador de la red. |

1. Haga clic en uno de los vínculos de método.
2. Aparece la página de prueba para el método.
3. Haga clic en **Agregar referencia**.
4. Visual Web Developer crea una carpeta App\_WebReferences y le agrega una carpeta para la nueva referencia Web. De manera predeterminada, se asigna un espacio de nombres a las referencias Web correspondiente al nombre de su servidor (en este caso, **localhost**).Anote el nombre del espacio de nombres de la referencia Web. En la carpeta, Visual Web Developer agrega un archivo .wsdl que hace referencia al servicio Web. También agrega archivos de compatibilidad, como archivos de descubrimiento (.disco y .discomap), que incluyen información sobre la ubicación del servicio Web.

|  |
| --- |
| Nota**Nota** |
| Si el nombre de servidor del servicio Web contiene caracteres que no se pueden utilizar para un nombre de clase, por ejemplo un guión (-), Visual Web Developer los convierte en un carácter de subrayado (\_). Por consiguiente, el espacio de nombres de Visual Web Developer para el servicio Web podría no coincidir exactamente con el nombre del servidor. |

Ahora puede utilizar el servicio Web. En este tutorial, agregará controles a Default.aspx y, a continuación, programará los controles para convertir una temperatura especificada en Fahrenheit o Celsius. Cuando ejecute la página, tendrá la apariencia siguiente:

Página de conversión de temperatura



### Para llamar a los métodos del servicio Web

1. Abra la página Default.aspx y cambie a la vista Diseño.
2. En el grupo **Estándar** del Cuadro de herramientas, arrastre los controles siguientes a la página y establezca sus propiedades como se indica:

|  |  |
| --- | --- |
| 1. **Control** | 1. **Propiedades** |
| 1. **Textbox** | 1. **ID**: TemperatureTextbox 2. **Text**: (vacío) |
| 1. **Button** | 1. **ID**: ConvertButton 2. **Text**: Convertir |
| 1. **Label** | 1. **ID**: FahrenheitLabel 2. **Text**: (vacío) |
| 1. **Label** | 1. **ID**: CelsiusLabel 2. **Text**: (vacío) |

1. Opcionalmente, agregue texto a la página para los títulos.
2. En este tutorial, el diseño de la página no es importante.
3. Haga doble clic en **ConvertButton** a fin de crear un controlador de eventos para el evento **Click**.
4. Asegúrese de que el código del controlador de eventos coincida con el código del siguiente ejemplo.
5. **C#**
6. **[VB](https://msdn.microsoft.com/es-es/library/8wbhsy70(v=vs.100).aspx?cs-save-lang=1&cs-lang=vb#code-snippet-2)**
7. protected void ConvertButton\_Click(object sender, EventArgs e)  
   {  
    localhost.Convert wsConvert = new localhost.Convert();  
    double temperature =  
    System.Convert.ToDouble(TemperatureTextbox.Text);  
    FahrenheitLabel.Text = "Fahrenheit To Celsius = " +  
    wsConvert.FahrenheitToCelsius(temperature).ToString();  
    CelsiusLabel.Text = "Celsius To Fahrenheit = " +  
    wsConvert.CelsiusToFahrenheit(temperature).ToString();  
   }
8. Presione CTRL+F5 para ejecutar la página.
9. En el cuadro de texto, escriba un valor, por ejemplo 100, y a continuación, haga clic en **Convertir**.
10. La página presenta el resultado de convertir el valor de temperatura en Fahrenheit o Celsius.

CÓDIGO HTML DE UNA WEB FORM:

//Para resaltar un div:

<div class="jumbotron">

<h1>ASP.NET</h1>

<p class="lead">ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS, and JavaScript.</p>

<p><a href="http://www.asp.net" class="btn btn-primary btn-lg">Learn more &raquo;</a></p>

</div>

<div class="row"> //otra fila

<div class="col-md-4"> /Se va a dividir en 3 sectores verticales

RUTA GLOBAL DE UN CONTROL EN ASP:

System.Web.UI.WebControls.TextBox TextBox1;

OTRA FORMA DE VER UNA TABLA EN UN GRIDVIEW:

1. Insertar el GridView en el formulario
2. Insertar un SqlDataSource
3. In el SqlDataSource configurar el origen de Datos
4. Actualizar esquemas
5. Editar Columnas -> CommandField ->Editar, actualizar, Cancelar.
6. Insertar en la propiedad Update Query del SQLDataSource->

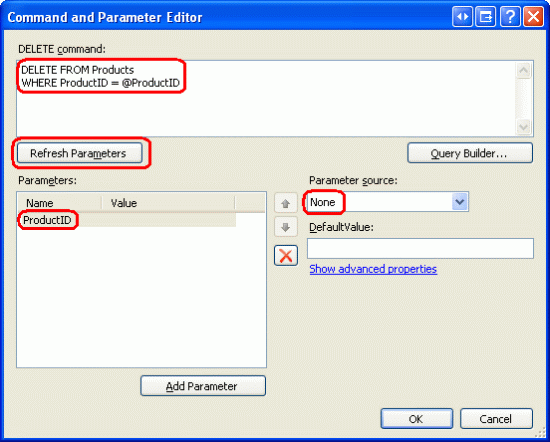
UPDATE usuarios SET password = @password, correo = @correo WHERE usuario=@usuario;

1. De manera similar se puede insertar una columna para eliminar en el GridView
2. En el SqlDataSource en la propiedad DeleteQuery insertar:

DELETE FROM usuarios WHERE usuario=@usuario

1. Hacer clic en Actualizar Parámetros.

El parámetro seleccionado debe coincidir con la clave principal de la tabla.



Página Maestra:

De esta forma se especifica la página maestra en la primera línea de cada página:

<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="WebApplication2.\_Default" %>

Observar:

MasterPageFile="~/Site.Master"

rest api

WEB API

(Application Programing Interface)

<http://www.tutorialsteacher.com/webapi/create-web-api-project>

API is some kind of interface which has a set of functions that allow programmers to access specific features or data of an application, operating system or other services.

Web API as the name suggests, is an API over the web which can be accessed using HTTP protocol. It is a concept and not a technology. We can build Web API using different technologies such as Java, .NET etc. For example, Twitter's [REST APIs](https://dev.twitter.com/rest/public) provide programmatic access to read and write data using which we can integrate twitter's capabilities into our own application.

## ASP.NET Web API

The ASP.NET Web API is an extensible framework for building HTTP based services that can be accessed in different applications on different platforms such as web, windows, mobile etc. It works more or less the same way as ASP.NET MVC web application except that it sends data as a response instead of html view. It is like a webservice or WCF service but the exception is that it only supports HTTP protocol.

1. ASP.NET Web API supports different formats of response data. Built-in support for JSON, XML, BSON format.

* Choose WCF if your service needs to support multiple protocols such as HTTP, TCP, Named pipe.
* Choose WCF if you use .NET Framework 3.5. Web API does not support .NET 3.5 or below.
* Choose Web API if you are using .NET framework 4.0 or above.
* Choose Web API if you want to build a service that supports only HTTP protocol.
* Choose Web API to build RESTful HTTP based services.

JSON stands for JavaScript Object Notation.

http://www.tutorialsteacher.com/webapi/create-web-api-project

<https://www.tutorialspoint.com/json/json_overview.htm>

JSON or JavaScript Object Notation is a lightweight text-based open standard designed for human-readable data interchange. Conventions used by JSON are known to programmers, which include C, C++, Java, Python, Perl, etc.

* It is used while writing JavaScript based applications that includes browser extensions and websites.
* JSON format is used for serializing and transmitting structured data over network connection.
* It is primarily used to transmit data between a server and web applications.
* Web services and APIs use JSON format to provide public data.

we will try another example. Let's save the below code as json.htm −

<html>  
 <head>  
 <title>JSON example</title>  
   
 <script language = "javascript" >  
   
 var object1 = { "language" : "Java", "author" : "herbert schildt" };  
 document.write("<h1>JSON with JavaScript example</h1>");  
 document.write("<br>");  
 document.write("<h3>Language = " + object1.language+"</h3>");   
 document.write("<h3>Author = " + object1.author+"</h3>");   
  
 var object2 = { "language" : "C++", "author" : "E-Balagurusamy" };  
 document.write("<br>");  
 document.write("<h3>Language = " + object2.language+"</h3>");   
 document.write("<h3>Author = " + object2.author+"</h3>");   
   
 document.write("<hr />");  
 document.write(object2.language + " programming language can be studied " + "from book written by " + object2.author);  
 document.write("<hr />");  
   
 </script>  
   
 </head>  
   
 <body>  
 </body>  
   
</html>

Below is a simple example −

{  
 "book": [  
  
 {  
 "id": "01",  
 "language": "Java",  
 "edition": "third",  
 "author": "Herbert Schildt"  
 },  
  
 {  
 "id": "07",  
 "language": "C++",  
 "edition": "second",  
 "author": "E.Balagurusamy"  
 }  
  
 ]  
}

JSON supports the following two data structures −

* Collection of name/value pairs − This Data Structure is supported by different programming languages.
* Ordered list of values − It includes array, list, vector or sequence etc.

JSON format supports the following data types −

|  |  |
| --- | --- |
| **S. No.** | **Type & Description** |
| 1 | Number  double- precision floating-point format in JavaScript |
| 2 | String  double-quoted Unicode with backslash escaping |
| 3 | Boolean  true or false |
| 4 | Array  an ordered sequence of values |
| 5 | Value  it can be a string, a number, true or false, null etc |
| 6 | Object  an unordered collection of key:value pairs |
| 7 | Whitespace  can be used between any pair of tokens |
| 8 | null  empty |

### Syntax

var json-object-name = { string : number\_value, .......}

### Example

Example showing Number Datatype, value should not be quoted −

var obj = {marks: 97}

The table shows various special characters that you can use in strings of a JSON document −

|  |  |
| --- | --- |
| **S. No.** | **Type & Description** |
| 1 | "  double quotation |
| 2 | \  backslash |
| 3 | /  forward slash |
| 4 | b  backspace |
| 5 | f  form feed |
| 6 | n  new line |
| 7 | r  carriage return |
| 8 | t  horizontal tab |
| 9 | u  four hexadecimal digits |

## Boolean

It includes true or false values.

### Syntax

var json-object-name = { string : true/false, .......}

### Example

var obj = {name: 'Amit', marks: 97, distinction: true}

## Array

* It is an ordered collection of values.
* These are enclosed in square brackets which means that array begins with .[. and ends with .]..
* The values are separated by , (comma).
* Array indexing can be started at 0 or 1.
* Arrays should be used when the key names are sequential integers.

### Syntax

[ value, .......]

### Example

Example showing Object −

{  
 "id": "011A",  
 "language": "JAVA",  
 "price": 500,  
}

This is an example that shows creation of an object in javascript using JSON, save the below code as json\_object.htm −

<html>  
 <head>  
 <title>Creating Object JSON with JavaScript</title>  
   
 <script language = "javascript" >  
  
 var JSONObj = { "name" : "tutorialspoint.com", "year" : 2005 };  
   
 document.write("<h1>JSON with JavaScript example</h1>");  
 document.write("<br>");  
 document.write("<h3>Website Name = "+JSONObj.name+"</h3>");   
 document.write("<h3>Year = "+JSONObj.year+"</h3>");   
  
 </script>  
   
 </head>  
   
 <body>  
 </body>  
   
</html>

## **JSON vs XML**

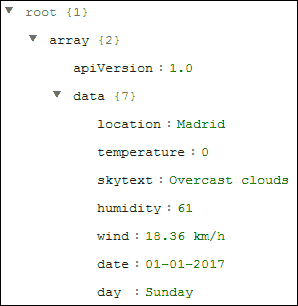
Algunas ventajas por las cuales se ha popularizado mucho JSON frente a su principal formato competidor hasta el momento (XML) son:

* El uso de una sintaxis más reducida, lo que implica intercambio de datos más pequeños
* La facilidad para poder ser leído por una persona
* La sencillez de su generación (y uso) utilizando tecnologías web

## **Buscando un proveedor de datos**

Para intentar que el artículo te resulte lo más útil posible si necesitas conectar tu aplicación .NET a una API que devuelva JSON, vamos a realizar un ejemplo realista paso a paso.

Para ello hemos seleccionado la API [Weathers](http://weathers.co/api) como base para el ejemplo que veremos a continuación. Esta API, como su propio nombre indica, nos va a proporcionar información meteorológica de la ciudad que le indiquemos (Madrid en nuestro ejemplo).

El código para llamar a la API consiste en realizar una petición GET a la URL http://weathers.co/api.php?city=Madrid. Como resultado nos devuelve la siguiente información en formato JSON (en la figura la vemos desplegada):

Esta es la API de clima de YAHOO:

[https://query.yahooapis.com/v1/public/yql?q=select%20\*%20from%20weather.forecast%20where%20woeid%20in%20(select%20woeid%20from%20geo.places(1)%20where%20text%3D%22nome%2C%20ak%22)&format=json&env=store%3A%2F%2Fdatatables.org%2Falltableswithkeys](https://query.yahooapis.com/v1/public/yql?q=select%20*%20from%20weather.forecast%20where%20woeid%20in%20(select%20woeid%20from%20geo.places(1)%20where%20text%3D%22nome%2C%20ak%22)&format=json&env=store%3A%2F%2Fdatatables.org%2Falltableswithkeys)

Y EN <http://json2csharp.com/> ARROJA:

public class Units  
{  
 public string distance { get; set; }  
 public string pressure { get; set; }  
 public string speed { get; set; }  
 public string temperature { get; set; }  
}  
  
public class Location  
{  
 public string city { get; set; }  
 public string country { get; set; }  
 public string region { get; set; }  
}  
  
public class Wind  
{  
 public string chill { get; set; }  
 public string direction { get; set; }  
 public string speed { get; set; }  
}  
  
public class Atmosphere  
{  
 public string humidity { get; set; }  
 public string pressure { get; set; }  
 public string rising { get; set; }  
 public string visibility { get; set; }  
}  
  
public class Astronomy  
{  
 public string sunrise { get; set; }  
 public string sunset { get; set; }  
}  
  
public class Image  
{  
 public string title { get; set; }  
 public string width { get; set; }  
 public string height { get; set; }  
 public string link { get; set; }  
 public string url { get; set; }  
}  
  
public class Condition  
{  
 public string code { get; set; }  
 public string date { get; set; }  
 public string temp { get; set; }  
 public string text { get; set; }  
}  
  
public class Forecast  
{  
 public string code { get; set; }  
 public string date { get; set; }  
 public string day { get; set; }  
 public string high { get; set; }  
 public string low { get; set; }  
 public string text { get; set; }  
}  
  
public class Guid  
{  
 public string isPermaLink { get; set; }  
}  
  
public class Item  
{  
 public string title { get; set; }  
 public string lat { get; set; }  
 public string @long { get; set; }  
 public string link { get; set; }  
 public string pubDate { get; set; }  
 public Condition condition { get; set; }  
 public List<Forecast> forecast { get; set; }  
 public string description { get; set; }  
 public Guid guid { get; set; }  
}  
  
public class Channel  
{  
 public Units units { get; set; }  
 public string title { get; set; }  
 public string link { get; set; }  
 public string description { get; set; }  
 public string language { get; set; }  
 public string lastBuildDate { get; set; }  
 public string ttl { get; set; }  
 public Location location { get; set; }  
 public Wind wind { get; set; }  
 public Atmosphere atmosphere { get; set; }  
 public Astronomy astronomy { get; set; }  
 public Image image { get; set; }  
 public Item item { get; set; }  
}  
  
public class Results  
{  
 public Channel channel { get; set; }  
}  
  
public class Query  
{  
 public int count { get; set; }  
 public DateTime created { get; set; }  
 public string lang { get; set; }  
 public Results results { get; set; }  
}  
  
public class RootObject  
{  
 public Query query { get; set; }  
}

Como podemos ver en la imagen anterior, nos informa de cuál es la temperatura en Madrid, la humedad y otros datos climatológicos.

## **Creando la clase Meteorología**

Para tratar en C# el resultado anterior, estaremos de acuerdo en que lo mejor sería almacenar cada uno de los datos como una propiedad en una clase. En este caso he decidido llamar a la clase Meteorologia, pero esta clase no la vamos a crear nosotros a mano, sino que se creará de manera automática siguiendo estos pasos:

1. Vamos a realizar una petición a la API, por ejemplo desde el navegador, para copiar la respuesta que nos da.
2. Vamos a entrar en [json2csharp](http://json2csharp.com/) y vamos a copiar la respuesta de la API. http://json2csharp.com/
3. Le damos al botón de generar, para que nos cree la clase o clases necesarias para mapear cada uno de los campos del JSON, en propiedades de las clases.
4. Copiamos el código de las clases resultantes en Visual Studio, teniendo en cuenta que **renombraremos la clase RootObject, con el nombre que nosotros queramos usar**, en este caso Meteorologia.
5. Yo normalmente renombro las propiedades para que tengan la primera letra en mayúsculas, pero esto es una cuestión de estilo y no es obligatorio.

El código resultante de nuestras clases será, en este caso, el siguiente:

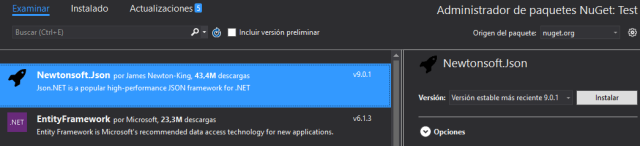
public class Meteorologia  
{  
 public string ApiVersion { get; set; }  
 public Data Data { get; set; }  
}  
  
public class Data  
{  
 public string Location { get; set; }  
 public string Temperature { get; set; }  
 public string Skytext { get; set; }  
 public string Humidity { get; set; }  
 public string Wind { get; set; }  
 public string Date { get; set; }  
 public string Day { get; set; }  
}

## **Realizando la petición y el *parseo* del JSON**

El formato JSON está soportado de forma nativa por C#. Sin embargo, en este caso vamos a utilizar una biblioteca externa denominada [JSON.Net](http://www.newtonsoft.com/json). Ésta ofrece muchas ventajas frente a la nativa, como por ejemplo:

* Es mucho más rápida en el procesado de ficheros JSON. Tanto es así que la propia Microsoft la utiliza en sus desarrollos.
* Es muy sencilla de usar.
* Permite *serializar* y *deserializar* cualquier objeto de .Net desde JSON y viceversa.

Para añadir esta biblioteca a nuestro proyecto, lo podemos hacer bajando el archivo desde su página o bien usando el administrador de paquetes de NuGet:



Finalmente realizamos la petición GET para recibir la información en JSON y usamos el método JsonConvert.DeserializeObject<>() para *deserializar* el JSON en nuestra clase, como se ve a continuación:

Meteorologia meteorologia;  
HttpWebRequest request = (HttpWebRequest)WebRequest.Create(@"http://weathers.co/api.php?city=Madrid");  
using (HttpWebResponse response = (HttpWebResponse)request.GetResponse())  
using (Stream stream = response.GetResponseStream())  
using (StreamReader reader = new StreamReader(stream))  
{  
 var json = reader.ReadToEnd();  
 meteorologia = JsonConvert.DeserializeObject<Meteorologia>(json);  
}  
Console.WriteLine("La temperatura en Madrid es: " + meteorologia.Data.Temperature);

Espero que este artículo te parezca interesante y te ayude a incorporar servicios web que utilizan JSON en tus desarrollos.

**TUTORIAL CREACIÓN DE UNA API:**

clic derecho en la carpeta Controllers -> Insertar Nuevo elemento scaffolder

Web API Controller with actions using Entity Framework

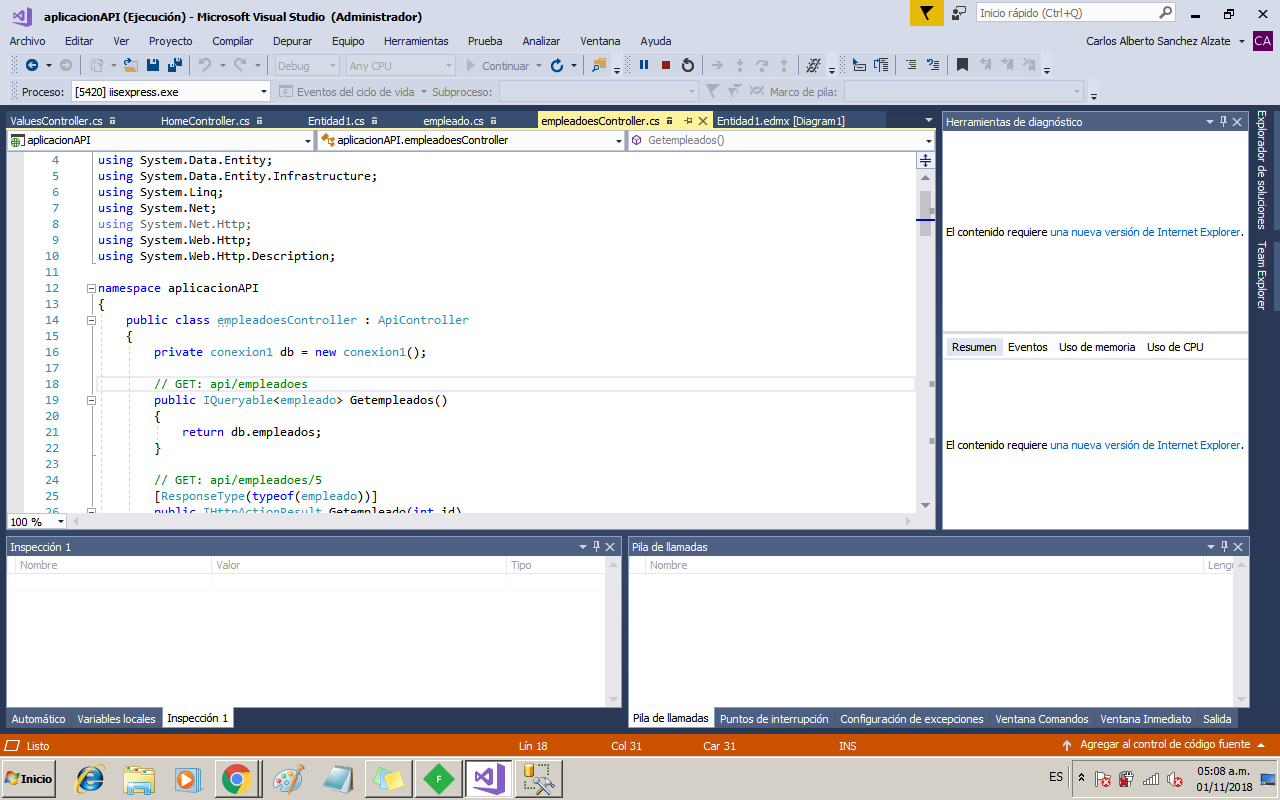
Las llamadas se realizarían a api/empleadoes

Elegir contexto: Empleados

Model class: empleado

Controller Name: EmpleadosController

El sistema crea automáticamente los métodos GET, POST, PUT, DELETE



Instalar Fiddler para realizar pruebas

Abrir Fiddler y seleccionar composer.

Ejecutar la API (F5) desde el visual studio

En fidler seleccionar la mira, arrastrarla y colocarla sobre la página

Seleccionar la dirección del navegador (incluyendo el puerto) y pegarla sobre la barra de fiddler en la pestaña Parsed y complementarla con /api/empleadoes (PESTAÑA COMPOSER)

Hacer clic en Execute (está seleccionado GET por defecto)

Para que esto funcione, debe estar corriendo el SQL SERVER

Para ver los resultados vamos a la pestaña Inspector y allí podemos ver JSON o RAW (Objeto en formato JSON)

Para actualizar:

colocamos el objeto con los datos actualizados en formato JSON en Request Body

Seleccionamos PUT

Insertamos Content-type: application/json en: Composer ->Parsed

Ejecutamos

Error: Cannot deserialize the current JSON array (e.g. [1,2,3]) into type 'aplicacionAPI.empleado' because the type requires a JSON object (e.g. {"name":"value"}) to deserialize correctly.

To fix this error either change the JSON to a JSON object (e.g. {"name":"value"}) or change the deserialized type to an array or a type that implements a collection interface (e.g. ICollection, IList) like List<T> that can be deserialized from a JSON array. JsonArrayAttribute can also be added to the type to force it to deserialize from a JSON array.

Path '', line 1, position 1.

**habilitación de CORS en mi API**.

permitir que se hagan llamadas AJAX desde JavaScript, desde direcciones (dominios) diferentes a la del servidor

La forma más sencilla de habilitar CORS es utilizando **las capacidades de NuGet para localizar e instalar el código necesario**.

Clic derecho en la solución ->Administrar paquetes Nuget

Examinar -> cors en Actualizar y seleccionar:

Microsoft.AspNet.WebApi.Cors y seleccionar todos los proyectos

abro la carpeta *App\_Start* y edito el fichero *WebApiConfig.cs*.

Inserto estas líneas:

//Habilitando CORS

var UrlPermitidas = new EnableCorsAttribute("\*", "\*", "\*");

config.EnableCors(UrlPermitidas);

Se necesita este using:

using System.Web.Http.Cors;

En la segunda línea estoy habilitando las comunicaciones Cross Domain, y en la primera línea le indico **que le dé acceso a cualquier dominio de origen** (el primer asterísco). Aquí podría indicar cada uno de los dominios permitido, separandolos por una coma ("www.midominio1.com, www.midominio2.es").

Y con esto **ya tengo lista mi API para ser consumida desde cualquier punto de Internet**.

///////////////////////////////////////////////////////////////////////////////////////////////

///////////////////////////////////////////////////////////////////////////////////////////////

Fuente Libro: Pro ASP NET MVC

**WEB API DE UNA SOLA PÁGINA:**

[SI FUNCIONA]

I am going to rely on three NuGet packages in this chapter: jQuery, Bootstrap and Knockout

Knockout is the library that Microsoft has adopted for single-page applications. It

install the NuGet packages. Select Package Manager Console from the

Visual Studio Tools ➤ Library Package Manager menu and enter the following commands:

Install-Package jquery –version 1.10.2

Install-Package bootstrap –version 3.0.0

Install-Package knockoutjs –version 3.0.0

1. Crear un PROYECTO: ASP.NET project

-> Seleccionar plantilla vacía -> chuliar Web API Y MVC

2. CREAR UNA CLASE Reservation

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace Service.Models

{

public class Reservation

{

public int ReservationId { get; set; }

public string ClientName { get; set; }

public string Location { get; set; }

}

}

2. Crear un controlador HomeController:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using Service.Models;

using System.Web.Mvc;

namespace Service.Controllers

{

public class HomeController : Controller

{

private ReservationRepository repo = ReservationRepository.Current;

public ViewResult Index()

{

return View(repo.GetAll());

}

public ActionResult Add(Reservation item)

{

if (ModelState.IsValid)

{

repo.Add(item);

return RedirectToAction("Index");

}

else

{

return View("Index");

}

}

public ActionResult Remove(int id)

{

repo.Remove(id);

return RedirectToAction("Index");

}

public ActionResult Update(Reservation item)

{

if (ModelState.IsValid && repo.Update(item))

{

return RedirectToAction("Index");

}

else

{

return View("Index");

}

}

}

}

3. CREAR UNA CLASE ReservationRepository

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace Service.Models

{

public class ReservationRepository

{

private static ReservationRepository repo = new ReservationRepository();

public static ReservationRepository Current

{

get

{

return repo;

}

}

private List<Reservation> data = new List<Reservation> {

new Reservation {

ReservationId = 1, ClientName = "Adam", Location = "Board Room"},

new Reservation {

ReservationId = 2, ClientName = "Jacqui", Location = "Lecture Hall"},

new Reservation {

ReservationId = 3, ClientName = "Russell", Location = "Meeting Room 1"},

};

public IEnumerable<Reservation> GetAll()

{

return data;

}

public Reservation Get(int id)

{

return data.Where(r => r.ReservationId == id).FirstOrDefault();

}

public Reservation Add(Reservation item)

{

item.ReservationId = data.Count + 1;

data.Add(item);

return item;

}

public void Remove(int id)

{

Reservation item = Get(id);

if (item != null)

{

data.Remove(item);

}

}

public bool Update(Reservation item)

{

Reservation storedItem = Get(item.ReservationId);

if (storedItem != null)

{

storedItem.ClientName = item.ClientName;

storedItem.Location = item.Location;

return true;

}

else

{

return false;

}

}

}

}

Explicación:

Private List<Reservatinon> data = new List<Reservation>{

//data es una lista de Reservation

new reservation{

ReservationId=1, ClientName=”Adam”, Location=”Medellin”}

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Adicionar Layout y vistas

\_LAYOUT.cshtml

@{

Layout = null;

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title>@ViewBag.Title</title>

<link href="~/Content/bootstrap.css" rel="stylesheet" />

<link href="~/Content/bootstrap.min.css" rel="stylesheet" />

@RenderSection("Scripts")

</head>

<body>

@RenderSection("Body")

</body>

</html>

INDEX

@using WebServices.Models

@model IEnumerable<Reservation>

@{

ViewBag.Title = "Reservations";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

@section Scripts {

}

@section Body {

<div id="summary" class="section panel panel-primary">

@Html.Partial("Summary", Model)

</div>

<div id="editor" class="section panel panel-primary">

@Html.Partial("Editor", new Reservation())

</div>

}

Summary.cshtml

@model IEnumerable<WebServices.Models.Reservation>

<div class="panel-heading">Reservation Summary</div>

<div class="panel-body">

<table class="table table-striped table-condensed">

<thead>

<tr><th>ID</th><th>Name</th><th>Location</th><th></th></tr>

</thead>

<tbody>

@foreach (var item in Model) {

<tr>

<td>@item.ReservationId</td>

<td>@item.ClientName</td>

<td>@item.Location</td>

<td>

@Html.ActionLink("Remove", "Remove",

new { id = item.ReservationId },

new { @class = "btn btn-xs btn-primary" })

</td>

</tr>

}

</tbody>

</table>

</div>

Creating the Web API Controller: WebController

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Http;

using System.Web.Mvc;

using Service.Models;

namespace Service.Controllers

{

public class WebController : ApiController

{

private ReservationRepository repo = ReservationRepository.Current;

// GET: Web

public IEnumerable<Reservation> GetAllReservations()

{

return repo.GetAll();

}

public Reservation GetReservation(int id)

{

return repo.Get(id);

}

public Reservation PostReservation(Reservation item) {

return repo.Add(item);

}

public bool PutReservation(Reservation item) {

return repo.Update(item);

}

public void DeleteReservation(int id) {

repo.Remove(id);

}

}

}

PARA PROBAR LA API:

<http://localhost:60977/api/Web>

/api/web/3

La configuración de la ruta de la API se puede ver en:

/App\_Start/WebApiConfig.cs

La ruta es:

routeTemplate: "api/{controller}/{id}",

Este es uno de los archivos que se crean cuando se selecciona API desde el inicio.

La aplicación queda con dos rutas: una de la API y otra del MVC.

En la vista para acceder a la API:

<p>Acceda a la

@Html.ActionLink("Api", "Web", "API")

</p>

En la vista, ActionLink llama al método Remove del HomeController, y mostrarlo como un botón con estilo bootstrap:

@Html.ActionLink("Remove", "Remove",

new { id = item.ReservationId },

new { @class = "btn btn-xs btn-primary" })

COMENTARIO en una VISTA:

'Libro: Pro ASP.NET MVC 5 . Pagina: 756'

ActionLink (string linkText, String ActionName, Object RouteValues, Object HtmlAttributes)

Tambien es possible:

ActionLink (String linkText, String ActionName, String ControllerName)

@Html.ActionLink("Remove", "Remove",

new { id = item.ReservationId },

new { @class = "btn btn-xs btn-primary" })

btn : es un botón

btn-xs : botón pequeño

btn-primary: color azul

btn-danger: rojo

Llamar en el Index a una vista parcial:

@section Body {

<div id="summary" class="section panel panel-primary">

@Html.Partial("Summary", Model)

</div>

Configurar página de inicio antes de ejecución:

->Clic derecho en el proyecto -> propiedades -> Web -> Seleccionar Página específica y dejarla en blanco

CREAR UN FORMULARIO RELACIONADO CON UN MODELO:

<body>

@using (Html.BeginForm())

{

<p>Numero reservacion: @Html.TextBoxFor(x=>x.ReservationId)</p>

<p>Cliente:@Html.TextBoxFor(x=>x.ClientName)</p>

<p>Ubicación:@Html.TextBoxFor(x=>x.Location)</p>

<input type="button" value="Enviar grafico" />

}

<div>

</div>

</body>

Para ejecutarlo:<http://localhost:60977/Home/grafico>

Reformular el WebController.cs y queda así:

using System.Collections.Generic;

using System.Web.Http;

using WebServices.Models;

namespace WebServices.Controllers {

public class WebController : ApiController {

private ReservationRespository repo = ReservationRespository.Current;

public IEnumerable<Reservation> GetAllReservations() {

return repo.GetAll();

}

public Reservation GetReservation(int id) {

return repo.Get(id);

}

[HttpPost]

public Reservation CreateReservation(Reservation item) {

return repo.Add(item);

}

[HttpPut]

public bool UpdateReservation(Reservation item) {

return repo.Update(item);

}

public void DeleteReservation(int id) {

repo.Remove(id);

}

}

}

Para aplicaciones de una sola página microsoft adoptó la Librería knockouts:

<http://knockoutjs.com/>

Para aplicaciones más complejas es preferible utilizar:

<http://angularjs.org/>

Se utilizará jquery para manejar las solicitudes ajax.

insertar esto en al Layout:

<script src="~/Scripts/jquery-1.10.2.js"></script>

<script src="~/Scripts/knockout-3.0.0.js"></script>

<div id="Summary" class="section panel panel-primary">

@Html.Partial("Summary", Model)

</div>

CONTINUAR EN 766

//////////////////////////////////////////////////////////////////////////////////////////////

**ENTITY FRAMEWORK:**

**Entity framework**. Es una tecnología desarrollada por [Microsoft](https://www.ecured.cu/Microsoft), que a través de ADO.NET genera un conjunto de objetos que están directamente ligados a una [Base de Datos](https://www.ecured.cu/Base_de_Datos), permitiendo a los desarrolladores manejar dichos objetos en lugar de utilizar lenguaje [SQL](https://www.ecured.cu/SQL)contra la [Base de Datos](https://www.ecured.cu/Base_de_Datos).

Es importante tener en cuenta que **las capacidades de Entity Framework en .NET "tradicional" (EF6) y en .NET Core (EF Core) son completamente diferentes**. Así, los tres modos de trabajo descritos a continuación están completamente soportados en EF6, pero EF Core solamente soporta "Code First" y muy poquito de "Database First". "Model First" en .NET Core ni está ni se le espera. A continuación lo detallaremos más.

## **Database First**

En este modo se parte de una base de datos pre-existente con la que queremos trabajar. Es decir, tenemos la base de datos ya diseñada y probablemente con datos y lo que queremos es que EF se encargue de generar las clases necesarias y toda la "fontanería" interna para trabajar con ella.

Se suele utilizar si **aprovechamos una base de datos existente y queremos crear una nueva aplicación sobre ésta**. Por ejemplo, vamos a modernizar una aplicación antigua y queremos aprovechar todo lo que hay en la capa de datos, o si le añadimos una nueva API por encima a una aplicación que usa otra tecnología para el acceso a datos.

## **Model First**

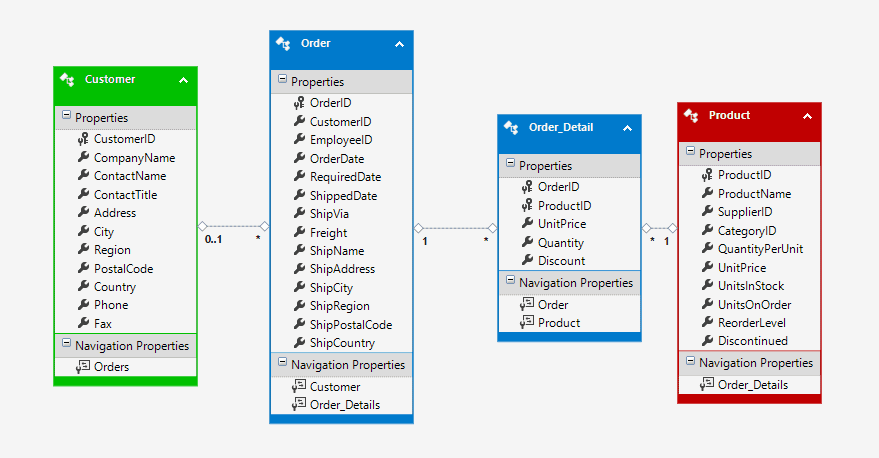
Este nombre puede despistarte, sobre todo si vienes del mundo MVC, en el que el "Modelo" de la "M" en su nombre, es código que escribes para modelar tus datos. En este caso se refiere a **crear tu modelo de datos visualmente**, usando el Diseñador de Modelos de Visual Studio. No tienes que escribir código alguno.

## **Code First**

Si en *Database First* empezábamos por la base de datos generando EF todo Solo demás, y en *Model First* estábamos en el "medio" definiendo el modelo para que EF generase tanto las clases como la base de datos... en *Code First* nos situamos en el otro extremo del espectro: **definimos nuestras clases mediante código, y EF se encarga de generar la base de datos y todo lo necesario para encajar las clases en ellas**.

Se trata de un enfoque muy orientado al programador. Es para gente a la que le gusta escribir código y tener control total sobre cómo se comportan sus clases. En este caso EF no genera código, ni necesita complejos y pesados archivos EDMX para *mapear* objetos a la base de datos, por lo que todo es más ágil.

El programador define su modelo usando clases normales (POCO). Relaciona unas con otras simplemente haciendo referencia entre ellas en propiedades. A mayores puede decorar algunas propiedades de estas clases de forma especial (las llamadas "Annotations", "Anotaciones" en inglés) si quiere indicar cosas concretas, como por ejemplo el nombre que quiere darle al campo correspondiente en la base de datos, y cosas así. Y puede implementar algunos métodos de la clase dbContext para indicar cómo mapear ciertas cosas con la base de datos.



Describes visualmente el modelo de objetos que quieres crear, con las entidades, sus propiedades, relaciones entre ellas (que generan propiedades de navegación entre las entidades), etc... y luego **Entity Framework se encarga de generar la base de datos** subyacente o modificarla **y también las entidades POCO** (*Plain Old C# Object*, o sea, clases normales y corrientes de C#) que representan dicho modelo, usando plantillas T4.

# Programación por capas

[Ir a la navegación](https://es.wikipedia.org/wiki/Programaci%C3%B3n_por_capas#mw-head)[Ir a la búsqueda](https://es.wikipedia.org/wiki/Programaci%C3%B3n_por_capas#p-search)

|  |  |
| --- | --- |
| [Commons-emblem-question book orange.svg](https://es.wikipedia.org/wiki/Programaci%C3%B3n_por_capas#p-search) | Este artículo o sección necesita [**referencias**](https://es.wikipedia.org/wiki/Wikipedia:VER) que aparezcan en una [publicación acreditada](https://es.wikipedia.org/wiki/Wikipedia:FF).  Este aviso fue puesto el 23 de febrero de 2013. |

La **programación por capas** es un modelo de desarrollo *software* en el que el objetivo primordial es la separación (desacoplamiento) de las partes que componen un sistema *software* o también una arquitectura [cliente-servidor](https://es.wikipedia.org/wiki/Cliente-servidor): lógica de negocios, capa de presentación y capa de datos. De esta forma, por ejemplo, es sencillo y mantenible crear diferentes interfaces sobre un mismo sistema sin requerirse cambio alguno en la capa de datos o lógica.



La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de que sobrevenga algún cambio, solo afectará al nivel requerido sin tener que revisar entre el código fuente de otros módulos, dado que se habrá reducido el [Acoplamiento informático](https://es.wikipedia.org/wiki/Acoplamiento_inform%C3%A1tico) hasta una interfaz de paso de mensajes.

<https://handlingresistance.wordpress.com/2011/09/14/creating-the-onion-model/>

ATR

**WPF**

Windows Presetation Foundation. Es una parte del framework de desarrollo .NET. Pero no una parte cualquiera, es la más grande de todas ellas. Abarca muchos conceptos desde interfaces de usuario con XAML hasta patrones de diseño como MVVM.

Es una API (Application Programming Interface) para crear interfaces de usuario (UI) para aplicaciones de escritorio con el framework .NET.

**WCF**

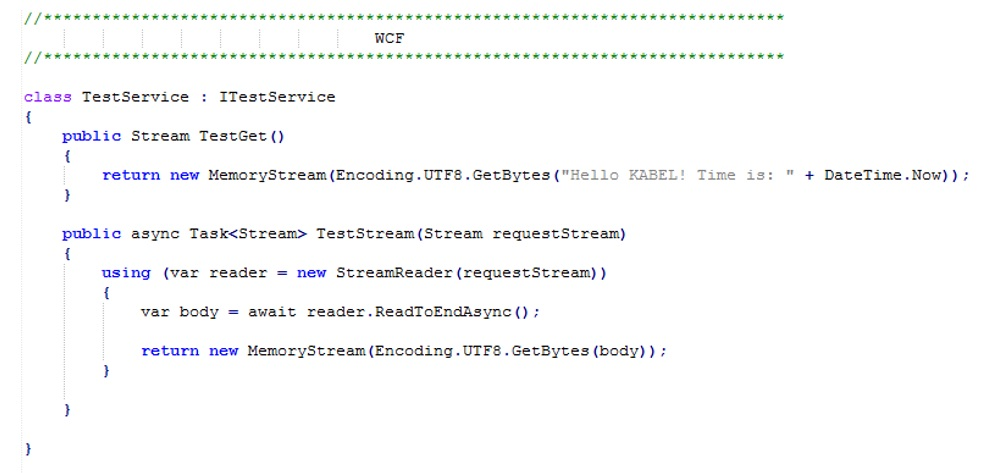
Ideado para construir:

* Servicios *SOAP* que devolvieran datos en *XML*
* Servicios seguros, fiables y transaccionales que pudieran soportar mensajería y ser accesibles a través de una serie de transportes

Es más versátil que **Web API**, ya que puede utilizar más protocolos: TCP, HTTP, HTTPS, Named Pipes, MSMQ

**WCF** para *REST* soporta formatos de datos como *XML, JSON* y *ATOM*.

Ejemplo:



**WEB API**

Fue pensado para construir:

* Servicios *REST a través de HTTP*

Ejemplo:



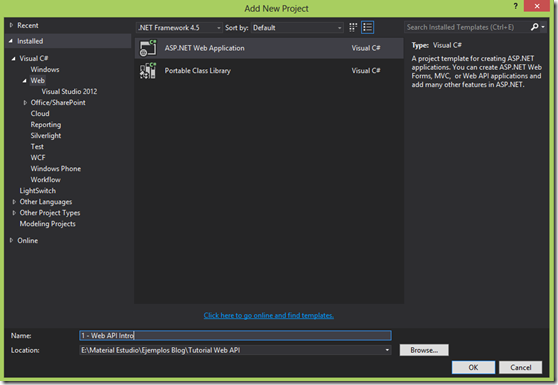
**Web Services**

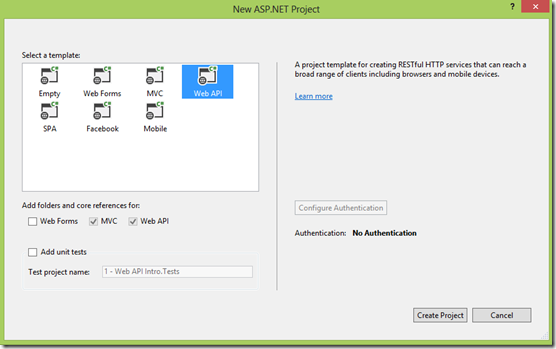
Un web service es una capa de lógica de negocio accesible a través de protocolos web, uno de ellos SOAP, que usa XML como estándar de datos para enviar y recibir datos.

Puede ser usado internamente por cualquier aplicación o externamente, a través de la internet por cualquier número de aplicaciones usando estándares como **XML** (Lenguaje de Marca Extensible) y **HTTP** (Protocolo de Transferencia de Hipertexto).

CÓMO CREAR UNA WEB API:

<https://msdn.microsoft.com/es-es/communitydocs/web-dev/webapi/mi-primer-proyecto>





Todos los controladores en Web API deben heredar de la clase **ApiController**, disponible en el namespace **System.Web.Http**, los nombres de los controladores deben finalizar con el texto Controller, así entonces si creamos un controlador llamado Test, su nombre real será TestController.

***CREAR UN CRUD EN LA API:***

<https://msdn.microsoft.com/es-es/communitydocs/web-dev/webapi/creando-un-crud>

<https://www.genbetadev.com/formacion/haciendo-una-api-rest-en-1-hora-con-asp-net-web-api-iv>

TUTORIAL MUY BUENO: Crear una API con CRUD:

<https://www.genbeta.com/desarrollo/haciendo-una-api-rest-en-1-hora-con-asp-net-web-api-ii>

El recurso seleccionado no admite el método PUT

Fiddler

**Windows Communication Foundation**

Es un conjunto de librerías que provee Microsoft en el Framework .NET para la construcción de aplicaciones orientadas a servicios.

Es también un modelo de programación unificado, el cual se define como un simple vía para escribir servicios y por tanto unifica elementos como Servicio Web (\*.asmx), .NET Remoting, Message Queue (MSMQ), Enterprise Services (COM+) y Servicios web Mejorados. WCF no remplaza estas tecnologías sobre una base individual, más bien suministra un modelo simple de programación que puede usar para aprovechar todo estos elementos a la vez.

* WCF es la evolución de las tecnologías web service de Microsoft de años anteriores.
* WCF provee un amplio rango de funcionalidad por encima de web services, con mejores características en aspectos de calidad como flexibilidad, portabilidad y mantenibilidad.
* Los web services sólo pueden ser accedidos a través de HTTP, mientras que WCF se puede hospedar en un servidor web, WAS, puede ser un servicio de Windows, y usa una variedad de protocolos más amplia.

