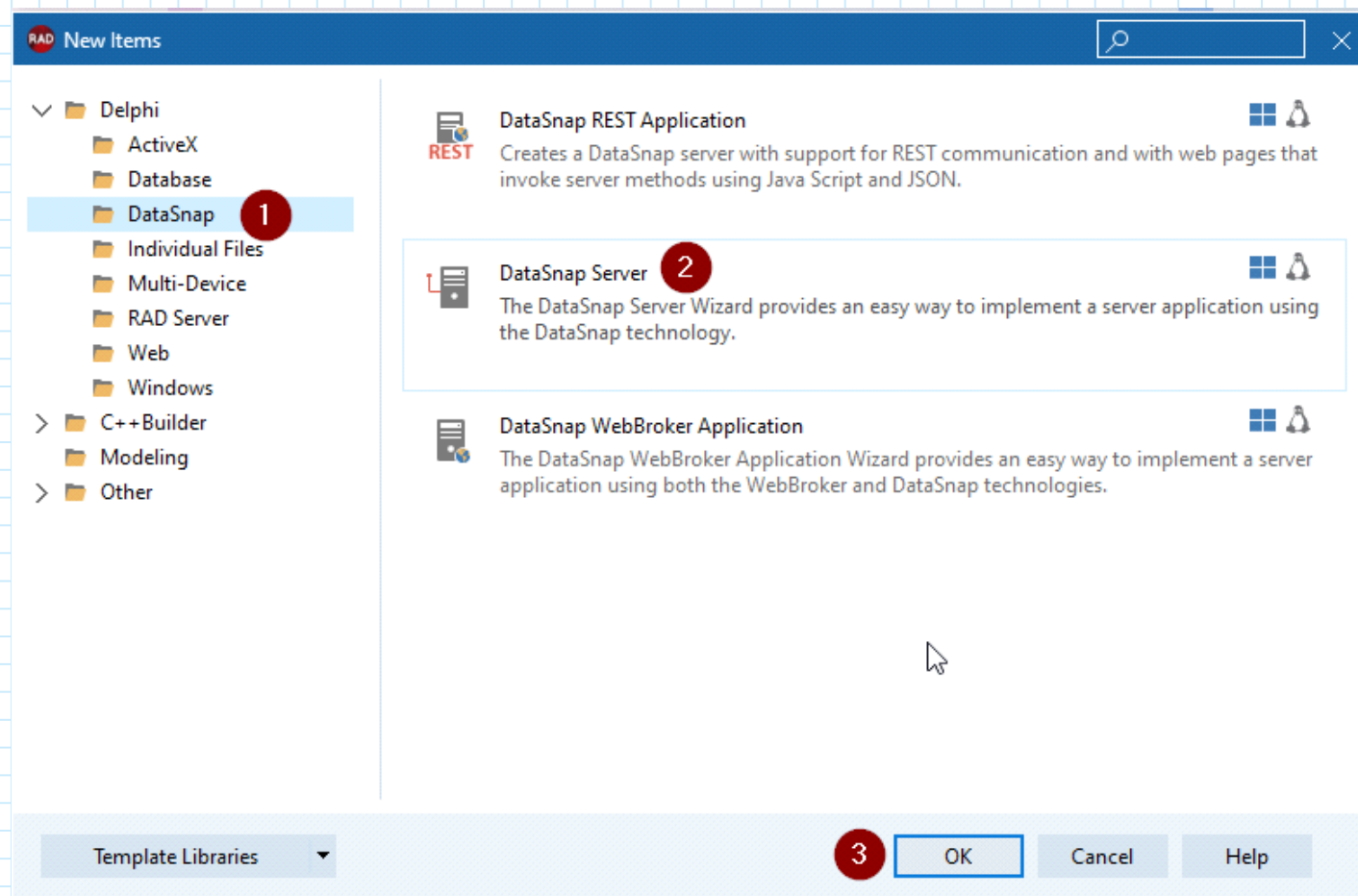



# CREACION

Lunes, 9 de setiembre de 2024 09:36

## Pasos para crear una aplicación DataSnapServer

1. File / New / Other






 New DataSnap Server

Platform

Select the platform for the application to be created

Specify the type of platform for the application



  
1100011000000001110  
1110000101010100111  
111111010000000111  
101001010101000111

☒ Windows

☐ Linux

1 of 6


<< Back

Next >> 1

Finish

Cancel


Help



 New DataSnap Server

Project type

Specify the type of this application

A Forms application displays a form



  
1100011000000001110  
1110000101010100111  
111111010000000111  
101001010101000111

☒ Forms Application

☐ Console Application

☐ Service Application

2 of 6

<< Back

Next >> 1

Finish

Cancel


Help

New DataSnap Server

## Application Type

Select the type of application to be created

Specify the type of application that will be created



☒ VCL application

☐ FireMonkey application

3 of 6

<< Back

Next >>

1 Finish

Cancel


Help

New DataSnap Server

## Server Features

Check the features to add to the DataSnap server

Choose the protocols that can be used to connect to the DataSnap server.



✓ ☒ Protocols

☒ TCP/IP

☐ HTTP

☐ HTTPS

✓ ☐ Authentication

☐ Authorization

✓ ☒ Server Methods Class

☒ Sample Methods

✓ ☐ Filters

☐ Encryption

☐ Compression

☐ JavaScript Files

☐ Mobile Connectors

☒ Select / deselect all

4 of 6