Tutorial WCF:

<https://www.tutorialspoint.com/wcf/wcf_consuming_service.htm>

CONSUMIR UN SERVICIO EN ASP.NET MVC

<https://www.c-sharpcorner.com/article/consuming-web-service-in-asp-net-mvc/>

* Crear un nuevo proyecto ASP.NET vacío seleccionando (chuliando) la referencia MVC.
* Click drecho en la solución ->Agregar -> referencia de servicio.
* <http://www.w3.org/2003/05/soap-envelope>
* <https://test.placetopay.com/soap/pse/?wsdl>
* <http://ws.cdyne.com/NotifyWS/phonenotify.asmx>
* <http://ws.cdyne.com/NotifyWS/phonenotify.asmx?WSDL>
* <https://celcer.sri.gob.ec/comprobantes-electronicos-ws/AutorizacionComprobantes?wsdl>
* <https://geoservices.tamu.edu/Services/Geocode/WebService/GeocoderService_V04_01.asmx?WSDL>
* ->Avanzadas ->Agregar Referencia web ->Digitar la URL ->ir ->Agregar Referencia. Y notamos que se agrega la referencia en la carpeta: Web References
* En la carpeta Controlador -> Agregar con scaffold ->Agregar Controlador MVC 5: En blanco
* **using** System.Linq;
* **using** System.Web.Mvc;
* **namespace** ConsumingWebServiceInASPNETMVC.Controllers
* {
* **public** **class** HomeController : Controller
* {
* // GET: Home
* **public** ActionResult Index()
* {
* **return** View();
* }
* [HttpGet]
* **public** JsonResult BookingStatus()
* {
* //Creating Web Service reference object
* HotepBookingPayReference.PaymentWebService objPayRef = **new** HotepBookingPayReference.PaymentWebService();
* //calling and storing web service output into the variable
* var BookingStatusInfo = objPayRef.RoomBookingStatus().ToList();
* //returning josn result
* **return** Json(BookingStatusInfo, JsonRequestBehavior.AllowGet);
* }

* }
* }
* Crear una vista “Index”
* Insertar el código JQuery para vincular el Html usando JSON:
* @{
* ViewBag.Title = "Index";
* }
* <script src="~/Scripts/jquery-1.10.2.min.js"></script>
* <script src="~/Scripts/jquery-1.10.2.intellisense.js"></script>
* <script type="text/javascript">
* $(document).ready(**function** () {
* $.getJSON("/Home/BookingStatus", **function** (data) {

* **var** tr;
* $.each(data, **function** (d,i) {

* tr = $('<tr/>');
* tr.append("<td>" + i.RoomId + "</td>");
* tr.append("<td>" + i.RooType + "</td>");
* tr.append("<td>" + i.BookingStatus + "</td>");
* $('#tblBookingStatus').append(tr);


* });
* });


* });

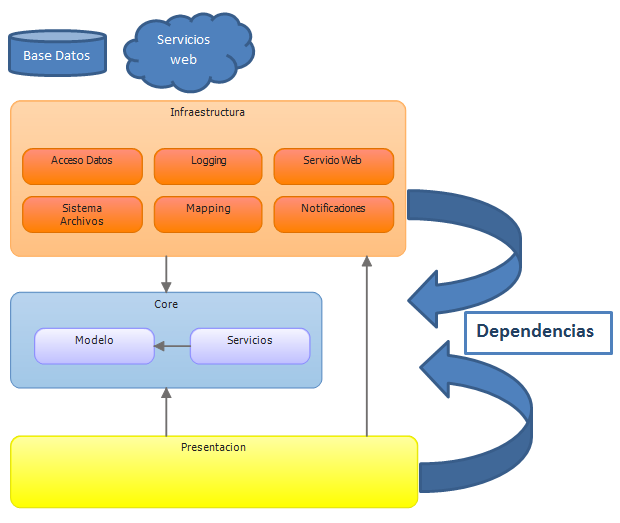
* </script>
* <div **class**="form-horizontal">
* <hr />
* <div **class**="form-group">
* <table id="tblBookingStatus" **class**="table table-responsive" style="width:400px">
* <tr>
* <th>
* Room Id
* </th>
* <th>
* Room Type
* </th>
* <th>
* Booking Status
* </th>
* </tr>
* <tbody>
* </tbody>
* </table>


* </div>
* </div>

VER CURSO: .NET Y MVC

<http://www.pragimtech.com/Home.aspx>

**ARQUITECTURA CEBOLLA**



#### 

#### **Finalidad:**

La finalidad de este estilo de arquitectura es poder construir aplicaciones que sean fáciles de mantener, probar y sobre todo que se encuentren desacopladas de elementos de infraestructura tales como base de datos o servicios.

* **Independencia de frameworks:** La arquitectura no debe depender de ninguna librería. Esto permite usar frameworks como herramientas.
* **Testeable:** Las reglas de negocio pueden ser testeadas sin necesidad de conocer la interfaz de usuario, la base de datos o algún servicio externo.
* **Independiente de la UI:** La interfaz de usuario puede cambiar fácilmente sin tener que afectar al resto del sistema. Un tipo de interfaz de usuario puede ser reemplazada por otra sin cambiar las reglas de negocio.
* **Independiente de la base de datos:** Se puede cambiar de motor de base de datos sin ningún problema. Las reglas de negocio no deben estar amarradas a esta.
* **Independiente de agentes externos:** Las reglas de negocio no deben saber nada que este fuera de su contexto.

#### **Elementos:**

El número de elementos puede variar, pero siempre se debe tener en cuenta que el Domain Model es el centro de todo.

* **Core:** Domain model, servicios, interfaces de repositorio, otras interfaces.
* **UI:** Mvc, Servicios Web, etc.
* **Test:** Pruebas unitarias o de integración.
* **Infraestructura:** Implementación de las interfaces definidas en el Core.

WPF

Aplicación de escritorio:

<https://msdn.microsoft.com/es-co/library/jj153219.aspx>

1. Crear nuevo proyecto C# Aplicacion de escritorio: wpf
2. Se crea automàticamente un formulario (.xaml)
3. Para cambiar el formulario inicial de la aplicaciòn modifcarlo en App.xaml ->StartupUri
4. Insertar los controles en el formulario e ingresar el còdigo:

private void Button\_Click(object sender, RoutedEventArgs e)

{

if (radio1.IsChecked == true)

{

MessageBox.Show("Holaaa");

}

else if (radio2.IsChecked == true) {

MessageBox.Show("Adiosss");

}

}

1. Ejecutar

/////////////////////////////////////////////////

NET CORE

Fuente: LiteleASPNetCoreBook.pdf

(Funciona O.K)

ASP.NET Core is a web framework created by Microsoft for building web applications, APIs, and microservices. It uses common patterns like MVC (Model-View-Controller), dependency injection, and a request pipeline comprised of middleware.

.NET Standard is a platform-agnostic interface that defines features and APIs.

.NET Core is the .NET runtime that can be installed on Windows, Mac, or Linux. It implements the APIs defined in the .NET Standard interface with the appropriate platform-specific code on each operating system.

Se basa en un conjunto de paquetes NuGet granulares y bien factorizados. Esto **te permite optimizar tu aplicación** para incluir solo los paquetes NuGet que necesitas.

[**ASP.NET Core**](https://docs.microsoft.com/en-us/aspnet/core/) es un nuevo **framework de código abierto y multiplataforma** para la creación de aplicaciones modernas conectadas a Internet, como aplicaciones web y APIs Web.

PRIMERA APLICACIÓN EN ASP.NET CORE

En la consola digitar:

-> dotnet --version

y debe aparecer la versión:

2.1.500

Crear un nuevo proyecto desde la consola:

-> dotnet new console -o CsharpHelloWorld

Crear un nuevo proyecto MVC con autenticación en la carpeta AspNetCoreTodo:

-> dotnet new mvc --auth Individual -o AspNetCoreTodo

cd AspNetCoreTodo

En el menú:

-> Archivo-Nuevo -Proyecto -> Aplicación Web ASP.NET CORE -> Aplicacion WEB MVC

Autenticación: Cuentas de usuario individuales

1. Crear un proyecto WEB - .Net Core -> Aplicaciòn WEB ASP.NET CORE
2. Seleccionar Web Application con Autenticación Individual
3. El sistema crea una estructura de un sitio:

-pages: vistas

-wwwroot: documentos estáticos

-appsettings.json: configuraciòn de la aplicación. Lo invalida: appsetings.Development.json

1. a

Program.cs : Configurar el Web server

Startup.cs : se puede adicionar las instrucciones que manejan las solicitudes entrantes y entrega páginas estáticas como las páginas de errores.

También sirve para adicionar tu propio servicio al contenedor de dependency Injection

wwwroot: guarda los archivos js, css, imágenes,

appsettings.json : Contiene configuración que ASP.NET CORE carga al inicio. Se utiliza para almacenar connection string de base de datos

->Insertar un nuevo controlaror: peliculaController

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

namespace Cinema.Controllers

{

public class PeliculaController : Controller

{

public IActionResult Index()

{

return View();

}

}

}

Crear un Modelo Pelicula.cs en la carpeta Models:

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Threading.Tasks;

namespace Cinema.Models

{

public class Pelicula

{

public int id { get; set; }

public DateTime fecha { get; set; }

public string titulo { get; set; }

[Required]

public bool enCartelera {get; set;}

}

}

En un controlador, un Action puede retornar un View, un JSON data, o Http status como el 404 (Not found)

-> Insertar un nuevo controlador: PeliculaViewModel

namespace Cinema.Models

{

public class PeliculaViewModel

{

//Esta es una colección de peliculas

public Pelicula[] coleccion { get; set; }

}

}

Insertar una vista Index.cshtml en una carpeta Pelicula de Views:

@model PeliculaViewModel

@{

ViewData["Title"] = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<table class="table table-hover">

<thead>

<tr>

<td>&#x2714;</td>

<td>Titulo</td>

<td>En cartelera?</td>

</tr>

</thead>

@foreach (var item in Model.coleccion)

{

<tr>

<td>@item.fecha</td>

<td>@item.titulo</td>

@{var r=””;}

@{if (item.enCartelera==true) {r=”Si”;}else{r=”No”;}

<td>@r</td>

</tr>

}

</table>

<h2>Index</h2>

Insertar esto en la hoja de estilos:

div.todo-panel {

margin-top: 15px;

}

table tr.done {

text-decoration: line-through;

color: #888;

}

Estudiar: Vistas partialView y server-rendered view

<https://docs.asp.net>

Normalmente una aplicación tiene:

Presentation Layer; Business Logic Layer, DataRepository Layer (maneja la base de datos).

INTERFACE:

La definición de los métodos y propiedades de un objeto es separada del código de los métodos y propiedades.

Esto hace más fácil mantener las clases desacopladas y fácil de probar (con pruebas automáticas).

Por convención el nombre de una interfaz empieza con la letra I.

Se creará una interfaz para representar el servicio:

->Crear una carpeta Servicios

->Insertar una interfaz en la nueva carpeta llamada: IPeliculaServicios.cs

Como es una interfaz, no lleva código, solo la definición de un método:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Cinema.Models;

namespace Cinema.Servicios

{

interface IpeliculaServices

{

Task<Pelicula[]> GetIncompleteItemsAsync();

}

}

Este método (GetIncompleteItemsAsync) no requiere parámetros y retorna un Task<Pelicula[]>

Crear una clase (FakePeliculaServicios) en la carpeta Servicios que implementa la interfaz IpeliculaServices:

public class FakePeliculaServicios:IpeliculaServices

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Cinema.Models;

namespace Cinema.Servicios

{

public class FakePeliculaServicios:IpeliculaServices

{

public Task<Pelicula[]> GetIncompleteItemsAsync() {

var item1 = new Pelicula

{

enCartelera = true,

titulo = "El viejo y elmar",

fecha = DateTime.Now,

id = 1

};

var item2 = new Pelicula();

item2.enCartelera = false;

item2.titulo = "Terminator";

item2.fecha = DateTime.Now;

item2.id = 2;

return Task.FromResult(new[] { item1, item2 });

}

}

}

Usar Inyección de Dependencia:

En el controlador PeliculaController insertar el siguiente código para interactuar con la Interfaz:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Mvc;

using Cinema.Servicios;

namespace Cinema.Controllers

{

public class PeliculaController : Controller

{

private IpeliculaServices \_ipeliculaServices;

public PeliculaController(IpeliculaServices ipeliculaServices)

{

\_ipeliculaServices = ipeliculaServices;

}

public async Task<IActionResult> Index()

{

var items = await \_ipeliculaServices.GetIncompleteItemsAsync();

var model = new PeliculaViewModel()

{

coleccion = items

};

return View(model);

}

}

}

--------------------------

public PeliculaController(IpeliculaServices ipeliculaServices) es un constructor que utiliza como parámetro una interfaz: IpeliculaServices

The Task pattern is common when your code calls out to a database or an API service, because it won't be able to return a real result until the database (or network) responds.

await lets your code pause on an async operation, and then pick up where it left off when the underlying database or network request finishes.

In the meantime, your application isn't blocked, because it can process other requests as needed.

Es necesario declarar que el FakePeliculaServicio será la clase que implementará la interfaz. Esto se hace en la clase startup

Cualquier servicio que se quiera utilizar en la aplicación debe ser ingresada al ConfigureServices del startup:

public void ConfigureServices(IServiceCollection services)

{

services.AddSingleton<IpeliculaServices,FakePeliculaServicios>();

This line tells ASP.NET Core to use the FakeTodoItemService whenever the ITodoItemService interface is requested in a constructor (or anywhere else).

AddSingleton adds your service to the service container as a singleton. This means that only one copy of the FakeTodoItemService is created, and it's reused whenever the service is requested. Later, when you write a different service class that talks to a database, you'll use a different

approach (called scoped) instead. I'll explain why in the Use a database chapter.

Ejecutar: Ctrl +F5

dotnet run

REFRESCAR PÁGINA: Ctrl + F5

REFRESCAR PÁGINA: Ctrl + R

Editar el Layout.cshtml para insertar en la barra la nueva opción para Película:

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

<li><a asp-area="" asp-controller="Home" asp-action="Index">Home</a></li>

<li><a asp-area="" asp-controller="Home" asp-action="About">About</a></li>

<li><a asp-area="" asp-controller="Home" asp-action="Contact">Contact</a></li>

<li><a asp-area="" asp-controller="Pelicula" asp-action="Index">Pelicula</a></li>

</ul>

The asp-controller and asp-action attributes on the <a> element are called tag helpers. This replaced the former @Html.ActionLink()

INSERTAR UN PAQUETE NUGET:

dotnet add package Humanizer

Permite ver las fechas más amigablemente.

Cuando está instalado debe aparecer en ->Dependencias ->Nuget ->Humanizer.

En la vista:

Insertar el using: @using Humanizer;

y en el campo de la fecha de la vista: @item.fecha.Humanize()

BASES DE DATOS:

No es buena idea escribir consultas SQL en la aplicación. Para eso existen los ORM (Object-Relational Mapper) que se encargan de hacer el código que interactúa con la base de datos. Ejemplos de ORM: Hibernate en Java y ActiveRecord en Ruby.

El ORM creado por Microsoft es EntityFramework, el cual se puede conectar a SQL Server, PostgreSQL, MYSQL, Mongo (NoSQL), SQLite.

Cómo se creo un proyecto con MVC y autenticación individual, el proyecto ya tiene:

El EntityFramework Core Package

Una base de datos app.db en SQLite

Una clase databaseContext que provee un punto de ingreso a la base de datos y expresa como interactuar con la bd para leer o escribir. Por defecto esa clase está en: Data/ApplicationDbContext.cs

Un ConnectionString: que contiene la ruta de la base de datos. Por defecto está en: appsettings.json y específicamente dice: DataSource=app.db .

Es necesario decirle a EntityFramework cual connectionString y cual Context y cual proveedor de Base de Datos se va a utilizar en el método: ConfigureServices de la clase: Startup

Por defecto:

services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlite(

Configuration.GetConnectionString("DefaultConnection")));

En el visual studio por defecto aparece:

services.AddDbContext<ApplicationDbContext>(options =>

options.UseSqlServer(

Configuration.GetConnectionString("DefaultConnection")));

services.AddDefaultIdentity<IdentityUser>()

.AddEntityFrameworkStores<ApplicationDbContext>();

En Data/ApplicationDbContext aparece lo siguiente:

public ApplicationDbContext(

DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

Insertar debajo:

public DbSet<Pelicula> Items {get;set;}

DbSet representa una tabla o colección de una base de datos.

DbSet<Pelicula> tblPeliculas

Significa que en la tabla tblPeliculas se van a guardar entidades Pelicula.

Ahora se debe crear la tabla en la base de datos mediante migración:

En la consola (ubicado en el directorio raiz):

dotnet ef migrations add AddItems

Y debe aparecer #######\_AddItems en Data/Migration

Para deshacer:

To undo this action, use 'dotnet ef migrations remove'

dotnet ef migrations list : para ver el listado de migraciones

dotnet ef database update : para aplicar los cambios en la base de datos y crear la nueva tabla. Funciona O.K.

dotnet ef database update CreateIdentitySchema : para deshacer la última migración, se debe proveer el nombre de la migración anterior (CreateIdentitySchema)

Para eliminar la Base de Datos:

dotnet ef database drop

dotnet ef database update

CREAR UNA NUEVA CLASE DE SERVICIO

La nueva clase se llama: PeliculaServicio.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Cinema.Data;

using Cinema.Models;

using Microsoft.EntityFrameworkCore;

namespace Cinema.Servicios

{

public class PeliculaServicio : IpeliculaServices

{

private readonly ApplicationDbContext \_context;

public PeliculaServicio(ApplicationDbContext context)

{

\_context = context;

}

public async Task<Pelicula[]> GetIncompleteItemsAsync()

{

return await \_context.TblPeliculas

.Where(x=>x.enCartelera==true)

.ToArrayAsync();

}

}

}

El método where implementa: LINQ (language integrated

query) .

En esta clase el método: ApplicationDbContext está inyectado.

El método: ToArrayAsync le dice al EntityFramework que entregue los datos como un array. Ese método es asíncrono y por tanto debe usarse un await.

Es necesario actualizar el ConfigureServices (en Starup) con el nuevo servicio:

public void ConfigureServices(IServiceCollection services)

{

services.AddScoped<IpeliculaServices,PeliculaServicio>();

AddScoped adds your service to the service container using the scoped lifecycle. This means that a new instance of the TodoItemService class will be created during each web request. This is required for service classes that interact with a database.

INGRESAR ITEMS A UNA BASE DE DATOS:

En Index.cshtml:

<div class="panel-footer add-item-form">

@await Html.PartialAsync("IngresarPeliculaParcial",new Pelicula())

</div>

Ingresar una vista parcial en Views/Pelicula:

@model Pelicula;

<form asp-action="AddItem" method="post">

<label asp-for="titulo" >Ingresar Película</label>

<input asp-for="titulo" >

<label asp-for="enCartelera">En cartelera?</label>

<input asp-for="enCartelera" />

<label asp-for="fecha">fecha</label>

<input asp-for="fecha" />

<button type="submit" >Enviar</button>

</form>

En el controlador PeliculaController INGRESAR el método: AddItemAsync

public async Task<IActionResult> AddItem(Pelicula newItem)

{

if (!ModelState.IsValid)

{

return RedirectToAction("Index");

}

var exitoso = await \_ipeliculaServices.AddItemAsync(newItem);

if (!exitoso)

{

return BadRequest("No se pudo ingresar item");

}

return RedirectToAction("Index");

}

Model binding: Cuando el usuario presiona el botón Enviar, ASP.NET CORE relaciona cada campo recibido en el controlador con los campos del modelo Pelicula y los coloca en la variable newItem

[ValidateAntiForgeryToken]: le dice a ASP.NET CORE que debe verificar el token de verificación oculto que inserta el asp-action.

Intertar el nuevo método (AddItemAsync) en la interfaz:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Cinema.Models;

namespace Cinema.Servicios

{

public interface IpeliculaServices

{

Task<Pelicula[]> GetIncompleteItemsAsync();

Task<bool> AddItemAsync(Pelicula newItem);

}

}

Implementar el código del nuevo método en PeliculaServicio:

public async Task<bool> AddItemAsync(Pelicula newItem)

{

//newItem.id = Guid.NewGuid();

\_context.TblPeliculas.Add(newItem);

var saveRetult = await \_context.SaveChangesAsync();

return saveRetult == 1;

}

INSERTAR UN CHECKBOX EN LA TABLA DE DATOS:

<td>

<form asp-action="MarkDone" method="POST">

<input type="checkbox" class="done-checkbox">

<input type="hidden" name="id" value="@item.Id">

</form>

</td>

Add JavaScript code

Find the site.js file in the wwwroot/js directory and add this code:

wwwroot/js/site.js

$(document).ready(function() {

// Wire up all of the checkboxes to run markCompleted()

$('.done-checkbox').on('click', function(e) {

markCompleted(e.target);

});

});

function markCompleted(checkbox) {

checkbox.disabled = true;

var row = checkbox.closest('tr');

$(row).addClass('done');

var form = checkbox.closest('form');

form.submit();

}

This code first uses jQuery (a JavaScript helper library) to attach some code to the click even of all the checkboxes on the page with the CSS class done-checkbox .

Insertar el método MarkDone en el controlador PeliculaController:

[ValidateAntiForgeryToken]

public async Task<IActionResult> MarkDone(Guid id)

{

if (id == Guid.Empty)

{

return RedirectToAction("index");

}

var exitoso = await \_ipeliculaServices.MarkDoneAsync(id);

if (!exitoso)

{

return BadRequest("No es posible marcar el item");

}

return RedirectToAction("index");

}

**SEGURIDAD E IDENTIDAD**

Seguridad:

Prevenir SQL Injection

Prevenir Cross Domain (CSFR) Forms

Usar encriptación HTTPS

Dar la opción al usuario de tener cuenta, password, recuperar password,

Autenticación: Si el usuario se loguió o no.

Autorización: Si el usuario tiene permiso para acceder a alguna función después de loguearse

ASP.NET CORE trae un paquete Nuget de identificación que se instala con la opción Autenticación Individual cuando se crea el proyecto.

[Authorize] : Es una anotación que requiere que esté logueado. En el controlador:

[Authorize]

public class TodoController : Controller

{

// ...

}

Pero requiere este using:

using Microsoft.AspNetCore.Authorization;

Se puede filtrar la base de datos dependiendo de quien está logueado:

[Authorize]

public class TodoController : Controller

{

private readonly ITodoItemService \_todoItemService;

private readonly UserManager<ApplicationUser> \_userManager;

public TodoController(ITodoItemService todoItemService,

UserManager<ApplicationUser> userManager)

{

\_todoItemService = todoItemService;

\_userManager = userManager;

}

// ...

}

Se requiere este using:

using Microsoft.AspNetCore.Identity;

<https://jakeydocs.readthedocs.io/en/latest/security/authentication/identity.html>

Funciona o.k.

///////////////////////////////////////////////////

NET CORE

Buscar Tutorial de Web site CRUD con Base de Datos

**Tutorial de MVC**

CREAR UN MVC

<https://youtu.be/gHJdHwCYhi8>

1. Crear un proyecto WEB - .Net Core -> Aplicaciòn WEB ASP.NET CORE
2. Seleccionar MVC
3. Insertar con clic derecho en solución -> New Project ->.NET CORE -> Biblioteca de clases
4. Clic derecho en el proyecto ->Agregar ->referencia y seleccionar model y service
5. Herramientas -> Administrador de packetes Nuget -> Administrar paquetes Nuget

Instalar: microsoft.entityframeworkcore.sqlserver y

microsoft.entityframeworkcore.tools

1. Agregar una nueva clase llamada: AlbumDbContext

insertar en el nombre de la clase: : DbContext

y queda asì: **AlbumDbContext** : DbContext

en DbContext sale un error y se presiona: CTR + .

Y se selecciona: using Microsoft.EntityFrameworkCore;

insertar este constructor:

public **AlbumDbContext**(DbContextOptions<**AlbumDbContext**> options) : base(options)

{

}

7. Crear una clase Album en una nueva carpeta Domain

con las siguientes propiedades:

public int Id { get; set; }

[Required]

[StringLength(200)]

public string Name { get; set; }

public string Description { get; set; }

public string Photolink { get; set; }

public DateTime CreatedAt { get; set; }

//[Required] : Se utiliza para que un campo sea NOT NULL en la base de datos.

//[StringLength(200)] : especificar la longitud. Si no se hace entonces asume el máximo tamaño de campo VARCHAR(MAX)

8. En Startup registrar la dependencia en

app\_start

RouteConfig.cs : para especificar las rutas: controlador/Accion/parametros y controlador por defecto

La ruta que trae la plantilla por defecto es la más general y se debe dejar pero de último.

Antes de ella se debe dejar las reglas particulares:

routes.MapRoute(

name: “Ejemplo”,

url:”Ejemplo”,

default: new{

controller=”home”,

action=”contact”

}

);

PASARLE UN MODELO A LA VISTA:

En el controlador:

public ActionResult Index()

{

var serviciosPeliculas = new serviciosPelicula();

var model = serviciosPeliculas.ListadoPeliculas();

return View(model);

}

En la nueva carpeta services crear una clase: serviciosPelicula

namespace MVC1.services

{

public class serviciosPelicula

{

public List<Movie> ListadoPeliculas() {

List<Movie> Listado = new List<Movie>() ;

Movie pel1 = new Movie()

{

Title = "El depredador2",

Id\_Movie = 1,

Available = true,

Genre = "Accion",

Price = 10

};

Movie pel2 = new Movie()

{

Title = "El depredador3",

Id\_Movie = 2,

Available = true,

Genre = "Accion",

Price = 11

};

//Listado ={pel1,pel2 };

return new List<Movie> { pel1, pel2 };

}

}

}

En la vista:

@model List<Movie>

@{

Layout = null;

}

<!DOCTYPE html>

<html>

<head>

<meta name="viewport" content="width=device-width" />

<title></title>

</head>

<body>

<div>

@foreach (var pelicula in Model)

{

<H4>Nombre:@pelicula.Title</H4>

<h5>Genero: @pelicula.Genre</h5>

}

</div>

</body>

</html>

En la pàgina de Microsoft se puede ver un listado de todo lo que puede retornar un action adicional al ActionResult (ViewResult, ContentResult, JsonResult, FileContentRetuls, FileResult, FilePatchResult, FileStreamResult)

Pasar contenido (ContentResult) a una vista:

En el controlador:

public ContentResult Muestra() {

return Content("Felipe");

}

INSTANCIAR:

var p = new persona();

StudentName student1 = new StudentName("Craig", "Playstead");

StudentName student2 = new StudentName

{

FirstName = "Craig",

LastName = "Playstead",

};

Pasar contenido JSON:

public ActionResult Muestra() {

var p = new persona() {nombre="Felipe",Edad=55 };

return Json(p,JsonRequestBehavior.AllowGet);

return Content("Felipe");

}

REDIRECCIONAR:

public RedirectResult Muestra() {

return Redirect (“http://www.google.com”);

}

public RedirectToRouteResult Muestra(){

return RedirectToAction(“Accion”,”Controlador”)

}

public ActionResult Muestra() {

return RedirectToRoute(“RutaEjemplo”);

}

RutaEjemplo: Nombre de Ruta configurada en el RouteConfig

STATUSCODE:

PUBLIC HttpStatusCodeResult Index()

{

return new HttpStatusCodeResult (301,”Hola”)

}

DESCARGAR UN ARCHIVO:

public ActionResult Index(){

var ruta=Server.MapPath(“Ejemplo.pdf”)

return File(ruta,”application/pdf”,”Ejemplo6.pdf”)

}

QUERYSTRING

[http://localhost:4969](http://localhost:49699)9?nombre1=valor1&edad=99

En el controlador:

public ActionResult Contact(string nombre)

{

ViewBag.Message = "Your contact page. "+ nombre;

return View();

}

HTTPGET Y HTTPPOST

[HttpGet]

public ActionResult Contact(

{ViewBag.message =”Formulario Contacto”;

Return View();

}

[HttpPost]

public ActionResult Contact(int edad){

ViewBag.message =”Formulario Contacto”+edad;

Return View();

}

FORMULARIO CON HTML PURO:

<form action="/home/contact" method="post">

<input type="number" id="edad" name="edad"/>

<input type="submit" value="enviar">

</form>

ViewBag y ViewData

public ActionResult Contact(){

ViewBag.Nombre=”Carlos”;

ViewBag.id=01;

ViewBag.fecha=new DateTime(2018,4,7);

}

En la vista:

<h2>@ViewBag.Nombre;</h2>

ViewData:

ViewData[“mensaje”]=”mensaje ejemplo”;

en la vista: ViewData[“mensaje”]

PARTIAL VIEW:

En una vista se puede ingresar un partial:

@Html.Partial(“\_loginPartial”)

//inserta la vista \_loginPartial en otra vista

@Html.ActionLink (controlador, accion);

[AllowAnonymous] : Antes de la clase, correo con cualquier usuario

[Authorize] : Antes de la clase y solo corre con usuarios registrados.

RAZOR: en la vista

@model List<Movie>

@\*Esto es un comentario\*@

<table class="table table-striped">

<thead>

<tr>

<th>Titulo</th>

<th>GEnero</th>

</tr>

</thead>

<tbody>

@foreach (var pelicula in Model)

{

var claseFila=(!pelicula.EstaenCartelera)? “danger” : string.Empty

<tr class=”@claseFila”>

<td>@pelicula.Title</td>

<td>@pelicula.Genre</td>

</tr>

}

</tbody>

</table>

@Html.Raw (campo de una tabla): Permite ejecutar el còdigo html o script que ya trae el campo. Pero es peligroso porque podría ser usado por hackers

@Html.ActionLink(“link text”,”Action Name”,”Controller Name”)

Si no se especifica el controlador entonces el asume que es el controlador actual.

Otra forma:

@Html.ActionLink(“link text”,”Action Name”,”Controller Name”,Null,new {Atributo=”valor1”})

Otra forma (muestra un botòn):

@Html.ActionLink(“link text”,”Action Name”,”Controller Name”,Null,new {Atributo=”valor1”,@class=”btn btn-danger”})

@Html.ActionLink(“link text”,”Action Name”,”Controller Name”,Null,new {edad=88} {Atributo=”valor1”,@class=”btn btn-danger”})

PARA VINCULAR UN CONTROL CON UN CAMPO DE LA BASE DE DATOS:

<div class="editor-label">

@Html.LabelFor(model => model.Participantes)

</div>

<div class="editor-field">

@Html.EditorFor(model => model.Participantes)

@Html.ValidationMessageFor(model => model.Participantes)

</div>

App\_Star config

identity\_config se puede editar las validaciones requeridas para el password

Cuando se selecciona LOGIN (cuando se está creando el proyecto)

LocalDb es una versión de prueba de SQLServer

Modelo

En WebConfig hay un connection string y se puede ver el nombre de la base de datos.

ENTITY FRAMEWORK:

CONNECTIONSTRING

En el archivo web.config

<connectionStrings>  
 <add name="DefaultConnection" connectionString="Data Source=(LocalDb)\MSSQLLocalDB;Initial Catalog=aspnet-MvcMovie-fefdc1f0-bd81-4ce9-b712-93a062e01031;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\aspnet-MvcMovie-fefdc1f0-bd81-4ce9-b712-93a062e01031.mdf" providerName="System.Data.SqlClient" />  
 <add name="MovieDBContext" connectionString="Data Source=(LocalDb)\MSSQLLocalDB;Initial Catalog=aspnet-MvcMovie;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\Movies.mdf" providerName="System.Data.SqlClient" />  
</connectionStrings>

Initial Catalog: Es el nombre de la base de datos

DataDirectory|\\*.mdf : es el nombre del archivo

En Modelos Crea una clase: PeliculaContext.cs

public class PeliculaContext : DbContext {

public PeliculaContext() : base (“DefaultConnection”){

//DefaultConnection: Es el nombre que aparece en web.config

}

public DbSet<Pelicula>Peliculas {get;set;}

Para verificar si está instalado EntityFramework se lee el archivo References

Si no aparece entonces instalarlo en Herramientas - -> Administrador de packetes Nuget -> Administrar paquetes Nuget

->Examinar ->Entity FrameWork ->Buscar ->Instalar

Console:

Herramientas - -> Administrador de packetes Nuget -> Consola de Administrador de paquetes

Digitar en la consola: enable-migrations

Esto crea automáticamente un folder Migrations con una clase: Configuration

En Configuration, la propiedad:

AutomaticMigrationsEnabled=false //Si se coloca a true entonces actualiza la Base de Datos cuando se modifica el modelo.

AutomaticMigrationDataLossAllowed=true //permite truncar datos cuando actualice la base de datos, si se ha modificado la extensión de algún campo.

En la consola digitar (para crear o actualizar la base de datos):

update-database

En server explorer se puede ver la base de datos creada.

si cambió la longitud de algún campo ([StringLength(200)]) entonces se debe ingresar este comando (porque de lo contrario muestra un error):

update-database -force

CONTROLADOR:

Agregar un controlador con acciones de Leer y Escribir.

En el código del index del controlador -> clic derecho ->Add view Para crear una vista.

<https://msdn.microsoft.com/es-es/library/dd410597(v=vs.100).aspx>

La estructura de carpetas de un proyecto de MVC difiere de la de un proyecto de sitio web de ASP.NET. El proyecto de MVC contiene las carpetas siguientes:

* Content, para los archivos de compatibilidad del contenido. Esta carpeta contiene la hoja de estilos en cascada (archivo .css) para la aplicación.
* Controllers, para los archivos de controlador. Esta carpeta contiene los controladores de ejemplo de la aplicación, que se denominan AccountController y HomeController. La clase AccountController contiene la lógica de inicio de sesión para la aplicación. La clase HomeController contiene la lógica a la que se llama de forma predeterminada cuando la aplicación se inicia.
* Models, para los archivos de modelo de datos como los archivos .dbml de LINQ to SQL o los archivos de entidad de datos.
* Scripts, para los archivos de script, como los compatibles con ASP.NET AJAX y jQuery.
* Views, para los archivos de paginación de vistas. Esta carpeta contiene tres subcarpetas: Account, Home y Shared. La carpeta Account contiene vistas que se utilizan como interfaz de usuario para el inicio de sesión y para cambiar las contraseñas. La carpeta Home contiene una vista Index (la página de inicio predeterminada para la aplicación) y una vista de página Acerca de. La carpeta Shared contiene la vista de la página maestra para la aplicación.

En el **Explorador de soluciones**, haga clic con el botón secundario en la carpeta **Controllers**, haga clic en **Agregar** y, a continuación, haga clic en **Controlador**.

## Crear un código auxiliar de método de acción

Para aplicar técnicas de desarrollo basado en pruebas (TDD) a este proyecto debe escribir la prueba unitaria para un método de acción antes de escribir el propio método de acción. Sin embargo, si desea que su prueba unitaria se compile, debe tener un código auxiliar para el método de acción planeado, que en este tutorial es ViewMaps.

## Crear un código auxiliar de método de acción

Para aplicar técnicas de desarrollo basado en pruebas (TDD) a este proyecto debe escribir la prueba unitaria para un método de acción antes de escribir el propio método de acción. Sin embargo, si desea que su prueba unitaria se compile, debe tener un código auxiliar para el método de acción planeado, que en este tutorial es ViewMaps.

**C#**

public ActionResult ViewMaps()  
{  
 // Add action logic here  
 throw new NotImplementedException();  
}

### Para agregar pruebas unitarias para los métodos de acción

1. En el proyecto de pruebas, haga clic con el botón secundario del mouse en la carpeta **Controllers**, haga clic en **Agregar** y después haga clic en **Clase**.
2. En el cuadro de texto **Nombre**, escriba **MapsControllerTest**.
3. Haga clic en **Agregar**.
4. Visual Studio agrega la clase MapsControllerTest al proyecto de prueba.
5. Abra la clase MapsControllerTest y escriba el código que se muestra a continuación:

**C#**

using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using Microsoft.VisualStudio.TestTools.UnitTesting;  
using System.Web.Mvc;  
using MvcBasicWalkthrough;  
using MvcBasicWalkthrough.Controllers;  
  
namespace MvcBasicWalkthrough.Tests.Controllers  
{  
 [TestClass]  
 public class MapsControllerTest  
 {  
 [TestMethod]  
 public void ViewMaps()  
 {  
 // Arrange  
 MapsController controller = new MapsController();  
  
 // Act  
 ViewResult result = controller.ViewMaps() as ViewResult;  
  
 // Assert  
 Assert.IsNotNull(result);  
 }  
 }  
}

En **Explorador de soluciones**, seleccione el proyecto de prueba y presione CTRL+F5 para ejecutar las pruebas unitarias.

La prueba produce un error, porque el método de acción ViewMaps produce una excepción actualmente.

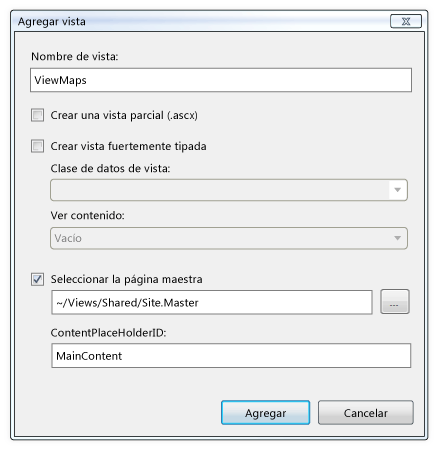
### Para agregar el código al método de acción

1. Abra la clase MapsController y reemplace el código auxiliar de método de acción ViewMaps con el código siguiente para representar la vista Maps:
2. **C#**
3. public ActionResult ViewMaps()  
   {  
    return View();  
   }

### **Para agregar una vista de contenido de página al proyecto de MVC**

1. Abra la clase MapsController, haga clic con el botón secundario del mouse dentro del método de acción ViewMaps y, a continuación, haga clic en **Agregar vista**.
2. Se mostrará el cuadro de diálogo **Agregar vista**.

<https://msdn.microsoft.com/es-es/library/dd410597(v=vs.100).aspx>

1. 
2. En el cuadro **Nombre de vista**, escriba **ViewMaps**.
3. Desactive las casillas **Crear una vista parcial (.ascx)** y **Crear una vista fuertemente tipada**.
4. Active la casilla **Seleccionar la página maestra** y establezca la página maestra en **~**/Views/Shared/Site.Master.
5. Establezca **ContentPlaceHolder ID** en "MainContent".
6. Haga clic en **Agregar**.
7. La nueva vista se agrega al proyecto en la carpeta Maps.

Crear una base de datos: <https://docs.microsoft.com/en-us/aspnet/visual-studio/overview/2013/aspnet-scaffolding-overview>

TUTORIAL: Crear un Proyecto MVC

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/getting-started>

Run your application and browse to the example URL ([http://localhost:xxxx/HelloWorld/Welcome?name=Scott&numtimes=4](about:blank))

## **Passing Data from the Controller to the View**

Currently, the Welcome action method in the HelloWorldController class takes a name and a numTimes parameter and then outputs the values directly to the browser. Rather than have the controller render this response as a string, let's change the controller to use a view template instead. The view template will generate a dynamic response, which means that you need to pass appropriate bits of data from the controller to the view in order to generate the response. You can do this by having the controller put the dynamic data (parameters) that the view template needs in a ViewBag object that the view template can then access.

Return to the *HelloWorldController.cs* file and change the Welcome method to add a Message and NumTimes value to the ViewBag object. ViewBag is a dynamic object, which means you can put whatever you want in to it; the ViewBag object has no defined properties until you put something inside it. The [ASP.NET MVC model binding system](http://odetocode.com/Blogs/scott/archive/2009/04/27/6-tips-for-asp-net-mvc-model-binding.aspx) automatically maps the named parameters (name and numTimes) from the query string in the address bar to parameters in your method. The complete *HelloWorldController.cs* file looks like this:

C#Copy

using System.Web;  
using System.Web.Mvc;  
  
namespace MvcMovie.Controllers  
{  
 public class HelloWorldController : Controller  
 {  
 public ActionResult Index()  
 {  
 return View();  
 }  
  
 public ActionResult Welcome(string name, int numTimes = 1)  
 {  
 ViewBag.Message = "Hello " + name;  
 ViewBag.NumTimes = numTimes;  
  
 return View();  
 }  
 }  
}

Now the ViewBag object contains data that will be passed to the view automatically. Next, you need a Welcome view template! In the Build menu, select Build Solution (or Ctrl+Shift+B) to make sure the project is compiled. Right click the *Views\HelloWorld* folder and click Add, then click MVC 5 View Page with Layout (Razor).

EN LA VISTA:

<ul>

@for (int i=0; i<=ViewBag.numtimes; i++) {

<li>ViewBag.message;</li>

}

</ul>

COMO ESTABLECER LA RUTA:

/[Controller]/[ActionName]/[Parameters]

You set the format for routing in the *App\_Start/RouteConfig.cs* file.

C#Copy

public static void RegisterRoutes(RouteCollection routes)  
{  
 routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");  
  
 routes.MapRoute(  
 name: "Default",  
 url: "{controller}/{action}/{id}",  
 defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }  
 );  
}

Run the application and browse to /localhost:XXX/HelloWorld/Welcome/Scott/3.

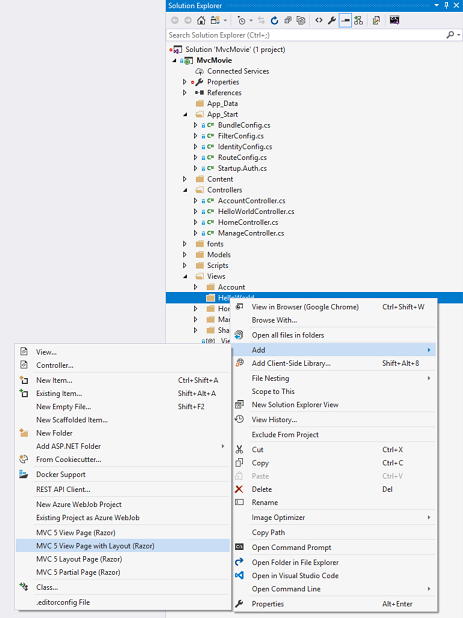
*App\_Start\RouteConfig.cs* file:

public class RouteConfig  
{  
 public static void RegisterRoutes(RouteCollection routes)  
 {  
 routes.IgnoreRoute("{resource}.axd/{\*pathInfo}");  
  
 routes.MapRoute(  
 name: "Default",  
 url: "{controller}/{action}/{id}",  
 defaults: new { controller = "Home", action = "Index", id = UrlParameter.Optional }  
 );  
  
 routes.MapRoute(  
 name: "Hello",  
 url: "{controller}/{action}/{name}/{id}"  
 );  
 }  
}

Crear una Vista:

Controller methods (also known as [action methods](http://rachelappel.com/asp.net-mvc-actionresults-explained)), such as the Index method above, generally return an [ActionResult](https://msdn.microsoft.com/library/system.web.mvc.actionresult.aspx) (or a class derived from [ActionResult](https://msdn.microsoft.com/library/system.web.mvc.actionresult.aspx)), not primitive types like string.

Right click the *Views\HelloWorld* folder and click Add, then click MVC 5 View Page with Layout (Razor).



Para mostrar la vista, el controlador debe tener:

public ActionResult Index()   
{   
 return View();   
}

ASIGNAR EN CADA PAGINA:

@{  
 ViewBag.Title = "Movie List";  
}

PARA LEER Y MOSTRAR EN EL LAYOUT:

<title>@ViewBag.Title - Movie APP</title>

TRABAJAR CON BASE DE DATOS:

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/creating-a-connection-string>

BASES DE DATOS:

Crear el modelo: Crear una clase con la estructura de la base de datos.

using System;  
using System.Data.Entity;  
  
namespace MvcMovie.Models  
{  
 public class Movie  
 {  
 public int ID { get; set; }  
 public string Title { get; set; }  
 public DateTime ReleaseDate { get; set; }  
 public string Genre { get; set; }  
 public decimal Price { get; set; }  
 }  
  
 public class MovieDBContext : DbContext  
 {  
 public DbSet<Movie> Movies { get; set; }  
 }  
}

Esto necesita el using System.Data.Entity; pero ese viene con el Nuget

[ScaffoldColumn(false)]:

[Range(0, 9999)]

[StringLength(50),Required]

Para crear una base de datos con el Entity Framework se debe usar:

using System.Data.Entity;

Para usarlo se debe adicionar esta referencia:

Another way to fix this problem (which does the same thing) is to right-click on your project, and select "Add Reference". Make sure you are on the .NET tab, and scroll down till you find "System.Data.Entity" and press OK.

Instalar el ENTITY DATA FRAMEWORK (si sirvió):

Para que funcione se debe insertar el paquete Entity Framework del Nuget.

Herramientas -> Administrador de packetes Nuget -> Administrar paquetes Nuget

->Examinar ->Entity FrameWork ->Buscar ->Instalar

Instalar el ENTITY DATA FRAMEWORK (No lo usé).

https://www.nuget.org/packages/EntityFramework/

By default, the Entity Framework looks for a connection string named the same as the object context class (MovieDBContext for this project).

Abrir el Web Config que aparece en el root:

Insertar esto dentro de <configuration>:

<connectionStrings>

<add name="MovieDBContext"   
 connectionString="Data Source=(LocalDb)\MSSQLLocalDB;Initial Catalog=aspnet-MvcMovie;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\Movies.mdf"   
 providerName="System.Data.SqlClient"   
/>

</connectionStrings>

The name of the connection string must match the name of the [DbContext](https://msdn.microsoft.com/library/system.data.entity.dbcontext(v=vs.103).aspx) class.

You don't actually need to add the MovieDBContext connection string. If you don't specify a connection string, Entity Framework will create a LocalDB database in the users directory with the fully qualified name of the [DbContext](https://msdn.microsoft.com/library/system.data.entity.dbcontext(v=vs.103).aspx) class (in this case MvcMovie.Models.MovieDBContext).

**ADICIONAR UN CONTROLADOR PARA LA BASE DE DATOS (esto no funcionó):**

* Select Movie (MvcMovie.Models) for the Model class.
* Select MovieDBContext (MvcMovie.Models) for the Data context class.
* For the Controller name enter MoviesController.

## **Creating the Movie Sample Database**

The Movie Database application uses a database table named Movies that contains the following columns:

|  |  |  |  |
| --- | --- | --- | --- |
| **Column Name** | **Data Type** | **Allow Nulls?** | **Is Primary Key?** |
| Id | int | False | True |
| Title | nvarchar(100) | False | False |
| Director | nvarchar(100) | False | False |

You can add this table to an ASP.NET MVC project by following these steps:

1. Right-click the App\_Data folder in the Solution Explorer window and select the menu option Add, New Item.
2. From the Add New Item dialog box, select SQL Server Database, give the database the name MoviesDB.mdf, and click the Add button.
3. Double-click the MoviesDB.mdf file to open the Server Explorer/Database Explorer window.
4. Expand the MoviesDB.mdf database connection, right-click the Tables folder, and select the menu option Add New Table.
5. In the Table Designer, add the fields: Id, Title, and Director columns.
6. Click the Save button (it has the icon of the floppy) to save the new table with the name Movies.
7. Clic en Actualizar
8. Clic en Actualizar Base de Datos

INSERTAR EL ADO.NET ENTITY DATA MODEL

Clic derecho en el proyecto.

->Agregar ->Nuevo elemento -> Data ->Entity Data Model -> seleccionar base de datos y tabla

cache

down vote

The error about not being able to load Microsoft.VisualStudio.Web.CodeGeneration.Utils assembly is usually due to NuGet package cache or due to version mismatches of packages in the project.

Can you show your *.csproj* file?

A few things to try:

1. Ensure the package reference for the scaffolding packages Microsoft.VisualStudio.Web.CodeGeneration.Design has the right version: a. 1.0.x for ASP.NET Core 1.0 and such.
2. Other scaffolding packages that shouldn’t be referenced directly are a. Microsoft.VisualStudio.Web.CodeGeneration.Utils b. Microsoft.VisualStudio.Web.CodeGeneration.Templating c. Microsoft.VisualStudio.Web.CodeGeneration.EntityFrameworkCore d. Microsoft.VisualStudio.Web.CodeGeneration.Core e. Microsoft.VisualStudio.Web.CodeGeneration f. Microsoft.VisualStudio.Web.CodeGeneration.MsBuild g. Microsoft.VisualStudio.Web.CodeGeneration.Contracts h. Microsoft.VisualStudio.Web.CodeGenerators.Mvc i. Microsoft.VisualStudio.Web.CodeGeneration.Tools -- **This one needs to be a DotNetCliToolReference**
3. Clear the local NuGet cache.
4. Clear the bin and obj folders of the project and restore the project.

///////////////////////////////////////////////////

PROCEDIMIENTO MVC (Desde el CÓDIGO):

1. Crear el MVC

2. Crear el modelo (carpeta models, add class)

3. compilar

4. Crear el controlador: clic derecho en controller ->add -> New scaffolded item

MVC 5 WITH

Tomar la clase creada antes en modelo

Insertar el contexto: Proyecto.Models.ProyectoContext

5. Agregar ->Nuevo elemento scaffolded -MVC ->MVC 5 Dependencies ->Full Dependencies. Aquì se crean las vistas del crud.

6. Instalar el ENTITY DATA FRAMEWORK (si sirvió):

Para que funcione se debe insertar el paquete Entity Framework del Nuget.

Herramientas -> Administrador de packetes Nuget -> Administrar paquetes Nuget

->Examinar ->Entity FrameWork ->Buscar ->Instalar

7. Insertar links a los nuevos items em el INDEX:

<div>  
 @Html.ActionLink("List of students", "Index", "Students")<br />  
 @Html.ActionLink("List of enrollments", "Index", "Enrollments")  
</div>

8. INSERTAR EL ADO.NET ENTITY DATA MODEL

Clic derecho en el proyecto.

->Agregar ->Nuevo elemento -> Data ->Entity Data Model -> seleccionar base de datos y tabla

[Empty EF Designer model:](http://www.entityframeworktutorial.net/model-first-with-entity-framework.aspx)

[EF Designer from Database:](https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/database-first-development/creating-the-web-application)

[Empty code first](https://www.youtube.com/watch?v=kbH-rqMl8cE)

Crear las clases para cada objeto

Public class Employee{}

Crear una clase:

Public class EmployeeDBContext : DbContext {

que necesita un: using System.Data.Entity;

public DbSet <Department>Departments {get;set;}

public DbSet <Employee>Employees {get;set;}

}

Insertar una nueva clase:

public class EmployeeRepository{

}

8. In the application Web.config file,

<connectionStrings>  
 <add name="SchoolContext" connectionString="Data Source=(LocalDb)\v11.0;Initial Catalog=ContosoUniversity2;Integrated Security=SSPI;" providerName="System.Data.SqlClient" />  
</connectionStrings>

MYSQL:

<connectionStrings> <add name="MyEntities" connectionString="metadata=res://\*/Data.Entity.Model.csdl|res://\*/Data.Entity.Model.ssdl|res://\*/Data.Entity.Model.msl;provider=MySql.Data.MySqlClient;provider connection string=&quot;Data Source=localhost;User Id=dbadmin;Password=password;Initial Catalog=database\_name;&quot;" providerName="System.Data.EntityClient"/> </connectionStrings>

TRABAJAR CON MYSQL EN VISUAL STUDIO .NET 2017

* Descargar conector MySQL

<https://dev.mysql.com/downloads/file/?id=480300>

* Agrega la referencia al conector de MySQL desde tu proyecto.

da click derecho en **Referencias** y luego en **Agregar referencias** -> Extensiones en el menú contextual:

En el menú que aparecerá, navega a Extensiones y selecciona la casilla de MySql.Data (MySql.Data.dll) y oprime en OK:

* Crea tu base de datos (ignora esto si ya tienes una) en MySQL con PHPMyAdmin.
* Comienza a ejecutar consultas.

Herramientas -> Administrador de paquetes de Nuget -> Consola de administrador de paquetes:

Enable-Migrations -ContextTypeName eManager.Web.Infrastructure.DepartmentDb.

enable-migrations -Force

enable-migrations

///////////////////////////////////////////////////

//////////////////////////////////////////////////

PROCEDIMIENTO MVC (Desde la Base de Datos): (Funciona O.K)

1. Crear el MVC

2. En la carpeta APP\_Data insertar Nuevo elemento ->Data -> Base de datos

3. Insertar las tablas a la base de datos

FOREIGN KEY:

Clic en Foreign Key -Add New

Y En la parte inferior se editan los campos a vincular:

CONSTRAINT [FK\_Reservas\_ToPeliculas] FOREIGN KEY (*Id\_Pelicula*) REFERENCES [*Peliculas*]([*Id\_Pelicula*])

Las bases de datos se editan en el: Explorador de objetos de SQL

4. ->Actualizar ->Actualizar Base de Datos

5. Instalar el ENTITY DATA FRAMEWORK (Funciona O.K.):

Para que funcione se debe insertar el paquete Entity Framework del Nuget.

Herramientas -> Administrador de packetes Nuget -> Administrar paquetes Nuget

->Examinar ->EntityFrameWork ->Buscar ->Instalar

6. Insertar el ADO Entity Data Model. EF Designer Desde Base de Datos. En la carpeta Models.

->Clic derecho en Models ->Agregar ->Nuevo Elemento->Data->ADO.NET Entity Data Model

Para ADICIONAR una nueva entidad (Nueva tabla en la BD) se debe ir a “Model.edmx” y hacer clic derecho en el fondo blanco y clic en *Actualizar Modelo desde Base de Datos*. ->Seleccionar la tabla nueva

7. Compilar

8.Insertar Controlador. Clic derecho en controller ->add -> New scaffolded item MVC 5 Controller WITH Views Using Entity Framework.

El Layout está en: ~/Views/Shared/\_Layout.cshtml

Tomar la clase creada antes en modelo

Insertar el contexto:

PeliculasEntities1 (MVC3.Models)

**EJECUTAR: Ctr+F5**

<https://docs.microsoft.com/en-us/aspnet/visual-studio/overview/2013/aspnet-scaffolding-overview>

//////////////////////////////////////////////////////////////////////////////////////////////////////////////

Fuente Libro: Pro ASP NET MVC

CAPITULO 6. MVC ,pag 119

Archivo: CARLOS\NET\Soluciones\Solucion1

Se utilizará Ninject como dependency injection(DI)

For unit testing, I am going to be using the built-in Visual Studio testing tools.

The third tool I selected is Moq, which is a mocking tool kit. Another alternative is Rhino Mocks, which is a nice alternative.

1. Crear un ASP.NET MVC Web Application con MVC.
2. Crear el modelo: Product.cs

namespace EssentialTools.Models

{

public class Product

{

public int ProductID { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public decimal Price { get; set; }

public string Category { set; get; }

}

}

1. Agregar otro modelo: LinqValueCalculator.cs que tien un método para calcular la suma de precios de productos.

using System.Collections.Generic;

using System.Linq;

namespace EssentialTools.Models {

public class LinqValueCalculator {

public decimal ValueProducts(IEnumerable<Product> products) {

return products.Sum(p => p.Price);

}

}

}

1. Agregar otro modelo: ShoppingCart

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace EssentialTools.Models

{

public class ShoppingCart

{

private LinqValueCalculator calc;

public ShoppingCart(LinqValueCalculator calcParam)

{

calc = calcParam;

}

public IEnumerable<Product> Products { get; set; }

public decimal CalculateProductTotal()

{

return calc.ValueProducts(Products);

}

}

}

1. Editar el HomeController

using System.Web.Mvc;

using System.Linq;

using EssentialTools.Models;

namespace EssentialTools.Controllers {

public class HomeController : Controller {

private Product[] products = {

new Product {Name = "Kayak", Category = "Watersports", Price = 275M},

new Product {Name = "Lifejacket", Category = "Watersports", Price = 48.95M},

new Product {Name = "Soccer ball", Category = "Soccer", Price = 19.50M},

new Product {Name = "Corner flag", Category = "Soccer", Price = 34.95M}

};

public ActionResult Index() {

LinqValueCalculator calc = new LinqValueCalculator();

ShoppingCart cart = new ShoppingCart(calc) { Products = products };

decimal totalValue = cart.CalculateProductTotal();

return View(totalValue);

}

}

}

1. Insertar la vista Index de HomeController:

@model decimal

...

<body>

<div>

Total value is $@Model

</div>

</body>

1. Usar Dependency Injection cuando las clases están estrechamente acopladas y es difícil reemplazar una clase por otra:

Crear la Interface:

Listing 6-6. The Contents of the IValueCalculator.cs File

using System.Collections.Generic;

namespace EssentialTools.Models {

public interface IValueCalculator {

decimal ValueProducts(IEnumerable<Product> products);

}

}

Modificar las otras clases:

7.1 public class LinqValueCalculator : IValueCalculator {

7.2 public class ShoppingCart {

private IValueCalculator calc;

public ShoppingCart(IValueCalculator calcParam) {

calc = calcParam;

}

Listing 6-9. Applying the Interface to the HomeController.cs File

...

public ActionResult Index() {

IValueCalculator calc = new LinqValueCalculator();

Instalar NINJECT:

Herramientas ->Administrador de paquetes de Nuget ->Consola de administrador de paquetes:

Install-Package Ninject -version 3.0.1.10

Install-Package Ninject.Web.Common -version 3.0.0.7

Install-Package Ninject.MVC3 -Version 3.0.0.6

Insertar: using Ninject; en el HomeController

insertar

public ActionResult Index()

{

IKernel ninjectKernel = new StandardKernel(); //Crea una instancia del Ninject kernel que crea objetos.

ninjectKernel.Bind<IValueCalculator>().To<LinqValueCalculator>();//Las dependencias de la interfaz IValueCalculator deben ser resueltas creando una instancia de la clase LinqValueCalculator

IValueCalculator calc = ninjectKernel.Get<IValueCalculator>();//Crea un objeto calc que es una instancia de la implementación de la interfaz:IValueCalculator -> LinqValueCalculator

Crear una carpeta Infrastructure y adicionar una clase: NinjectDependencyResolver.cs que es un Dependency resolver y sirve para crear una instancia de una clase.

using System;

using System.Collections.Generic;

using System.Web.Mvc;

using EssentialTools.Models;

using Ninject;

namespace EssentialTools.Infrastructure {

public class NinjectDependencyResolver : IDependencyResolver {

private IKernel kernel;

public NinjectDependencyResolver(IKernel kernelParam) {

kernel = kernelParam;

AddBindings();

}

public object GetService(Type serviceType) {

return kernel.TryGet(serviceType);

}

public IEnumerable<object> GetServices(Type serviceType) {

return kernel.GetAll(serviceType);

}

private void AddBindings() {

kernel.Bind<IValueCalculator>().To<LinqValueCalculator>();

}

}

}

Registrar el Dependency Resolver en la carpeta App\_Start archivo: NinjectWebCommon.cs

private static void RegisterServices(IKernel kernel) {

System.Web.Mvc.DependencyResolver.SetResolver(new

EssentialTools.Infrastructure.NinjectDependencyResolver(kernel));

}

Realizar cambios en el HomeController:

using System.Web.Mvc;

using EssentialTools.Models;

namespace EssentialTools.Controllers {

public class HomeController : Controller {

private IValueCalculator calc;

private Product[] products = {

new Product {Name = "Kayak", Category = "Watersports", Price = 275M},

new Product {Name = "Lifejacket", Category = "Watersports", Price = 48.95M},

new Product {Name = "Soccer ball", Category = "Soccer", Price = 19.50M},

new Product {Name = "Corner flag", Category = "Soccer", Price = 34.95M}

};

public HomeController(IValueCalculator calcParam) {

calc = calcParam;

}

public ActionResult Index() {

ShoppingCart cart = new ShoppingCart(calc) { Products = products };

decimal totalValue = cart.CalculateProductTotal();

return View(totalValue);

}

}

}

IMPLEMENTAR UNA INTERFAZ:

public class LinqValueCalculator: IValueCalculator {}

LinqValueCalculator implementa la interfaz: IValueCalculator

PUBLICAR UN MENSAJE EN CONSOLA:

System.Diagnostics.Debug.WriteLine(

string.Format("Instance {0} created", ++counter));

1. UNIT TEST:

En la Solución -> Agregar ->Nuevo Proyecto - Prueba ->Proyecto de prueba unitaria

1. Continuar en 145 con pruebas con Moq.

APLICACIÓN REAL DE ASP.NET MVC

Fuente Libro: Pro ASP NET MVC

CAPITULO 7. MVC ,pag 119

Archivo: CARLOS\NET\Soluciones\Solucion2

INYECCIÓN DE DEPENDENCIAS:

Sirve para construir aplicaciones con módulos independientes (clases, métodos, etc). Esto facilita las pruebas unitarias. Ninject es un “Contenedor de dependencias” que se encarga de inyectar las dependencias entre dichos módulos.

Explicación de Ninject y pruebas unitarias:

<http://pildorasdotnet.blogspot.com/2015/06/ninject-inyeccion-dependencias.html>

1. Insertar una solución en blanco: Achivo -Nuevo Proyecto-Otros tipos de Proyectos-Soluciones de Visual Studio
2. Crear tres proyectos:

**SportsStore.Domain:** Class Library (se encuentra en el menú .Net Standard) Holds the domain entities and logic; set up

for persistence via a repository created with

the Entity Framework.

**SportsStore.WebUI:** ASP.NET MVC Web Application (choose

Empty when prompted to choose a project template and check the MVC option)

Holds the controllers and views; acts as the UI for the SportsStore application.

**SportsStore.UnitTests:** Unit Test Project Holds the unit tests for the other two projects

Clic derecho sobre **SportsStore.WebUI** y seleccionar: Establecer como proyecto de inicio.

Clic derecho sobre **SportsStore.WebUI** y seleccionar propiedades ->Web ->Seleccionar la opción Página Específica y dejar en blanco.. Para que siempre inicie en la página index.

1. Instalar estos paquetes:

Install-Package Ninject -version 3.0.1.10 -projectname SportsStore.WebUI

Install-Package Ninject.Web.Common -version 3.0.0.7 -projectname SportsStore.WebUI

Install-Package Ninject.MVC3 -Version 3.0.0.6 -projectname SportsStore.WebUI

Install-Package Ninject -version 3.0.1.10 -projectname SportsStore.UnitTests

Install-Package Ninject.Web.Common -version 3.0.0.7 -projectname SportsStore.UnitTests

Install-Package Ninject.MVC3 -Version 3.0.0.6 -projectname SportsStore.UnitTests

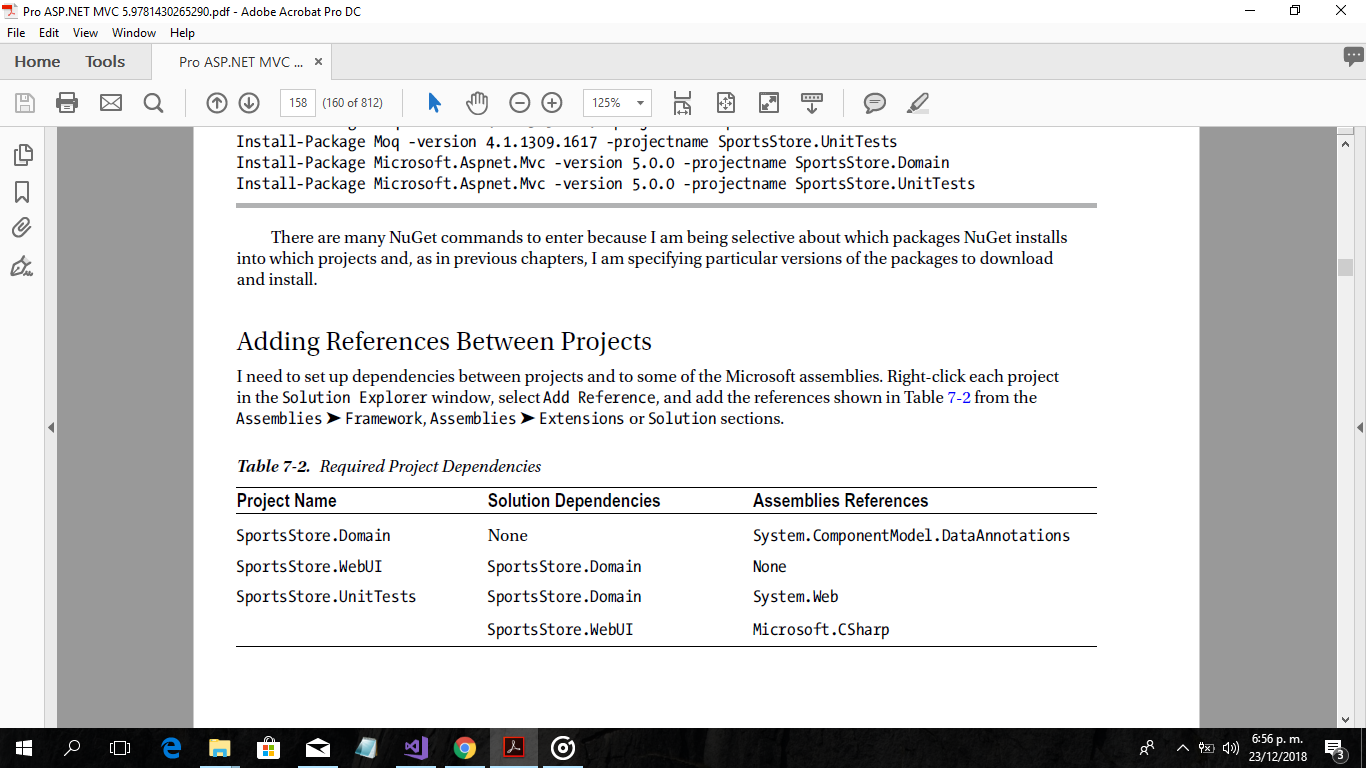
Install-Package Moq -version 4.1.1309.1617 -projectname SportsStore.WebUI

Install-Package Moq -version 4.1.1309.1617 -projectname SportsStore.UnitTests

Install-Package Microsoft.Aspnet.Mvc -version 5.0.0 -projectname SportsStore.Domain

Install-Package Microsoft.Aspnet.Mvc -version 5.0.0 -projectname SportsStore.UnitTests

1. Agregar las dependencias para cada proyecto:



1. En el proyecto SportsStore.WebUI project crear una carpeta infrastructure y crear una clase: NinjectDependencyResolver.cs

using System;

using System.Collections.Generic;

using System.Web.Mvc;

using Ninject;

namespace SportsStore.WebUI.Infrastructure {

public class NinjectDependencyResolver : IDependencyResolver {

private IKernel kernel;

public NinjectDependencyResolver(IKernel kernelParam) {

kernel = kernelParam;

AddBindings();

}

public object GetService(Type serviceType) {

return kernel.TryGet(serviceType);

}

public IEnumerable<object> GetServices(Type serviceType) {

return kernel.GetAll(serviceType);

}

private void AddBindings() {

// put bindings here

}

}

}

1. Añadir en la carpeta App\_start -> NinjectWebCommon.cs

...

private static void RegisterServices(IKernel kernel) {

**System.Web.Mvc.DependencyResolver.SetResolver(new**

**SportsStore.WebUI.Infrastructure.NinjectDependencyResolver(kernel));**

}

...

1. El Dominio de Modelo va a quedar en un proyecto independiente. En el proyecto SportsStore.Domain crear una carpeta Entities y un archivo: Product.cs

namespace SportsStore.Domain.Entities {

public class Product {

public int ProductID { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public decimal Price { get; set; }

public string Category { get; set; }

}

}

1. Crear una carpeta Abstract y una Interface: IProductsRepository.cs

using System.Collections.Generic;

using SportsStore.Domain.Entities;

namespace SportsStore.Domain.Abstract {

public interface IProductRepository {

IEnumerable<Product> Products { get; }

}

}

1. Añadir un repositorio ficticio (Mock) Mediante la implementación de la interfaz: IProductRepository en NinjectDependencyResolver.cs

**using System.Collections.Generic;**

**using System.Linq;**

**using Moq;**

**using SportsStore.Domain.Abstract;**

**using SportsStore.Domain.Entities;**

private void AddBindings() {

Mock<IProductRepository> mock = new Mock<IProductRepository>();

mock.Setup(m => m.Products).Returns(new List<Product> {

**new Product { Name = "Football", Price = 25 },**

**new Product { Name = "Surf board", Price = 179 },**

**new Product { Name = "Running shoes", Price = 95 }**

});

**kernel.Bind<IProductRepository>().ToConstant(mock.Object);**

}

Más información acerca de Moq:

<http://code.google.com/p/moq/wiki/QuickStart>.

1. Crear un controlador llamado: ProductController en el proyecto: SportStores.WebUI.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using SportsStore.Domain.Abstract;

using SportsStore.Domain.Entities;

namespace SportsStore.WebUI.Controllers {

public class ProductController : Controller {

private IProductRepository repository;

public ProductController(IProductRepository productRepository) {

this.repository = productRepository;

}

public ViewResult List()

{

return View(repository.Products);

}

}

}

1. Crear una vista para el método List

@using SportsStore.Domain.Entities

@model IEnumerable<Product>

@{

ViewBag.Title = "Productos";

}

<h2>List</h2>

@foreach (var p in Model) {

<div>

<h3>@p.Name</h3>

@p.Description

<h4>@p.Price.ToString("c")</h4>

</div>

}

1. En App\_start RouteConfig.cs editar el defaultRoute:

defaults: new { controller = "**Product**", action = "**List**",

id = UrlParameter.Optional }

HASTA AQUÍ FUNCIONA O.K.

1. INSTALAR:

Install-Package EntityFramework -projectname SportsStore.Domain

Install-Package EntityFramework -projectname SportsStore.WebUI

1. Crear un contexto para asociar el Modelo a la Base de Datos:

Crear una nueva carpeta en: SportsStore.Domain llamada Concrete e insertar una nueva clase: EFDbContext.cs

using SportsStore.Domain.Entities;

using System.Data.Entity;

namespace SportsStore.Domain.Concrete

{

public class EFDbContext : DbContext

{

public DbSet<Product> Products { get; set; }

}

}

Esta clase es derivada de System.Data.Entity.

DbContext. Y el DbSet tendrá una propiedad que es el nombre de cada tabla de la DB, el tipo será el modelo asociado a los datos de la tabla.

En este caso la propiedad es: Products y el tipo es Product.

La conexión a la DB se hace en SportsStore.WebUI en

el Web.config, adicionando un database

connection string

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<configSections>

<!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->

<section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />

</configSections>

<connectionStrings>

<add name="EFDbContext" connectionString="Data Source=(localdb)\v11.0;Initial

Catalog=SportsStore;Integrated Security=True"

providerName="System.Data.SqlClient"/>

</connectionStrings>

1. Crear el Repositorio Product en el SportsStore.Domain.Concrete:

using System.Collections.Generic;

using System.Text;

using SportsStore.Domain.Abstract;

using SportsStore.Domain.Entities;

namespace SportsStore.Domain.Concrete

{

class EFProductRepository:IProductRepository

{

private EFDbContext context = new EFDbContext();

public IEnumerable<Product> Products {

get { return context.Products; }

}

}

}

Esta clase es el repositorio, implementa la interfaz IProductRepository y usa una instancia de EFDbContext para obtener datos de la Base de Datos usando EntityFramework.

1. Editar en el NinjectDependencyResolver.cs el Mockrepository, cambiando el método: AddBindings

using SportsStore.Domain.Concrete;

private void AddBindings() {

**kernel.Bind<IProductRepository>().To<EFProductRepository>();**

}

1. error: No muestra los datos y

Para depurar en RAZOR: @ObjectInfo.Print(Model)

NINJECT WITH MVC

<https://www.c-sharpcorner.com/article/how-to-use-ninject-framework-with-mvc/>

CONNECTION STRING IN AN ASP.NET WEB FORM PROJECT:

<https://www.c-sharpcorner.com/UploadFile/7d3362/various-ways-to-specify-connection-string-in-Asp-Net-web-app/>

1. Instalar DAPPER desde el menú Nuget en la SportsStore.Domain

Dapper es un ORM

1. TRABAJAR CON LA PLANTILLA MvcNinject, solo es necesario crear los procedimientos almacenados y configurar el connectionString en el webConfig y las vistas.
2. A

CAPÍTULO SP

STORED PROCEDURE:

CREATE PROCEDURE seleccionarAyB

AS

SET NOCOUNT ON;

SELECT IdAlimentoYBebida, Representante, Participantes,FechaActividad

FROM Sales.SalesPerson AS sp

JOIN HumanResources.vEmployee AS e ON e.BusinessEntityID = sp.BusinessEntityID

RETURN

GO

\_\_\_\_\_\_\_\_

CREATE PROCEDURE [dbo].[Procedure]

@param1 int = 0,

@param2 int

AS

SELECT @param1, @param2

RETURN 0

CREATE PROCEDURE seleccionarAyB

AS

SET NOCOUNT ON;

SELECT IdAlimentoYBebida, Representante, Participantes,FechaActividad

FROM db\_RTVSS.dbo.tbAlimentoYBebida AS sp

RETURN

GO

DEVOLVER UNA TABLA VACÍA

SELECT TOP 0 \* FROM TableDepartamento

SP QUE SIEMPRE DEVUELVE UNA TABLA:

CREATE PROCEDURE [rtvss].[RTVSSSp\_IngresarCiudad]

@IdPais SMALLINT,

@IdDepartamento SMALLINT,

@Nombre VARCHAR (50),

@IdUsuarioRegistro SMALLINT

AS

BEGIN

IF ((SELECT COUNT([IdCiudad]) FROM [rtvss].[tbCiudad] WHERE Nombre=@Nombre AND IdPais= @IdPais AND IdDepartamento=@IdDepartamento)>0)

BEGIN

SELECT TOP 0 \* FROM [rtvss].[tbCiudad]

END

ELSE

BEGIN

INSERT INTO [rtvss].[tbCiudad] ([Codigo],[Nombre],[IdDepartamento],[IdPais],[Estado],[IdUsuarioRegistro],[FechaRegistro])

VALUES(CAST(SCOPE\_IDENTITY() AS VARCHAR(10)),@Nombre,@IdDepartamento,@IdPais,1,@IdUsuarioRegistro,GETDATE());

SELECT \* FROM [rtvss].[tbCiudad] WHERE IdPais=@IdPais

END

END

FECHA ACTUAL (CURRENT DATE)

GETDATE()

SP CONVERTIR UN SCOPE\_IDENTITY EN VARCHAR

CAST(SCOPE\_IDENTITY() AS VARCHAR(10))

STORED PROCEDURE QUE DEVUELVE UN REGISTRO

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ActualizarCiudad] Script Date: 4/06/2019 2:42:30 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [rtvss].[RTVSSSp\_ActualizarCiudad]

@IdCiudad INT,

@IdDepartamento SMALLINT,

@IdPais SMALLINT,

@Nombre VARCHAR(50),

@IdUsuarioActualizacion SMALLINT

AS

BEGIN

DECLARE @new\_identity INT

UPDATE [rtvss].[tbCiudad] SET [Nombre] = @Nombre, [IdUsuarioActualizacion]=@IdUsuarioActualizacion,

[FechaActualizacion] = GETDATE()

WHERE [IdCiudad] = @IdCiudad

SELECT \* FROM [rtvss].[tbCiudad] WHERE [Nombre] = @Nombre AND [IdCiudad]=@IdCiudad

ENDUSE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ActualizarCiudad] Script Date: 4/06/2019 2:42:30 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [rtvss].[RTVSSSp\_ActualizarCiudad]

@IdCiudad INT,

@IdDepartamento SMALLINT,

@IdPais SMALLINT,

@Nombre VARCHAR(50),

@IdUsuarioActualizacion SMALLINT

AS

BEGIN

DECLARE @new\_identity INT

UPDATE [rtvss].[tbCiudad] SET [Nombre] = @Nombre, [IdUsuarioActualizacion]=@IdUsuarioActualizacion,

[FechaActualizacion] = GETDATE()

WHERE [IdCiudad] = @IdCiudad

SELECT \* FROM [rtvss].[tbCiudad] WHERE [Nombre] = @Nombre AND [IdCiudad]=@IdCiudad

END

STORED PROCEDURE PARA INGRESAR USUARIO NUEVO: CALL STORED PROCEDURE (LLAMAR PROCEDIMIENTO DESDE OTRO:

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_InsertarUsuario] Script Date: 22/01/2019 4:44:49 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Carlos Alberto Sánchez>

-- Create date: <15/01/2019>

-- Description: <Guarda la informacion de un usuario Nuevo>

-- =============================================

ALTER PROCEDURE [rtvss].[RTVSSSp\_InsertarUsuario]

@NombresApellidos varchar(100),

@Usuario varchar(30),

@Correo varchar(100),

@IdTipoIdentificacion tinyint,

@NumeroIdentificacion varchar(30),

@IdCiudad smallint,

@IdPerfilUsuario smallint,

@IdUsuarioRegistro smallint,

@FechaRegistro datetime,

@GerenteRepresentante varchar(100),

@Clave varchar(50),

@new\_identity INT = NULL OUTPUT

AS

BEGIN

INSERT INTO rtvss.tbUsuarios

([NombresApellidos], [Usuario],[Correo],[IdTipoIdentificacion], [NumeroIdentificacion],[IdCiudad],[IdPerfilUsuario],[IdUsuarioRegistro],[FechaRegistro],[GerenteRepresentante],[Estado])

VALUES (@NombresApellidos, @Usuario, @Correo,@IdTipoIdentificacion,@NumeroIdentificacion, @IdCiudad,@IdPerfilUsuario,@IdUsuarioRegistro,@FechaRegistro,@GerenteRepresentante,1);

SET @new\_identity = SCOPE\_IDENTITY();

END

IF @new\_identity>0

BEGIN

EXEC [rtvss].[RTVSSSp\_ClaveUsuarioNuevo] @new\_identity, @Clave

END

GO

LEER UN VALOR DENTRO DEL SP

DECLARE @EXISTE INT

SELECT @EXISTE = iddtllconceptos

FROM rtvss.tbdtllconceptos

WHERE idcharlas = @IdCharlas

AND idconceptos = @IdConceptos

GENERAR CAMPO AUTONUMÉRICO EN SELECT

ROW\_NUMBER() OVER (ORDER BY fechatranferencia,tipotransferencia) as 'consecutivo'

Esta instrucción ordena y luego genera un campo autonumérico.

DEVOLVER VALOR EN CAMPO STORED PROCEDURE

SET @IdCharlas = @@IDENTITY;

select @IdCharlas AS IdCharlas;

y recibe:

\_Charlas = ConvertData.ConvertirDtoToList<DTCharlas>(DTCharlas);

Resultado.Respuesta = true;

Resultado.Resultado = \_Charlas.FirstOrDefault();

STORED PROCEDURE CON VARIABLE INTERNA:

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_CambiarClave] Script Date: 18/01/2019 11:27:02 a. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Carlos Alberto Sánchez>

-- Create date: <18/01/2019>

-- Description: <Consulta la informacion del usuario>

-- =============================================

ALTER PROCEDURE [rtvss].[RTVSSSp\_CambiarClave]

@Usuario VARCHAR(30),

@NuevaClave VARCHAR(30)

AS

DECLARE @IdUser SMALLINT

BEGIN

SET @IdUser= (SELECT TOP 1 U.[IdUsuario] FROM [db\_RTVSS].[rtvss].[tbUsuarios] AS U WHERE U.Usuario = @Usuario);

IF EXISTS(SELECT U.[IdUsuario] FROM [db\_RTVSS].[rtvss].[tbUsuarios] AS U WHERE U.Usuario = @Usuario)

UPDATE [rtvss].[tbClaves] SET [Estado]=0 WHERE [IdUsuario]=@IdUser

INSERT INTO [rtvss].[tbClaves] (Clave,IdUsuario,Estado,IntentosFallido,FechaCreacion)VALUES (@NuevaClave,@IdUser,1,0,GETDATE())

END

<https://docs.microsoft.com/es-es/sql/relational-databases/stored-procedures/specify-parameters?view=sql-server-2017>

CONCATENAR

SELECT IdCiudad, {fn CONCAT(P.Nombre +' ', {fn CONCAT(D.Nombre+' ' ,C.Nombre)}) } as Nombre ,C.Codigo, C.Nombre AS Ciudad, D.Nombre AS Departamento,

P.Nombre AS Pais, C.Descripcion, C.IdDepartamento, C.IdPais AS IdPais, C.Estado, C.IdUsuarioRegistro, C.FechaRegistro, C.IdUsuarioActualizacion, C.FechaActualizacion

FROM rtvss.tbCiudad C

(C.[Nombre]+ ' '+UPPER(LEFT(D.Nombre,9))) AS 'Descripcion'

## The SQL SELECT DISTINCT syntax

The general syntax is:

1. **SELECT DISTINCT column-name**
2. **FROM table-name**

Can be used with COUNT and other aggregates

1. **SELECT COUNT (DISTINCT column-name)**
2. **FROM table-name**

COMENTARIOS EN SP

UPDATE [rtvss].[tbDtllConceptos]

SET [Valor]=ISNULL(@Valor\_P,[Valor]),[FechaActualizacion]=GETDATE()

WHERE [IdPatrocinio]=@IdPatrocinio AND [IdConceptos]=@IdConceptos

--ESTO ES UN COMENTARIO EN SQL

--ESTO ES UN COMENTARIO EN SP

END

PARÁMETRO OPCIONAL EN STORED PROCEDURE

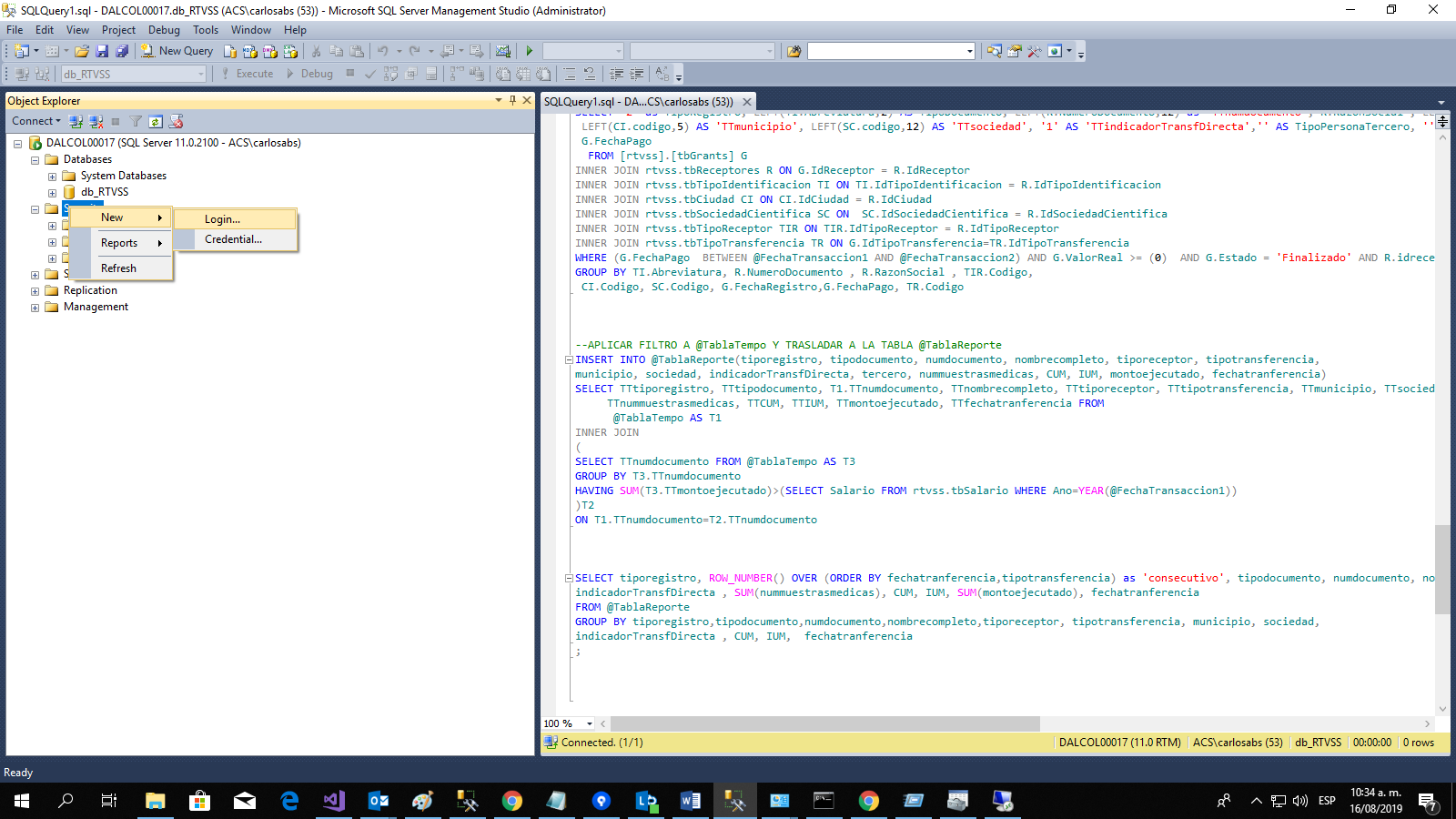
[IdTiposEventos]=**ISNULL**(@IdTiposEventos,[IdTiposEventos])

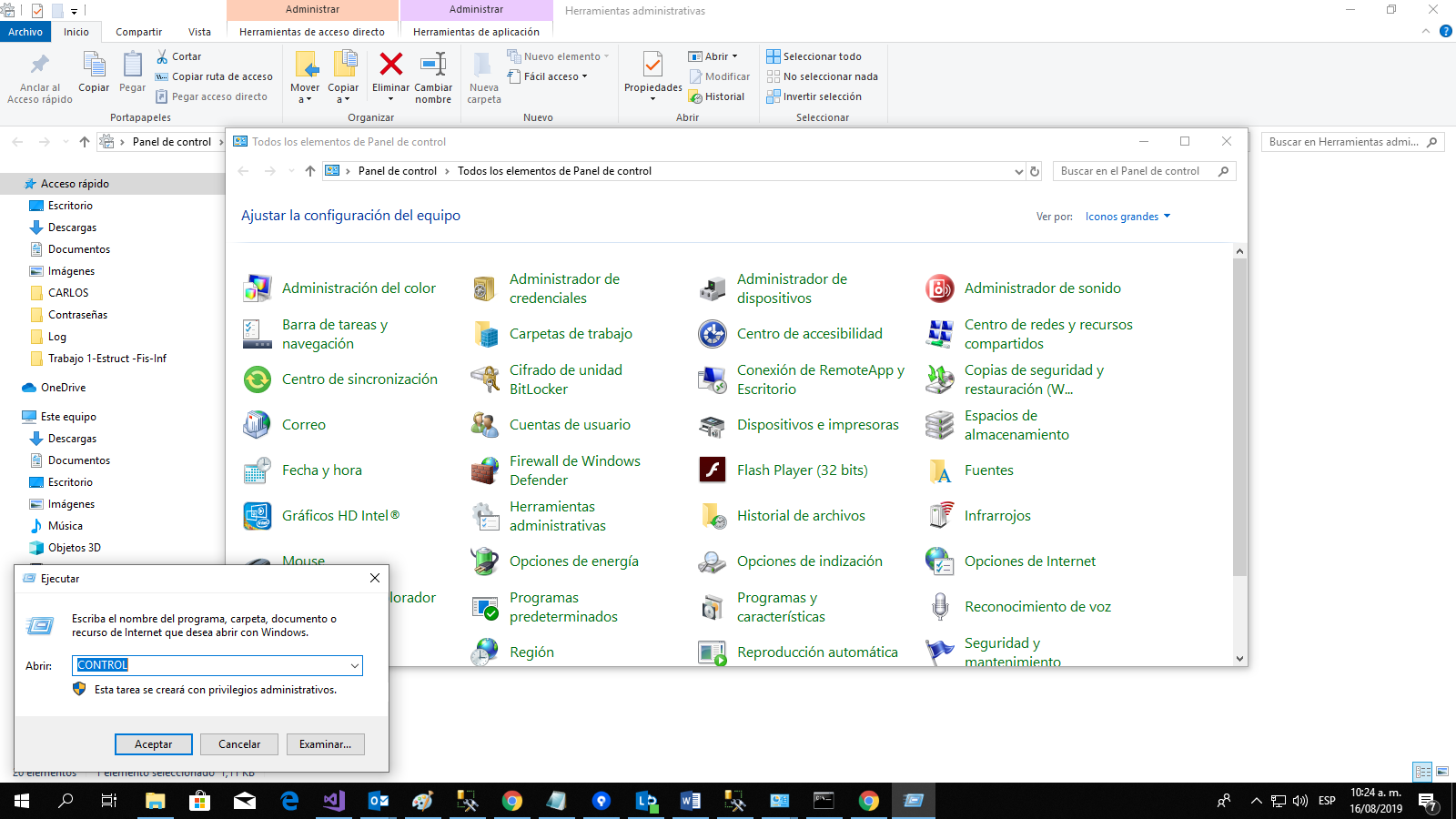
CONNECTION STRING CON AUTENTICACIÓN DE WINDOWS

Integrated Security=SSPI;

<add name="strConnectionRVSST" connectionString="Data Source=DALCOL00017;Initial Catalog=db\_RTVSS;Integrated Security=SSPI;MultipleActiveResultSets=True;" providerName="System.Data.SqlClient" />

USUARIO NUEVO EN SQL SERVER





STORED PROCEDURE CON IF ANIDADO:

<https://stackoverflow.com/questions/7595100/if-else-nesting-in-stored-procedures>

DEPURAR UN STORED PROCEDURE:

1. Abre una nueva ventana, escribe la ejecución del procedimiento almacenado, por ejemplo suponiendo que tu procedmiento se llama ObtieneUsuarios:
2. EXEC ObtieneUsuarios
3. Posiciona el cursor y teclea F9, esto para poner un punto de interrupción.
4. Dale clic al botón ► Depurar, o en su caso, ALT + F5
5. El proceso de depuración comenzará y ahora el texto EXEC ObtieneUsuarios se sombrea en color amarillo.
6. Para entrar a hacer depuración al *Stored Procedure* F11
7. El depurador entrará al *Stored Procedure*, para ir paso a paso deberás teclear F10
8. Para dejar correr la ejecución será nuevamente con ALT + F5
9. Puedes poner los puntos de interrupción que desees, pero cuando dejes correr la ejecución con la intrucción del punto 7, el depurador se detendrá justo en donde encuentre el siguiente punto de interrupción.
10. Para salir del modo de depuración, será necesario que termine la ejecución del *Stored Procedure*, o en su caso dar clic al botón detener pulsando ■

PROBAR UN STORED PROCEDURE CON FECHA

DECLARE @tmp DATETIME

SET @tmp = GETDATE()

EXEC [rtvss].[RTVSSSp\_InsertarUsuario]

@NombresApellidos='Daniela',

@Usuario='Dani',

@Correo='danielausu@algartec.com',

@NumeroIdentificacion=754425,

@IdPerfilUsuario=2,

@IdCiudad=2,

@IdUsuarioRegistro=3,

@FechaRegistro= @tmp;

PREGUNTAR SI UN CAMPO ES NULO (NULL) EN U PROCEDIMIENTO ALMACENADO

WHEN MATCHED AND (ISNULL(TIdentificacionReemplazante,'')!='') THEN

PROCEDIMIENTO ALMACENADO CON PARAMETROS OPCIONALES

ALTER PROCEDURE [dbo].[sp\_ClientNotes\_update]  
 @id uniqueidentifier,  
 @ordering smallint = NULL,  
 @title nvarchar(20) = NULL,  
 @content text = NULL  
AS  
BEGIN  
 SET NOCOUNT ON;  
 UPDATE tbl\_ClientNotes  
 SET ordering=ISNULL(@ordering,ordering),   
 title=ISNULL(@title,title),   
 content=ISNULL(@content, content)  
 WHERE id=@id  
END

**SP STORED PROCEDURE QUE CALCULA CAMPO Y LO CONVIERTE DE FORMATO (CAST)**

SELECT DLL.[IdPatrocinio], CAST(COUNT(DISTINCT DLL.[IdReceptor]) AS TINYINT) AS 'ParticipantesReales' FROM [rtvss].[tbDtllConceptos] DLL WHERE DLL.[IdPatrocinio]='35' AND DLL.[Aplica]=1 AND DLL.[EsValorFinal]=1 GROUP BY DLL.[IdPatrocinio]

**SP STORED PROCEDURE QUE DEVUELVE UN REGISTRO NUEVO**

USE [db\_RTVSS]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [rtvss].[RTVSSSp\_GuardarSalario]

@Ano INT,

@Salario DECIMAL,

@IdUsuarioRegistro SMALLINT

AS

DECLARE @new\_identity INT

DECLARE @CuentaActual INT

BEGIN

SET @CuentaActual = (SELECT COUNT (IdSalario) FROM [rtvss].[tbSalario] WHERE Ano=@Ano AND Estado=1);

IF @CuentaActual=0

BEGIN

INSERT INTO [db\_RTVSS].[rtvss].[tbSalario]([Salario],[Ano],[Estado],[IdUsuarioRegistro],[FechaRegistro])VALUES(@Salario,@Ano,1,@IdUsuarioRegistro,GETDATE());

SET @new\_identity = SCOPE\_IDENTITY();

-- RETURN @new\_identity;

END

ELSE

BEGIN

UPDATE [rtvss].[tbSalario] SET Salario=@Salario WHERE Ano=@Ano;

SET @new\_identity = SCOPE\_IDENTITY();

END

SELECT SCOPE\_IDENTITY() AS [IdSalario];

END

SP QUE GENERA UN NÚMERO ALEATORIO

UPDATE [db\_RTVSS].[rtvss].[tbProductosMedicamentos]

SET [IUM]='019994557-'+CAST(CAST(RAND(RIGHT(CodigoSap, 2))\*(10) AS INT) AS VARCHAR(2))

SP PARA CAMBIAR PRIMERA LETRA MAYUSCULA Y LAS OTRAS MINÚSCULA

USE [db\_RTVSS]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

DECLARE @intFlag INT

SET @intFlag = 1

WHILE (@intFlag <=1130)

BEGIN

UPDATE [rtvss].[tbCiudad]

SET [Nombre] = UPPER(LEFT([Nombre],1))+LOWER(SUBSTRING([Nombre],2,LEN([Nombre])))

WHERE [IdCiudad]=@intFlag

SET @intFlag = @intFlag + 1

END

GO

SP PARA ELIMINAR ESPACIOS EN CAMPO (INICIALES Y FINALES)

USE [db\_RTVSS]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

DECLARE @intFlag INT

DECLARE @Letra CHAR

DECLARE @Nombre VARCHAR(50)

SET @intFlag = 1

WHILE (@intFlag <=17)

BEGIN

UPDATE [rtvss].[tbCiudad]

SET [Nombre] = LTRIM(RTRIM([Nombre]))

WHERE [IdCiudad]=@intFlag

SET @intFlag = @intFlag + 1

END

GO

STORED PROCEDURE QUE DEVUELVE NÚMERO (ESCALAR):

Alter PROCEDURE [dbo].[sp\_Test]  
 @Action VARCHAR(1)  
 ,@No int = Null  
 ,@Code VARCHAR(10) = NULL  
 ,@Status VARCHAR(1) = Null  
 ,@Result INT OUTPUT  
AS  
DECLARE @Returns BIT   
  
BEGIN  
 SET NOCOUNT ON;  
   
 SET @Returns = 0  
 --INSERT  
 IF @Action = 'I'  
 BEGIN  
 INSERT INTO dbo.Test(Code,Status)  
 VALUES (@Code,'A')  
 SET @Returns = 1  
 END  
   
 --UPDATE  
 IF @Action = 'U'  
 BEGIN  
 UPDATE dbo.Test  
 SET Code = @Code WHERE [No] = @No  
 SET @Returns = 1  
 END  
 RETURN @Returns  
END

and in your code you can get it below like

string ReturnValue = com.ExecuteNonQuery().ToString();

### Re: Return Value from Stored Procedure After Insert/Update

Aug 20, 2018 05:41 AM|[LINK](https://forums.asp.net/post/6225231.aspx)

Instead of using ExecuteNonQuery() use ExecuteScaler(), will give you primary key as return,

cmd.CommandText = "SELECT COUNT(\*) FROM dbo.region";   
Int32 count = (Int32) cmd.ExecuteScalar();

SELECT IFNULL(TaskCode, 'N/A') AS Result   
FROM Tasks;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

TRY, CATCH y COMMIT.

view sourceprint?

01

INSERT INTO tblPruebas VALUES( 1 )

02

03

BEGIN TRY

04

INSERT INTO tblPruebas VALUES( 2 ) , ( 3 ) --b

05

BEGIN TRAN nombreTransaccion

06

INSERT INTO tblPruebas VALUES( 4 ) --c

07

INSERT INTO tblPruebas VALUES( 5 ) , ( 'a' ) --d

08

COMMIT TRAN nombreTransaccion

09

END TRY

10

BEGIN CATCH

11

SELECT ERROR\_NUMBER() AS errNumber

12

, ERROR\_SEVERITY() AS errSeverity

13

, ERROR\_STATE() AS errState

14

, ERROR\_PROCEDURE() AS errProcedure

15

, ERROR\_LINE() AS errLine

16

, ERROR\_MESSAGE() AS errMessage

17

18

ROLLBACK TRAN nombreTransaccion

19

20

SELECT \* FROM tblPruebas

21

22

END CATCH

**TRY CATCH**

Se utiliza para el manejo de errores detectados dentro de un bloque.

**COMMIT**

Es la finalización de nuestro bloque de código y todas las instrucciones dentro de él se llevarán a cabo siempre y cuando no se haya generado error alguno.

**ROLLBACK**

Sirve para deshacer todos los movimientos realizados dentro del bloque de la transacción del mismo nombre

INCLUIR FOREIGN KEY

USE [db\_RTVSS]

GO

ALTER TABLE [rtvss].[tbAlimentoYBebida] WITH CHECK ADD CONSTRAINT [FK\_IdCiudadAB] FOREIGN KEY([IdCiudad])

REFERENCES [rtvss].[tbCiudad] ([IdCiudad])

GO

ALTER TABLE [rtvss].[tbAlimentoYBebida] CHECK CONSTRAINT [FK\_IdCiudadAB]

GO

INCLUIR CLAVE PRIMARY

ALTER TABLE [rtvss].[tbCiudad] ADD PRIMARY KEY CLUSTERED

(

[IdCiudad] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, IGNORE\_DUP\_KEY = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

GO

MODIFICAR CAMPO EN TABLA:

USE [db\_RTVSS]

ALTER TABLE [rtvss].[tbReceptores] ALTER COLUMN [IdCiudad] INT NOT NULL

USE [db\_RTVSS]

GO

STORED PROCEDURE CON FUNCIÓN AGREGADA (SUM)

USE [db\_RTVSS]

GO

SELECT

P.[IdPatrocinio],

P.FechaActualizacion,

SUM(DLL.Valor) stock\_count

FROM

[rtvss].[tbPatrocinios] P

LEFT JOIN [rtvss].[tbDtllConceptos] DLL ON P.IdPatrocinio=DLL.IdPatrocinio

GROUP BY

P.[IdPatrocinio],P.FechaActualizacion

ORDER BY

P.[IdPatrocinio] DESC;

CONVERTIR FORMATO DE NÚMEROS

CONVERT(TINYINT,SOURCE.TIdTipoReceptor)

SP CON FUNCIÓN AGREGADA DE VARIAS TABLAS:

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ConsultaGeneralPatrocinios] Script Date: 29/05/2019 9:21:56 a. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Carlos Alberto Sánchez>

-- Create date: <06-Febrero-2019>

-- Description: <Consulta General de Patrocinios>

-- =============================================

ALTER PROCEDURE [rtvss].[RTVSSSp\_ConsultaGeneralPatrocinios]

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

SELECT

[IdPatrocinio]

,P.[NombreEvento]

,[GerenteResponsable]

,[IdLinea]

,P.[IdTiposEventos]

,[NombreEvento]

,[NParticipantesPresupuestales]

,P.[IdCiudad]

,[Especialidad]

,CONVERT(varchar(10),[FechaInicio]) as FechaInicio

,CONVERT(varchar(10),[FechaFin]) as FechaFin

,CONVERT(varchar(10),[FechaLimiteInscripcion]) as FechaLimiteInscripcion

,(select NumSolpedSap from [rtvss].[tbDtllSolped] WHERE IdPatrocinio = P.IdPatrocinio) AS NumSolped

,[NumSEC]

,[AplicaStand]

,[Status]

,[ValorStand]

,[NombreGerente]

,P.[FechaRegistro]

,[Ciudad]

,[Pais]

,[TipoEvento]

,Transporte

,Tiquetes\_Presup

FROM (SELECT P.[IdPatrocinio], P.[FechaRegistro],[NombreEvento],[GerenteResponsable],[IdLinea]

,P.[IdTiposEventos] ,[NParticipantesPresupuestales] ,P.[IdCiudad]

,[Especialidad], [FechaInicio],[FechaFin],[FechaLimiteInscripcion],[NumSEC] ,[AplicaStand]

,[Status] ,[ValorStand] ,U.[NombresApellidos] as 'NombreGerente'

,C.[Nombre] as Ciudad, PA.Nombre AS Pais, TE.Nombre as 'TipoEvento', Transporte, Tiquetes\_Presup

FROM

[rtvss].[tbPatrocinios] P

INNER JOIN [rtvss].[tbUsuarios] U ON P.[GerenteResponsable]= U.IdUsuario

INNER JOIN [rtvss].[tbCiudad] C ON P.IdCiudad = C.IdCiudad

INNER JOIN [rtvss].[tbTiposEventos] TE ON P.IdTiposEventos=TE.IdTiposEventos

INNER JOIN [rtvss].[tbDepartamento] DE ON C.IdDepartamento=DE.IdDepartamento

INNER JOIN [rtvss].[tbPais] PA ON DE.IdPais = PA.IdPais

LEFT JOIN (SELECT DLL.[IdPatrocinio],SUM( DLL.Valor) AS Transporte FROM [rtvss].[tbDtllConceptos] DLL WHERE DLL.[IdConceptos]='3' GROUP BY DLL.[IdPatrocinio]) AS A

ON A.[IdPatrocinio]= P.IdPatrocinio

LEFT JOIN (SELECT DLL.[IdPatrocinio],SUM( DLL.Valor) AS Tiquetes\_Presup FROM [rtvss].[tbDtllConceptos] DLL WHERE DLL.[IdConceptos]='4' AND DLL.EsValorFinal='0' GROUP BY DLL.[IdPatrocinio]) AS B

ON B.[IdPatrocinio]= P.IdPatrocinio

)AS P

ORDER BY P.[FechaRegistro] DESC

END

STORED PROCEDURE CON MANEJADOR DE ERRORES

AS

BEGIN TRY

SELECT TOP 4 [IdClave]

,[Clave]

,[IdUsuario]

,[Estado]

,[IntentosFallido]

,[FechaCreacion]

FROM [db\_RTVSS].[rtvss].[tbClaves]

WHERE [IdUsuario] = @IdUsuario ORDER BY [FechaCreacion] DESC;

END TRY

BEGIN CATCH

SELECT ERROR\_NUMBER() AS errNumber

END CATCH

SP CON TABLA INTERMEDIA

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ReporteMinSalud] Script Date: 20/03/2019 4:17:58 p. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [rtvss].[RTVSSSp\_ReporteMinSalud]

@FechaTransaccion1 varchar(20),

@FechaTransaccion2 varchar(20)

as

DECLARE @TablaReporte TABLE

(

tiporegistro CHAR(1) NOT NULL,

consecutivo INT NOT NULL PRIMARY KEY IDENTITY(1, 1),

tipodocumento CHAR(2),

numdocumento VARCHAR(12),

nombrecompleto VARCHAR(300),

);

DECLARE @TablaTempo TABLE

(

TTtiporegistro CHAR(1) NOT NULL,

TTtipodocumento CHAR(2),

TTnumdocumento VARCHAR(12),

TTnombrecompleto VARCHAR(300),

TTtiporeceptor CHAR(2),

TTtipotransferencia CHAR(2),

);

INSERT INTO @TablaReporte

SELECT '2', TI.abreviatura, R.numerodocumento, R.razonsocial, TIR.codigo, TR.codigo, C.codigo, SC.codigo, RT.indicadorrecibiotransferencia, RT.tipopersonatercero,

M.nummuestrasentregadas, PM.cum, PM.ium, '0', M.fechatransferencia

FROM rtvss.tbmuestrasmedicas M INNER JOIN rtvss.tbreceptores R

ON R.idreceptor = M.idreceptor INNER JOIN rtvss.tbtipoidentificacion TI

ON TI.idtipoidentificacion = R.idtipoidentificacion INNER JOIN rtvss.tbreportetercero RT

WHERE M.FechaTransferencia BETWEEN @FechaTransaccion1 AND @FechaTransaccion2;

);

-- Pasar información de una tabla a otra:

INSERT INTO @TablaReporte

select \* from @TablaTempo

where TTmontoejecutado > (select Salario \* 3 from rtvss.tbSalario) AND TTfechatranferencia BETWEEN @FechaTransaccion1 AND @FechaTransaccion2;

-- Retornar información:

SELECT \* from @TablaReporte;

SP CON SUBCONSULTA

SELECT Products.\* FROM Products

INNER JOIN

(

SELECT ProductID, Sum(Quantity) as QuantitySum from ( SELECT ProductID, Quantity FROM BasketItems UNION ALL SELECT ProductID, Quantity FROM OrderItems ) v

GROUP BY ProductID

) ProductTotals ON Products.ID = ProductTotals.ProductID

ORDER BY QuantitySum DESC

ISNULL

Use ISNULL() for sql server.

select s.IDENTIFIERS, s.store\_id, isnull(d.SALES\_PRICE,isnull(u.SALES\_PRICE,s.SALES\_PRICE)) as SALES\_PRICE from BOOK\_SALES as s

SP CON CAMPO CALCULADO EN FUNCIÓN DE OTRO CAMPO:

ALTER PROCEDURE [rtvss].[RTVSSSp\_IngresarPais]

@Nombre VARCHAR (50),

@Descripcion VARCHAR (10),

@IdUsuarioRegistro SMALLINT,

@new\_identity\_SM SMALLINT = NULL OUTPUT

AS

BEGIN

SET NOCOUNT ON;

SET @new\_identity\_SM = (SELECT TOP 1 [IdPais] FROM [rtvss].[tbPais] ORDER BY [IdPais] DESC);

--SET @new\_identity\_SM = CAST(@new\_identity\_VC AS SMALLINT);

SET @new\_identity\_SM = @new\_identity\_SM +1

INSERT INTO [rtvss].[tbPais] ([Nombre],[Codigo],[Descripcion],[Estado],[IdUsuarioRegistro],[FechaRegistro])

VALUES(@Nombre,CAST(@new\_identity\_SM AS VARCHAR(10)),UPPER(LEFT(@Descripcion,3)),1,@IdUsuarioRegistro,GETDATE());

--SET @new\_identity = SCOPE\_IDENTITY();

SELECT \* FROM [rtvss].[tbPais] WHERE [Nombre]=@Nombre;

END

CONTAR CAMPO CON VALORES NULOS

SELECT COUNT(consecutivo) AS 'consecutivo' FROM @TablaReporte WHERE ([numidentificacion] IS NULL)

CONTAR CAMPO CON CONTENIDO VACÍO

SELECT COUNT(consecutivo) AS 'consecutivo' FROM @TablaReporte WHERE LEN([numidentificacion])=0

BORRAR CONTENIDO DE TABLA

TRUNCATE TABLE tbl\_usuarios;

BORRAR CONTENIDO DE BASE DE DATOS

**BORRAR TABLAS:**

USE [db\_RTVSS]

EXEC sp\_MSForEachTable 'ALTER TABLE ? NOCHECK CONSTRAINT ALL'

GO

EXEC sp\_MSforeachtable @command1 = "DROP TABLE ?"

-- Ahora volver a habilitar la integridad referencial

EXEC sp\_MSForEachTable 'ALTER TABLE ? CHECK CONSTRAINT ALL'

GO

**BORRAR TODOS LOS SP:**

1. Click on Stored Procedures Tab.
2. Press f7 to Display All Stored Procedures.
3. Select All Procedure By Ctrl + A except System Table.
4. Press Delete button and Click OK.

INSERTAR CAMBIOS EN TABLA TEMPORAL:

DECLARE @SummaryOfChanges TABLE(Change VARCHAR(20));

--Luego de la actualización o inserción:

OUTPUT $action INTO @SummaryOfChanges;

--Entregar Cantidad de registros:

SELECT COUNT(\*) as ContRegistros

FROM @SummaryOfChanges

GROUP BY Change;

GENERAR SCRIPT DE BASE DE DATOS CON DATOS:

En SQL server:

1. Right-click on your database and select Tasks > Generate Scripts
2. In the Generate and Publish Scripts wizard, select the "Select specific database objects" option
3. Expand the "Tables" tree and select all of the tables you wish to export the scheme and data for, then click Next
4. In the next screen choose how you wish to save the script (the Output Type must remain set as "Save scripts to a specific location"), then click the Advanced button in the top right corner
5. Click on Advenced:



Choose Type of data to script: **schema and data**

1. In the newly opened window, under the General section is a setting called "Types of data to script", set this to "Scheme and data" and click OK
2. Click Next, review the export summary and click Next again. This will generate the script to your selected destination.

CAMBIAR NULL EN UN CAMPO DE UNA TABLA

USE [db\_RTVSS]

GO

ALTER TABLE [rtvss].[tbPatrocinios]

ALTER COLUMN [Especialidad] VARCHAR(100) NULL;

BORRAR REGISTROS DE TABLA:

delete from usuarios;

INSERTAR XML EN BD:

1. Leer el archivo XML en EXCEL y guardarlo como texto delimitado por Tabulaciones
2. En SQL Server Hacer clic en la BD → Task → Import Data → Next → Seleccionar Flat File Source → Ingresar la ubicación del archivo.

ESTABLECER UNA NUEVA RELACIÓN ENTRE DOS TABLAS:

ALTER TABLE [rtvss].[tbGrants] WITH CHECK ADD CONSTRAINT [FK\_IdTipoTransferenciaGrants] FOREIGN KEY([IdTipoTransferencia])

REFERENCES [rtvss].[tbTipoTransferencia] ([IdTipoTransferencia])

GO

ALTER TABLE [rtvss].[tbGrants] CHECK CONSTRAINT [FK\_IdTipoTransferenciaGrants]

GO

**ELIMINAR RESTRICCIÓN DE CLAVE ÚNICA (UNIQUE KEY)**

System.Data.SqlClient.SqlException: 'Violation of UNIQUE KEY constraint 'UQ\_\_tbPatroc\_\_35AFFDDF7FE0DB9D'. Cannot insert duplicate key in object 'rtvss.tbPatrocinios'.

The statement has been terminated.'

alter table [rtvss].[tbPatrocinios]

drop constraint 'UQ\_\_tbPatroc\_\_35AFFDDF7FE0DB9D;

**ELIMINAR RESTRICCIÓN DE NOT NULL**

USE [db\_RTVSS]

ALTER TABLE [db\_RTVSS].[rtvss].[tbCiudad] ALTER COLUMN [Codigo] varchar(10) NULL;

Si el campo es PRIMARY KEY entonces se debe desplegar la carpeta ***Keys*** y eliminar la restricción.

# **RETURN (Transact-SQL)**

SE APLICA A: síSQL Server (a partir de 2008) síAzure SQL Database noAzure SQL Data INSERT INTO XMLwithOpenXML(XMLData, LoadedDateTime)

SELECT CONVERT(XML, BulkColumn) AS BulkColumn, GETDATE()

FROM OPENROWSET(BULK 'D:\OpenXMLTesting.xml', SINGLE\_BLOB) AS x;

SELECT \* FROM XMLwithOpenXMLWarehouse noAlmacenamiento de datos paralelos

Sale incondicionalmente de una consulta o procedimiento. RETURN es inmediata y completa, y se puede utilizar en cualquier punto para salir de un procedimiento, lote o bloque de instrucciones. Las instrucciones que siguen a RETURN no se ejecutan.

RETORNAR ERRORES DE UN SP:

<https://sqlpsykrest.wordpress.com/2009/01/22/sql-procedimiento-almacenado-para-insertar-datos-con-transaccin-y-control-de-errores/>

**How to return the total records count returned by a stored procedure?**

SELECT COUNT(\*) As Cant FROM usuarios where usuario=?usuario AND password=?pass";

SP DE BÚSQUEDA ORDENADA

ALTER PROCEDURE

[rtvss].[RTVSSSp\_FiltrarReceptores]

@RazonSocial VARCHAR(40)=NULL,

@NumeroDocumento VARCHAR(30)=NULL

AS

IF (@RazonSocial IS NOT NULL)

BEGIN

SELECT [IdReceptor],[NumeroDocumento],[RazonSocial],[IdSociedadCientifica], C.[Nombre]AS 'Ciudad',R.[Estado], I.Abreviatura AS 'TipoDocumento'

FROM [rtvss].[tbReceptores] AS R

INNER JOIN [rtvss].[tbCiudad] AS C ON R.[IdCiudad]=C.[IdCiudad]

INNER JOIN [rtvss].[tbTipoIdentificacion] AS I ON R.IdTipoIdentificacion=I.IdTipoIdentificacion

WHERE ([RazonSocial] LIKE @RazonSocial+'%' AND R.[Estado]=1) ORDER BY [RazonSocial] ASC;

END

ELSE

BEGIN

SELECT [IdReceptor],[NumeroDocumento],[RazonSocial],[IdSociedadCientifica], C.[Nombre]AS 'Ciudad',R.[Estado], I.Abreviatura AS 'TipoDocumento'

FROM [rtvss].[tbReceptores] AS R

INNER JOIN [rtvss].[tbCiudad] AS C ON R.[IdCiudad]=C.[IdCiudad]

INNER JOIN [rtvss].[tbTipoIdentificacion] AS I ON R.IdTipoIdentificacion=I.IdTipoIdentificacion

WHERE ([NumeroDocumento] LIKE @NumeroDocumento+'%' AND R.[Estado]=1) ORDER BY [NumeroDocumento] ASC;

END

CREAR ESQUEMA EN BASE DE DATOS

CREATE SCHEMA db\_NovoTransferenciaValor

En la Pestaña: Security - Schemas es posible crear un nuevo esquema:

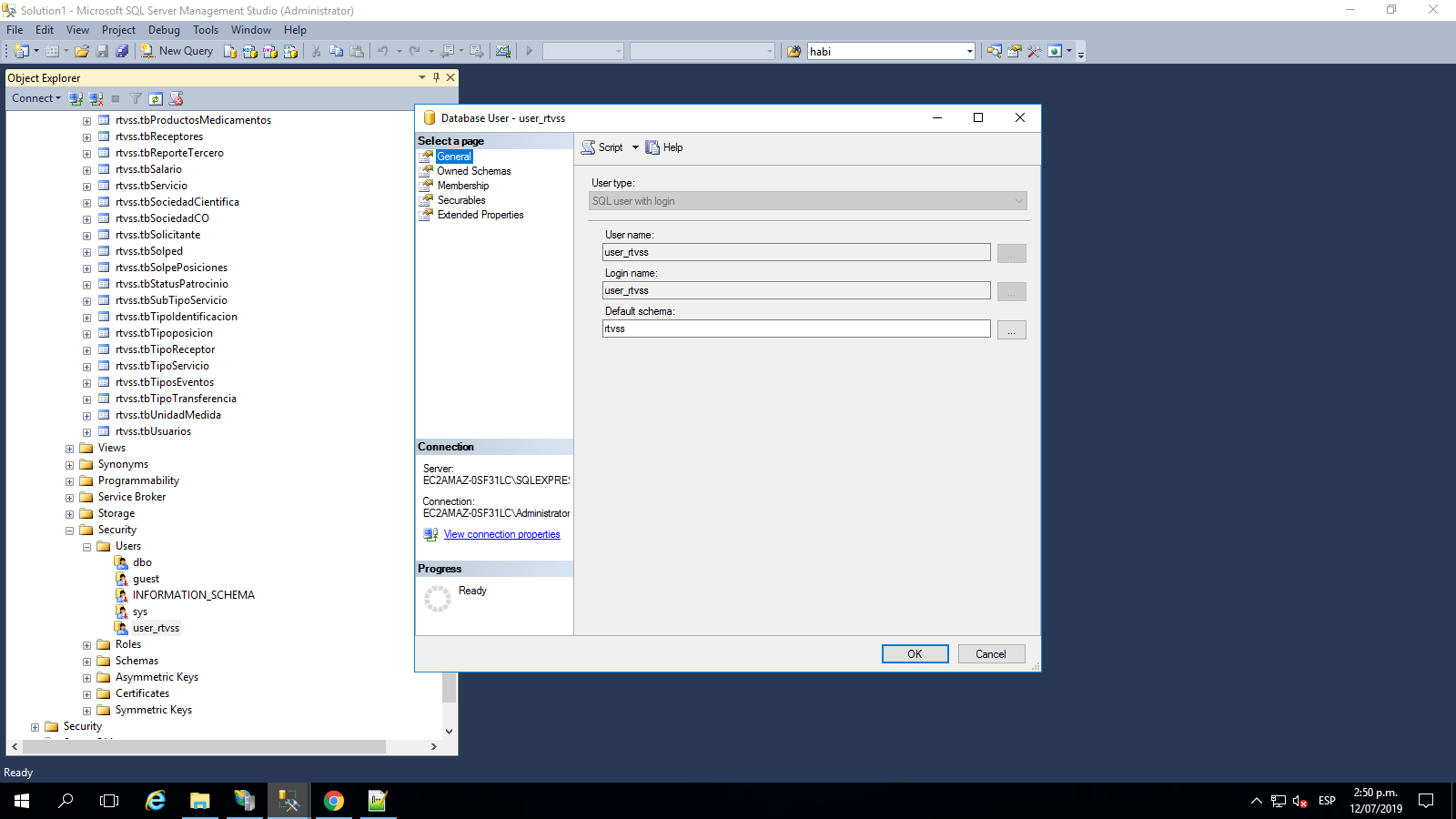
Ejemplo:

[rtvss].[tbAlimentoYBebida]

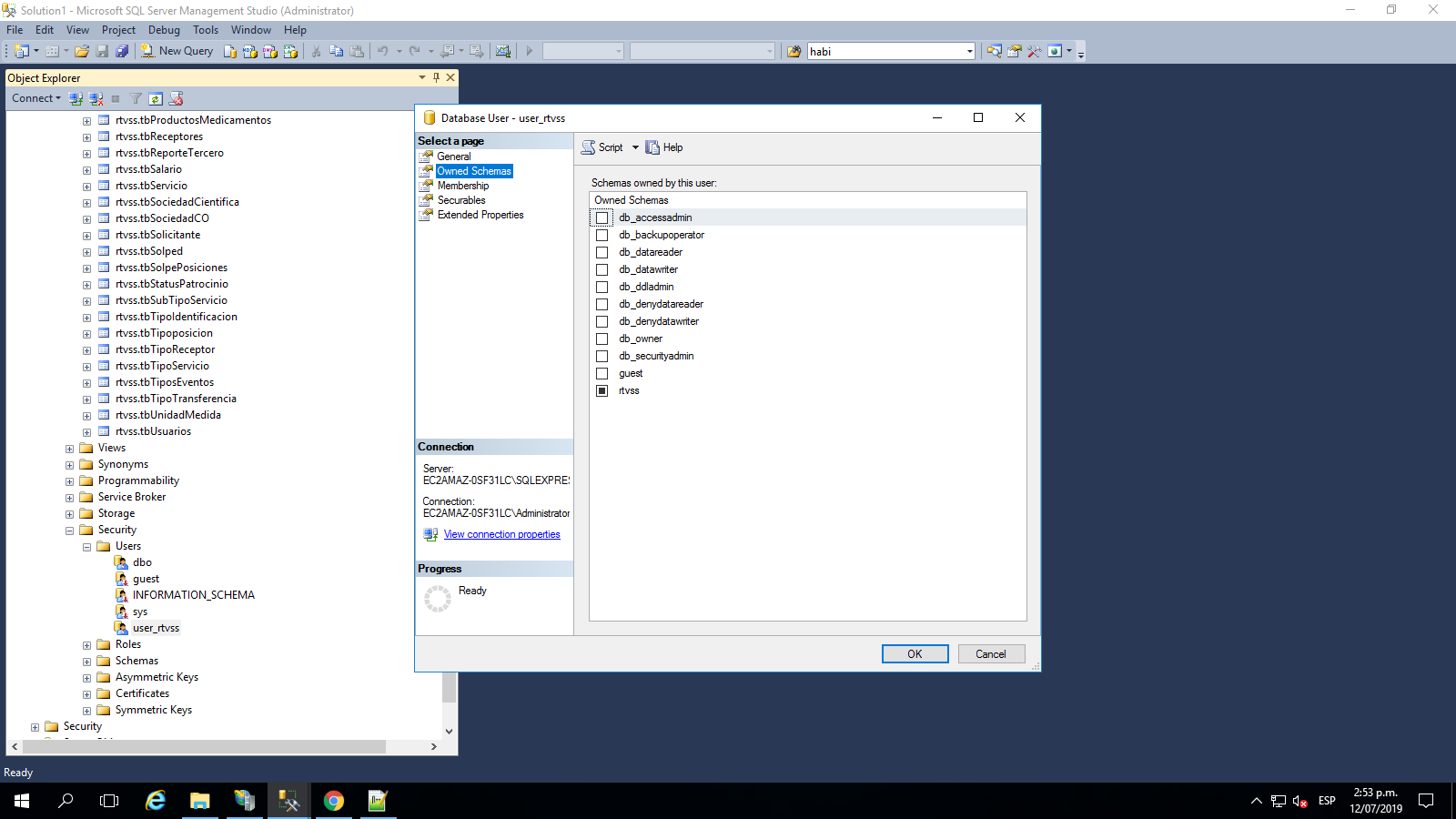
[rtvss]: Es el esquema

USUARIO DE LA BASE DE DATOS:

Security - Users - New User:



Owned Schema: rtvss



ELIMINAR CONEXIONES ABIERTAS EN BASE DE DATOS:

USE master

GO

SET NOCOUNT ON

DECLARE @DBName varchar(50)

DECLARE @spidstr varchar(8000)

DECLARE @ConnKilled smallint

SET @ConnKilled=0

SET @spidstr = ''

Set @DBName = 'dB\_RTVSS222XX'

IF db\_id(@DBName) < 4

BEGIN

PRINT 'Las conexiones a la base de datos de sistema no puedes ser eliminadas.'

RETURN

END

SELECT @spidstr=coalesce(@spidstr,',' )+'kill '+convert(varchar, spid)+ '; '

FROM master..sysprocesses WHERE dbid=db\_id(@DBName)

IF LEN(@spidstr) > 0

BEGIN

EXEC(@spidstr)

SELECT @ConnKilled = COUNT(1)

FROM master..sysprocesses WHERE dbid=db\_id(@DBName)

END

OTORGAR PERMISOS A UN USUARIO EN BASE DATOS

use BD\_RTVSS\_XX

GO

GRANT EXECUTE TO user\_qet

GO

# [Exit single-user mode](https://stackoverflow.com/questions/18965980/exit-single-user-mode)

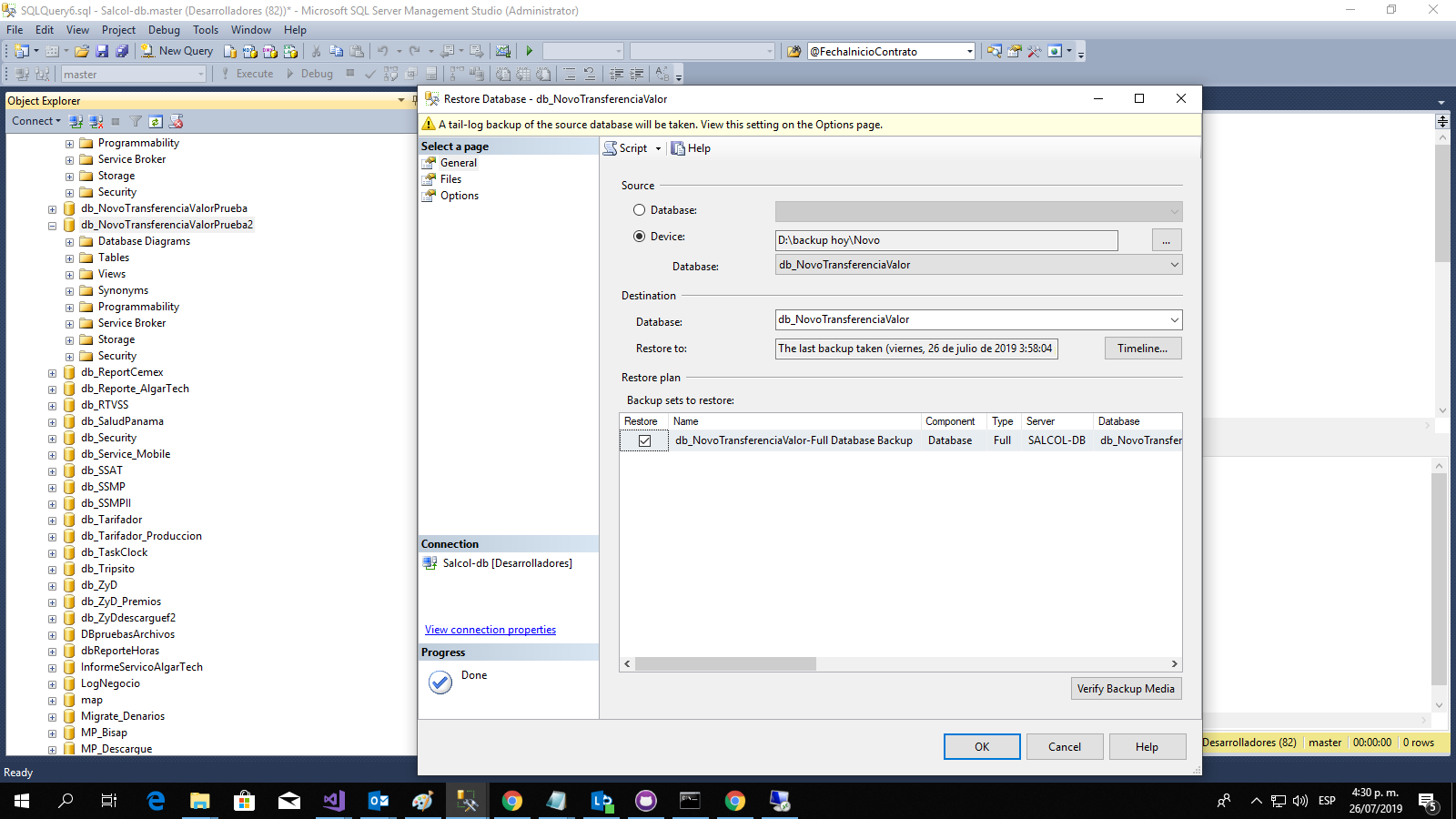
USE MASTER;

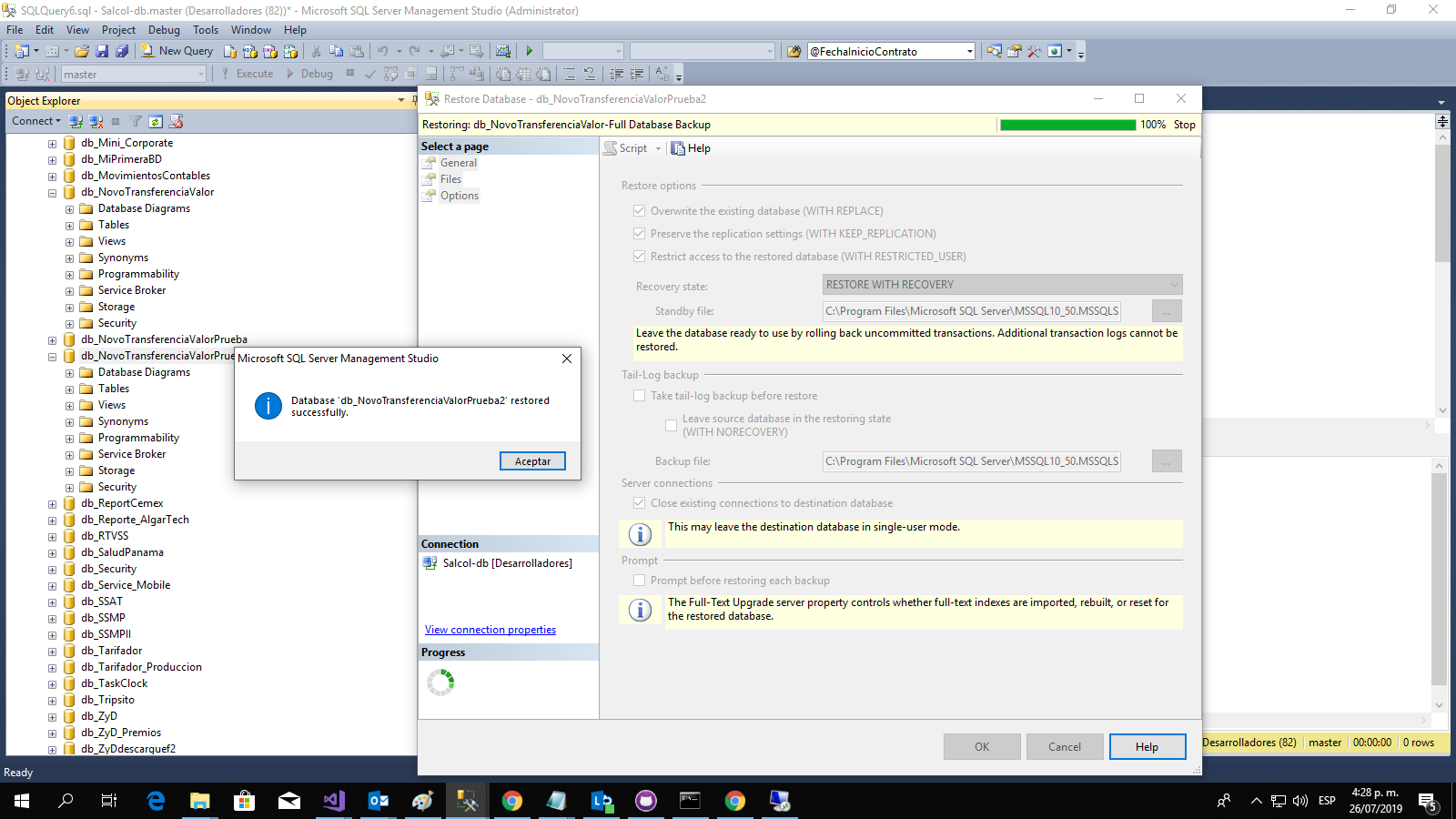
-- Add users

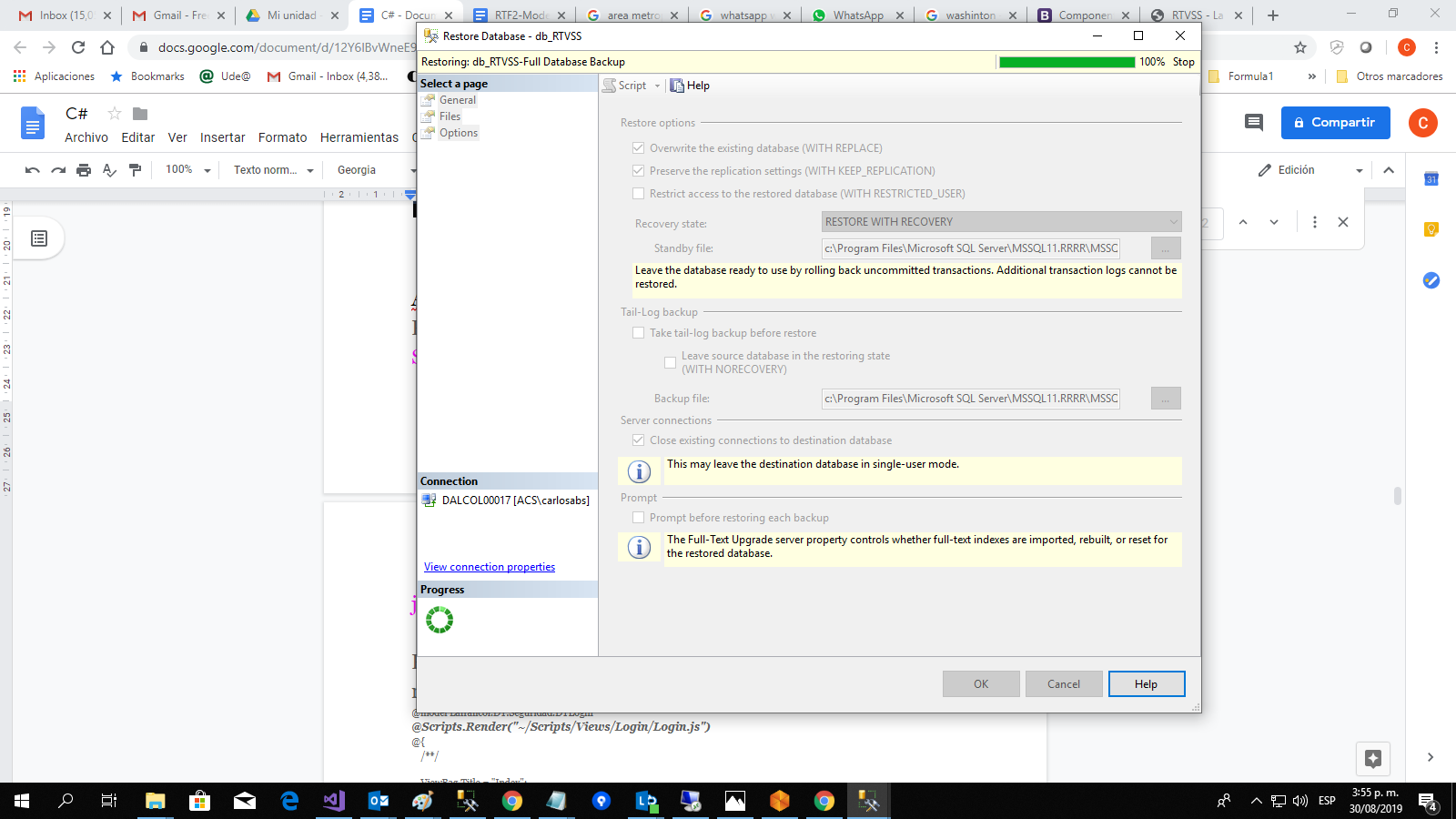
ALTER DATABASE LAFRANCOL SET MULTI\_USER

GO

RESTAURAR BD DESDE BACK UP







ALGARTECH:

Para las validaciones de la vista:

Se debe instalar la librería de Validaciones desde Nuget:

jQuery Validation.

Para vincular un archivo js, En la vista Se debe insertar la ruta:

@model Lafrancol.DT.Seguridad.DTLogin

***@Scripts.Render("~/Scripts/Views/Login/Login.js")***

@{

/\*\*/

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_LayoutLogin.cshtml";

}

El archivo js debe estar en la carpeta Scripts.

# Para que las librerías de clase se puedan ver en otros proyectos:

# Se debe instalar:

# Microsoft.Extensions.Configuration

NewTonsoft.Json

Referencia a system.data.datasetExtensions

Referencia a system.Rumtime.data.serialization

System.Numerics

System.Numerics.Vector

System.xml

System.xml.Linq

JAVASCRIPT:

<http://www.codedigest.com/posts/35/adding-javascript-in-aspnet-mvc-views>

function saveData() {

var pLogin = new Object();

pLogin.UsuarioLFC = $("#UsuarioLFC").val();

pLogin.ClaveLFC = $("#claveLFC").val();

}

@section scripts {

<script type="text/javascript">

$(document).ready(function () {

$("#btnLogin").click(function () {

var pLogin = new Object();

pLogin.UsuarioLFC = $("#UsuarioLFC").val();

pLogin.ClaveLFC = $("#ClaveLFC").val();

alert('You clicked ' + pLogin.UsuarioLFC + "!" + pLogin.ClaveLFC);

if (VentaAdicional.ValidarFormulario()) {

alert('Validación Exitosa ');

VentaAdicional.ValidarCredenciales();

} else {

alert('Por favor verifique las credenciales');

}

window.location.href = '/Home/Index/';

});

});

</script>

}

<script src="http://code.jquery.com/jquery-3.3.1.min.js"></script>

$('#btnSubmit').on('click', myFunction);

$('#btnSubmit').triggerHandler('click');

<script>

$(document).ready(function () {

initialize();

});

</script>

Para redireccionar:

Response.Redirect("~/Home/Index");

<https://www.codecademy.com/courses/introduction-to-javascript/lessons/introduction-to-javascript/exercises/console?action=resume_content_item>

console.log(): Imprime un valor en la consola.

COMENTARIOS:

// Este es un comentario

/\* Comentario \*/

TIPOS DE DATOS:

*Number*: Any number, including numbers with decimals: 4, 8, 1516, 23.42.

*String*: Any grouping of characters on your keyboard (letters, numbers, spaces, symbols, etc.) surrounded by single quotes: ' ... ' or double quotes " ... ". Though we prefer single quotes. Some people like to think of string as a fancy word for text.

*Boolean*: This data type only has two possible values— either true or false(without quotes). It’s helpful to think of booleans as on and off switches or as the answers to a "yes" or "no" question.

*Null*: This data type represents the intentional absence of a value, and is represented by the keyword null(without quotes).

*Undefined*: This data type is denoted by the keyword undefined (without quotes). It also represents the absence of a value though it has a different use than null.

*Symbol*: A newer feature to the language, symbols are unique identifiers, useful in more complex coding. No need to worry about these for now.

*Object*: Collections of related data.

OPERADORES:

Add: +

Subtract: -

Multiply: \*

Divide: /

Remainder: %

console.log('hi' + 'ya'); // Prints 'hiya'

console.log('Hello'.length); // Prints 5

console.log('hello'.toUpperCase()); // Prints 'HELLO'  
console.log('Hey'.startsWith('H')); // Prints true

console.log(' Remove whitespace '.trim());

console.log(Math.random()); // Prints a random number between 0 and 1

Math.floor() takes a decimal number, and rounds down to the nearest whole number.

#### **[Math.ceil()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math/ceil)**

*The Math.ceil() function returns the* ***smallest******integer*** *greater than or equal to a given number.*

#### **[Number.isInteger()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number/isInteger)**

*The* ***Number****.is****Integer****() method determines whether the passed value is an* ***integer****.*

var food=’Pescado’;

let changeMe=true;

const entree='Enchiladas'; //Declara una constante

let x = 20;  
x -= 5; // Can be written as x = x - 5  
console.log(x); // Output: 15

let a = 10;  
a++;  
console.log(a); // Output: 11

a--;

console.log(`My name is ${myName}. My favorite city is ${myCity}.`)

typeof

*DELAY EN JAVASCRIPT (espera en js unos segundos)*

function timeoutAsync(milliseconds) {

var deferred = $.Deferred();

setTimeout(function () { deferred.resolve(); }, milliseconds);

return deferred.promise();

}

function abcAsync() {

var promise = timeoutAsync(2000);

promise.done(function () { alert('done!') });

return promise;

}

*TIEMPO MÁXIMO DE INACTIVIDAD:*

idleTimer = null;

idleState = false;

idleWait = 900000; //Tiempo máximo de inactividad.

$(document).ready(function () {

//Tiempo de Inactividad

$('\*').bind('mousemove keydown scroll', function () {

clearTimeout(idleTimer);

if (idleState === true) {

// Reactivated event

$("body").append("<p>Bienvenido de Vuelta.</p>");

}

idleState = false;

idleTimer = setTimeout(function () {

// Idle Event

//Este mensaje es el que se muestra al usuario.

var Message = {};

Message.Tipo = 1;

Message.Texto = "Superado tiempo máximo de inactividad";

objMensajes.MostrarMensaje(Message);

//Con este código se sale de la sesión.

Message.Tipo = 6;

delay(Message);

//Message.Codigo = ;

idleState = false;

}, idleWait);

});

$("body").trigger("mousemove");

});

function delay(Message) {

var promise = timeoutAsync(2000);

promise.done(function () { objMensajes.MostrarMensaje(Message); });

return promise;

}

function timeoutAsync(milliseconds) {

var deferred = $.Deferred();

setTimeout(function () { deferred.resolve(); }, milliseconds);

return deferred.promise();

}

*MOSTRAR BARRA PROGRESO:*

objMensajes.MensajeProgreso("Buscando información");

*OCULTAR BARRA PROGRESO:*

objMensajes.OcultarMensaje();

*EXECUTESCALAR: EJECUTARSCALAR*

Int32 count = Convert.ToInt32(DMAccesoDatos.EjecutarScalar(ProcedimientosAlmacenados.RTVSSSp\_EstadoCuenta, Parametros));

El procedimiento almacenado correspondiente:

ALTER PROCEDURE [rtvss].[RTVSSSp\_EstadoCuenta]

@Usuario VARCHAR(30)

AS

DECLARE @Idusuario int=0

BEGIN

SET @Idusuario=(SELECT [IdUsuario] FROM [db\_RTVSS].[rtvss].[tbUsuarios] WHERE [Usuario]=@Usuario AND [Estado]=1);

IF EXISTS (SELECT [IdAccionMovimiento] FROM [db\_RTVSS].[rtvss].[tbDtllMovimientoUsuario] WHERE [Estado]=1 AND [IdUsuarioMovimiento]=@Idusuario AND ([IdAccionMovimiento]=3 OR [IdAccionMovimiento]=4))

BEGIN

SELECT TOP 1 [IdAccionMovimiento] FROM [db\_RTVSS].[rtvss].[tbDtllMovimientoUsuario] WHERE [Estado]=1 AND [IdUsuarioMovimiento]=@Idusuario AND ([IdAccionMovimiento]=3 OR [IdAccionMovimiento]=4) ORDER BY [IdAccionMovimiento] DESC;

END

END

*MOSTRAR MENSAJE:*

objMensajes.MostrarMensaje(Message)

var Mensaje = {

Texto: "Por favor diligencie todos los campos obligatorios. La información no se guardó.",

Tipo:1

};

objMensajes.MostrarMensaje(Mensaje);

JQUERY

<https://www.codecademy.com/courses/learn-jquery/lessons/jquery-setup/exercises/event-handlers?action=resume_content_item>

const login = document.getElementById('login');  
const loginMenu = document.getElementById('loginMenu');  
  
login.addEventListener('click', () => {  
 if(loginMenu.style.display === 'none'){  
 loginMenu.style.display = 'inline';  
 } else {  
 loginMenu.style.display = 'none';  
 }  
});

In this example, JavaScript is used to apply behavior to an HTML element with id login. The behavior allows a user to click a **LOGIN**button that toggles a login form.

The code below accomplishes the same behavior with jQuery.

$('#login').click(() => {  
 $('#loginMenu').toggle()  
});

In this example, the same toggle functionality is accomplished using just three lines of code.

METHODS:

.hide(), .show(), .fadeIn(), .fadeOut()

# **Adding jQuery**

To include jQuery, we use a <script> tag as follows:

<script  
 src="https://code.jquery.com/jquery-3.2.1.min.js"  
 integrity="sha256-hwg4gsxgFZhOsEEamdOYGBf13FyQuiTwlAQgxVSNgt4="  
 crossorigin="anonymous"></script>

Se usa llamar al JQuery y a JavaScript al fina de la página para no interferir con la carga de los estilos.

JQUERY se debe cargar antes de cualquier JavaScript

JQUERY se puede descargar de: <https://code.jquery.com/>

The jQuery .ready() method waits until the HTML page's DOM is ready to manipulate.

$(document).ready(() => {  
  
});

Target by Class name:

We can reference elements by class name with the following syntax:

$('.someClass').handlerMethod();

Ejemplo:

apunta a todos los elementos con clase: product-photo

$('.product-photo').show();

Target by id:

$('#someId').hide();

$('#nav-dropdown').hide();

Developers often save jQuery objects in variables, like so:

const $jQueryObject = $('.someClass');

$(document).ready(() => {

const $navDropdown = $('#nav-dropdown');

$('#nav-dropdown').hide();

});

$('#login').on('click', () => {  
 $loginForm.show();  
})

$(document).ready(() => {

const $menuButton = $('.menu-button');

const $navDropdown = $('#nav-dropdown');

$('#menuButton').on('click',()=>{

$('#navDropdown').show();

});

})

$navDropdown.on('mouseleave', () => {

$navDropdown.hide();

})

$('.menu-button').on('click',() => {

$('#nav-dropdown').slideToggle('slow'); //Inserta el menú a mostrar y Desliza lo existente hacia abajo. Se oculta con otro click.

$(".shoe-details-one").toggle(); //Se despliega hacia abajo y luego se desaparece con el mismo click.

$(".text-one").fadeIn(1000); //Aparece lentamente.

$(".text-one").fadeOut(1000); //Desaparece lentamente. 1000 milisegundos. Si no se digita nada lo estándar es 400 ms

$(".text-two").fadeIn();

$(".text-three").fadeIn(1000);

$('.fade-image').fadeToggle('fast'); //El mismo botón sirve para cambiar de estado: mostrar - desaparecer

$('.menu-content').slideDown('slow'); //Desliza hacia abajo

$('.slide-image').slideUp(100); //oculta hacia arriba con animación.

$('.slide-image').slideDown('slow'); //Muestra hacia abajo con animación.

});

# **On 'mouseenter'**

he mouseenter event triggers a callback function when a user enters the area that a targeted element occupies.

# **On 'mouseleave'**

The mouseleave event listener can detect when a user's mouse leaves the area that an element occupies.

**Concatenar EVENTOS: y hacer zoom (acercar y alejar foto):**

$('.product-photo').on('mouseenter', (event) => {

$(event.currentTarget).addClass('photo-active')

}).on('mouseleave', event => {

$(event.currentTarget).removeClass('photo-active')

})

RELLENAR CON COLOR UN BOTÓN (Cuando se coloca el mouse sobre él) $('.btn').on('mouseenter',()=>{

$('.btn').addClass('btn-hover');

<script type="text/x-kendo-template" id="template">

<a class="btn btn-success k-grid-excel" href="\#"><span class="k-icon k-i-excel"></span>Exportar Excel </a>

<a class="btn btn-danger k-grid-pdf" href="\#"><span class="k-icon k-i-pdf"></span>Exportar PDF</a>

<a class="btn btn-warning" id="boton\_nuevo" **href="javascript:hola()**;"><span class="k-icon next"></span>Nuevo Usuario</a>

</script>

OBJETOS EN JAVASCRIPT

Probably the simplest way to create an object in JavaScript is to use the object literal syntax.

This starts with a set of curly braces to indicate an object. Inside the curly braces is a comma-

separated list of name/value pairs to define each property. Object literals create an object

from nothing, so these objects contain precisely what you assign to them and nothing more.

No prototype object is associated with the created object. The following example demonstrates the creation of two objects that represent vehicles.

var car1 = {

year: 2000,

make: 'Ford',

model: 'Fusion',

getInfo: function () {

return 'Vehicle: ' + this.year + ' ' + this.make + ' ' + this.model;

}

};

If you want to define an array of items and assign it to a property, you can use square brackets as shown in the following example.

var car1 = {

year: 2000,

make: 'Ford',

model: 'Fusion',

repairs: ['repair1', 'repair2', 'repair3'],

getInfo: function () {

return 'Vehicle: ' + this.year + ' ' + this.make + ' ' + this.model;

}

};

Creating dynamic objects by using the factory pattern

■■ constructor The function that is called to initialize a new object

■■ hasOwnProperty Returns a Boolean indicator of whether the current object has the

specified property

■■ isPrototypeOf Returns a Boolean indicator of whether the current object is in the

specified object’s prototype object chain

■■ propertyIsEnumerable Returns true if the object can be enumerated in a for...in

loop

■■ toLocalString Converts a date to a string value based on the current local

■■ toString Returns the string representation of the current object

■■ valueOf Returns the value of the current object converted to its most meaningful

primitive value

# **Lista dinámica con Javascript**

<http://www.forosdelweb.com/f13/lista-dinamica-con-javascript-690277/>

<!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>

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

<title>Documento sin t&iacute;tulo</title>

<script>

function t(x){return document.getElementById(x);}

var lista=[];

function almacenar(){

lista.push({n:t('nombre').value,a:t('anio').value});

t('nombre').value=t('anio').value='';

}

function mostrar(){

t('e').innerHTML='';

for(var i=0,m;m=lista[i];i++)

t('e').innerHTML+=lista[i].n+'--'+lista[i].a+'<br />';

}

</script>

</head>

<body>

<div id="e"></div>

<form id="form1" name="form1" method="post" action="">

<input name="nombre" type="text" id="nombre" />

<input name="anio" type="text" id="anio" />

<input type="button" name="Submit" value="almacenar" onclick="almacenar()" />

<input type="button" name="Submit2" value="mostrar" onclick="mostrar()" />

</form>

</body>

</html>

REALIZADO:

//Carga de títulos:

$('#listaFechas').append('<li type="disc" >' + '<b>Numero Clave' + ' Fecha </b>' + '</li>');

$.each(data.Datos, function (i, v) {

//Carga de Valores:

$('#listaFechas').append('<li type="disc" ><b>' + v.IdClave + '</b>&nbsp;'+ moment(v.FechaCreacion).format('MMMM Do YYYY, h:mm:ss a') + '</li>');

});

KENDO ListBox / Basic usage

<https://demos.telerik.com/kendo-ui/listbox/index>

ADICIONAR PROPIEDADES A UN OBJETO:

function getVehicle(theYear, theMake, theModel) {

var vehicle = new Object();

vehicle.year = theYear;

vehicle.make = theMake;

vehicle.model = theModel;

vehicle.getInfo = function () {

return 'Vehicle: ' + this.year + ' ' + this.make + ' ' + this.model;

};

return vehicle;

}

<https://demos.telerik.com/kendo-ui/listbox/index>

If you want to implement encapsulation. Then you need to create objects, each with its own data. To implement encapsulation, use the var keyword for the year, make, and model. This will make these variables private to the function.

function Vehicle(theYear, theMake, theModel) {

var year = theYear;

var make = theMake;

var model = theModel;

this.getInfo = function () {

return 'Vehicle: ' + year + ' ' + make + ' ' + model;

};

}

INTELLISENSE: (AYUDA)

To activate IntelliSense, you must set a reference to the jQuery file (not the IntelliSense file)

in every JavaScript file that requires IntelliSense. The following is an example of the default.js

file with the reference set.

**/// <reference path="jquery-1.8.2.js" />**

var txtInput;

var txtResult;

Triggering event handlers (Para Pruebas)

$('#btnSubmit').triggerHandler('click');

Initializing code when the browser is ready

You will often need to execute initialization code after the HTML document is loaded and ready, and jQuery executes with a browser-independent way to execute code when the document is loaded as follows.

<script>

$(document).ready(function () {

initialize();

});

</script>

RESETEAR CONTROL:

$("#ComboCiudadNuevo").html("");

# **Añadir option a select y guardarlo en base de datos**

<http://www.forosdelweb.com/f18/anadir-option-select-guardarlo-base-datos-1122656/>

CAPÍTULO KENDO

# MultiColumnComboBox

$("#ComboNombreReceptorNuevo").kendoMultiColumnComboBox({

placeholder: "Seleccione un Receptor...",

dataTextField: "RazonSocial",

dataValueField: "IdReceptor",

dropDownWidth: 350,

dataSource: PatrociniosMostrar5,

columns: [

{

field: "RazonSocial",

title: "Nombre",

headerTemplate: "<strong>Nombre</strong>",

width: 300

}, {

field: "TipoDocumento",

title: "TipoDocumento",

width: 300

}, {

field: "NumeroDocumento",

title: "NumeroDocumento",

template: "<strong>#:NumeroDocumento#</strong>",

width: 300

}, {

field: "Ciudad",

title: "Ciudad",

width: 300

}, {

field: "SociedadCientifica",

title: "SociedadCientifica",

width: 300

}

]

});

KENDO TOOL BAR:

<https://demos.telerik.com/kendo-ui/toolbar/events>

DESCARGAR PLANTILLAS MENSAJES KENDO ESPAÑOL:

<https://demos.telerik.com/kendo-ui/grid/localization>

kendo.messages.es-ES.min.js

<http://www.newforestgateway.org/DesktopModules/OnyakTechComments/jsos/kendo/js/messages/>

ICONO DE CARGA EN grid KENDO

kendo.ui.progress($("#gridListaDivision"), true);

kendo.ui.progress($("#gridListaDivision"), false);

RESET KENDO COMBOBOX

$("#cmbPais").data("kendoComboBox").value("");

$('#cmbPais').html('');

ESTABLECER VALOR A KENDO COMBO BOX

$("#SlcCentroSociedad").data("kendoComboBox").value(data.Datos[0].IdCentroSociedad);

OCULTAR COLUMNA (CAMPO) EN GRID TELERIK

,{ title: "Gerente Representante",

field: "GerenteRepresentante",

hidden: true

}, {...

FILTRAR KENDO CON UN VALOR INGRESADO POR EL USUARIO:

var kgrid = $("#grid").data("kendoGrid"); var orfilter = { logic: "or", filters: [] }; orfilter.filters.push({ field: "lastName", operator: "startswith", value: viewModel.get('searchValue') }, { field: "address", operator: "startswith", value: viewModel.get('searchValue') }); kgrid.dataSource.filter(orfilter);

FILTRAR GRID KENDO CON CUADRO DE TEXTO

**EN LA VISTA (HTML)**

<script type="text/x-kendo-template" id="template">

<a class="btn btn-success k-grid-excel" href="\#"><span class="k-icon k-i-excel"></span>Exportar Excel </a>

<input type="search" id="searchBox" class="k-textbox" style="width: 250px" />

<a class="k-button" onclick="javascript:Buscar()" id="btnSearch">Buscar</a>

<a class="k-button" onclick="javascript:Reset()" id="btnReset">Reset</a>

</script>

**EN EL JS:**

function Buscar() {

var searchValue = $('#searchBox').val();

$("#gridReporte").data("kendoGrid").dataSource.filter({

logic: "or",

filters: [

{

field: "municipio",

operator: "contains",

value: searchValue

},

{

field: "nombrecompleto",

operator: "contains",

value: searchValue

},

{

field: "numdocumento",

operator: "contains",

value: searchValue

},

{

field: "tipodocumento",

operator: "contains",

value: searchValue

}

]

});

}

BOTÓN RESETEAR FILTRO EN GRID KENDO

function Reset() {

$("#gridReporte").data("kendoGrid").dataSource.filter({});

$("#searchBox").val('');

}

GENERAR ARCHIVO PDF KENDO:

<style>

#export-container {

position: absolute;

width: 800px;

left: -10000px;

top: 0; }

</style>

<div id="export-container">

<div id="content">

Lorem ipsum dolor sit amet, ad dolores conceptam eos, et his quodsi blandit. Quas sanctus omittam eos no, alia laoreet oportere eos no. Ex eum munere tritani atomorum, id natum assentior vis. Ad vitae prompta omnesque nam, nonumy sapientem principes cu est. Usu molestie vivendum ut, reque tation primis in sit, cum apeirian iudicabit te. Ad atqui luptatum pertinacia est.

</div>

</div>

<script>

kendo.drawing.drawDOM("#content", {

paperSize: "A4",

margin: "2cm"

}).then(function (group) {

kendo.drawing.pdf.saveAs(group, "filename.pdf");

});

</script>

GENERAR ARCHIVO TXT

function GenerarSAP() {

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//Genera un archivo de texto con la información del pedido SAP

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

var mensaje;

mensaje = "Hello world";

var blob = new Blob([mensaje], { type: "text/plain;charset=utf-8" });

kendo.saveAs({

dataURI: blob,

fileName: "test.txt"

});

}

DIÁLOGO CONFIRMAR ACEPTAR CANCELAR

O.K. CANCEL DIALOG (Telerik)

$("#confirmBtn").on("click", function () {  
 kendo.confirm("Are you sure that you want to proceed?").then(function () {  
 kendo.alert("You chose the Ok action.");  
 }, function () {  
 kendo.alert("You chose to Cancel action.");  
 });  
 });

PROBLEMA: NO DETIENE EL FLUJO.

Ajustar el ancho de cada columna:

dataBound: function (e) {

for (var i = 0; i < this.columns.length; i++) {

this.autoFitColumn(i);

}

},

KENDO TOOL TIP TEXT:

$(".k-grid-content").kendoTooltip({

filter: "td",

content: toolTip,

width: 400,

height: 100,

position: "top"

});

----

Flecha apuntando a la columna 2:

$("#gridReporte").kendoTooltip({

filter: "td:nth-child(2)", //this filter selects the second column's cells

position: "right",

content: function (e) {

var dataItem = $("#gridReporte").data("kendoGrid").dataItem(e.target.closest("tr"));

var content = dataItem.Text;

return content;

}

}).data("kendoTooltip");

MENÚ CONTEXTUAL KENDO

<div id="container">Container</div>

<ul id="context-menu">

<li>

Item 1

<ul>

<li>Sub Item 1</li>

<li>Sub Item 2</li>

<li>Sub Item 3</li>

</ul>

</li>

<li>

Item 2

<ul>

<li>Sub Item 1</li>

<li>Sub Item 2</li>

<li>Sub Item 3</li>

</ul>

</li>

</ul>

<script>

$("#context-menu").kendoContextMenu({

appendTo: "#container"

});

</script>

-----------------------------------

<https://www.sitepoint.com/building-custom-right-click-context-menu-javascript/>

HTML

<nav class="context-menu">

<ul id="context-menu" class="context-menu\_\_item">

<li id="Editar" class="context-menu\_\_item">

<a href="#" class="context-menu\_\_link" id="Edit">

<i class="fa fa-eye"></i> Editar

</a>

</li>

</ul>

</nav>

ESTILO:

<style>

#context-menu:hover {

background: rgba(6, 6, 248, 0.90);

color: white;

}

#Edit:hover {

color: white;

}

#context-menu {

list-style: none;

margin: 0;

padding: 0;

margin-bottom: -1px;

min-height: 30px;

min-width: 20px;

text-align: start;

display: none;

position: absolute;

z-index: 10;

box-shadow: 0px 0px 4px rgba(0,0,0,.5);

border: solid 1px #000;

background: #fff;

cursor: pointer;

}

.context-menu--active {

display: block;

}

JS:

noContext = document.getElementById('combo1');

noContext.addEventListener('contextmenu', e => {

e.preventDefault();

var x = $("#combo1").position();

var XX = document.getElementById('combo1');

var h = $('.navbar-header').height();

var panel = document.getElementById("Panel"); //panel es el id del body.

document.getElementById("context-menu").style.top = mouseY(event)-panel.scrollHeight + 'px';

document.getElementById("context-menu").style.left = mouseX(event) + 'px';

toggleMenuOn();

});

function mouseX(evt) {

if (evt.pageX) {

return evt.pageX;

} else if (evt.clientX) {

return evt.clientX + (document.documentElement.scrollLeft ?

document.documentElement.scrollLeft :

document.body.scrollLeft);

} else {

return null;

}

}

function mouseY(evt) {

if (evt.pageY) {

return evt.pageY;

} else if (evt.clientY) {

return evt.clientY + (document.documentElement.scrollTop ?

document.documentElement.scrollTop :

document.body.scrollTop);

} else {

return null;

}

}

function toggleMenuOn() {

$("#context-menu").show();

if (menuState !== 1) {

menuState = 1;

menu.classList.add(active);

}

}

$("#Editar").on('click', function () {

alert("Usted seleccionó Editar");

})

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

MOSTRAR FECHA CON FORMATO EN GRID KENDO

{

title: "Solicitud",

width: "105px",

template: function (dataItem) {

return moment(dataItem.FechaRegistro).format('DD/MMM/YYYY');}

LEER FECHA DE KENDO DATE PICKER

var year1 = $("#start").data("kendoDatePicker").value()

LEER AÑO DE UN KENDO DATE PICKER

var year1 = $("#start").data("kendoDatePicker").value().getFullYear();

GET SELECTED VALUE OF KENDO COMBO BOX

ObjHome.PaisActual = $("#cmbPais").data("kendoComboBox").text();

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CAPÍTULO HTML

CONVERTIR ARCHIVO XML

<https://codebeautify.org/xmlviewer>

GENERAR XML CON DATOS DE BASE DE DATOS

DataSet ds = DMAccesoDatos.EjecutarDataSet(ProcedimientosAlmacenados.RTVSSSp\_ConsultarMuestrasMedicasXML, Parametros);

ds.WriteXml("C:\\Windows\\Temp\\dataEmployee.xml");

SOLO PERMITIR NÚMEROS EN CUADRO DE TEXTO

<input type="number">

ESTABLECER LONGITUD MÁXIMA EN TEXT

maxlength="10"

FINAL DE LÍNEA HTML

<BR/>

# HTML <span> tag

Updated: 04/02/2019 by Computer Hope

**<span>**

When writing in [HTML](https://www.computerhope.com/jargon/h/html.htm), the **<span> tag** is used as an [inline](https://www.computerhope.com/jargon/i/inline.htm)container for grouping or phrasing content.

OCULTAR UN ELEMENTO

<ul id="context-menu">

<li>

Item 1

</li></ul>

<style>

#context-menu {

display:none;}

MENU CONTEXTUAL

<https://www.sitepoint.com/building-custom-right-click-context-menu-javascript/>

CLASS:

container body-content section panel panel-primary

panel panel-primary: Marco externo

<table center="center" class="table table-responsive table-striped table-hover" style="width:400px">

TOOL TIP

BOTÓN OCULTAR:

En la vista:

<div>

<input id="btnOcultarSummary" type="button" value="Ocultar" class="btn btn-success" />

</div>

<style type="text/css">

.inv {

visibility: hidden;

}

.vis {

visibility: visible;

}

</style>

MÉTODO GLOBAL

VARIABLE GLOBAL

Insertar la variable o los métodos en un JS y luego referenciarlos en el Head del Layout:

<script src="~/Scripts/Views/VariablesGlobales/VariableGlobal.js"></script>

PANEL COLAPSIBLE:

<div class="panel-group">

<div class="panel panel-default">

<div class="panel-heading">

<h4 class="panel-title">

<a data-toggle="collapse" href="#collapse2">SEC</a>

</h4>

</div>

<div id="collapse2" class="panel-collapse collapse">

<div class="panel-body">

Panel Body

<div id="sec">

@{Html.RenderPartial("\_SEC");}

</div>

</div>

<div class="panel-footer">Panel Footer</div>

</div>

</div>

</div>

COMBO CON CHECKBOXES

<http://jsfiddle.net/bDvkQ/>

CASCADIN COMBOBOX

<https://demos.telerik.com/kendo-ui/combobox/cascadingcombobox>

BOTÓN CON >>

<p><a class="btn btn-default" id="boton01">Learn more &raquo;</a></p>

ICONOS GRATIS

glyphicon: incluye 250 icónos bootstrap

<https://getbootstrap.com/docs/3.3/components/>

EQUIVALENCIAS

|  |  |  |
| --- | --- | --- |
| Varchar (#) | string |  |
| tinyint | Byte | 0 to 255 |
| bit | bool |  |
| smallint | Int16 | -32,768 to 32,767 |
| datetime | DateTime |  |
| date | DateTime |  |
| int | int | -2,147,483,648 to 2,147,483,647 |
| bigint | long | 9,223,372,036,854,775,807 |
| money | decimal |  |

LLENAR DIV CON TEXTO

var divRe2 = document.getElementById('txtReceptorPDF\_2');

divRe2.innerHTML = v.Nombre

JUSTIFICAR TEXTO

<P ALIGN="justify"> Esto es **texto justificado**.

MODAL

<div class="modal fade in" id="modal-Ciudad" tabindex="-1" data-keyboard="false" data-backdrop="static">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h4 class="modal-title"><label id="titulomensaje" class="modal-titel">Ciudades</label>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</h4>

</div>

<div class="modal-body">

<div class="form-group col-md-12">

<label><b>Nombres y Apellidos: </b> <span style="color:red; height:6em;">(\*)</span></label><br>

<input type="text" class="form-control" id="NombresApellidosNuevo" maxlength="100" />

</div>

<div class="form-group col-md-12">

<label><b>Usuario: </b> <span style="color:red; height:6em;">(\*)</span></label><br>

<input type="text" class="form-control" id="UsuarioNuevo" maxlength="30" />

</div>

<div class="form-group col-md-12">

<label><b>Tipo Documento: </b><span style="color:red; height:6em;">(\*)</span></label><br>

<select class="form-control" id="ComboTipoIdentificacion"></select>

</div>

<div class="form-group col-md-12">

<label><b>Número Identificación: </b><span style="color:red; height:6em;">(\*)</span></label><br>

<input type="text" class="form-control justNumbers" id="NumeroIdentificacionNuevo" maxlength="30" />

</div>

<div class="form-group col-md-12">

<label><b>Correo: </b><span style="color:red; height:6em;">(\*)</span></label><br>

<input type="text" class="form-control" id="CorreoNuevo" maxlength="100" />

</div>

<div class="form-group col-md-12">

<label><b>Ciudad: </b><span style="color:red; height:6em;">(\*)</span></label><br>

<select class="form-control" id="ComboCiudadNuevo"></select>

</div>

<div class="form-group col-md-12">

<label><b>Gerente Representante: </b><span style="color:red; height:6em;"></span></label><br>

<select class="form-control" id="ComboGerenteNuevo"></select>

</div>

<div class="form-group col-md-12">

<label><b>Perfil: </b><span style="color:red; height:6em;">(\*)</span></label><br>

<select class="form-control" id="ComboPerfilNuevo"></select>

</div>

<div class="form-group form-inline">

<label style="width:120px" id="IdUsuarioNuevo" hidden><b> </b></label>

</div>

</div>

<div class="modal-footer" style="border-top: 0px;">

<table class="botones-modal">

<tr>

<td>

<input id="btnGuardarNuevo" type="button" value="Agregar" class="btn btn-success" />

<input id="btnCancelarCiudad" type="button" value="Cancelar" class="btn btn-danger" />

</td>

</tr>

</table>

</div>

</div>

</div>

</div>

Teoría de MODAL: <https://www.sitepoint.com/understanding-bootstrap-modals/>

IMAGEN EN BACKGROUND

<body style="background: url(~/../Content/Imagens/fondoBaground2.jpg); min-height: 100vh; background-size: cover; background-position: center top; background-repeat: no-repeat;">

ESPACIO EN HTML

&nbsp;

ESTABLECER ANCHO RESPONSIVE EN CONTROL EN LÍNEA

<input class="form-control" id="txtDepartamento" style="max-width:100%; width:80%;" type="text"/>

COLOCAR \* INDICANDO CAMPO OBLIGATORIO

<label Usuario <span style="color:red; height:6em;">(\*)</span> </label>

Publicar PROYECTO:

<https://www.kyocode.com/2018/08/publicar-aplicacion-web-equipo-local-visual-studio-2017/>

1. En el explorador de Soluciones buscar el Proyecto Web
2. Clic derecho en el Proyecto Web y seleccionar Publicar
3. Clic en Configurar para seleccionar carpeta
4. Clic en Publicar

QUITAR RAYA AZUL DE GRID:

Se debe remover:

<hr />

COMENTARIO EN HTML

<!--Modals: Nuevo Integrante→

COLORES EN HTML

**Lista de códigos de colores HTML comunes**

|  |  |  |  |
| --- | --- | --- | --- |
| **COLOR** | **COLOR NAME** | **HEX COLOR CODE** | **RGB COLOR CODE** |
|  | **WHITE** | **#FFFFFF** | **RGB(255, 255, 255)** |
|  | **SILVER** | **#C0C0C0** | **RGB(192, 192, 192)** |
|  | **GRAY** | **#808080** | **RGB(128, 128, 128)** |
|  | **BLACK** | **#000000** | **RGB(0, 0, 0)** |
|  | **RED** | **#FF0000** | **RGB(255, 0, 0)** |
|  | **MAROON** | **#800000** | **RGB(128, 0, 0)** |
|  | **YELLOW** | **#FFFF00** | **RGB(255, 255, 0)** |
|  | **OLIVE** | **#808000** | **RGB(128, 128, 0)** |
|  | **LIME** | **#00FF00** | **RGB(0, 255, 0)** |
|  | **GREEN** | **#008000** | **RGB(0, 128, 0)** |
|  | **AQUA** | **#00FFFF** | **RGB(0, 255, 255)** |
|  | **TEAL** | **#008080** | **RGB(0, 128, 128)** |
|  | **BLUE** | **#0000FF** | **RGB(0, 0, 255)** |
|  | **NAVY** | **#000080** | **RGB(0, 0, 128)** |
|  | **FUCHSIA** | **#FF00FF** | **RGB(255, 0, 255))** |
|  | **PURPLE** | **#800080** | **RGB(128, 0, 128))** |

Lista de códigos de colores HTML comunes

|  |  |  |
| --- | --- | --- |
| **Color Name** | **Hex Color Code** | **RGB Color Code** |
| Red | #**FF0000** | rgb(255, 0, 0) |
| Maroon | #800000 | rgb(128, 0, 0) |
| **Yellow** | #FFFF00 | rgb(255, 255, 0) |
| **Olive** | #808000 | rgb(128, 128, 0) |

CAMBIAR ESTILO DE BOTON (CAMBIAR COLOR DE BOTÓN):

EN LA VISTA:

<style>

.botonSeleccionado {

background-color: #0000FF;

color: #FFFFFF;

font-size: 11px;

border: 1px solid #333333;

}

</style>

EN JS:

document.getElementById("btnFinalizado").className = "botonSeleccionado";

Eliminar SP de BD:

Clic en Stored Procedures -> F7

Seleccionar todos los SP y hacer clic en Suprimir

BOTÓN BONITO:

class="btn btn-primary btn-lg"

<button class="k-button k-primary" id="get">View Order</button>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<button id="ActualizarAlojamientoPRE" class="btn btn-info"><i class="fa fa-bars"></i> Actualizar</button>

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

<button id="ActualizarAlojamientoPRE" class="btn btn-info"><span class="glyphicon glyphicon-search text-success" style="color:blue"></span> Actualizar</button>

<a title="Editar" data-toggle="tooltip" href="../Contratos/ActualizacionContratos?Id=' + Id + '" class="btn btn-warning"><span class="glyphicon glyphicon-check"></span></a>

# FORMULARIO EN VISTA CON ACCIÓN (enviar algo):

<form name="formulario" id="formulario">

<input type="file"/>

<button type="submit" formaction=@Url.Action("UploadButton") formmethod="post" name="UploadButton" >Enviar</button>

</form>

<a href="@Url.Action("ActionName", "ControllerName", new { paramName1 = item.paramValue1, paramName2 = item.paramValue2 })" > Link text </a>

CÓDIGO QUE RECIBE:

public ContentResult UploadButton(string ruta) {

return Content("<script language='javascript' type='text/javascript'>alert(' Exitosa ');</script>");

}

TOOLTIPTEXT EN ETIQUETA HTML

placeholder="Texto indicativo que desaparecerá"

TOOLTIP EN BOTÓN

data-toggle="tooltip"

title="Tooltip on right"

<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="right" title="Tooltip on right">

Tooltip on right

BOTON LINK CONTROLADOR MVC (BOTON CSS)

LLAMAR AL CONTROLADOR

<div class="col-xs-12 ctn-btn-login">

<input id="btnCambiarClave" onclick="location.href='**@Url.Action("Accion", "Controlador")'**" class="input-btn btn-login" type="button" value="Cambiar Clave" />

</div>

LINK REDIRECCIONAR EN JAVASCRIPT (REDIRIGIR):

REDIRECCIONAR DESDE JS

window.location.href = '/Home/Index/';

Response.Redirect("~/Home/Index");

MOSTRAR MENSAJE DESDE C# EN JAVASCRIPT:

return Content("<script language='javascript' type='text/javascript'>alert( file. );</script>");

MOSTRAR DOS BOTONES EN LÍNEA (EN EL MISMO RENGLÓN HORIZONTALMENTE)

Insertar una en cada botón:

style='float:left;'

style='float:right;'

Tamaño de un Botón

<button style="height:200px;width:200px"></button>

BUTON SIN BORDE:

<button class="btn-without-border"> Submit </button>

BOOTSTRAP BUTTON:

https://getbootstrap.com/docs/4.0/components/buttons/

CUADRO DE TEXTO SIN BORDE:

<input id="txtCiudadPDF" type="text" style="border:0"/>

ESTABLECER VALOR SELECCIONADO A SELECT (SET SELECTED VALUE TO SELECT CONTROL)

document.getElementById("mySelect").value = "banana";

document.getElementById("ComboPerfilModal").value = data.Datos[0].IdPerfilUsuario;

LEER ARCHIVOS DE EXCEL

Instalar DocumentFormat.OpenXml de Nuget

using DocumentFormat.OpenXml

MÉTODO 1:

string fileName = @"c:\path\to\my\file.xlsx"; using (FileStream fs = new FileStream(fileName, FileMode.Open, FileAccess.Read, FileShare.ReadWrite)) { using (SpreadsheetDocument doc = SpreadsheetDocument.Open(fs, false)) { WorkbookPart workbookPart = doc.WorkbookPart; SharedStringTablePart sstpart = workbookPart.GetPartsOfType<SharedStringTablePart>().First(); SharedStringTable sst = sstpart.SharedStringTable; WorksheetPart worksheetPart = workbookPart.WorksheetParts.First(); Worksheet sheet = worksheetPart.Worksheet; var cells = sheet.Descendants<Cell>(); var rows = sheet.Descendants<Row>(); Console.WriteLine("Row count = {0}", rows.LongCount()); Console.WriteLine("Cell count = {0}", cells.LongCount()); // One way: go through each cell in the sheet foreach (Cell cell in cells) { if ((cell.DataType != null) && (cell.DataType == CellValues.SharedString)) { int ssid = int.Parse(cell.CellValue.Text); string str = sst.ChildElements[ssid].InnerText; Console.WriteLine("Shared string {0}: {1}", ssid, str); } else if (cell.CellValue != null) { Console.WriteLine("Cell contents: {0}", cell.CellValue.Text); } } // Or... via each row foreach (Row row in rows) { foreach (Cell c in row.Elements<Cell>()) { if ((c.DataType != null) && (c.DataType == CellValues.SharedString)) { int ssid = int.Parse(c.CellValue.Text); string str = sst.ChildElements[ssid].InnerText; Console.WriteLine("Shared string {0}: {1}", ssid, str); } else if (c.CellValue != null) { Console.WriteLine("Cell contents: {0}", c.CellValue.Text); } } } } } }

PASAR XML A SQL

<https://stackoverflow.com/questions/3600091/how-to-pass-xml-from-c-sharp-to-a-stored-procedure-in-sql-server-2008>

<https://medium.com/@israelcortesflores/c-m%C3%A9todos-de-lectura-de-ficheros-excel-xlsx-9f627be4a740>

**video:**

<https://youtu.be/vy2Nv26UzAU>

CAPÍTULO KENDO

KENDO

## **Displaying a Date (Mostrar una Fecha)**

Utilizado finalmente en módulo de Usuarios.

template: "#= kendo.toString(kendo.parseDate(FechaRegistro, 'yyyy-MM-dd h:mm:ss'), 'dd/MM/yyyy h:mm:ss') #"

Otras alternativas para fechas con KENDO:

//template: "#= kendo.parseDate(FechaRegistro, 'yyyy-mm-dd') #"

//"FechaRegistro":new Date(field: "FechaRegistro")

// moment(field:"FechaRegistro").format('MMMM Do YYYY, h:mm:ss a')

//field: "FechaRegistro",

//field = kendo.parseDate(field: "FechaRegistro", "yyyy-MM-ddTHH:mm:ss.fffZ"),

//format: "{0:MM/dd/yyyy h:mm tt}"

//format: "{0:MMMM Do YYYY, h:mm:ss a}"

//format: "{0:MM/dd/yyyy h:mm tt}"

//date: new Date(field= "FechaRegistro"),

//template: "#= kendo.toString(kendo.parseDate(FechaRegistro, 'yyyy-MM-dd'), 'MM/dd/yyyy') #"

KENDO TOOL BAR:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8"/>

<title>Kendo UI Snippet</title>

<link rel="stylesheet" href="http://kendo.cdn.telerik.com/2017.1.118/styles/kendo.common.min.css"/>

<link rel="stylesheet" href="http://kendo.cdn.telerik.com/2017.1.118/styles/kendo.rtl.min.css"/>

<link rel="stylesheet" href="http://kendo.cdn.telerik.com/2017.1.118/styles/kendo.silver.min.css"/>

<link rel="stylesheet" href="http://kendo.cdn.telerik.com/2017.1.118/styles/kendo.mobile.all.min.css"/>

<script src="http://code.jquery.com/jquery-1.12.4.min.js"></script>

<script src="http://kendo.cdn.telerik.com/2017.1.118/js/kendo.all.min.js"></script>

<style>

.k-toolbar {

overflow: visible;

}

</style>

</head>

<body>

<div id="toolbar"></div>

<script>

$("#toolbar").kendoToolBar({

items: [

{ type: "button", text: "<<" },

{ type: "separator" },

{ template: "<label>pág:</label>" },

{ type: "button", icon: "tick", showIcon: "toolbar" },

{ type: "button", text: ">>"}

],

click: function(e) {

console.log("click", e.target.text());

}

});

//initialze the Menu after ToolBar

var datasource = [

{

text: "Item 1",

},

{

text: "Item 2",

},

{

text: "Item 3",

items: [{text: "Sub Item 1"},{text: "Sub Item 2"}]

}

]

$("#menu").kendoMenu({

dataSource: datasource

})

</script>

</body>

</html>

GRID CON PAGINADOR DINÁMICO DESDE EL SERVIDOR

GET PAGE CHANGE EVENT ON GRID KENDO

<div id="pager"></div>

<script>

function pager\_change() {

console.log("pager change event");

}

var dataSource = new kendo.data.DataSource({

data: [

{ productName: "Tea", category: "Beverages" },

{ productName: "Coffee", category: "Beverages" },

{ productName: "Ham", category: "Food" },

{ productName: "Bread", category: "Food" }

],

pageSize: 2

});

dataSource.read();

var pager = $("#pager").kendoPager({

dataSource: dataSource

}).data("kendoPager");

pager.bind("change", pager\_change);

</script>

<style>

#pager {

margin-top: 100px;

}

</style>

//////////////////////////

<https://www.telerik.com/forums/grid-page-change-event>

AUTOBIND:inhabilita el llenado automático:

$("#grid").kendoGrid({

autoBind: false,

dataSource: dataSource

});

datasource.read();

<https://docs.telerik.com/kendo-ui/api/javascript/ui/grid/configuration/autobind>

PAGE:

<https://docs.telerik.com/kendo-ui/api/javascript/ui/grid/events/page>

solo muestra una página:

<https://dojo.telerik.com/aqEZiraK>

KENDO DATEPICKER

DESHABILITAR KENDO DATE PICKER

$('#txtFechaOtrosi').data('kendoDatePicker').enable(false);

Mostrar Fecha en un KendoDatePicker FUNCIONA:

En la vista:

<input id="FechaInicio" title="datepicker" style="width:150px !important;"/>

Se inicializa el KendoDatePicker

$('#txtFechaInicio').kendoDatePicker({

value: new Date(),

language: "es",

todayHighlight: true,

autoclose: true,

format: "dd-MM-yyyy"

});

Leer las fechas:

var FI = $(‘#txtFechaInicio').val();

En el DT:

public DateTime FI {get;set;}

BLOQUEAR EL INGRESO DE CARACTERES EN UN DATEPICKER

$('#txtFechaInicio).kendoDatePicker({

value: new Date(),

language: "es",

todayHighlight: true,

autoclose: true

});

**$("#txtFechaInicio").attr("readonly", "readonly");**

Se muestra la fecha leída en BD En un KendoDatePicker:

$("#txtFechaInicio").val(moment(data.Datos[0].FechaInicio).format('DD-MM-YYYY'));

EN EL DT SE DECLARA EL FORMATO FECHA:

public DateTime FechaPago {get;set;}

public DateTime FechaPactada { get; set; }

PARA LEER LA FECHA SELECCIONADA Y GUARDARLA EN BD:

DT\_Grant.FechaInicio = document.getElementById("txtFechaInicio").value;

------------------------------------------------------------

en el SQL:

SELECT CONVERT(varchar(10),YourDatetimeColumn,111) AS StringDate1 --YYYY/MM/DD ,CONVERT(varchar(10),YourDatetimeColumn,121) AS StringDate2 --YYYY-MM-DD FROM ...

En el Ajax:

var \_meetStartTime = dateFormat(now, "mm/dd/yyyy HH:MM:ss");

Función para convertir en JS:

1. function ToJavaScriptDate(value) {
2. var pattern = /Date\(([^)]+)\)/;
3. var results = pattern.exec(value);
4. var dt = new Date(parseFloat(results[1]));
5. return (dt.getMonth() + 1) + "/" + dt.getDate() + "/" + dt.getFullYear();
6. }

~~var fecha1 = DateTime.ParseExact(document.getElementById("txtFechaPagoEG").value, "dd/MM/yyyy HH:mm:ss", CultureInfo.InvariantCulture);~~

~~var fecha2 = (document.getElementById("txtFechaPagoEG").value).toJSON();~~

RESTRINGIR SELECCIÓN EN UN KENDO COMBOBOX

How to restrict combobox to not allow manually edit value but allow only dropdown selection

El usuario tiene que seleccionar un valor:

$("#cmbCiudad").kendoComboBox({

filter: "contains",

placeholder: "Seleccione una Ciudad...",

dataTextField: "Nombre",

dataValueField: "IdCiudad"

}).data("kendoComboBox");

$("#cmbCiudad").blur(function () {

var setme = $("#cmbCiudad").data("kendoComboBox");

if (setme.selectedIndex === -1) {

alertify.error("Por favor seleccione una ciudad");

setme.select(-1);

setme.value('');

}

});

GET VALUE OF KENDO COMBOBOX (TEXT)

$("#cmbPais").data("kendoComboBox").text();

GET INDEX OF KENDO COMBOBOX (Id)

document.getElementById("cmbPaisModal").value;

Mostrar Fecha de Base de datos en formato largo

FI = new Date(moment(data.Datos[0].FechaInicio).format('MM-DD-YYYY'));

$("#txtFechaInicioPDF").val(FI.getDate() + " de " + ObjHome.meses[FI.getMonth()] + " de " + FI.getFullYear());

Extraer valor de KendoDatePicker:

var Fecha = DateTimePicker1.Value.ToString("yyyy-MM-dd")

## **Displaying a Date**

var entries = [

{ "name":"Dave Packard", "born":new Date(1912, 8, 7) },

{ "name":"Bill Hewlett", "born":new Date(1913, 4, 20) },

{ "name":"Larry Ellison","born":new Date(1944, 7, 17) },

{ "name":"Steve Jobs", "born":new Date(1955, 1, 24) },

{ "name":"Bill Gates", "born":new Date(1955, 9, 28) },

{ "name":"Jeff Bezos", "born":new Date(1964, 0, 12) },

{ "name":"Larry Page", "born":new Date(1973, 5, 26) },

{ "name":"Sergey Brin", "born":new Date(1973, 7, 21) }

];

And I use the following grid for displaying the data:

|  |  |
| --- | --- |
| 1  2  3  4  5 | var grid = $("#auth\_tbl").kendoGrid({  dataSource: { data: entries },  editable: true,  navigatable: true  });  SET VALUE TO KENDO COMBOBOX  function SeleccionarCiudad(idCiudad) {  var select = $("#cmbCiudad").data("kendoComboBox");  select.value(idCiudad);  }  OBTENER VALOR  SI NO ES KENDO (ES UN SELECT NORMAL)  function SeleccionarDepartamento(IdDepartamento) {  document.getElementById("cmbDepartamento").value = IdDepartamento;  }  MOSTRAR FECHA EN GRID KENDO  {  title: "Fecha Registro",  field: “FechaRegistro”,  template: "#= kendo.toString(kendo.parseDate(FechaRegistro, 'yyyy-MM-dd h:mm:ss'), 'dd/MM/yyyy HH:mm:ss') #"  }  {  title: "FechaFin",  template: function (dataItem) {  return moment(dataItem.FechaFin).format('MM-DD-YYYY');  }  template: function (dataItem) {  return moment(dataItem.FechaFinContrato).format('MM-DD-YYYY');  }  {  title: "Inicio",  field: "FechaInicio",  format: "{0:dd/MM/yyyy}"  }  SI EN EL DT EL CAMPO FECHA es string:  En entonces en el SP de lectura se debe ingresar:  CONVERT(varchar(10),[FechaInicio]) as FechaInicio  CLEAR VALUE TO KENDO COMBO BOX  var cmb = $("#cmbPaisModal").data("kendoComboBox");  if (cmb) {  cmb.value('');  } |

SELECCIONAR FILA EN GRID KENDO

<https://docs.telerik.com/kendo-ui/api/javascript/ui/grid/events/change>

selectable: "row",

change: function(e) {

var selectedRows = this.select();

var selectedDataItems = [];

for (var i = 0; i < selectedRows.length; i++) {

var dataItem = this.dataItem(selectedRows[i]);

selectedDataItems.push(dataItem);

$('#Documento').val(dataItem.NumeroDocumento);

}

}

MOSTRAR Y OCULTAR COLUMNAS EN GRID KENDO

function Ocultar() {

var grid = $("#gridMenus").data("kendoGrid");

grid.hideColumn("TipoEvento");

}

function Mostrar() {

var grid = $("#gridMenus").data("kendoGrid");

grid.showColumn("TipoEvento");

}

TÍTULO DE MÚLTIPLE LÍNEA EN GRID KENDO

<style>

#gridMenus k-grid .k-grid-header .k-header .k-link {

height: auto;

}

.k-grid .k-grid-header .k-header {

white-space: normal;

}

</style>

GRID KENDO APARIENCIA

<https://docs.telerik.com/kendo-ui/controls/data-management/grid/appearance/height#height>

GRID KENDO CON EDITOR (DROPDOWN)

$("#grid").kendoGrid({  
 dataSource: dataSource,  
 pageable: true,  
 height: 550,  
 toolbar: ["create"],  
 columns: [  
 { field:"ProductName",title:"Product Name" },  
 { field: "Category", title: "Category", width: "180px", editor: categoryDropDownEditor, template: "#=Category.CategoryName#" },  
 { field: "UnitPrice", title:"Unit Price", format: "{0:c}", width: "130px" },  
 { command: "destroy", title: " ", width: "150px" }],  
 editable: true  
 });  
 });  
  
 function categoryDropDownEditor(container, options) {  
 $('<input required name="' + options.field + '"/>')  
 .appendTo(container)  
 .kendoDropDownList({  
 autoBind: false,  
 dataTextField: "CategoryName",  
 dataValueField: "CategoryID",  
 dataSource: {  
 type: "odata",  
 transport: {  
 read: "https://demos.telerik.com/kendo-ui/service/Northwind.svc/Categories"  
 }  
 }  
 });  
 }  
  
 </script>

FORMATO MONEDA EN KENDO GRID

{

title: "Alojamiento",

field: "Alojamiento",

format: "{0:c0}",

attributes: { class: "text-right" },

footerTemplate: "<div style='text-align:right'>" + SumAlojamiento(data)

}

GRID KENDO CON BOTÓN DE NUEVO

editable:true,

toolbar: ["create"],

GRID KENDO CON BARRA:

toolbar: kendo.template($("#template").html()),

EXTRAER VALOR DE JSON PARA ALIMENTAR CONTROL:

$.each(Dato.Datos, function (i, v) {

$("#ComboPerfilNuevo").append("<option value='" + v.IdPerfilUsuario + "'>" + v.Nombre + "</option>"); });

EXTRAER FILA EN GRID KENDO (SELECCIONAR FILA)

change: function (e) {

var rows = e.sender.select();

rows.each(function (e) {

var grid = $("#gridMenusNG").data("kendoGrid");

var dataItem = grid.dataItem(this);

$('#Documento').val(dataItem.NumeroDocumento);

console.log(dataItem);

});

}

DISABLED AN INPUT

<form> <label for="disabled">Disabled</label><br> <input name="disabled" value="You cannot edit me" disabled> </form>

JS EJECUTAR FUNCIÓN DESPUES DE QUE SE HA CARGADO UNA PÁGINA

$(document).ready(function () {

window.addEventListener('load', Mostrar, false);

function Mostrar() {

ObjHome.consultarPatrocinios();

};})

## ready()

.ready() es un evento que tiene como finalidad el ejecutar alguna función inmediatamente despúes de cargar todo el documento **HTML** y su **DOM** correspondiente, garantizando que nuestro código sea ejecutado sobre elementos que ya han sido desplegados.

Podemos invocar a .ready() de tres formas diferentes:

$(document).ready(handler)  
$().ready(handler)  
$(handler)

La principal ventaja del método .ready de jQuery es que, a diferencia del metodo **load** nativo de **javascript**, esperará a la carga del contenido adicional al HTML (imágenes, hojas de estilo) para ser ejecutado; esto se traduce a procesamiento post-carga como la búsqueda de elementos por estilos aplicados.

El metodo .ready() suele ser usado junto con una función anónima, esto es:

$(document).ready(**function**() {  
 *// Instrucciones a ejecutar al terminar la carga*  
});

OTRA FORMA:

<script>

window.onload = function () {

alert('OK'+@ViewData["IdPatrocinio"]);

}

</script>

GET VALUE FROM COMBO

OBTENER LEER VALOR SELECCIONADO DE COMBO

Establecer valor a Combo:

document.getElementById("cmbStatus").value = data.Datos[0].Status;

Extraer valor de combo:

var City = document.getElementById('cmbCiudad');

var Nombre\_Ciudad = City.options[City.selectedIndex].text;

DT\_Patrocinio.Status =document.getElementById("cmbStatus").value;

var idReceptor = $("#ComboNombreReceptorNuevo option:selected").val();

HALLAR EL USUARIO ACTUAL (AVERIGUAR USUARIO)

DTUsuario \_usuario = (DTUsuario)Session[DTNombreSesion.DTUsuario];

DT\_Patrocinio.IdUsuarioRegistro = \_usuario.IdUsuario;

PONER EN BLANCO UN CUADRO DE TEXTO:

$("#TxtIdPatrocinioAP").html('');

CUADRO BLANCO TIPO CONSOLA

<div class="box wide">  
 <h4>Console log</h4>  
 <div class="console"></div>  
 </div>  
 <style>  
 .console div {  
 height: 3.3em;  
 }  
 </style>

FORMATO DE CHECKBOXES:

<https://demos.telerik.com/kendo-ui/styling/checkboxes>

<https://www.csslab.cl/2014/03/17/checkboxes-con-estilo/>

<https://paulund.co.uk/how-to-style-a-checkbox-with-css>

Grid: Checkbox selection

https://demos.telerik.com/kendo-ui/grid/checkbox-selection

INCLUIR SUBTOTAL EN GRID

{

title: "Tiquetes",

field: "Tiquetes",

footerTemplate:

SumTiquetes(data)

}

AFUERA DEL GRID:

function SumTiquetes(data) {

var sum = 0;

for (idx = 0; idx < data.length; idx++) {

if (typeof data[idx].Tiquetes !== "undefined") {

sum += (data[idx].Tiquetes === null || data[idx].Tiquetes === '' || data[idx].Tiquetes === 'indefined' ? 0 : data[idx].Tiquetes);

}

}

return kendo.toString(sum, "c");

}

BOTON REDIRIGIR EN KENDOGRID

{

title: "",

width: "80px",

template: function (dataItem) {

return "<button onclick='BuscarIntegrante(" + dataItem.IdReceptor + ")' data-toggle='modal' data-target='#modal-EditarAplicacion' type='button' class='btn btn-light' placeholder='Detalle del patrocinio' border='none' outline-style:'none'>+</button>";

}

}

KENDO TOOL BAR

<https://demos.telerik.com/kendo-ui/toolbar/index>

SEARCH BOX:

<https://www.telerik.com/forums/search-box-kendo-grid>

capítulo c#

CAPÍTULO C#

CONTROLADOR REST:

Por defecto el controlador API devuelve xml pero insertando estas líneas en el WebApiConfig del (App\_start) ya devuelve json por defecto:

GlobalConfiguration.Configuration.Formatters.JsonFormatter.MediaTypeMappings

.Add(new System.Net.Http.Formatting.RequestHeaderMapping("Accept", "text/html", StringComparison.InvariantCultureIgnoreCase,true,"application/json"));

config.Formatters.XmlFormatter.SupportedMediaTypes.Remove(appXmlType);

ENVIAR ARGUMENTOS POR URL:

ENVIAR VALOR DESDE EL CONTROLADOR A LA VISTA

Envía (En la vista):

Envía (En el Controlador):

public ActionResult LeerUnPatrocinio(DTPatrocinio DT\_Patrocinio) {

ViewData["IdPatrocinio"] = DT\_Patrocinio.IdPatrocinio;

return View("DetallePatrocinios");

}

Recibe en la vista:

<input id='IdP' value=@ViewData["IdPatrocinio"] style="visibility: hidden" ;>

ELIMINAR ESPACIOS EN BLANCO AL FINAL DE UN STRING

String trimmed = cadena.TrimEnd();

CREAR ARCHIVO EXCEL

librerías: OpenXml y EEPPLUS

OTRA FORMA: ENMASCARAR ARGUMENTO URL

EN EL GRID:

{

title: "Cotizar",

field: "IdContratos",

template: function (dataItem) {

var Id = window.btoa(unescape(encodeURIComponent(dataItem.IdContratos)));

return '<center><a href="../Contratos/Cotización?Id=' + Id + '" class="btn btn-info"><span class="glyphicon glyphicon-list-alt"></span></a></center>';

}

}

Enviar Varios argumentos:

var IdR\_Coded = window.btoa(unescape(encodeURIComponent(boton)));

var IdP\_Coded = window.btoa(unescape(encodeURIComponent(IdP)));

window.location.href = "/Patrocinios/CartaPDF?IdReceptor=" + IdR\_Coded + "&IdPatrocinio=" + IdP\_Coded;

RECIBIR ARGUMENTOS EN EL JS (CODIFICADOS): ENMASCARADO

**ENVÍA:**

var IdP = $(boton).data('id');

var Id = window.btoa(unescape(encodeURIComponent(IdP)));

window.location.href = "/Patrocinios/LeerUnPatrocinio?IdPatrocinio=" + Id;

**RECIBE:**

$(document).ready(function () {

var URLactual = window.location.href;

let params = new URL(URLactual).searchParams;

IdContrato = decodeURIComponent(escape(window.atob(params.get('Id'))));

if (IdContrato !== 0) {

ConsultarContrato(IdContrato);

ObjHome.ConsultarDtllConcepto();

}

});

LLAMAR MÉTODO ESTÁTICO

EL MÉTODO ESTÁTICO A LLAMAR:

public static HashSet<DTReporteNoRegistrados> ConsultaNoReportadosStatic(DTReporteNoRegistrados Dtos)

LLAMADO:

HashSet<DTReporteNoRegistrados> consultaNR = BMReporteMinisterio.ConsultaNoReportadosStatic(DT\_Reporte);

RECIBIR VARIOS OBJETOS DESDE JS

public string AgregarContrato(DTContratos objContrato, List<DTCharlas> ObjCharlas, DtllConcepto DatosConcepto, List<DtllConcepto> ObjDtllConcepto)

{}

PARÁMETRO TIPO GENÉRICO

[Generic Type Parameter](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/generics/generic-type-parameters).

Ejemplo:

public T Resultado { get; set; }

public T[] Reverse<T>(T[] array)

{ var result = new T[array.Length]; int j=0;

for(int i=array.Length - 1; i>= 0; i--)

{ result[j] = array[i]; j++; }

return result; }

A generic type parameter allows you to specify an arbitrary type T to a method at compile-time, without specifying a concrete type in the method or class declaration.

The key point here is that the array elements can be of any type, and the function will still work.

The following example only allows you to invoke the method with types that are classes:

void Foo<T>(T item) where T: class { }

void Foo<T>(T item) where T: class

{ Type type = typeof(T); }

LBRERÍA PARA CREAR GRILLA C#

# ImageGrid lib

IMPLEMENTAR GRIDVIEW CON XML

# [**Gridview displaying XML file data [duplicate]**](https://stackoverflow.com/questions/20245229/gridview-displaying-xml-file-data)

<https://stackoverflow.com/questions/1988813/gridview-binding-to-xml?rq=1>

DECLARAR NULL (NULLABLE) UNA VARIABLE

public Nullable<DateTime> FechaPago { get; set; }

GRIDVIEW

**EN LA VISTA:**

<script src="~/Scripts/grid-0.4.3.min.js" type="text/javascript"></script>

<link href="~/Content/grid-0.4.3.min.css" rel="stylesheet" type="text/css">

<div class="container fill">

<table id="grid" data-source="@Url.Action("GetPlayers")"></table>

</div>

**EN EL JS:**

<script type="text/javascript">

var grid;

function Add() {

$("#playerId").val("");

$("#name").val("");

$("#placeOfBirth").val("");

$("#dateOfBirth").val("");

$("#playerModal").modal("show");

}

function Edit(e) {

$("#playerId").val(e.data.id);

$("#name").val(e.data.record.Name);

$("#placeOfBirth").val(e.data.record.PlaceOfBirth);

$("#dateOfBirth").val(e.data.record.DateOfBirth);

$("#playerModal").modal("show");

}

function Save() {

var player = {

ID: $("#playerId").val(),

Name: $("#name").val(),

PlaceOfBirth: $("#placeOfBirth").val(),

DateOfBirth: $("#dateOfBirth").val()

};

$.ajax({ url: "Home/Save", type: "POST", data: { player: player } })

.done(function () {

grid.reload();

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

})

.fail(function () {

alert("Unable to save.");

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

});

}

function Remove(e) {

$.ajax({ url: "Home/Remove", type: "POST", data: { id: e.data.id } })

.done(function () {

grid.reload();

})

.fail(function () {

alert("Unable to remove.");

});

}

function Search() {

grid.reload({ searchString: $("#search").val() });

}

$(document).ready(function () {

grid = $("#grid").grid({

dataKey: "ID",

uiLibrary: "bootstrap",

columns: [

{ field: "ID", width: 50, sortable: true },

{ field: "Name", sortable: true },

{ field: "PlaceOfBirth", title: "Place Of Birth", sortable: true },

{ field: "DateOfBirth", title: "Date Of Birth", sortable: true },

{ title: "", field: "Edit", width: 34, type: "icon", icon: "glyphicon-pencil", tooltip: "Edit", events: { "click": Edit } },

{ title: "", field: "Delete", width: 34, type: "icon", icon: "glyphicon-remove", tooltip: "Delete", events: { "click": Remove } }

],

pager: { enable: true, limit: 5, sizes: [2, 5, 10, 20] }

});

$("#btnAddPlayer").on("click", Add);

$("#btnSave").on("click", Save);

$("#btnSearch").on("click", Search);

});

</script>

**EN EL CONTROLADOR:**

[HttpGet]

public JsonResult GetPlayers(int? page, int? limit, string sortBy, string direction, string searchString = null)

{

int total;

var records = new GridModel().GetPlayers(page, limit, sortBy, direction, searchString, out total);

return Json(new { records, total }, JsonRequestBehavior.AllowGet);

}

[HttpPost]

public JsonResult Save(Player player)

{

new GridModel().Save(player);

return Json(true);

}

[HttpPost]

public JsonResult Remove(int id)

{

new GridModel().Remove(id);

return Json(true);

}

}

[AttributeUsage(AttributeTargets.Class | AttributeTargets.Method)]

public sealed class NoCacheAttribute : ActionFilterAttribute

{

public override void OnResultExecuting(ResultExecutingContext filterContext)

{

filterContext.HttpContext.Response.Cache.SetExpires(DateTime.UtcNow.AddDays(-1));

filterContext.HttpContext.Response.Cache.SetValidUntilExpires(false);

filterContext.HttpContext.Response.Cache.SetRevalidation(HttpCacheRevalidation.AllCaches);

filterContext.HttpContext.Response.Cache.SetCacheability(HttpCacheability.NoCache);

filterContext.HttpContext.Response.Cache.SetNoStore();

base.OnResultExecuting(filterContext);

}

}

<https://blog.goyello.com/2013/01/07/8-most-common-mistakes-c-developers-make/>

HASHSET

class Program

{

static void Main(string[] args)

{

const int COUNT = 100000;

HashSet<int> hashSetOfInts = new HashSet<int>();

Stopwatch stopWatch = new Stopwatch();

for (int i = 0; i < COUNT; i++)

{

hashSetOfInts.Add(i);

}

stopWatch.Start();

for (int i = 0; i < COUNT; i++)

{

hashSetOfInts.Contains(i);

}

stopWatch.Stop();

Console.WriteLine(stopWatch.Elapsed);

stopWatch.Reset();

List<int> listOfInts = new List<int>();

for (int i = 0; i < COUNT; i++)

{

listOfInts.Add(i);

}

stopWatch.Start();

for (int i = 0; i < COUNT; i++)

{

listOfInts.Contains(i);

}

stopWatch.Stop();

Console.WriteLine(stopWatch.Elapsed);

Console.Read();

}

}

After executing it you will see the difference:

100000 ‘Contains’ operations using HashSet: **0.002 sec**

100000 ‘Contains’ operations using List: **28.74 sec**

Of course HashSets or SortedSets are not golden means for every possible scenario. **HashSet** should be used in case when you care about the performance (especially if you know that you will operate on a large set of items) but do not care about the order. **SortedSet** is a bit slower than HashSet but will give you a sorted collection and is much faster than List.

LEER INFORMACIÓN DEL WEB CONFIG:

**//LEER UN KEY: RutaEncriptada**

string userName = System.Configuration.ConfigurationManager.AppSettings["RutaEncriptada"];

var smtpSection = (SmtpSection)ConfigurationManager.GetSection("system.net/mailSettings/smtp");

string strHost = smtpSection.Network.Host;

int port = smtpSection.Network.Port;

string strUserName = smtpSection.Network.UserName;

string strFromPass = smtpSection.Network.Password;

<add key="Ftp" value="RlRQOg==" />

<add key="FTP:Host" value="MTkwLjIxNy4xMTMuMjUz" />

<add key="FTP:Port" value="MjI=" />

<add key="FTP:Usuario" value="bGFmcmFuY29s" />

<add key="FTP:Contrasena" value="QWJiMHR0MjAxOSE=" />

ENCODING Y DECODING (CODIFICAR Y DECODIFICAR):

<https://onlineutf8tools.com/convert-utf8-to-base64>

//Leer del WebConfig

string Ruta2= "C:Archivos de Programa";

string Ruta = System.Configuration.ConfigurationManager.AppSettings["RutaEncriptada"];

byte[] toEncodeAsBytes = System.Text.Encoding.UTF8.GetBytes(Ruta2);

string returnValue = System.Convert.ToBase64String(toEncodeAsBytes);

string returnValue2 = "QzpcQXJjaGl2b3MgZGUgUHJvZ3JhbWE=";

byte[] encodedDataAsBytes = System.Convert.FromBase64String(returnValue2);

string returnValue3 = System.Text.Encoding.UTF8.GetString(encodedDataAsBytes);

Encriptar y Desencriptar:

//ENCRIPTAR:

string plainText = "c:\archivos de programa";

string passPhrase = "hola";

byte[] initVectorBytes = Encoding.UTF8.GetBytes(initVector);

byte[] plainTextBytes = Encoding.UTF8.GetBytes(plainText);

PasswordDeriveBytes password = new PasswordDeriveBytes(passPhrase, null);

byte[] keyBytes = password.GetBytes(keysize / 8);

RijndaelManaged symmetricKey = new RijndaelManaged();

//symmetricKey.Mode = CipherMode.CBC;

//ICryptoTransform encryptor = symmetricKey.CreateEncryptor(keyBytes, initVectorBytes);

//MemoryStream memoryStream = new MemoryStream();

//CryptoStream cryptoStream = new CryptoStream(memoryStream, encryptor, CryptoStreamMode.Write);

//cryptoStream.Write(plainTextBytes, 0, plainTextBytes.Length);

//cryptoStream.FlushFinalBlock();

//byte[] cipherTextBytes = memoryStream.ToArray();

//memoryStream.Close();

//cryptoStream.Close();

//DESENCRIPTAR

//string cipherText = Convert.ToBase64String(cipherTextBytes);

//string cipherText = "S+qaV1og3WxBQHxm+ZlPZX1NZZJ4+yhIdgmNguXFlbZQfejSTpBbTwz3ADMXUc+mSGFNXNTo+NjbL8eoUuiZ6A==";

//initVectorBytes = Encoding.UTF8.GetBytes(initVector);

//byte[] cipherTextBytes = Convert.FromBase64String(cipherText);

//password = new PasswordDeriveBytes(passPhrase, null);

//keyBytes = password.GetBytes(keysize / 8);

//symmetricKey = new RijndaelManaged();

//symmetricKey.Mode = CipherMode.CBC;

//ICryptoTransform decryptor = symmetricKey.CreateDecryptor(keyBytes, initVectorBytes);

//MemoryStream memoryStream = new MemoryStream(cipherTextBytes);

//CryptoStream cryptoStream = new CryptoStream(memoryStream, decryptor, CryptoStreamMode.Read);

//plainTextBytes = new byte[cipherTextBytes.Length];

//int decryptedByteCount = cryptoStream.Read(plainTextBytes, 0, plainTextBytes.Length);

//memoryStream.Close();

//cryptoStream.Close();

//string Desencriptado = Encoding.UTF8.GetString(plainTextBytes, 0, decryptedByteCount);

HALLAR DIRECCIÓN VIRTUAL (VIRTUAL PATH) O DIRECCIÓN DEL SERVIDOR

string SubRutaActual = Request.RawUrl;

RutaPlantilla = Request.Url.OriginalString.ToString();

var RutaFilal = RutaPlantilla.Replace(SubRutaActual, "");

DIRECTORIO FÍSICO DEL SERVIDOR

Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory)

VERIFICAR SI UN ARCHIVO EXISTE EN EL SERVIDOR:

if (System.IO.File.Exists(Path.GetDirectoryName(AppDomain.CurrentDomain.BaseDirectory)+RutaPlantilla))

INSERTAR UN ICONO GLOBAL

El archivo requerido debe tener este nombre: favicon.ico y debe ser de 16x16.

Se puede convertir en:

<https://icoconvert.com/>

Se inserta en la carpeta de imágenes del proyecto y en el Layout se agrega la siguiente línea en el HEAD:

<link rel="shortcut icon" href="~/Content/Images/favicon.ico" type="image/x-icon" />

TILDES EN JS

CARACTERES ESPECIALES

<http://www.etnassoft.com/2010/12/28/cadenas-de-escape-como-poner-tildes-en-javascript/>

Las cadenas de escape más conocidas son:

\n: Salto de línea

\r: Retorno de carro

\t: Tabulación horizontal

\v: Tabulación vertical

\’: Comilla simple

\”: Comilla doble

\\: Barra invertida

\xdd: Carácter especial especificado mediante dos dígitos hexadecimales dd

\u00F3 : ó



GENERAR ARCHIVO PLANO:

Opción #1: C#

using System.IO;

using System;

// para crear el archivo

string rutaCompleta = @" E:\mi archivo.txt";

string texto = "REGISTRO TIPO 1 NIT RAZÓN SOCIAL ";

using (StreamWriter mylogs = new StreamWriter(rutaCompleta, true)) //se crea el archivo

{

//se adiciona alguna información y la fecha

DateTime dateTime = new DateTime();

dateTime = DateTime.Now;

string strDate = Convert.ToDateTime(dateTime).ToString("yyMMdd");

mylogs.WriteLine(texto + strDate);

mylogs.Close();

}

long consecutivo=1;

using (StreamWriter file = new StreamWriter(rutaCompleta, true))

{

foreach (DTReporteMinisterio item in ResultadoReporte)

{

file.WriteLine(consecutivo+"|"+item.tipodocumento+"|"+item.numdocumento+'|'+item.nombrecompleto+'|'+item.tiporeceptor+'|'+item.tipotransferencia); //se agrega información al documento

consecutivo++;

}

file.Close();

}

// para leer la información el archivo

void LeerInfoTxt()

{

string rutaCompleta = @" D:\mi archivo.txt";

string line = "";

using (StreamReader file = new StreamReader(rutaCompleta))

{

while ((line = file.ReadLine()) != null) //Leer linea por linea

{

Console.WriteLine(line);

}

// OTRA FORMA DE LEER TODO EL ARCHIVO

line = file.ReadToEnd();

Console.WriteLine(line);

file.close();

}

}

Opción #2: JS

document.getElementById("dwn-btn").addEventListener("click", function () {

if ($("#start").val() !== "" && $("#end").val() !== "") {

var Datos = {};

Datos.FechaInicial = $("#start").val();

Datos.FechaFinal = $("#end").val();

objMensajes.MensajeProgreso("Generando Archivo.txt");

$.ajax({

type: "POST",

url: "/AlimentosYBebidas/ExportTextFile",

data: JSON.stringify(Datos),

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

success: function (data) {

if (data !== "") {

console.log(data);

var filename = "Archivo.txt";

download(filename, data);

objMensajes.OcultarMensaje();

}

else {

ObjHome.CargarGrid(data.Datos, MostrarMensajeDeGridVacio);

if (MostrarMensajeDeGridVacio === 1)

objMensajes.MostrarMensaje(data.Mensaje);

}

}

});

} else {

alert("Mal");

}

}, false);

function download(filename, text) {

var element = document.createElement('a');

element.setAttribute('href', 'data:text/plain;charset=utf-8,' + encodeURIComponent(text));

element.setAttribute('download', filename);

element.style.display = 'none';

document.body.appendChild(element);

element.click();

document.body.removeChild(element);

}

ENVIAR E-MAIL

string respuesta = "";

try

{

//Obtenemos el servidor smtp del archivo de configuración (Web.config)

var smtpSection = (SmtpSection)ConfigurationManager.GetSection("system.net/mailSettings/smtp");

string strHost = smtpSection.Network.Host;

int port = smtpSection.Network.Port;

string strUserName = smtpSection.Network.UserName;

string strFromPass = smtpSection.Network.Password;

MailMessage Msg = new MailMessage();

Msg.From = new MailAddress(smtpSection.From);

Msg.To.Add(DT\_Usuario.Correo);

Msg.IsBodyHtml = true;

Msg.BodyEncoding = Encoding.GetEncoding("iso-8859-1");

Msg.Subject = "Nueva contraseña en RTVSS";

Msg.Body += "<p>MENSAJE DE RTVSS:</p> <BR/>";

Msg.Body += "<P>HOLA.</P><br/> <P>Se ha generado una nueva contraseña para el usuario: <b>" + DT\_Usuario.Usuario + "</b> en el sistema RTVSS.</P><br/>";

Msg.Body += "<P>La nueva contraseña es: " + clave.Original + "</P><br/>";

Msg.Body += "Por favor recuerde que tiene cinco (5) días para acceder al sistema y generar una nueva contraseña<br/>";

SmtpClient smtp = new SmtpClient();

smtp.Host = "smtp.gmail.com";

smtp.Port = 587;

smtp.Credentials = new System.Net.NetworkCredential(strUserName, strFromPass);

smtp.EnableSsl = true;

smtp.Send(Msg);

FORMATO FECHA

CONVERTIR FECHAS EN C#

**CONVERTIR FECHA dd/mm/yyyy a fecha SQL (yyyy/mm/dd)**

string FI = DT\_Patrocinio.FechaInicio;

string[] tempsplit = FI.Split('/');

string joinstring = "/";

string newdate = tempsplit[2] + joinstring + tempsplit[1] + joinstring + tempsplit[0];

**CONVERTIR FECHA DEL SQL EN dd/MM/yyyy (JS)**

$("#txtFechaInicio").val(moment(data.Datos[0].FechaInicio).format('DD/MM/YYYY'));

**LEER FECHA DE KENDO DATE PICKER**

var year1 = $("#start").data("kendoDatePicker").value()

**LEER AÑO DE UN KENDO DATE PICKER**

var year1 = $("#start").data("kendoDatePicker").value().getFullYear();

**CONVERTIR FECHA DEL SQL EN dd/MM/yyyy**

(Modelo: NuevoGrant)

string FechaInicial = Convert.ToDateTime(DT\_Reporte.FechaInicial).ToString("yyyy-MM-dd");

----

EN EL JS:

Datos.FechaInicial = $("#start").val();

-----

var start = $("#start").kendoDatePicker({

max: new Date(),

min: new Date(2018, 0, 1),

format: "dd-MM-yyyy",

language: "es",

autoclose: true

}).data("kendoDatePicker");

----

string I = (Datos.FechaInicial).ToString("dd/MM/yyyy") + " 00:00:00.000";

DateTime Fecha\_I = Convert.ToDateTime(I);

----------------------------- MAS FECHAS -----------------

//string FI = Datos.FechaInicial.ToString();

//string[] tempsplit = FI.Split('/');

//string joinstring = "/";

//string newdate = tempsplit[2] + joinstring + tempsplit[1] + joinstring + tempsplit[0]+ " 00:00:00.000";

//SOLUCIÓN ANTERIOR:

//string I = (Datos.FechaInicial).ToString("dd/MM/yyyy") + " 00:00:00.000";

//string F = (Datos.FechaFinal).ToString("dd/MM/yyyy") + " 23:59:59.997";

//DateTime Fecha\_I = Convert.ToDateTime(I);

//DateTime Fecha\_F = Convert.ToDateTime(F);

//String FechaI = Convert.ToDateTime(Datos.FechaInicial).ToString("yyyy-MM-dd") + " 00:00:01";

//String FechaF = Convert.ToDateTime(Datos.FechaFinal).ToString("yyyy-MM-dd") + " 23:59:59";

//String FechaI = Convert.ToDateTime(Datos.FechaInicial).ToString("yyyy-MM-dd");

//String FechaF = Convert.ToDateTime(Datos.FechaFinal).ToString("yyyy-MM-dd");

//string FechaI = Datos.FechaInicial.ToString("yyyy-MM-dd HH:mm:ss.fff");

//string FechaF = Datos.FechaFinal.ToString("yyyy-MM-dd HH:mm:ss.fff");

//string date = "14/05/2019";

//DateTime result = DateTime.ParseExact(FechaI, "yyyy-MM-dd",

//CultureInfo.InvariantCulture);

//DateTime FechaInicial = DateTime.Parse(FechaI, null, DateTimeStyles.RoundtripKind);

//DateTime FechaFinal = DateTime.Parse(FechaF, null, DateTimeStyles.RoundtripKind);

//DateTime FechaInicial = DateTime.Parse(FechaI, null, DateTimeStyles.AdjustToUniversal);

//DateTime FechaFinal = DateTime.Parse(FechaF, null, DateTimeStyles.AdjustToUniversal);

//String FechaInicial = Convert.ToDateTime(Datos.FechaInicial).ToString("yyyyMMdd");

//String FechaFinal = Convert.ToDateTime(Datos.FechaFinal).ToString("yyyyMMdd");

//DateTime FIni = DateTime.ParseExact(FechaI, "yyyy-MM-dd hh:mm:ss", CultureInfo.InvariantCulture);

//DateTime FechaFin = Convert.ToDateTime(FechaFinal);

//DateTime start\_date;

//DateTime.TryParseExact(FechaI, "yyyy-MM-dd HH:mm:ss",

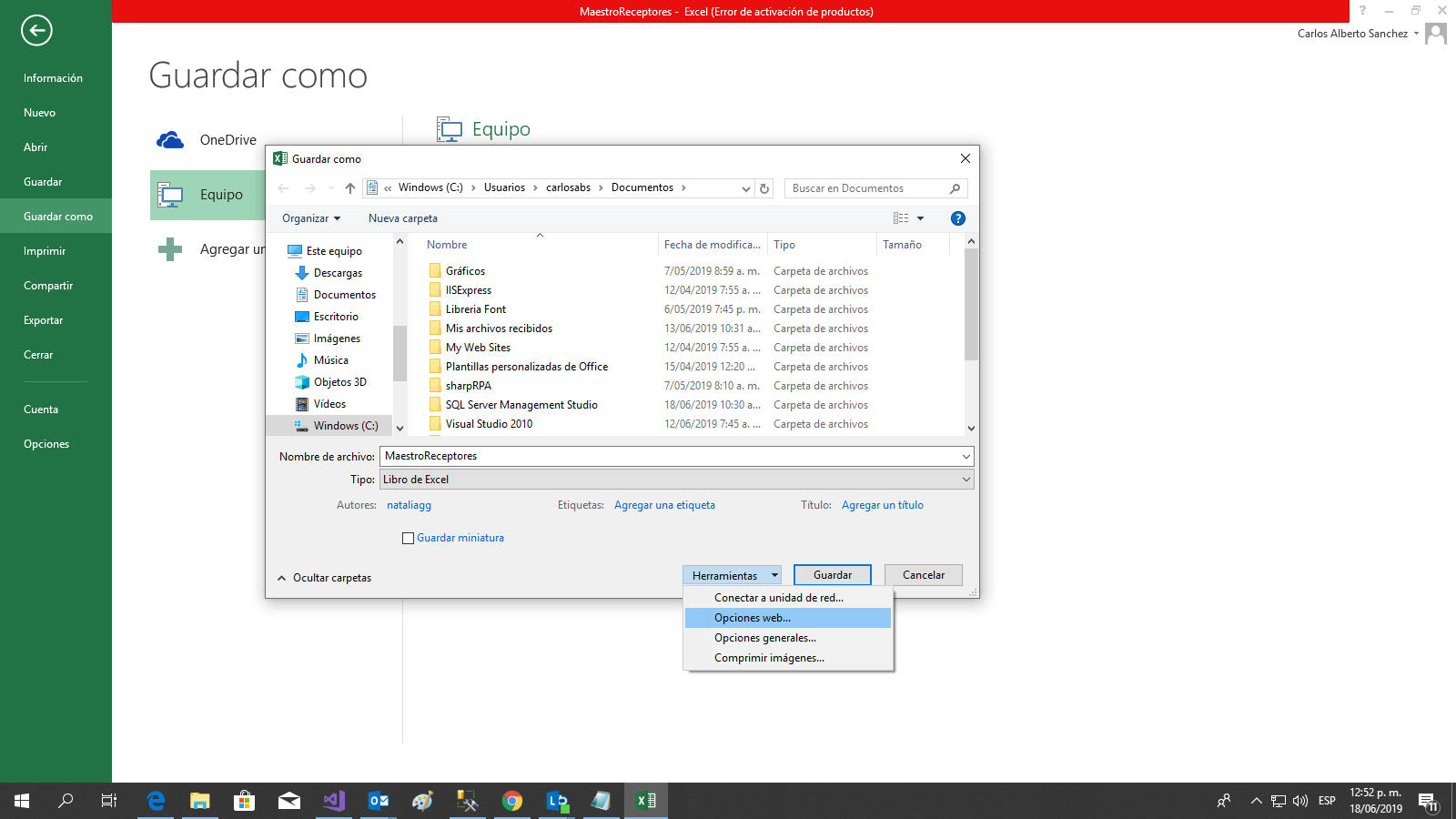
System.Globalization.CultureInfo.InvariantCulture,

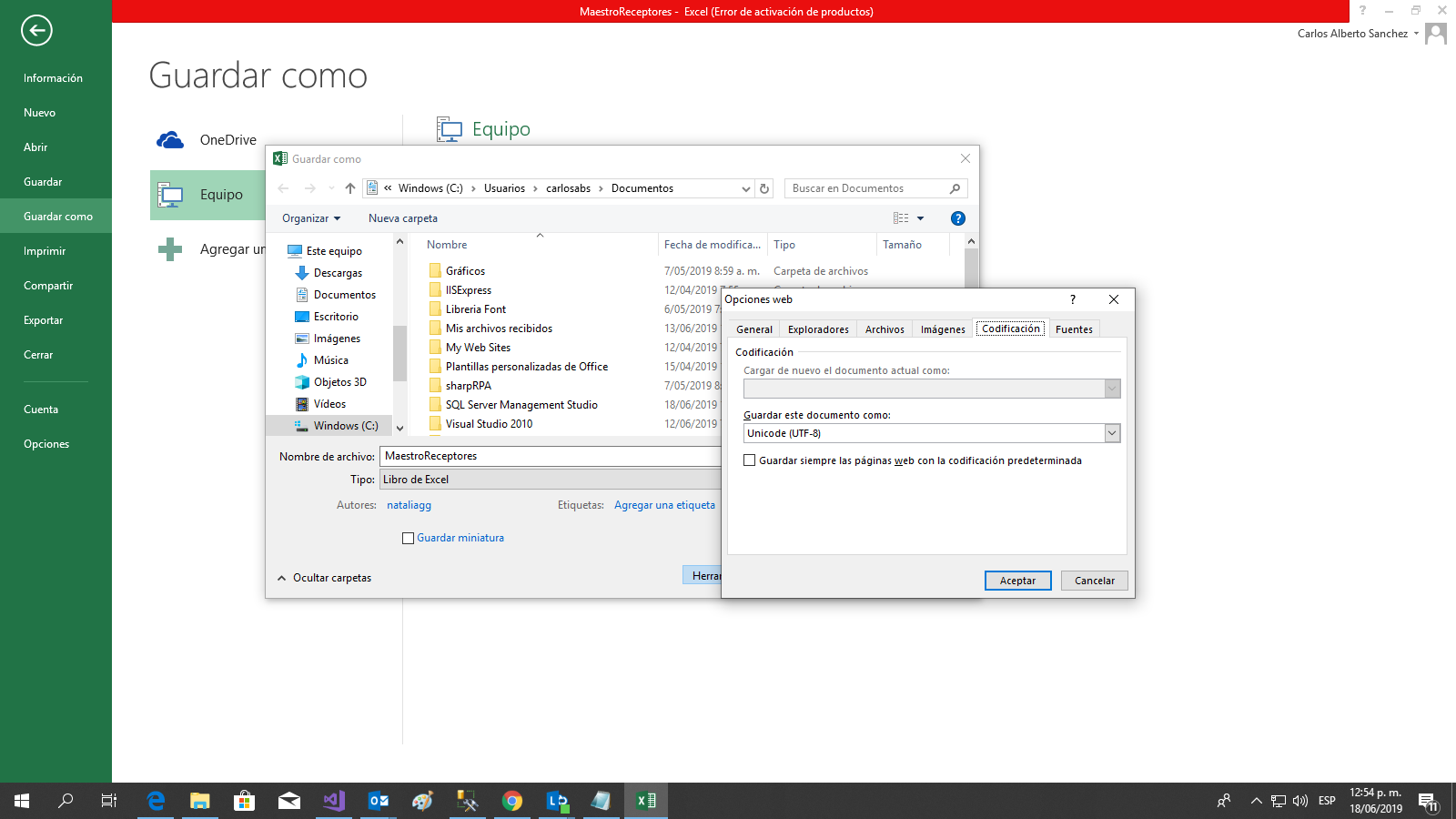
System.Globalization.DateTimeStyles.None, out start\_date);

//DateTime Fecha\_F=DateTime.Now;

SUBIR ARCHIVO EXCEL

El archivo debe estar guardado en codificación UTF-8





<https://stackoverflow.com/questions/23259173/unexpected-character-encountered-while-parsing-value/42313090>

ADICIONAR HORAS A FECHA

dateValue.AddHours(hour))

EXTRAER SUBSTRIG DE CADENA

string newString = str.Substring(0,10)

ACTUALIZAR ARCHIVO EN EL BIN

En propiedades del archivo, clic en la Propiedad:

Copiar en el Directorio de Salida: **Copiar siempre**

CLASES

## **Accesibilidad**

Cada miembro de una clase tiene asociada una accesibilidad, que controla las regiones del texto del programa que pueden tener acceso al miembro. Existen seis formas de accesibilidad posibles. Se resumen a continuación.

* public
  + Acceso no limitado
* protected
  + Acceso limitado a esta clase o a las clases derivadas de esta clase
* internal
  + Acceso limitado al ensamblado actual (.exe, .dll, etc.)
* protected internal
  + Acceso limitado a la clase contenedora, las clases derivadas de la clase contenedora, o bien las clases dentro del mismo ensamblado
* private
  + Acceso limitado a esta clase
* private protected
  + Acceso limitado a la clase contenedora o las clases derivadas del tipo contenedor con el mismo ensamblado

CLASE ESTÁTICA

Los requisitos a la hora de crear una clase estática son los siguientes:

* La clase debe derivar directamente de System.Object porque la herencia se aplica a clases de instancia y como hemos dicho, una clase estática no realiza intanciación alguna.
* La clase no debe implementar ningún interfaz ya que que los métodos del interfaz son llamados en una instancia de la clase.
* La clase debe definir únicamente miembros estáticos. Cualquier miembro de instancia provocará un error de compilación.
* La clase no puede usarse como un campo, como parámetro de un método o como variable local ya que todos estos conceptos refieren a una instancia.
* Sólo puede contener [constructores estáticos](https://msdn.microsoft.com/en-us/library/k9x6w0hc.aspx) y privados.

Se usa el modificador [static](https://msdn.microsoft.com/en-us/library/98f28cdx.aspx) para definir una clase o miembro estático.

VINCULAR UNA LIBRERÍA DE CLASES (BIBLIOTECA DE CLASES)

-> Click derecho en el proyecto

-> Agregar Biblioteca de Clases

-> Agregar las carpetas y clases necesarias

-> En las clases eliminar la instrucción: namespace

-> Las clases deben ser públicas

-> Clic Derecho en el Proyecto Web -> Agregar -> Referencias

-> Seleccionar la biblioteca creada

LEER INFO DE JSON

public JsonResult MostrarJson(DTPersona P) {

JsonResult Resultado = new JsonResult();

DTPersona Persona = new DTPersona();

JavaScriptSerializer ser = new JavaScriptSerializer();

string outputJSON = System.IO.File.ReadAllText("C:\CARLOS\NET\SOLUCIONES\JAVASCRIPT\JS1\JS1\Json\json.json");

Persona = JsonConvert.DeserializeObject<DTPersona>(outputJSON);

Console.WriteLine(Persona.ToString());

Console.ReadLine();

ADMINISTRAR CACHE

<https://www.codeproject.com/Articles/12117/Simulate-a-Windows-Service-using-ASP-NET-to-run-sc>

When the application is started the ASP.Net Framework create an instace of Global class defined in the Global.asax.cs and invoke two important methods: Application\_start and Application\_end(src, args)

When an application stops, we get a callback at Application\_End

PROBLEMA: Se inserta el item al cache pero se borra.

CacheItemPriority.NotRemovable

LISTAS EN C#

List<string> LoggedInUsers = new List<string>();

LoggedInUsers.Add(Usuario); //Insertar item a Lista

DICTIONARY IN C #

**Dictionary**<string, int> data = new Dictionary<string, int>()

{

{"cat", 2},

{"dog", 1},

{"llama", 0},

{"iguana", -1}

};

// Loop over pairs with foreach.

foreach (**KeyValuePair**<string, int> pair in data)

{

Console.WriteLine("FOREACH KEYVALUEPAIR: {0}, {1}", pair.Key, pair.Value);

}

// Use var keyword to enumerate Dictionary.

foreach (var pair in data)

{

Console.WriteLine("FOREACH VAR: {0}, {1}", pair.Key, pair.Value);

}

ELIMINAR DE UNA LISTA.

public ActionResult Logout(string id)

{

//NUEVO --------------------------------------------------

Dictionary<string, string> UsuariosRTVSS = MvcApplication.UsuariosActivos;

DTUsuario \_usuario = (DTUsuario)Session[DTNombreSesion.DTUsuario];

string User = \_usuario.Usuario;

try

{

foreach (var item in UsuariosRTVSS)

{

if (item.Value == User)

UsuariosRTVSS.Remove(item.Key);

continue;

}

}

catch (Exception e) {

}

OTRA FORMA:

public static void RemoveAllByValue<K, V>(this Dictionary<K, V> dictionary, V value) { foreach (var key in dictionary.Where( kvp => EqualityComparer<V>.Default.Equals(kvp.Value, value)). Select(x => x.Key).ToArray()) dictionary.Remove(key); }

OTRA FORMA:

var key=dict.FirstOrDefault(m => m.Value == s).Key; dict.Remove(key);

OTRA FORMA:

You can use the following as extension method

public static void RemoveByValue<T,T1>(this Dictionary<T,T1> src , T1 Value) { foreach (var item in src.Where(kvp => kvp.Value.Equals( Value)).ToList()) { src.Remove(item.Key); } }

LISTAS EN C#

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* LISTA LOCAL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

List<Usuarios> parts = new List<Usuarios>();

// Add parts to the list.

parts.Add(new Usuarios() { PartName = "crank arm", PartId = 1234 });

Console.WriteLine("\nContains(\"1734\"): {0}",

parts.Contains(new Usuarios { PartId = 1734, PartName = "" }));

//Remove Parts from the list

parts.Remove(new Usuarios() { PartId = 1534, PartName = "cogs" });

////\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* LISTA GLOBAL \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class Usuarios : IEquatable<Usuarios>

{

public string PartName { get; set; }

public int PartId { get; set; }

public override string ToString()

{

return "ID: " + PartId + " Name: " + PartName;

}

public override bool Equals(object obj)

{

if (obj == null) return false;

Usuarios objAsPart = obj as Usuarios;

if (objAsPart == null) return false;

else return Equals(objAsPart);

}

public override int GetHashCode()

{

return PartId;

}

public bool Equals(Usuarios other)

{

if (other == null) return false;

return (this.PartId.Equals(other.PartId));

}

private static List<Usuarios> \_carList = new List<Usuarios>();

public static List<Usuarios> CarList

{

get

{

return \_carList;

}

}

}

<https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.list-1?view=netframework-4.8>

GetHashCode

TRATAMIENTO DE VALORES NULOS

DTUsuario \_usuario = (DTUsuario)Session[DTNombreSesion.DTUsuario];

string User = \_usuario == null ? null : \_usuario.Usuario;

if (User!=null)

{

//User no es Nulo

}

<https://es.stackoverflow.com/questions/75999/qu%C3%A9-es-una-nullreferenceexception-y-c%C3%B3mo-solucionarla>

VARIABLES SESION EN C#

Las variables Sesion están disponibles solo en el ámbito de un usuario.

The session data is used for a user, the application data is used for all users.

<https://stackoverflow.com/questions/12365705/asp-net-mvc-c-sharp-global-variable>

SESSION END (GLOBAL ASAX)

<http://www.elguille.info/colabora/NET2005/GRP_Redireccion_en_Session_End.htm>

## CONVERTIR EN XML

public static string ConvertObjectToXML<T>(T Obj)

{

XmlSerializer ser = new XmlSerializer(typeof(T));

string xml = "";

XmlDocument \_XmlDocument = new XmlDocument();

//Serializar objeto

using (StringWriter sww = new StringWriter())

{

using (XmlWriter writer = XmlWriter.Create(sww))

{

ser.Serialize(writer, Obj);

xml = sww.ToString(); // Your XML

}

}

//Cargar en objeto xmldocument

\_XmlDocument.LoadXml(xml);

//eliminar primer nodo de declaracion xml

if (\_XmlDocument.FirstChild.NodeType == XmlNodeType.XmlDeclaration)

\_XmlDocument.RemoveChild(\_XmlDocument.FirstChild);

return \_XmlDocument.InnerXml;

}

## Eventos del global.asax

Entre todos los eventos global.asax, cabe destacar como principales los siguientes:

* Application\_Init: Inicio de la aplicación
* Application\_Disposed: Se dispara justo antes de que una aplicación sea destruida. Este método es ideal para limpiar anteriormente recursos que fueron utilizados.
* Application\_Start: Se dispara cuando se ha creado la primera instancia de la clase HttpApplication. Esto permite crear objetos que son accesibles a las todas las instancias de HttpApplication.
* Application\_End: Último evento lanzado antes de finalizar por completo la aplicación.
* Application\_Error: Se ejecuta cuando salta un error no controlado dentro de la aplicación.
* Application\_BeginRequest: Evento lanzado por cada petición de un cliente, ya sea el acceso a una url, una imagen, etc...
* Application\_EndRequ: Último evento antes de dar finalizada una petición.
* Session\_Start: Nueva sesión iniciada por un usuario. Muy usado junto al session\_end para establecer contadores de usuarios activos.
* Session\_End: Fin de sesión para un usuario.

you could write the values into the "Application" object.

void Application\_Start(object sender, EventArgs e) {

// Code that runs on application startup

Application.Add("Foo", "Bar"); }

Lastly, you can mark the property you want to expose from the Global as static.

public static string Abc { get; set; }

void Application\_Start(object sender, EventArgs e) {

// Code that runs on application startup

Abc = "123"; }

MULTIPLE INSTANCE OF APPLICATION

<https://ehikioya.com/multiple-instances-of-same-application/>

CURSO C# AVANZADO

<https://csharp-station.com/Tutorial/CSharp/SmartConsoleSetup.aspx>

<https://mva.microsoft.com/en-US/training-courses/c-fundamentals-for-absolute-beginners-16169?l=Lvld4EQIC_2706218949>

ENUM

public enum Volume

{

Low,

Medium,

High

}

// demonstrates how to use the enum

class EnumSwitch

{

static void Main()

{

// create and initialize

// instance of enum type

Volume myVolume = Volume.Medium;

// make decision based

// on enum value

switch (myVolume)

{

case Volume.Low:

Console.WriteLine("The volume has been turned Down.");

break;

EXCEPCIONES:

using System;

using System.IO;

class tryCatchDemo

{

static void Main(string[] args)

{

try

{

File.OpenRead("NonExistentFile");

}

catch(Exception ex)

{

Console.WriteLine(ex.ToString());

}

finally

{

if (outStream != null)

{

outStream.Close();

Console.WriteLine("outStream closed.");

}

if (inStream != null)

{

inStream.Close();

Console.WriteLine("inStream closed.");

}

}

}

LINQ

LINQ QUERY

List<car> myCars = new List<car>(){

new car (){Make=”bmw”, Model=”Cayman”, Year=1965 }

new car () {Make=”Toyota”, Model=”SUV”, Year=2018}

}

LINQ QUERY:

var bmw = from car in mycars

where car.Year=2018 && car.Make=”Toyota”

orderby car.Year descending

select car;

LINQ METHOD:

var bmw = mycars.where(p=> p.Make==”bmw” && p.year==2018)

var bmw = mycars.OrderByDescending(p=> p.Year);

var bmw = mycars.TrueForAll(p=> p.Year>2002);

mycars.ForEach(p=> Console.writeline(”{0}, {1,c}”,p.Year, p.price));

Console.WriteLine (mycars.Exist(p=>p.Model==”SUV”);

Console.WriteLine(mycars.sum (p=>p.Price);

class car{

public string Model {get;set;}

public int Year {get;set;}

public money Year {get;set;}

}

SERVIDOR DE PRUEBAS:

Página: [http://carlossan.somee.com](http://carlossan.somee.com/)

<https://somee.com/>

Username: **carlossan**

Email: carlossan@gmail.com

FTP: **198.37.116.30**

Puerto FTP: **21**

SERVICIOS (REST API)

Servicio para pruebas:

<https://demo.docusign.net/restapi/v2/login_information>

CAPÍTULO JS

MAYÚSCULAS EN JS TO UPPER CASE

if (String(x.toUpperCase() === String(NuevoPais.toUpperCase())) {

contador++;

}

Get number of options in Select

var x = document.getElementById("mySelect").length;

Pasar varios argumentos dinámicos a función string html template: function (dataItem) {

if (dataItem.Estado === true) {

return '<center><a title="Inactivar División" data-toggle="tooltip" class="btn btn-success" onclick="ActualizarEstadoDivision(\'' + dataItem.IdDivision + '\',\'' + dataItem.Nombre + '\', 0)"><span class="glyphicon glyphicon-ok-sign"></span></a></center>';

} else if (dataItem.Estado === false) {

return '<center><a title="Activar División" data-toggle="tooltip" class="btn btn-danger" onclick="ActualizarEstadoDivision(\'' + dataItem.IdDivision + '\',\'' + dataItem.Nombre + '\', 1)"><span class="glyphicon glyphicon-remove-sign"></span></a></center>';

}

}

VALIDAR NUMÉRICO

VALIDAR SI ES UN NÚMERO

numero = parseInt(criterioBusqueda.match(/\d+/), 10);

if (numero == criterioBusqueda) {

respuesta = true;

} else { alertify.error("Por favor ingrese un número”)}

Limpiar control (LIMPIAR INPUT):

$('#txtNuevoPais').val('');

$("#txtSEC").val("");

$('#txtValorReal').html('');

SET VALUE TO CHECK BOX (ESTABLECER VALOR A CHECKBOX)

$('#CheckTiquetesVR').attr('checked', 'checked');

**Estas dos se pueden combinar para check y Uncheck:**

document.getElementById("checkTransporteVR").checked = true;

document.getElementById("CheckAlojamientoVR").checked = false;

OBTENER VALOR DE CHECKBOX

var Tte = document.getElementById("CheckTransporte").checked;

if (Tte) {

}

CREAR UN ARCHIVO DE TEXTO

ESTABLECER VALOR A SELECT (SET VALUE TO SELECT)

function seleccionarValor(combo, val) {

var lc = document.getElementById(combo);

var max = lc.options.length;

for (var i = 0; i < max; i++) {

if (lc.options[i].innerText === val) {

lc.selectedIndex = lc.options[i].index;

}

}

}

CHEQUEAR DENTRO DE UN GRUPO DE CHECKBOX CUALES FUERON SELECCIONADOS:

1. var seleccionadas = new Array();
2. var indice = 0;
3. $('.checkbox').each(function(){
4. if( this.attr('checked'){
5. indice = indice + 1; seleccionadas[indice] = this.val();
6. }
7. });

O

1. $('#enviar').click(function()
2. {
3. var selectedItems = new Array();
5. $("input[@name='itemSelect[]']:checked").each(function(){
6. selectedItems.push($(this).val());
7. });
9. alert(selectedItems);
10. });

VARIABLES OBJETOS

This code assigns **many values** (Fiat, 500, white) to a **variable** named car:

var car = {type:"Fiat", model:"500", color:"white"};

CAMBIAR Z INDEX(FOCO)

$("#modal-Ciudad").css({ 'z-index': 10});

GENERAR ARCHIVO DE TEXTO DESDE JS

Para la generación del TXT en C#:

data += “Dato a publicar” + "\r\n";

**Parámetros:**

filename: Nombre del archivo

text: texto a insertar en el archivo txt

function download(filename, text) {

var element = document.createElement('a');

element.setAttribute('href', 'data:text/plain;charset=utf-8,' + encodeURIComponent(text));

element.setAttribute('download', filename);

element.style.display = 'none';

document.body.appendChild(element);

element.click();

document.body.removeChild(element);

}

MOSTRAR MODAL

$('#playerModal').modal('show');

OCULTAR MODAL

$('#playerModal').modal('hide');

CONVERTIR NÚMERO EN STRING

var numero=5;

var str=numero.toString();

ASIGNAR VALOR A LABEL

<label id="txtCapitan" style="width: 90%;" class="form-control"></label>

$("#txtNombreBarco").html(v.NombreBarco);

CONVERTIR STRING EN NÚMERO

var a = parseInt("10")

FIND

var obj = Charlas.find((o, i) => {

CONVERTIR STRING TO SHORT (INT16)

Convert.ToInt16

CONTROLES DINÁMICOS (CHECK DIMÁMICOS BASADOS EN CONSULTA)

var Valor = x;

$.ajax({

url: '/Conceptos/ConsultarConceptos',

type: "GET",

dataType: "JSON",

data: 0,

success: function (data) {

if (data.Resultado) {

$('#bodyConceptos' + Valor + '').html("");

$.each(data.Datos, function (i, v) {

IncrementableConcepto++;

if (v.Basico === true) {

$('#bodyConceptos' + Valor + '').append('<div class="col-md-6">' +

'<input type="checkbox" id = "check' + v.TipoConcepto + '' + x + '" name="check' + v.TipoConcepto + '" value="false" onclick = "CambiarValue(this.id, this.value, ' + v.IdConceptos + ', ' + Valor + ', ' + IncrementableConcepto + ')" >' +

'<label for="check' + v.TipoConcepto + '' + x + '">' + v.TipoConcepto + '</label>' +

'</div>');

}

});

} else {

$('#bodyConceptos' + Valor + '').append('<option value="0">No se encontraron conceptos</option>');

}

}

});

CONTENEDOR DIMÁMICO

$("#ContenedorCharlas").append('<div class="col-md-12 form-group">' +

'</div><div class="col-md-3"><input type="text" name="name" value="' + v.FechaEjecucion + '" class="form-control" disabled /></div>' +

'<div class="col-md-4"><input type="text" name="name" value="' + v.Observaciones + '" class="form-control" id="TxtObservaciones' + v.IdCharlas + '" /></div>' +

'<div class="col-md-4"><select class="form-control" id="TxtSelect' + v.IdCharlas + '">' +

'<option value="Pendiente">Pendiente</option >' +

'<option value="Cancelado">Cancelado</option>' +

'<option value="Finalizado">Finalizado</option></select></div>' +

'' + Btn + '</div>');

DETECTAR EVENTO CARGAR PÁGINA

document.addEventListener('load', function () {

alert("cargo sitio");

});

ESTABLECER CSS CLASS (CHANGE CSS CLASS)

document.getElementById("btnBorrador").className = "botonEstado";

document.getElementById("btnFinalizado").className = "botonEstado

CAMBIAR ESTILO DE ELEMENTO DINÁMICAMENTE CON JS

document.getElementById("Participacion").style["display"] = "block";

ASIGNAR VALOR A DIV O SPAN DINÁMICAMENTE

EN HTML:

<span id="TxtIdContrato"></span>

EN JS:

var divNombre01 = document.getElementById('txtNombreEventoPDF');

divNombre01.innerHTML = data.Datos[0].NombreEvento;

ASIGNARLE FUNCIONALIDAD A UN BOTÓN (CUANDO SE HACE CLIC EN UN BOTON (SPAN) Hacer clic en otro botón.

**EN LA VISTA:**

<button class="browse-btn"> Búsqueda de archivos </button>

<input type="file" id="flArchivo" accept=".xls, .xlsx" onchange="MostarBotonCargar();">

**EN JS:**

const uploadButton= document.querySelector('.browse-btn');

const realInput = document.getElementById('flArchivo');

uploadButton.addEventListener('click', (e) => {

realInput.click(e);

});

ENVIAR VARIOS OBJETOS DESDE JS A CONTROLADOR

var Charlas = Array();

var ObjConceptos = Array();

Charlas[i] = { IdCharlas: Num, FechaEjecucion: "", Observaciones: "", Estado: "false" };

var Datos = {};

Datos = {

TipoProveedor: $('#TipoProveedor').val(),

IdLinea: $('#SlctLinea').val(),

IdTipoServicio: $('#SlcTipoServicio').val(),

IdSubTipoServicio: $('#SlcSubServicio').val(),

SolpedHonorarios: $('#NumSopedHonorarios').val(),

NumSec: $('#NumeroSec').val(),

ObjCharlas: Charlas,

ObjDtllConcepto: ObjConceptos,

RECIBIR EN EL CONTROLADOR:

public string AgregarContrato(DTContratos objContrato, List<DTCharlas> ObjCharlas, DtllConcepto DatosConcepto, List<DtllConcepto> ObjDtllConcepto)

{

SWITCH JS

switch (v.Estado) {

case 'Borrador':

document.getElementById("btnBorrador").className = "botonSeleccionado";

break;

case 'Finalizado':

document.getElementById("btnFinalizado").className = "botonSeleccionado";

break;

case 'Anulado':

document.getElementById("btnAnulado").className = "botonSeleccionado";

break;

}

BUCLE FOR EN JS

for (paso = 0; paso < 5; paso++) { // Se ejecuta 5 veces, con valores desde paso desde 0 hasta 4. console.log('Dando un paso al Este'); };

MODAL LOST FOCUS (SCROLL ON BACK PAGE AFTER DIALOG WITH MODAL)

*// When the modal is shown, we want a fixed body*

document.body.style.position = 'fixed';

document.body.style.top = `-${window.scrollY}px`;

*// When the modal is hidden, we want to remain at the top of the scroll position*

document.body.style.position = '';

document.body.style.top = '';

RECORRER OPTIONS DE UN SELECT EN JS

for (var i = 0; i < control.length; i++) {

x = document.getElementById(combo).options[i].text;

}

BLUR METHOD

The **blur** event occurs when an element loses focus. The **blur**() method triggers the**blur** event, or attaches a function to run when a **blur** event occurs.

PROGRESS BAR EN JS

Código HTML:

<div class="modal fade in" id="basicModal" tabindex="-1" data-keyboard="false" data-backdrop="static">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-body">

***<div id="progressBar"></div>*** </div>

<div class="modal-footer" style="border-top: 0px;">

</div> </div> </div></div>

JS:

La función inicializarProgressBar se debe llamar en el $(document).ready:

function inicializarProgressBar() {

var maxNumber = 100;

var startNumber = 0;

$("#progressBar").kendoProgressBar({

min: startNumber,

max: maxNumber,

type: "chunk"

});

}

function iniciarBarra() {

ObjHome.progreso = 20; //Restablece la barra de progreso.

zeeprogress(); //Le actualiza el valor a la barra de progreso

$("#basicModal").show(); //Muestra la barra de progreso.

myVar = setInterval('contador()', 100); //Activar contador automático barra progreso

}

function finalizarBarra() {

ObjHome.progreso = 100; //Proceso finalizado.

zeeprogress(); //Muestra que el proceso se completó.

clearInterval(myVar); //Detiene el timer que alimenta la barra de progreso.

$("#basicModal").hide(); //Oculta el modal con la barra de progreso

}

RESPONSETYPE EN AJAX

<https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/responseType>

The values supported by responseType are the following:

""

An empty responseType string is treated the same as "text", the default type (therefore, as a [DOMString](https://developer.mozilla.org/en-US/docs/Web/API/DOMString).

arraybuffer

The [response](https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/response) is a JavaScript [ArrayBuffer](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/ArrayBuffer) containing binary data.

blob

The response is a [Blob](https://developer.mozilla.org/en-US/docs/Web/API/Blob) object containing the binary data.

document

The response is an [HTML](https://developer.mozilla.org/en-US/docs/Glossary/HTML) [Document](https://developer.mozilla.org/en-US/docs/Web/API/Document) or [XML](https://developer.mozilla.org/en-US/docs/Glossary/XML) [XMLDocument](https://developer.mozilla.org/en-US/docs/Web/API/XMLDocument), as appropriate based on the MIME type of the received data. See [HTML in XMLHttpRequest](https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/HTML_in_XMLHttpRequest) to learn more about using XHR to fetch HTML content.

json

The response is a JavaScript object created by parsing the contents of received data as [JSON](https://developer.mozilla.org/en-US/docs/Glossary/JSON).

text

Similar to "arraybuffer", but the data is received in a stream. When using this response type, the value in response is only available in the handler for the [progress](https://developer.mozilla.org/en-US/docs/Web/Events/progress) event, and contains the cumulative data received since the request was sent.

ms-stream

The response is part of a streaming download; this response type is only allowed for download requests, and is only supported by Internet Explorer.

PETICIÓN AJAX OCULTANDO EL CONTROLADOR

EN EL ROUTE CONFIG:

routes.MapRoute(

name: "guardarProveedor",

url: "guardarProveedor",

defaults: new { controller = "Proveedor", action = "guardarProveedor" }

);

Y SIGUE ABAJO:

PETICIÓN AJAX ASÍNCRONA (ASYNCRONOUS AJAX REQUEST)

function ConsultaNoReportados() {

objMensajes.OcultarMensaje();

var Datos = {};

Datos.f\_Inicio\_Seleccionada = $("#start").val();

Datos.f\_Final\_Seleccionada = $("#end").val();

var xhr = new XMLHttpRequest();

xhr.open("GET", "guardarProveedor" + "?f\_Inicio\_Seleccionada=" + $("#start").val() + "&f\_Final\_Seleccionada=" + $("#end").val(), true); //true: is processed asynchronously

xhr.responseType = 'json';

xhr.onload = function () {

poemDisplay.textContent = request.response;

};

xhr.send();

xhr.onload = function () {

var data = xhr.response;

generarExcel(data);

};

}

PETICIÓN AJAX ASÍNCRONA PARA DESCARGAR ARCHIVO

function descargarArhivoAsincrono2() {

//Funciona pero mientras el usuario no active otro proceso.

fetch(URLactual + "\\Reportes\\NoReportados.xlsx")

.then(resp => resp.blob())

.then(blob => {

const url = window.URL.createObjectURL(blob);

const a = document.createElement('a');

a.style.display = 'none';

a.href = url;

a.download = 'NoReportados.xlsx';// Nombre del arvhivo solicitado.

document.body.appendChild(a);

a.click();

window.URL.revokeObjectURL(url);

});

}

EJECUTA UN PETICIÓN AJAX ANTES QUE OTRA:

function abcAsync(IdPatrocinio) {

var promise = timeoutAsync(50);

promise.done(function () { FUNCIÓN A EJECUTAR LUEGO(IdPatrocinio, 1); });

return promise;

}

function timeoutAsync(milliseconds) {

var deferred = $.Deferred();

setTimeout(function () { deferred.resolve(); }, milliseconds);

return deferred.promise();

}

RECORRER VALORES LEÍDOS POR UN AJAX:

$.each(Dato.Datos, function (i, v) {

$("#ComboTipoEvento").append("<option value='" + v.IdTiposEventos + "'>" + v.Nombre + "</option>");

});

COMBO FECHA KENDO

En la Vista:

<label style="width:150px"><b>Fecha Fin: </b></label>

<input id="FechaFin" style="width:150px !important;" />

En el JS:

$(document).ready(function () {

$('#FechaFin).kendoDatePicker({

value: new Date(),

language: "es",

todayHighlight: true,

autoclose: true

});

Para establecer el valor:

$("#txtFechaInicio").val(moment(data.Datos[0].FechaInicio).format('DD-MM-YYYY'));

Para Leer el valor ingresado y Guardar

Patrocinio.FechaInicio = moment($("#FechaInicio").val()).format('YYYY-MM-DD');

MOSTRAR FECHA EN ESPAÑOL

Descargar e insertar la librería (moment-with-locales.js) en el proyecto

Relacionar la Librería en el Layout

<script src="~/Scripts/Librerias/moment/moment-with-locales.js"></script>

En la parte superior del JS y en la instrucción antes de utiliza la fecha insertar:

moment.locale('es-do');

‘es-do’ es como aparece relacionado el idioma en: moment-with-locales

moment(dataItem.FechaRegistro).format('DD/MMM/YYYY');

IF SOFISTICADO

var a = (data.Datos[0].AplicaStand === 1) ? 1 : 0;

ARRAY EN JAVASCRIPT:

var ObjConceptos = Array();

var points = [];

var cars = ["Saab", "Volvo", "BMW"];

cars.push({ codigo: "01", descripcion: "Manuel" });

ALIMENTAR Y RECORRER ARRAY:

var arrayAno = new Array();

$.each(data.Datos, function (i, v) {

arrayAno.push(v.Ano);

});

var s = false;

for (var i = 2020; i < 3000; i++) {

for (var y = 0; y <= arrayAno.length; y++) {

if (i === arrayAno[y]) {

s = true;

}

}

if (s === false) {

$("#SelectAno").append("<option value='" + i + "'>" + i + "</option>");

}

s = false;

}

ESCRIBIR EN CONSOLA JS

console.log(“Escribe este mensaje”);

ENVIAR ARRAY X JSON:

var DT\_DllePatrocinio = {};

DT\_DllePatrocinio.IdPatrocinio = IdPatrocinio;

DT\_DllePatrocinio.TipoConcepto = TipoConcepto;

DT\_DllePatrocinio.IdReceptor = IdReceptor;

DT\_DllePatrocinio.Aplica = Aplica;

DT\_DllePatrocinio.IdReporteTercero = IdReporteTercero;

DT\_DllePatrocinio.Valor = Valor;

DT\_DllePatrocinio.EsValorFinal = 1; //Todos Los valores reales se registran con EsValorFinal=1

var data = { DT\_DllePatrocinio };

$.ajax({

url: '../Patrocinios/EditarIntegrante',

type: 'POST',

data: JSON.stringify(data),

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

dataType: "json",

success: function (data) {

if (data.Respuesta === true) {

OTRA FORMA:

var ObjConceptos = Array();

ObjConceptos.push({ IdPatrocinio: IdPatrocinio, TipoConcepto: 'Transporte', IdReceptor: IdReceptor, Aplica: Tte, IdReporteTercero: 1 });

ObjConceptos.push({ IdPatrocinio: IdPatrocinio, TipoConcepto: 'Alimentación', IdReceptor: IdReceptor, Aplica: Alim, IdReporteTercero: 1 });

ObjConceptos.push({ IdPatrocinio: IdPatrocinio, TipoConcepto: 'Tiquetes', IdReceptor: IdReceptor, Aplica: Tiq, IdReporteTercero: 1 });

ObjConceptos.push({ IdPatrocinio: IdPatrocinio, TipoConcepto: 'Alojamiento', IdReceptor: IdReceptor, Aplica: Aloj, IdReporteTercero: 1 });

ObjConceptos.push({ IdPatrocinio: IdPatrocinio, TipoConcepto: 'Inscripción', IdReceptor: IdReceptor, Aplica: Ins, IdReporteTercero: 1 });

var Datos = {};

Datos = {

ObjConceptos

};

$.ajax({

url: '../Patrocinios/ActualizarAP',

type: 'POST',

dataType: "json",

data: JSON.stringify(Datos),

processData: false,

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

success: function (data) {

if (data.Respuesta === true) {

objMensajes.OcultarMensaje();

alertify.success("La información fue guardada exitosamente");

LimpiarNuevoIntegrante();

ConsultarAplicaciones(IdPatrocinio, 1);

} else {

objMensajes.MostrarMensaje(data.Mensaje);

}

}

});

EN EL CONTROLADOR QUE RECIBE:

public JsonResult ActualizarAP(List<DTDllePatrocinio> ObjConceptos)

{

foreach (var item in ObjConceptos)

{

ResultadoTemporal = new BMPatrocinios().ActualizarAplicacion(DllePatrocinio);

}

OBTENER AÑO DE FECHA

var year1 = $("#start").data("kendoDatePicker").value().getFullYear();

DAR FORMATO A VALORES EN GRID KENDO

CURRENCY:

, {

title: "Alojamiento",

field: "Alojamiento",

format: "{0:c0}",

attributes: { class: "text-right" },

footerTemplate: SumAlojamiento(data)

},

VALIDAR LONGITUD MÍNIMA DE CAMPO

ValidarLongitudMinima: function () {

var LCorreo = $("#Correo").val().length;

var LUsuario = $("#Usuario").val().length;

if (LCorreo < 3) {

alertify.error("Por favor ingrese un correo electrónico.");

return false;

}

EXTRAER SUBSTRING

Fecha.substring(6, 10;

6: caracter inicial (inicia en 0)

10: caracter final

PASAR A ENTERO

parseInt(Fecha,10);

10: para especificar decimal

VALIDAR FECHAS

var Fecha = $("#txtFechaInicio").val();

var Ano = parseInt(Fecha.substring(6, 10),10);

var Mes = parseInt(Fecha.substring(3, 5),10);

var Dia = parseInt(Fecha.substring(0, 2),10);

var FechaIni = new Date(Ano, Mes, Dia);

if (FechaFin < FechaIni) { }

VALIDAR CARACTER @ EN CAMPO:

ValidarCaracteres: function () {

var nMay = 0, nMin = 0, nEsp = 0;

Correo = $("#Correo").val();

var t1 = "@";

for (i = 0; i < Correo.length; i++) {

if (t1.indexOf(Correo.charAt(i)) !== -1) { nMay++; }

}

if (nMay === 0) {

alertify.error("La cuenta de correo no es válida.");

return false;

}

else {

return true;

}

}

Alinear texto a la derecha:

attributes: { class: "text-right" },

Alinear Footer en grid Kendo:

footerTemplate: "<div style='text-align:right'>" + SumAlojamiento(data)

ALERTAS ALERTIFY

En el LAYOUT se debe insertar:

<head>

<link href="~/Content/alertify.core.css" rel="stylesheet" />

<link href="~/Content/alertify.default.css" rel="stylesheet" />

</head>

<body>

<script src="~/Scripts/Librerias/alertify.js"></script>

</body>

En el JS:

alertify.success(“<b>Señor</b> usuario por favor...”);

alertify.error(“Este es un mensaje de error”);

alertify.alert(“Mensaje a mostrar”);

alertify.log(“Mensaje a mostrar”);

function datos() {

alertify.prompt("Esto es un <b>prompt</b>, introduce un valor:", function (e, str) {

if (e) {

alertify.success("Has pulsado '" + alertify.labels.ok + "'' e introducido: " + str);

} else {

alertify.error("Has pulsado '" + alertify.labels.cancel + "'");

}

});

return false;

}

alertify.confirm("Desea generar el archivo de registros <b>'No Reportados'</b> ?",

function (e) {

if (e) {

ConsultaNoReportados();

//alertify.success('Yes');

} else {

alertify.success('No');

}

});

ALERTA CON ESPERA (DELAY)

**var** msg = alertify.message('Default message');

msg.delay(3).setContent('Wait time updated to 3 Seconds');

ALERTA CON ESPERA

alertify.alert("<b>Señor usuario!</b></br>Su Cotización ha sido registrada exitosamente!", function () {

window.location.reload();

});

EVITAR QUE UN BOTON RECARGUE UNA PÁGINA

$('#btnInsertarDepartamento').on('click', function (e) {

e.preventDefault();

IngresarDepartamento();

});

Desactivar botones DEPENDIENDO de un campo:

function DesactivarBotones() {

if (ObjHome.DTPatrocinio.Status === 'Cancelado' || ObjHome.DTPatrocinio.Status === 'Finalizado') {

var buttons = document.getElementsByClassName('btn-without-border');

for (var i = 0; i < buttons.length; i++) {

buttons[i].disabled = true;

}

}

}

## CAMBIAR DISABLED Y RESETEAR VALOR DE UN TEXTBOX DEPENDIENDO DEL VALOR SELECCIONADO EN UN COMBO:

var txtVS = document.getElementById('txtVrStand');

document.getElementById('cmbAplicaStand').value === '1' ? txtVS.disabled = false : (txtVS.disabled = true, document.getElementById('txtVrStand').value='');

DETECTAR CUANDO EL USUARIO PRESIONA ENTER

$("#NombreSearch").on('keyup', function (e) {

if (e.keyCode === 13) {

//if (globalTimeout !== null) clearTimeout(globalTimeout);

//globalTimeout = setTimeout(FiltrarReceptores\_Nombre(), 200);

FiltrarReceptores\_Nombre();

}

});

REDONDEAR EN JS

Math.ceil(2.3) = 3;

Math.floor(2.6) = 2;

Math.round(2.2) = 2;

ASIGNAR VALOR A COMBO:

document.getElementById("ComboGerenteModal").value = data.Datos[0].GerenteRepresentante;

$('#NombreModal').val(data.Datos[0].Perfil);

BOTÓN PARA ACCEDER A JS:

<a class="btn btn-warning" id="boton\_nuevo" href="javascript:NuevoIntegrante();"><span class="k-icon next"></span>Nuevo Integrante</a>

DISPARAR EVENTO POR CÓDIGO (EJECUTAR EVENTO)

$('#TxtValor').trigger('keyup','my\_custom\_parameter');

FORMATO $ (FORMATO PESOS A CONTROL)

EVENTO EN EL $(document).ready(function(){:

$("#txtValorAprobado").on({

keyup: function () {

formatCurrency($(this));

},

blur: function () {

formatCurrency($(this));

}

});

})

LEER VALOR:

DT\_Grant.ValorReal = FormatValor(document.getElementById("txtValorReal").value);

FUNCIONES:

function formatCurrency(input, blur) {

// appends $ to value, validates decimal side

// and puts cursor back in right position.

// get input value

var input\_val = input.val();

// don't validate empty input

if (input\_val === "") { return; }

// original length

var original\_len = input\_val.length;

// initial caret position

var caret\_pos = input.prop("selectionStart");

input\_val = formatNumber(input\_val);

input\_val = "$" + input\_val;

// send updated string to input

input.val(input\_val);

// put caret back in the right position

var updated\_len = input\_val.length;

caret\_pos = updated\_len - original\_len + caret\_pos;

input[0].setSelectionRange(caret\_pos, caret\_pos);

}

function formatNumber(n) {

// format number 1000000 to 1,234,567

return n.replace(/\D/g, "").replace(/\B(?=(\d{3})+(?!\d))/g, ",");

}

function FormatValor(n) {

return n.replace("$", "").replace(/,/g, "");

}

PROBLEMA MODAL

SCROLL DE MODAL SE BLOQUEA DESPUÉS DE MENSAJE

max-height: 100%; overflow-y: auto;

EVITAR QUE UN MODAL SE CIERRE AL HACER CLIC FUERA DE ÉL

$('#modal-Presupuesto').modal({ backdrop: 'static', keyboard: false });

$('#modal-Presupuesto').modal('show');

OBTENER VALOR DE SELECT (COMBO)

GET VALUE (Index) from SELECT

<select onchange="showOptions(this)"> ...

this function will do the work

function showOptions(s) { console.log(s[s.selectedIndex].value); // get value console.log(s[s.selectedIndex].id); // get id }

CAPÍTULO XML

<https://www.red-gate.com/simple-talk/sql/database-administration/manipulating-xml-data-in-sql-server/>

-- create XML structure using FOR XML AUTO

SELECT BusinessEntityID,

PersonType,

Title,

FirstName,

MiddleName,

LastName,

Suffix

FROM Person.Person

WHERE BusinessEntityID = 10001

FOR XML AUTO

GO



By default, the AUTO argument organizes every non-XML field into a **node attribute**. To specify that the values be created as **node elements**, not attributes, we can additionally specify the ELEMENTS argument:

-- create XML structure using FOR XML AUTO

SELECT BusinessEntityID,

PersonType,

Title,

FirstName,

MiddleName,

LastName,

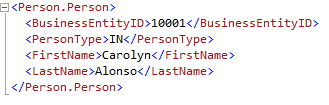
Suffix

FROM Person.Person

WHERE BusinessEntityID = 10001

FOR XML AUTO, ELEMENTS

GO



We notice that the root node name is the schema and table name (**Person.Person**). We would like to change this to ‘**Person**‘. To designate a custom root element, we’ll use the [PATH](http://msdn.microsoft.com/en-us/library/ms189885.aspx) argument instead of AUTO:

-- create XML using FOR XML PATH

SELECT BusinessEntityID,

PersonType,

Title,

FirstName,

MiddleName,

LastName,

Suffix

FROM Person.Person

WHERE BusinessEntityID = 10001

FOR XML PATH('Person')

GO

## **Combining Node Attributes and Elements**

**FUNCIONÓ***:*

DT:

SP

USE [db\_NovoTransferenciaValor]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [Novo].[NovoTVSp\_InsertarReceptores]

@IN\_Xml XML

AS

DECLARE @hDoc INT

SET NOCOUNT ON;

--Create an internal representation of the XML document

EXEC sys.sp\_xml\_preparedocument @hDoc OUTPUT ,@IN\_Xml;

INSERT INTO [Novo].[tbReceptores]([IdentificacionReceptor],[Tipo\_identificacion],[NombreReceptor],[CodigoMunicipio],[Municipio])

SELECT [IdentificacionReceptor], [Tipo\_identificacion],[NombreReceptor],[CodigoMunicipio],[Municipio]

FROM OPENXML(@hDoc, '/DTCargaReceptores/ListaCargaReceptores/Receptor', 2) --OPENXML (<handle>, <XPath rowpattern>, [, <mapping flags>]) [WITH (<Schema Declaration> | <Table Name>)]

WITH

(

IdentificacionReceptor VARCHAR(20),

Tipo\_identificacion VARCHAR(10),

NombreReceptor VARCHAR(100),

CodigoMunicipio VARCHAR(10),

Municipio VARCHAR(20)

);

EXEC sp\_xml\_removedocument @hDoc

SELECT \* FROM [Novo].[tbReceptores]

**TEORÍA OPENXML**

**OPENXML**

Once we have the XML document loaded in memory using sp\_xml\_preparedocument, we can then pass the document handle to OPENXML and use the XML document as traditional relational tabular rowset. The syntax of OPENXML looks like:

OPENXML (<handle>, <XPath rowpattern>, [, <mapping flags>]) [WITH (<Schema Declaration> | <Table Name>)]

The various clauses with OPENXML statement are explained here.

* <handle> parameter is the document handle of the internal representation of an XML document created by calling sp\_xml\_preparedocument.
* <XPath rowpattern> is the XPath pattern used to identify the nodes to be processed as rows.
* <mapping flags>, an optional input parameter, is the bit-mask flag value to indicate if attributes or sub-elements be used to do mapping between XML data and the relational rowset columns, and how the spill-over column should be filled. The values that this parameter can have include:

|  |  |
| --- | --- |
| * 0 | * Use the attribute-centric mapping. This is the default. |
| * 1 | * Use the attribute-centric mapping. This can be combined with XML\_ELEMENTS (2) to indicate use of element-centric mapping for all columns not yet dealt with. If XPath finds both an attribute and an element with the same name, the attribute wins. This can also be combined with option 8 or both (2 and 8). |
| * 2 | * Use the element-centric mapping. This can be combined with XML\_ATTRIBUTES (1) to indicate use of attribute-centric mapping for all columns not yet dealt with. If XPath finds both an element and an attribute with the same name, the element wins. This can also be combined with option 8 or both (1 and 8). |
| * 8 | * This flag indicates that the consumed data should not be copied to the overflow property @mp:xmltext. More about this and @mp:xmltext meta property discussed later in the article. This can be combined with XML\_ATTRIBUTES (1) or XML\_ELEMENTS (2). |

* Finally, the optional WITH clause provides the format for resulting rowset and additional mapping information, using either <Schema Declaration> or by specifying an existing <Table Name>.

The <Schema Declaration> is of the form:

<Column Name> <data type> [<Column XPath> | <Meta Property>]

[, <Column Name> <data type> [<Column XPath> | <Meta Property>] …]

* <Column Name> is the name of attribute or element being retrieved and the column name in the rowset. This will also serve as the name that we'll refer to when building the SELECT list, performing JOINs, etc.
* <data type> is any valid SQL Server data type of the column in the rowset. As XML can have data types that are not equivalents of those in SQL Server, an automatic coercion takes place in these cases if necessary. If the column is of type timestamp, the data value from XML document is disregarded when selecting from an OPENXML rowset, and the autofill values are returned.
* <Column XPath> is an optional relative XPath pattern that describes how the XML nodes should be mapped to columns. If this clause is not included, the default mapping (attribute-centric or element-centric as specified by <mapping flags>) is used.
* <Meta Property> is one of special Meta properties that OPENXML provides. Meta properties describe various aspects of whatever part of XML DOM you're interested in. Here is a list of available Meta properties:

|  |  |
| --- | --- |
| @mp:id | Provides system-generated, document-wide identifier of the DOM node. An XML ID of 0 indicates that the element is a root element and its @mp:parentid is NULL. |
| @mp:parentid | Same as @mp:id, only for the parent. |
| @mp:localname | Provides the non-fully qualified name of the node. It is used with prefix and namespace URI to name element or attribute nodes. |
| @mp:parentlocalname | Same as @mp:localname, only for the parent. |
| @mp:namespaceuri | Provides the namespace URI of the current element. If the value of this attribute is NULL, no namespace is present. |
| @mp:parentnamespaceuri | Same as @mp:namespaceuri, only for the parent. |
| @mp:prefix | The namespace prefix of the current element name. |
| @mp:prev | Stores the @mp:id of the previous sibling relative to a node. Using this, we can tell something about the ordering of the elements at the current level of the hierarchy. |
| @mp:xmltext | This Meta property is used for processing purposes, and contains the actual XML text for the current element as used in the overflow handling of OPENXML. |

JSON

<https://pro.arcgis.com/es/pro-app/tool-reference/conversion/an-overview-of-the-json-toolset.htm>

<https://www.codementor.io/andrewbuchan/how-to-parse-json-into-a-c-object-4ui1o0bx8>

<https://www.campusmvp.es/recursos/post/Como-usar-JSON-en-NET-facilmente.aspx>

CAPÍTULO RPA

**Robotic Process Automation (RPA) is a business automation technology that helps in automating the humdrum repetitive tasks (data entry, banking) without human intervention.**

* **It evolved from the three key predecessors: Screen scraping, workflow automation, and artificial intelligence.**

**Screen scraping is the process of collecting screen display data from one application and translating it so that another application can display it. This is normally done to capture data from a legacy application in order to display it using a more modern user interface.**

**Screen scraping usually refers to a legitimate technique used to translate screen data from one application to another. It is sometimes confused with content scraping, which is the use of manual or automatic means to harvest content from a website without the approval of the website owner.**

**Consider a browser automation tool like** [**Watin**](http://watin.sourceforge.net/)**. If you require complex interactions with the target website it might be easier to write your scraper from the point of view of the browser itself, rather than mucking with the HTTP requests and responses by hand. [Con Watin no permitió abrir una ventana del explorador]**

**Use** [**Html Agility Pack**](http://www.codeplex.com/htmlagilitypack)**. It handles poorly and malformed HTML. It lets you query with XPath, making it very easy to find the data you're looking for. DON'T write a parser by hand and DON'T use regular expressions, it's just too clumsy.**

**Para Screen Scraping se utiliza Selenium (Se instala desde el NuGet) y se debe instalar también Google ChromeDriver desde NuGet**

**[TestFixture]**

**public class MessagesController : Controller**

**{**

**// GET: Messages**

**public ActionResult Index()**

**{**

**return View();**

**}**

**[Test]**

**[STAThread]**

**public ActionResult LLamarGoogle()**

**{**

**IWebDriver driver = new ChromeDriver();**

**driver.Url = "http://www.google.com";**

**driver.Manage().Window.Maximize();**

**IWebElement emailTextField = driver.FindElement(By.Name("q"));**

**emailTextField.SendKeys("Selenium");**

**TimeSpan interval = new TimeSpan(50000000000); //Tiempo de espera**

**driver.Manage().Timeouts().ImplicitWait = interval;**

**IWebElement BotonBuscar = driver.FindElement(By.Name("btnK"));**

**BotonBuscar.Click();**

**return View("../Home/Index");**

**}**

**}**

**SELENIUM CURSE:**

[**https://www.udemy.com/cart/subscribe/course/776634/**](https://www.udemy.com/cart/subscribe/course/776634/)

[**https://automationstepbystep.com/selenium-introduction/**](https://automationstepbystep.com/selenium-introduction/)

# [**https://www.youtube.com/watch?v=udkiQSpXKw4**](https://www.youtube.com/watch?v=udkiQSpXKw4)

# **HtmlUnit es un Headless browser que puede emular el comportamiento de otros browsers**

# **KATALON AUTOMATION RECORDER: Es un plugin de Chrome que permite grabar la automatización y exportarla a otro lenguaje (C#).**

Ejercicios de Automatización:

<https://www.uio.no/studier/emner/matnat/ifi/IN3240/v19/seminarer/setup.pdf>

<http://automationpractice.com/index.php>

# [XLL Plus](https://www.planatechsolutions.com/xllplus/) is a toolkit to aid C/C++ programmers in the construction of Excel add-in libraries. According to their website it’s used by many of the top investment banks. It’s extremely expensive ($1,345 at the time of writing), so it’s not for casual users.

# Ejercicios RPA:

# [**https://www.linkedin.com/pulse/simple-exercises-get-familiar-automation-anywhere-rpa-steve-ng**](https://www.linkedin.com/pulse/simple-exercises-get-familiar-automation-anywhere-rpa-steve-ng)

# **Desde NUGET, instalar**

Selenium Web Driver

Selenium Support

Selenium WebDriver.ChromeDriver

# **Cómo utilizar la biblioteca de objetos de Microsoft Outlook para recuperar un mensaje de la Bandeja de entrada mediante Visual C#**

<https://support.microsoft.com/es-cl/help/310258/how-to-use-the-microsoft-outlook-object-library-to-retrieve-a-message>

**LÍDERES:**

**UiPath:**

**Automation Anywhere (AA):**

**Workfusion:**

**Kryon Systems:**

**Kofax:**

**OTROS:**

[**https://www.g1ant.com/**](https://www.g1ant.com/)

**SikuliX:** [RaiMan's SikuliX](http://sikulix.com/)

**TagUI:** [kelaberetiv/TagUI](https://github.com/kelaberetiv/TagUI)

A quick review about this tool: [The Democratization of Robotic Process Automation — A First Look at TagUI](https://medium.com/@cfb_bots/the-democratization-of-robotic-process-automation-a-first-look-at-tagui-f3c7b1e7a737)

**Taskt**: [saucepleez/taskt](https://github.com/saucepleez/taskt)

**Roro**: [arviedelgado/Roro](https://github.com/arviedelgado/Roro)

**Robot Framework : <http://robotframework.org/>**

I have seen people use Test Automation tools.

Selenium: [Browser Automation](https://www.seleniumhq.org/)

You can also just use common language like Python. This is a good place to start:

**[Automate the Boring Stuff with Python](https://automatetheboringstuff.com/)**

There are mainly **two kinds of RPA automation:**

##### [**https://ontrackhq.com/**](https://ontrackhq.com/)

##### **Teoría de RPA:**

##### [**https://www.guru99.com/robotic-process-automation-tutorial.html**](https://www.guru99.com/robotic-process-automation-tutorial.html)

##### **OPEN RPA: EN C#**

Lo descargué como repositorio GitHub:

<https://github.com/open-rpa/openrpa>

Usuario: [carlossan@gmail.com](mailto:carlossan@gmail.com)

Password: cl..0

Documentación:

<https://github.com/open-rpa/openrpa/wiki>

Video Primera aplicación:

<https://www.youtube.com/watch?v=SqLHlMpQhZA&feature=youtu.be>

##### 

## [AutoIt](https://www.autoitscript.com/site/autoit/)

##### **AutoIt v3 is a freeware BASIC-like scripting language designed for automating the Windows GUI and general scripting.**

# 

# **machine learning**

##### 

## **1. TensorFlow**

Initially released in 2015, [TensorFlow](https://www.tensorflow.org/) is an open source machine learning framework that is easy to use and deploy across a variety of platforms. It is one of the most well-maintained and extensively used frameworks for machine learning.

Created by Google for supporting its research and production objectives, TensorFlow is now widely used by several companies, including Dropbox, eBay, Intel, Twitter, and Uber.

TensorFlow is available in Python, C++, Haskell, Java, Go, Rust, and most recently, JavaScript. You can also find third-party packages for other programming languages.

The framework allows you to develop neural networks (and even other computational models) using flowgraphs.

## **4. Microsoft Cognitive Toolkit**

Initially released in 2016, the [Microsoft Cognitive Toolkit](https://www.microsoft.com/en-us/cognitive-toolkit/) (previously referred to as CNTK), is an AI solution that can empower you to take your machine learning projects to the next level.

Microsoft says that the open source framework is capable of "training deep learning algorithms to function like the human brain."

Some of the vital features of the Microsoft Cognitive Toolkit include highly optimized components capable of handling data from Python, C++, or BrainScript, ability to provide efficient resource usage, ease of integration with Microsoft Azure, and interoperation with NumPy.

## **8. Accord.NET**

Initially released in 2010, [Accord.NET](http://accord-framework.net/) is a machine learning framework entirely written in C#.

The open source framework is suitable for production-grade scientific computing. With its extensive range of libraries, you can build various applications in artificial neural networks, statistical data processing, image processing, and many others.

<https://towardsdatascience.com>

**Robot Framework** is a generic open source automation framework for acceptance testing, acceptance test driven development (ATDD), and robotic process automation (RPA).

<https://robotframework.org/>

##### **FlexRule RPA SAMPLE (Sample RPA for Google Search)**

<http://resource.flexrule.com/knowledge-base/modeling-procedural-rules/#how-to-model-a-procedural-rule>

##### **Attended automation**

These are the **assistants that run on your computer and help you complete a part of the tasks that you are performing**. For eg., if you usually search and copy paste data from one application to multiple applications, you can invoke an attended RPA to take over just these set of activities.

##### **Unattended Automation**

In this type of automation, workflows are self-triggered and run on servers. They run to **pre-determined schedules or real time. and are available 24 x 7**. In the case of unattended automation, all you would see of an automated task would be the outputs. For eg. For invoice processing, all you would see is a report indicating the invoices that could not be processed automatically.

# **Can you use code on Github for commercial purposes?**

Depends on what license it has. MIT, Apache, BSD, yes.

FREE OPEN SOURCE RPA IN C#

C:\Back Up Carlos Sanchez\PROGRAMACIÓN\GITHUB\sharpRPA

Investigar: Microsoft Robotics Developer Studio 4

**RORO: Licencia BSD 2-**

RORO necesita que se instale Fody en la consola de Nuget:

**Install-Package Fody -Version 4.2.1**

TASKT: <http://www.taskt.net/>

Investigar: Autolt

DESARROLLAR UN RPA CON C#: RPA Suite for VISUAL STUDIO. RIGEL Es el único motor RPA desarrollado en C#.

Tutorial (NotePad):

<https://www.youtube.com/watch?v=-tDwGBoqopw>

**RPA Suite is a Cognitive Artificial Intelligence**

1. Instalar desde el marketplace: RPA SUITE for visual studio
2. Nuevo Proyecto: Windows Forms App, eliminar Form1.cs y abrir Program.cs
3. Agregar la referencia: Cartes Library (en grupo COM)
4. En Herramientas -> RPA suite for VS crear un proyecto; Crear una variable ($EditorNotepad)
5. Clic en Captur ->Win32 -> Abrir Notepad -> Capture -> Barra espaciadora sobre Notepad. ->Save Project -> Run RPA developer.
6. Escribir el siguiente código en program.cs

using Cartes;

namespace RPA1

{

static class Program

{

[STAThread]

static void Main()

{

CartesObj Cartes = new CartesObj();

RPAWin32Component notepad = null;

Cartes.open(Path.Combine(Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location), @"Cartes\RPA1.cartes.rpa"));

notepad = (RPAWin32Component)Cartes.component("$EditorNotepad");

if (notepad.componentexist(0) == 0) {

Cartes.run("notepad.exe");

notepad.waitforcomponent(15);

Cartes.reset(notepad.api());

}

notepad.Value ="Hola Mundo";

}

}

}

EXCEL

<http://www.forumrpa.com/t/class-wrapper-for-cartes-in-c/51/2>

RPA DEVELOPER MANUAL (Pdf se instala con Cartes)

1. Click en New para Ingresar una nueva variable ($variable1)
2. Capture->Capture-> ingresar a aplicación -> recovery - O.K. -> Guardar Proyecto
3. En Testing ingresar $variable1.value(“Hola Mundo”);

for ($i = 0; $i < 10; $i = $i + 1){

if ($i > 4) break;

}

$i = 0;

do{

$i = $i + 1;

}while ($i < 10)

$i = 0;

while ($i < 10){

if ($i > 4) break;

else $i = $i + 1;

}

Seguidamente 3 ejemplos de la sentencia if.

if ($a > $b) $c = 1;

else $c = 0;

if ($a > $b) $c = 1;

if ($a > $b){

$c = 1;

}

else{

$c = 0;

}

throw new Exception(“Error Message”, $ErrorCode);

function increaseTop($top, &$counter, $step){

if ($counter < $top)

$counter = $counter + $step;

else $counter = $top;

}

function notepadWrite($text){

Global $Notepad;

merge($filename); Permite incluir un proyecto dentro de otro.

merge(ExtractFilePath(Projectfile) & “notepad.rpa”);

$notepad.value(“Hello world”);

open($filename);

close

class MyClass extends Object{

private $value;

\_\_construct($parameter1){

$parent.\_\_construct;

}

\_\_destruct{

$parent.\_\_destruct;

}

public override function assign($object){

$this.value = $Object.value;

}

## Microsoft Robotics Developer Studio 4: Microsoft Robotics Developer Studio is a Windows-based environment for robot control and simulation

Linx <https://linx.software/>

<https://ui.vision/> : Kantu

# TagUI: open source. PHP

WorkFusion’s [RPA Express](https://www.workfusion.com/rpaexpress) : Lo descargué.

<https://www.workfusion.com/>

SELENIUM (TEST AUTOMATION)

Selenium is an open source portable framework for automating web applications testing across different browsers (Chrome, Firefox, Safari) and not just the traditional browsers.

LIST OF RPA TOOLS:

<https://www.guru99.com/robotics-process-automation-tools.html>

<https://bpmtips.com/rpa-tools-list-2018/#RPA_free>

CAPÍTULO XAMARIN

<https://docs.microsoft.com/en-us/xamarin/android/get-started/hello-android/hello-android-quickstart?pivots=windows>

LISTA HTML CON VARIAS COLUMNAS:

1. ul {
2. column-count: 2;
3. }

CSS:

It's quite easy to change the way a Bootstrap modal appears. The basic premise is to replace the .modal.fade and .modal.fade.in classes.

## Modal Options

The data-keyboard attribute specifies whether the modal can be closed with the escape key.

## Modal Options

The data-backdrop attribute specifies whether the modal should have a dark overlay (the background color of the current page) or not. Sombra en la página actual.

<https://www.w3schools.com/bootstrap/tryit.asp?filename=trybs_ref_js_modal_backdrop_data&stacked=h>

## **El HTML del modal**

Ahora que tenemos cargadas las librerías, tenemos que crear el código HTML que hará que funcionen las funciones que nos ofrecen ambas.

En primer lugar, vamos a crear el modal en sí. Para que sea más fácil de explicar, dejamos aquí el código y, seguidamente, explicaremos línea a línea:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | <div class="modal fade" id="miModal" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">  <div class="modal-dialog" role="document">  <div class="modal-content">  <div class="modal-header">  <button type="button" class="close" data-dismiss="modal" aria-label="Close">  <span aria-hidden="true">&times;</span>  </button>  <h4 class="modal-title" id="myModalLabel">Esto es un modal</h4>  </div>  <div class="modal-body">  Texto del modal  </div>  </div>  </div>  </div> |

Línea 1: creamos el contenedor principal con un ID único (en nuestro caso “miModal”). Es importante quedarnos con este ID ya que será el que usemos a la hora de llamar a la función que muestra este contenedor. Es este el contenedor, que comentábamos antes, que tiene el fondo oscuro con cierta opacidad que hace que reste protagonismo al resto de la página.

Linea 2: es el contenedor donde está definido el tamaño del contenedor (por defecto 600px).

Línea 3: se trata de un contenedor donde están asociados estilos tales como fondo del modal, bordes, sombreados, etc.

Línea 4: en esta línea, Bootstrap mete el DIV que contiene el botón de cerrar el modal así como el título del modal en un encabezado (H4).

Línea 10: en el div con clase “modal-body” es donde insertaremos el contenido que queremos que haya en el modal.

INVESTIGAR (CONSULTAR):

Stored Procedure:

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CREATE PROCEDURE [dbo].[spUpdateConfirmation]

@ReportID int

AS --BEGIN

DECLARE @Confirmation varchar(30);

SET @Confirmation = replace(replace(replace(convert(varchar(16),CURRENT\_TIMESTAMP,120),'-',''),' ',''),':','')+convert(varchar(24),@ReportID) --PRINT @Confirmation

SET NOCOUNT ON; -- prevent rowcount messages

UPDATE Report

SET Confirmation = @Confirmation

WHERE ReportID = @ReportID; --RETURN @Confirmation

SET NOCOUNT OFF; -- re-enable for the following select

SELECT @Confirmation; -- this is the value you get --END

GO

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

BEGIN TRY  
 --Guardas en la tabla maestro.  
 --Guardas en la tabla detalle.  
 --Guardas en la tabla detalle.  
 --Guardas en la tabla detalle.  
 --Guardas en la tabla detalle.  
 END TRY  
 BEGIN CATCH  
 --Plop...!! haces algo por que ocurrió un error.  
 END CATCH

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Depuración en JS

Depuración en .NET

$.each(Dato.Datos, function (i, v) {

$('#ComboCiudadModal').append('<option value="' + v.IdCiudad + '">' + v.Nombre + '</option>');

});

ENVIAR CORREOS AUTOMÁTICAMENTE

( System.Net.Mail):

Servidor (SMTP) de Correo Saliente: smtp.gmail.com

Usar Autenticación: Sí

Usar Conexión Segura: Sí (puede TLS o SSL dependiendo de su cliente de email)

Nombre de usuario: cuenta GMail (email@gmail.com)

Contraseña: contraseña GMail

Puerto: 465 o 587

<https://programacion.net/articulo/enviar_email_usando_gmail_en_c_1073>

<https://social.msdn.microsoft.com/Forums/es-ES/fa40c039-2492-48f0-8e80-a8e583a0cf02/enviar-correo-electronico-gmail?forum=vcses>

Para que se pueda enviar un e-mail desde gmail se debe establecer la opción de:[permitir que apps menos seguras accedan a la cuenta](https://support.google.com/a/answer/6260879)

en SECURITY en: <https://myaccount.google.com/>

LÓGICA DEL COCO:

TRANSVERSAL: Librerías disponibles para toda la aplicación:

DT: Todas las DataTable (modelos)

Soporte: GestorLogs,

ACCESO A DATOS:

DM: Data Access Objects de cada módulo

Resources: Stored Procedures (SP)

AD: Adaptadoras

Al ingresar usuario llega al LoginController - Método: JsonResult ValidarCredenciales (String usuario)

->SeguridadRVSST.DecryptStringAES: Desencripta el usuario

->JsonConvert.DeserializeObject : Convierte el Json a DT y se almacena en un UsuarioSesion que es de tipo DTUsuario. Este DT está definido como clase pública en General en el DT en Transversal.

->Llama al método ValidarCredenciales del BMUsuario y guarda el resultado en una variable: ResultadoValidarCredenciales de tipo: DTResultadoOperacionModel<DTUsuario>

->Si ResultadoValidarCredenciales.Respuesta es true entonces Asigna variable de sesión e instancia BMUsuario().InforPerfilUsuario para saber el perfil y guarda la respuesta en ResultadoPerfilUsuario

y Consulta el Listado de menú y lo guarda en: Session[DTNombreSesion.DTMenu]

-

En BM/Usuario/BMUsuario: Método validarCredenciales

->Declara una respuesta de tipo DTResultadoOperacionModel<DTUsuario>

->Declara UsuarioRespuesta con DTUsuario

->UsuarioRespuesta recibe el resultado de: DMUsuario().ValidarCredenciales(Usuario)

->Si recibe un Usuario entonces retorna una respuesta true de lo contrario retorna False

AccesoADatos/Menu/Usuario/DMUsuario método ValidarCredenciales

->Crea una Lista de parámetros

->Le adiciona el parámetro @Usuario, con el nombre de usuario digitado

->Ejecuta el procedimiento Almacenado ConsultarUsuario y lo almacena en un DataTableResultado

->Utiliza ConvertData.ConvertirDtoToList para convertir el DataTable en un List: DTUsuario<DTUsuario>

->Retorna una Lista de < DTUsuario>.FirstOrDefault()

BMUsuario().InforPerfilUsuario

-> Declarar un ListaResultado como una Lista<DTPerfilUsuario>

->Insertar un elemento a la lista de parametros @IdPerfilUsuario y el valor: Usuario.IdPerfilUsuario

-> Ejecutar el SP RTVSSSp\_PerfilUsuario y guardar el resultado en DataTableResultado

-> Convetir el DataTable en una lista (ListaResultado) con: ConvertData.ConvertirDtoToList<DTPerfilUsuario>

->Retornar ListaResultado.FirstOrDefault().

Presentar datos de usuarios en una tabla con la opción de editar rol para cada usuario.

Presentar una vista con un Combo con el listado de roles para asignar un rol.

Mostrar labels, Cuadros de texto y botones:

@using Francol.RVSST.DT.Mensajes

@{

ViewBag.Title = "MenuUsuario";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>MenuUsuario</h2>

<div class="container-page-body">

<div class="row">

<div class="col-md-2 col-sm-2">

<h3>Buscar productos</h3>

</div>

</div>

<hr />

<div class="row">

<div class="form-group col-md-12">

<div class="col-md-5">

<input id="descripcion" class="form-control" placeholder="Descripción" style="width: 100%;" />

<input type="hidden" id="estado\_pedido" value="" />

<input type="hidden" id="pedidoId" value="" />

<input type="hidden" id="estadoId" value="" />

<input type="hidden" id="CantidadPedido" value="" />

<input type="hidden" id="TotalPedido" value="" />

</div>

<div class="col-md-3">

<input id="skureferencia" class="form-control justNumbers" maxlength="11" placeholder="SKU o Referencia" style="width: 100%;" />

</div>

<div class="col-md-1">

<input value="Buscar" class="btn btn-success" type="button" id="Buscar" />

</div>

<div class="col-md-2 col-md-offset-1">

<a class="btn btn-danger" id="MiPedido"><span class="glyphicon glyphicon-shopping-cart" style="font-size: 12px;"></span> Mi pedido</a>

</div>

</div>

</div>

<div class="row">

<div class="form-group col-md-12">

<script id="template" type="text/x-kendo-template">

<label class="search-label" for="searchBox">Filtrar:</label>

<input type="search" id="searchBox" class="k-textbox" style="width: 250px" />

@\*<input type="button" id="btnSearch" class="k-button k-button-transparent" value="Buscar" />\*@

<input type="button" id="btnReset" class="k-button k-button-transparent" value="Reestablecer" />

</script>

<div id="gridproductos"></div>

</div>

</div>

</div>

<!--Modals-->

<div class="modal fade in" id="modal-detalle-pedido" data-keyboard="false" data-backdrop="static">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h4 class="modal-title"><label id="titulomensaje" class="modal-titel">Agregar productos</label></h4>

</div>

<div class="modal-body">

<div class="form-group">

<label><b>Descripción: </b></label>

<label id="DescripcionModal"></label>

</div>

<div class="form-group">

<label><b>Referencia: </b></label>

<label id="ReferenciaModal"></label>

</div>

<div class="form-group">

<label><b>SKU: </b></label>

<label id="SKUModal"></label>

</div>

<div class="form-group">

<label><b>Color: </b></label>

<label id="ColorModal"></label>

</div>

<div class="form-group">

<label><b>Talla: </b></label>

<label id="TallaModal"></label>

</div>

<div class="form-group">

<label><b>Precio Unidad: </b></label>

<span>$</span>

<label id="PrecioModal" readonly class="moneda" aria-label="Amount (to the nearest dollar)"> </label>

<input type="hidden" id="preciomodalhidden" />

</div>

<div class="form-group form-inline CantidadDiv">

<label><b>Cantidad: </b></label>

<input type="text" class="form-control justNumbers" style="width:200px !important; " maxlength="2" id="CantidadModal" value="1" />

</div>

<div class="form-group">

<label><b>Precio Total: </b></label>

<span>$</span>

<label id="PrecioTotalModal" readonly class="moneda" aria-label="Amount (to the nearest dollar)"> </label>

<input type="hidden" id="preciototalhidden" />

</div>

</div>

<div class="modal-footer">

<table class="botones-modal">

<tr>

<td>

<input id="btnAgregarProducto" type="button" value="Agregar" class="btn btn-success" />

<input id="btnCancelarAgregarProducto" type="button" value="Cancelar" class="btn btn-danger" />

</td>

</tr>

</table>

</div>

</div>

</div>

</div>

<div class="modal fade in" id="modal-redirigir-mipedido" data-keyboard="false" data-backdrop="static">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h4 class="modal-title"><label id="titulomensaje" class="modal-titel">Mensaje</label></h4>

</div>

<div class="modal-body">

<div class="mensajes" style="display: block;">

<div>

<h3><span class="icon-confirmacion"></span><span id="">@DTMensaje.ObtenerMensaje(DTCodigoMensajes.MENSAJE004)</span></h3>

</div>

</div>

</div>

<div class="modal-footer">

<table class="botones-modal">

<tr>

<td>

<input id="btnRedirigirMiPedido" type="button" value="Si" class="btn btn-success" />

<input id="btnCancelarRedirigirMiPedido" type="button" value="No" class="btn btn-danger" />

</td>

</tr>

</table>

</div>

</div>

</div>

</div>

STORED PROCEDURE: RTVSSSp\_ConsultarUsuario

USE [db\_RTVSS]  
GO  
/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ConsultarUsuario] Script Date: 8/01/2019 11:23:05 a. m. \*\*\*\*\*\*/  
SET ANSI\_NULLS ON  
GO  
SET QUOTED\_IDENTIFIER ON  
GO  
-- =============================================  
-- Author: <Jose David Sanchez Lopez>  
-- Create date: <03/01/2018>  
-- Description: <Consulta la informacion del usuario>  
-- =============================================  
CREATE PROCEDURE [rtvss].[RTVSSSp\_ConsultarUsuario]  
 @Usuario VARCHAR(30)  
AS  
BEGIN  
 SELECT [IdUsuario]  
 ,[NombresApellidos]  
 ,[Usuario]  
 ,[IdTipoIdentificacion]  
 ,[NumeroIdentificacion]  
 ,[Correo]  
 ,[IdCiudad]  
 ,[Zona]  
 ,[Distrito]  
 ,[GerenteRepresentante]  
 ,[LineaProducto]  
 ,[Fuerza]  
 ,[CodidoSap]  
 ,[Division]  
 ,[IdPerfilUsuario]  
 ,[Estado]  
 ,[IdUsuarioRegistro]  
 ,[FechaRegistro]  
 ,[IdUsuarioActualizacion]  
 ,[FechaActualizacion]  
 FROM [db\_RTVSS].[rtvss].[tbUsuarios]  
 WHERE Usuario = @Usuario AND Estado = 1  
END

STORED PROCEDURE: RTVSSSp\_PerfilUsuario

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_PerfilUsuario] Script Date: 8/01/2019 11:25:18 a. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Jose David Sanchez Lopez>

-- Create date: <03/01/2018>

-- Description: <Consulta la informacion del perfil>

-- =============================================

CREATE PROCEDURE [rtvss].[RTVSSSp\_PerfilUsuario]

@IdPerfilUsuario INT

AS

BEGIN

SELECT [IdPerfilUsuario]

,[Nombre]

,[Descripcion]

,[Estado]

,[FechaRegistro]

,[UsuarioCreacion]

,[FechaActualizacion]

,[UsuarioActualizacion]

FROM [db\_RTVSS].[rtvss].[tbPerfilUsuario]

WHERE [IdPerfilUsuario] = @IdPerfilUsuario AND Estado = 1

END

STORED PROCEDURE:RTVSSSp\_ConsultarMenu

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ConsultarMenu] Script Date: 8/01/2019 11:38:37 a. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Jose David Sanchez Lopez>

-- Create date: <31/12/2018>

-- Description: <Consulta los menu de un Perfil>

-- =============================================

CREATE PROCEDURE [rtvss].[RTVSSSp\_ConsultarMenu]

@IdPerfilUsuario INT

AS

BEGIN

SELECT

Mo.Nombre,

Mo.Ruta,

Mo.Visible,

Mo.IdModuloPadre

FROM rtvss.tbModulos Mo

INNER JOIN [rtvss].[tbPerfilAcceso] PA ON Mo.IdModulo = PA.IdModulo

WHERE Mo.Estado = 1 AND PA.Estado = 1 AND PA.IdPerfilUsuario = @IdPerfilUsuario

END

GO

STORED PROCEDURE PARA GUARDAR REGISTRO EDITADO:

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_EditarUnUsuario] Script Date: 11/01/2019 11:23:05 a. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Carlos Alberto Sánchez>

-- Create date: <11/01/2019>

-- Description: <Guarda la informacion Editada de un usuario>

-- =============================================

CREATE PROCEDURE [rtvss].[RTVSSSp\_EditarUnUsuario]

@IdUsuario smallint,

@NombresApellidos varchar(100),

@Usuario varchar(30),

@Correo varchar(100),

@IdUsuarioActualizacion smallint,

@FechaActualizacion datetime

AS

BEGIN

UPDATE [rtvss].[tbUsuarios]

SET [NombresApellidos]=@NombresApellidos, [Usuario]=@Usuario,[Correo]=@Correo

WHERE [IdUsuario] = @IdUsuario AND Estado = 1

END

STORED PROCEDURE CONSULTARUSUARIO (ORIGINAL)

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ConsultarUsuario] Script Date: 17/01/2019 10:13:28 a. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Jose David Sanchez Lopez>

-- Create date: <03/01/2018>

-- Description: <Consulta la informacion del usuario>

-- =============================================

ALTER PROCEDURE [rtvss].[RTVSSSp\_ConsultarUsuario]

@Usuario VARCHAR(30),

@Clave VARCHAR(30)

AS

BEGIN

SELECT U.[IdUsuario]

,[NombresApellidos]

,[Usuario]

,[IdTipoIdentificacion]

,[NumeroIdentificacion]

,[Correo]

,[IdCiudad]

,[Zona]

,[Distrito]

,[GerenteRepresentante]

,[LineaProducto]

,[Fuerza]

,[CodidoSap]

,[Division]

,[IdPerfilUsuario]

,U.[Estado]

,[IdUsuarioRegistro]

,[FechaRegistro]

,[IdUsuarioActualizacion]

,[FechaActualizacion]

FROM [db\_RTVSS].[rtvss].[tbUsuarios] AS U

INNER JOIN [rtvss].[tbClaves] AS C ON U.IdUsuario=C.IdUsuario

WHERE Usuario = @Usuario AND U.Estado = 1 AND C.Clave=@Clave

ORDER BY C.[FechaCreación] DESC

END

STORED PROCEDURE CONSULTARCIUDADES

USE [db\_RTVSS]

GO

/\*\*\*\*\*\* Object: StoredProcedure [rtvss].[RTVSSSp\_ConsultarUsuario] Script Date: 11/01/2019 11:23:05 a. m. \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

-- =============================================

-- Author: <Carlos Alberto Sánchez>

-- Create date: <11/01/2019>

-- Description: <Consulta las ciudades>

-- =============================================

CREATE PROCEDURE [rtvss].[RTVSSSp\_ConsultarCiudades]

AS

BEGIN

SELECT IdCiudad,

[Codigo],

[Nombre],

[Descripcion],

[IdDepartamento]

,[Estado]

,[IdUsuarioRegistro]

,[FechaRegistro]

,[IdUsuarioActualizacion]

,[FechaActualizacion]

FROM [db\_RTVSS].[rtvss].[tbCiudad]

WHERE Estado = 1

END

VISTA ORIGINAL (CON TABLA) CON JS

La vista Tiene asociado un archivo js que realiza la consulta de los datos a mostrar en la tabla.

La consulta la realiza mediante una petición AJAX: POST a una URL.

Para que se ejecute en el JS:

var objPedido = new Pedido();

$(document).ready(function () {

objPedido.TraerInformacionDePedido();

//Busqueda de Productos

$('#Buscar').click(function () {

if (objPedido.validarBusqueda())

{

objPedido.ConsultarProductos(objPedido.objProducto);

}

});

PARA AGREGAR UN BOTÓN EN EL JS:

{

field: "Opción",

template: "<input onClick='objPedido.DetalleProducto(this)' value='Agregar' type='button' class='btn btn-default' data-id='#=Id#' data-sku='#=Codigo#' data-skuvariante='#=SkuVariante#' data-color='#=Color#' data-descripcion='#=Descripcion#' data-referencia='#=Referencia#' data-referenciasap='#=ReferenciaSAP#' data-talla='#=Talla#'>"

ENVIAR DATO (IdUsuario) POR AJAX:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

this.DetalleUsuario = function (boton) {

var IdU = $(boton).data('id');

var Datos = {};

Datos.IdUsuario = IdU;

ObjHome.ObjUsuario.IdUsuario = IdU;

DT\_Usuario = ObjHome.ObjUsuario;

confirm(IdU); // Recibe el ID del Usuario

$.ajax({

url: '../AdministracionUsuarios/LeerUnUsuario',

type: 'GET',

data: Datos,

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

dataType: "json",

success: function (data) {

if (data.Resultado === true) {

confirm("Ingreso");

}

}

});

$('#modal-EditarUsuario').modal('show');

};

MOSTRAR MENSAJE EN JS:

confirm(“Este es un mensaje”);

if (confirm == true)

{..}else

{}

//console.log(data.Datos[0].Nombre);

data1-Usuario='#=Usuario#' data1-Id='#=IdUsuario#' data1-NombresApellidos='#=NombresApellidos#'

SEARCH BOX:

En la vista:

<div class="row">

<div class="form-group col-md-12">

<script id="template" type="text/x-kendo-template">

<label class="search-label" for="searchBox">Filtrar:</label>

<input type="search" id="searchBox" class="k-textbox" style="width: 250px" />

@\*<input type="button" id="btnSearch" class="k-button k-button-transparent" value="Buscar" />\*@

<input type="button" id="btnReset" class="k-button k-button-transparent" value="Reestablecer" />

</script>

<div id="gridproductos"></div>

</div>

</div>

En el JS:

//Busca en la grid los valores del textbox

$("#searchBox").keyup(function () {

var searchValue = $('#searchBox').val();

//Agrega el evento a la grid

$("#gridproductos").data("kendoGrid").dataSource.filter({

logic: "or",

filters: [

{

field: "Descripcion",

operator: "contains",

value: searchValue

},

{

field: "Talla",

operator: "contains",

value: searchValue

},

{

field: "Color",

operator: "contains",

value: searchValue

},

{

field: "Codigo",

operator: "contains",

value: searchValue

},

{

field: "Referencia",

operator: "contains",

value: searchValue

}

]

});

});

//Reestablece la grid con los datos buscados.

$("#btnReset").click(function () {

$("#gridproductos").data("kendoGrid").dataSource.filter({});

$("#searchBox").val('');

});

};

VISTA CON TABLA (EJEMPLO KENDO)

@{

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<!DOCTYPE html>

<html>

<head>

<title></title>

<link rel="stylesheet" href="styles/kendo.common.min.css" />

<link rel="stylesheet" href="styles/kendo.default.min.css" />

<link rel="stylesheet" href="styles/kendo.default.mobile.min.css" />

<script src="js/jquery.min.js"></script>

<script src="js/kendo.all.min.js"></script>

</head>

<body>

<div id="example">

<div id="grid"></div>

<script>

$(document).ready(function () {

$("#grid").kendoGrid({

dataSource: {

type: "odata",

transport: {

read: "https://demos.telerik.com/kendo-ui/service/Northwind.svc/Customers"

},

pageSize: 20

},

height: 550,

groupable: true,

sortable: true,

pageable: {

refresh: true,

pageSizes: true,

buttonCount: 5

},

columns: [{

template: "<div class='customer-photo'" +

"style='background-image: url(../content/web/Customers/#:data.CustomerID#.jpg);'></div>" +

"<div class='customer-name'>#: ContactName #</div>",

field: "ContactName",

title: "Contact Name",

width: 240

}, {

field: "ContactTitle",

title: "Contact Title"

}, {

field: "CompanyName",

title: "Company Name"

}, {

field: "Country",

width: 150

}]

});

});

</script>

</div>

<style type="text/css">

.customer-photo {

display: inline-block;

width: 32px;

height: 32px;

border-radius: 50%;

background-size: 32px 35px;

background-position: center center;

vertical-align: middle;

line-height: 32px;

box-shadow: inset 0 0 1px #999, inset 0 0 10px rgba(0,0,0,.2);

margin-left: 5px;

}

.customer-name {

display: inline-block;

vertical-align: middle;

line-height: 32px;

padding-left: 3px;

}

</style>

</body>

</html>

////////////////////////////////////////////////////

PUBLICAR EN AZURE:

1. En visual Studio hacer clic derecho en el proyecto ->Clic en Publicar
2. Seleccionar AZURE
3. En Portal Azure crear una nueva ranura de Implementación (Deployment Slot).
4. Se puede clonar una ranura de implementación existente o se puede crear desde cero.

<https://mvc3-mvc3slot.azurewebsites.net>

PÁGINA CSHTML

links:

<div>

@Html.ActionLink("Insertar", "Create", "Peliculas")<br />

@Html.ActionLink("Ver Películas", "Index", "Peliculas")<br/>

@Html.ActionLink("Editar", "Edit", "Peliculas",new {id="1"},null)

</div>

BOTÓN:

<p><a class="btn btn-default" href="https://go.microsoft.com/fwlink/?LinkId=301866">Learn more &raquo;</a></p>

# **ASP.NET MVC: DIFFERENCE BETWEEN HTML.ACTIONLINK VS URL.ACTION**

Standard

***There is a difference.*** Html.ActionLink ***generates an*** <a href=".."></a> ***tag whereas*** Url.Action ***returns only an url.***

For example:

@Html.ActionLink("link text", "someaction", "somecontroller", new { id = "123" }, null)

generates:

<a href="/somecontroller/someaction/123">link text</a>

and

Url.Action("someaction", "somecontroller", new { id = "123" })

generates:

/somecontroller/someaction/123

JAVASCRIPT:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<title></title>

<script type="text/javascript" src="Scripts/funciones.js"></script>

</head>

<body>

<script type="text/javascript">

alert(calc(5, 4));

</script>

</body>

</html>

LISTAS EN JAVASCRIPT:

var fechas = [];

while (dia\_actual.isSameOrBefore(hasta)) { fechas.push(dia\_actual.format('DD-MM-YYYY'));

dia\_actual.add(1, 'days');

}

Insertar un archivo nuevo llamado: funciones.js

function calc(a, b) {

return a + b;

}

PROCEDIMIENTOS ALMACENADOS (STORED PROCEDURE)

INSTRUCCIÓN SQL PARA CREAR UNA TABLA:

Create table tbl\_usuarios{

id int,

UserName varchar(50),

Clave varchar(50),

Edad int

}

PROCEDIMIENTO ALMACENADO PARA INGRESAR UN REGISTRO:

Clic derecho sobre la Base de Datos en la ventana: Explorador de Objetos de SQLServer del Visual Studio 2017.

-> Nueva Consulta e ingresar:

Create proc insertarRegistroUsuario

(

@id int,

@UserName varchar(50),

@Clave varchar(50),

@Edad int

)

as

begin

SET IDENTITY\_INSERT tbl\_usuarios ON;

insert into tbl\_usuarios (id,UserName,Clave,Edad)

values (@id,@UserName,@Clave,@Edad)

end

->Al ejecutar esa consulta se genera automáticamente un procedimiento almacenado (llamado: insertarRegistroUsuario) que se puede ver en -> Programación -> procedimientos Almacenados

PARA EJECUTAR en SQL:

exec insertarRegistroUsuario 2,’Diego’, ‘Clave1’,55

PARA MODIFICAR UN PROCEDIMIENTO ALMACENADO:

ALTER procedure [dbo].[InsertarRegistroUsuario](

@id int,

@UserName varchar(50),

@Clave varchar(50)

)

as

begin

SET IDENTITY\_INSERT Usuarios ON;

insert into Usuarios (id,NombreUsuario,Clave)

values (@id,@UserName,@Clave)

end

PARA EJECUTAR EL PA EN VISUAL STUDIO 2017:

string connectionString = @"Data Source=(localdb)\MSSQLLocalDB; Initial Catalog=APLICACION1Context-20181124220847; Integrated Security=True; MultipleActiveResultSets=True; AttachDbFilename=|DataDirectory|APLICACION1Context-20181124220847.mdf";

//El connectionString se puede extraer del WebConfig

using (SqlConnection sql = new SqlConnection(connectionString))

{

sql.Open();

using (SqlCommand cmd = new SqlCommand("*InsertarRegistroUsuario*", sql)) {

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.Add(new SqlParameter("@id", usuario.ID));

cmd.Parameters.Add(new SqlParameter("@userName", usuario.NombreUsuario));

cmd.Parameters.Add(new SqlParameter("@clave", usuario.clave));

int filas=cmd.ExecuteNonQuery();

}

}

db.SaveChanges();

return RedirectToAction("Index");

}

ApplicationContext

he purpose of the ApplicationContext is to serve as an application start and termination notification link between the main Form in your application and the UI thread. The UI thread is the main thread that your application's user interface is running in, and it is the work horse that processes the application’s message loop. The message loop receives event messages from the operating system like ‘Mouse Right Click’ or ‘Space Bar Button Down’, and sends these messages to the form that needs to handle it.

En el controlador para enviar un listado de Usuarios a la vista:

private APLICACION1Context db = new APLICACION1Context();

public ActionResult Index()

{

return View(db.Usuarios.ToList());

}

en donde db es un ApplicationContext

AJAX

<https://msdn.microsoft.com/es-es/library/bb398874(v=vs.100).aspx>

### Conexión de red

La capa de conexión de red administra la comunicación entre el script del explorador y los servicios y aplicaciones web. También administra las llamadas asincrónicas a métodos remotos. En muchos escenarios habituales, como las actualizaciones parciales de la página que utilizan el control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx), la capa de conexión de red se utiliza automáticamente y no es necesario escribir ningún código.

La capa de conexión de red también proporciona compatibilidad para el acceso a autenticación de formularios basada en servidor, información de funciones e información del perfil en los scripts de cliente. Esta compatibilidad también está disponible para las aplicaciones web que no se crean utilizando ASP.NET, siempre que la aplicación tenga el acceso a la Microsoft AJAX Library. Para obtener más información, vea [Servicios web de AJAX en ASP.NET](https://msdn.microsoft.com/es-es/library/bb398785(v=vs.100).aspx).

### Servicios básicos

Las bibliotecas de scripts de cliente de AJAX en ASP.NET están compuestas por archivos JavaScript (.js)

### Controles de servidor

Los controles de servidor de AJAX en ASP.NET están compuestos por código de servidor y de cliente que se integra para generar un comportamiento de cliente enriquecido.

[ScriptManager](https://msdn.microsoft.com/es-es/library/system.web.ui.scriptmanager(v=vs.100).aspx)

Administra los recursos del script para los componentes de cliente,

[UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx)

Permite actualizar las partes seleccionadas de la página, en lugar de actualizar la página entera

[UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx)

Proporciona información del estado de las actualizaciones parciales de la página en controles [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx).

[Timer](https://msdn.microsoft.com/es-es/library/system.web.ui.timer(v=vs.100).aspx)

Realiza devoluciones de datos en intervalos definidos. Puede utilizar el control [Timer](https://msdn.microsoft.com/es-es/library/system.web.ui.timer(v=vs.100).aspx) para exponer la página entera, o utilizarlo con el control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx) para realizar actualizaciones parciales de la página a intervalos definidos.

PROCEDIMIENTO:

## Crear un sitio Web ASP.NET

Los sitios web que crea en ASP.NET incluyen de forma predeterminada compatibilidad con la funcionalidad de AJAX.

### Para crear un sitio web ASP.NET habilitado para AJAX

1. Inicie Visual Studio.
2. En el menú **Archivo**, haga clic en **Nuevo sitio Web**.
3. Se muestra el cuadro de diálogo **Nuevo sitio Web**.
4. En **Plantillas instaladas de Visual Studio**, seleccione **Sitio Web ASP.NET**.
5. Escriba una ubicación y un idioma y, después, haga clic en **Aceptar**.

2. Para crear una nueva página web ASP.NET

1. En el Explorador de soluciones, haga clic con el botón secundario en el nombre del sitio y haga clic en **Agregar nuevo elemento**.
2. Se abrirá el cuadro de diálogo **Agregar nuevo elemento**.
3. En **Plantillas instaladas de Visual Studio**, seleccione **Web Forms**.
4. Asigne a la nueva página el nombre Employees.aspx y desactive la casilla **Colocar el código en un archivo independiente**.
5. Seleccione el idioma que desea usar.
6. Haga clic en **Agregar**.
7. Cambie a la vista Diseño.
8. En la ficha **Extensiones AJAX** del cuadro de herramientas, haga doble clic en el control [ScriptManager](https://msdn.microsoft.com/es-es/library/system.web.ui.scriptmanager(v=vs.100).aspx) para agregarlo a la página.

### 3. Para agregar contenido a un control UpdatePanel

El control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx) realiza actualizaciones parciales de la página e identifica contenido que se actualiza independientemente del resto de la página.

1. En la ficha **Datos** del cuadro de herramientas, arrastre un control [GridView](https://msdn.microsoft.com/es-es/library/system.web.ui.webcontrols.gridview(v=vs.100).aspx) hasta el área editable del control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx).
2. En el menú **Tareas de GridView**, haga clic en **Formato automático**.
3. En el panel **Formato automático**, en **Seleccionar una combinación**, seleccione **Multicolor** y, a continuación, haga clic en **Aceptar**.
4. En el menú **Tareas de GridView**, seleccione **<Nuevo origen de datos>** en la lista **Elegir origen de datos**.
5. Aparece el **Asistente para la configuración de orígenes de datos**.
6. En **¿De dónde obtendrá la aplicación los datos?**, seleccione **Base de datos** y haga clic en **Aceptar**.
7. En el **Asistente para la configuración de orígenes de datos**, en el paso **Elegir la conexión de datos**, configure una conexión a la base de datos de AdventureWorks y, a continuación, haga clic en **Siguiente**.
8. En el paso **Configurar la instrucción Select**, seleccione **Especificar una instrucción SQL o un procedimiento almacenado personalizado**y, a continuación, haga clic en **Siguiente**.
9. En la ficha **SELECT** del paso **Definir instrucciones o procedimientos almacenados personalizados**, escriba la siguiente instrucción SQL:
10. SELECT FirstName, LastName FROM HumanResources.vEmployee ORDER BY LastName, FirstName
11. Haga clic en **Siguiente**.
12. Haga clic en **Finalizar**.
13. En el menú **Tareas de GridView**, active la casilla **Habilitar paginación**.
14. Guarde los cambios y presione CTRL+F5 para ver la página en un explorador.

## 4. Agregar un control UpdateProgress a la página

El control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) muestra un mensaje de estado mientras se solicita el nuevo contenido de un control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx).

### Para agregar un control UpdateProgress a la página

1. En la ficha **Extensiones AJAX** del cuadro de herramientas, arrastre un control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) a la página y colóquelo debajo del control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx).
2. Seleccione el control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) y, en la ventana Propiedades, establezca la propiedad [AssociatedUpdatePanelID](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress.associatedupdatepanelid(v=vs.100).aspx) en **UpdatePanel1**.**.**
3. De esta forma asocia el control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) al control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx) que agregó previamente.
4. En el área editable del control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx), escriba **Obteniendo empleados... .**
5. Guarde los cambios y presione CTRL+F5 para ver la página en un explorador.
6. Si hay un retraso mientras la página ejecuta la consulta SQL y devuelve los datos, el control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) muestra el mensaje que escribió en el control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx).

## 5. Agregar un retraso a la aplicación de ejemplo

Si la aplicación actualiza rápidamente cada página de datos, es posible que no vea el contenido del control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) en la página. El control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) admite una propiedad [DisplayAfter](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress.displayafter(v=vs.100).aspx) que permite establecer un retraso antes de que se muestre el control. De esta forma evita que el control parpadee en el explorador si la actualización se produce con mucha rapidez. De forma predeterminada, el retraso se establece en 500 milisegundos (0,5 segundos), lo que significa que el control [UpdateProgress](https://msdn.microsoft.com/es-es/library/system.web.ui.updateprogress(v=vs.100).aspx) no se mostrará si la actualización tarda menos de medio segundo.

### Para agregar un retraso a la aplicación de ejemplo

1. En el control [UpdatePanel](https://msdn.microsoft.com/es-es/library/system.web.ui.updatepanel(v=vs.100).aspx), seleccione el control [GridView](https://msdn.microsoft.com/es-es/library/system.web.ui.webcontrols.gridview(v=vs.100).aspx).
2. En la ventana Propiedades, haga clic en el botón **Eventos**.
3. Haga doble clic en el evento [PageIndexChanged](https://msdn.microsoft.com/es-es/library/system.web.ui.webcontrols.gridview.pageindexchanged(v=vs.100).aspx) para crear un controlador de eventos.
4. Agregue el código siguiente al controlador de eventos [PageIndexChanged](https://msdn.microsoft.com/es-es/library/system.web.ui.webcontrols.gridview.pageindexchanged(v=vs.100).aspx) para crear artificialmente un retraso de tres segundos:
5. **[C#](https://msdn.microsoft.com/es-es/library/bb398890(v=vs.100).aspx?cs-save-lang=1&cs-lang=csharp#code-snippet-2)**
6. 'Include three second delay for example only.  
   System.Threading.Thread.Sleep(3000)

# Tutorial: Agregar scripting de ASP.NET AJAX a un proyecto de MVC

<https://msdn.microsoft.com/es-es/library/dd381533(v=vs.100).aspx?cs-save-lang=1&cs-lang=csharp#code-snippet-1>

# El Poder de AJAX en Tiempo Real

<https://msdn.microsoft.com/es-es/library/bb972191.aspx>

# Crear script de cliente personalizado mediante el uso de Microsoft AJAX Library

LINQ

LINQ es un componente del framework .Net que permite hacer queries contra objetos. Esto incluye arreglos, listas, datatables, etc.. La mejor manera de demostrar el poder que LINQ te otorga es con varios ejemplos.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | string[] nombres = { "Juan", "Pedro", "Roberto", "Luis", "Lucy", "Patricia" };    string juan = (from n in nombres  where n == "Juan"  select n).First();    Console.WriteLine("Nombre: " + juan); |

Resultado:

Nombre: Juan

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | string[] nombres = { "Juan", "Pedro", "Roberto" };    var nombresOrdenados = from n in nombres  order by n  select n;    foreach(var nom in nombresOrdenados)  {  Console.WriteLine(juan);  } |

Resultado:

Juan

Pedro

Roberto

También podemos usar operadores agregados como por ejemplo count, sum, min, max, etc.

|  |  |
| --- | --- |
| 1  2  3  4  5 | string[] nombres = { "Juan", "Pedro", "Roberto" };    int cantidad = nombres.Count();    Console.WriteLine("Hay " + cantidad + " nombres en el arreglo"); |

Si necesitan más ejemplos hay una excelente página para eso:

<https://code.msdn.microsoft.com/101-LINQ-Samples-3fb9811b>

# NEWTONSOFT

# Json.NET

Popular high-performance JSON framework for .NET

* Flexible JSON serializer for converting between .NET objects and JSON
* LINQ to JSON for manually reading and writing JSON
* High performance: faster than .NET's built-in JSON serializers

POO

Para determinar la visibilidad el lenguaje cuenta con las siguientes palabras clave:

1. **public**: la clase puede ser utilizada en **cualquier** proyecto.
2. **internal**: la clase está limitada al proyecto en el cual está definida.
3. **private**: la clase sólo puede usarse en el módulo en la que está definida.
4. **protected**: la clase sólo puede ser utilizada en una subclase. Es decir sólo se puede utilizar protected para una clase declarada en otra clase.
5. **protected internal**: lo mismo que internal + protected.
6. **abstract**: no permite crear instancias de esta clase, sólo sirve para ser heredada como clase base. Suelen tener los métodos definidos pero sin ninguna operatividad con lo que se suele escribir estos métodos en las clases derivadas.
7. **sealed**: cuando una clase es la última de una jerarquía, por lo que no podrá ser utilizada como base de otra clase.

**Creación de métodos, sobrecarga y sobreescritura.**

Son procedimientos o funciones definidos dentro de una CLASE. Los métodos pueden manejar los campos de la clase incluso si son privados.

La sobrecarga es la creación dentro de la clase, de un grupo de métodos que tienen el mismo nombre pero con un número de parámetros distinto y/o bien distintos tipos de datos.

|  |  |
| --- | --- |
| 5  6  7  8  9  10 | public void visualización () {  MessageBox.Show("Sr. "+Apellido+" "+Nombre+" nacido el "+FechaNac);  }    public void visualización (string idioma) {  switch (idioma) {  case "es": MessageBox.Show("Sr. "+Apellido+" "+Nombre+" nacido el "+FechaNac); break;  case "en": MessageBox.Show("Mr. "+Apellido+" "+Nombre+" was born "+FechaNac); break;  }  } |

Sabemos que las clases derivadas heredan las propiedades y métodos de su clase base. Se pueden usar sin ninguna modificación, pero sí el método no está adaptado a la nueva clase podemos sobrescribirlo. Para ello utilizamos la palabra reservada ***override.***También es obligatorio que permitir la sobrescritura de mediante el de la palabra reservada ***virtual***.Esto se utiliza para asegurar el ***polimorfismo***entre las clases.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | public override void visualización () {  MessageBox.Show("Sr. "+Apellido+" "+Nombre+" nacido el "+FechaNac+" cobra "+Salario+".-€uros/mes.");  }    public sealead override void visualización () {  base.visualizacion();  MessageBox.Show("y cobra "+Salario+".-€uros/mes.");  } |

Ejemplo de método abstracto

C#

|  |  |
| --- | --- |
| 1  2  3  4  5 | ...  public abstract string EstadoCivil(); |

**Métodos de extensión**

Los métodos de extensión permiten añadir funcionalidades a una clase ya definida sin tener que modificar el código de esta clase. Se deben respetar las siguientes reglas:

* Pueden ser de tipo procedimientos o función. NUNCA propiedad.
* El primer parámetro irá precedido de la palabra **this**. La palabra clave **this** hace referencia a la instancia actual de la clase ,pero también se utiliza como modificador del primer parámetro de un método de extensión.
* El tipo del primer parámetro del método determina el tipo extendido por este método.
* En el momento de la ejecución, éste primer parámetro representa la instancia de la clase sobre la cual se llama el método.
* Se deben definir una clase **static**.
* Ellos mismos deben ser **static**.

|  |  |
| --- | --- |
| 4  5  6  7 | static class ExtesionPersona {  public static void presentacion(this Persona p) {  Console.WriteLine("Apellido: {0}", p.apellido);  Console.WriteLine("Nombre: {0}",p.nombre);  Console.writeLine("Fecha de nacimiento: {0}", p.fecha\_naci);  }  } |

CLASES Y POO:

<http://dis.um.es/~bmoros/privado/apuntes/Curso09-10/POO6-CSharp-0910.pdf>

CÓMO SABER SI UN PROYECTO ES .NET CORE O NET FRAMEWORK:

It is quite easy to determine Asp.net Core application.

There are some identifiers to it. There are few file in a project that helps to determine Core Project

Files like

* project.json
* appsetting.json
* startup.cs

Inside project.json, it include

"dependencies": { "Microsoft.NETCore.App": { "version": "1.0.0", "type": "platform" }, "frameworks": { "netcoreapp1.0": { "imports": [ "dotnet5.6", "portable-net45+win8" ] } },

Here by dependencies one can determine about the type of project..NET Core application do reference Microsoft.NETCore.App (either as "type": "platform" for portable apps or without it for self-contained apps).

Asp.net Core Application are platform independent, So a ASP.NET Core application do not express much about the platform .

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

ACTUALIZAR MIGRACIÓN:

> dotnet ef database update

LAFRANCOL:

1. Crear la base de datos: db\_RTVSS en el Explorador de Objetos de SQL Server.
2. Hacer clic derecho sobre la nueva Base de datos y seleccionar Nueva Consulta
3. Abrir el archivo con el script
4. Pegar el script en la Nueva Consulta y ejecutar.

¿Cómo saber la versión de MVC que estoy utilizando?

Ir a la carpeta References: System.Web.Mvc y en las propiedades consultar la versión

5.2.7

Ninject,

JavaScriptSerializer serializer = new JavaScriptSerializer();

trigger

RPA

[https://academy.uipath.com](https://academy.uipath.com/)

<https://www.uipath.com/kb-articles/start-process-from-.net-application>

A project is a graphical representation of a business process.

TUTORIAL:

<https://studio.uipath.com/docs/about-automation-projects>

The main types of supported projects are:

* [**Sequences**](https://studio.uipath.com/v2018.1/docs/sequences) - suitable to linear processes, enabling you to smoothly go from one activity to another, without cluttering your project.
* [**Flowcharts**](https://studio.uipath.com/v2018.1/docs/flowcharts) - suitable to a more complex business logic, enabling you to integrate decisions and connect activities in a more diverse manner, through multiple branching logic operators.
* [**State Machines**](https://studio.uipath.com/v2018.1/docs/state-machines) – suitable for very large projects; they use a finite number of states in their execution which are triggered by a condition (transition) or activity.

**-> New Process**

->Insert a Flowchart

-> Insert Input Dialog from System-> Dialog

ROBOT:

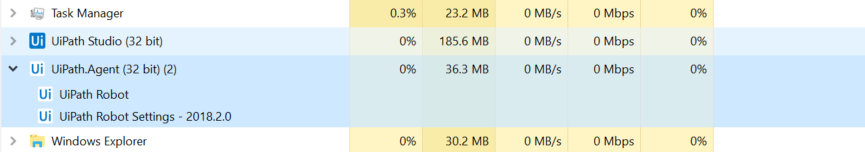
Robot Service, which are located in the %userprofile%\.nuget\packages folder.

The Robot is split into four components:

**Service** (UiPath.Service.Host.exe):



* **Executor** (UiPath.Executor.exe):
  + Runs the given jobs under a Windows session (executes workflows);
  + Is aware of per-monitor DPI settings.
* **Agent** (UiPath.Agent.exe, Robot Tray):
  + A WPF application which displays the available jobs in the system tray window;
  + Is a client of **Service**;
  + Can request to start or stop jobs and change settings.



**Command Line** (UiRobot.exe, Command line):

* Is a client of **Service**;
* A console application that can request to start jobs and waits for their output.

TUTORIAL PRÁCTICO:

<https://planetachatbot.com/tutorial-rpa-automatizando-el-llenado-de-un-formulario-con-uipath-244f11f90403>

TUTORIAL

<https://www.c-sharpcorner.com/article/an-introduction-to-robotic-process-automation-rpa-with-uipath/>

**GIT**

Para administrar el repositorio GIT se utiliza SOURCE TREE.

Una vez guardados los cambios → Stage All → Commit → Push.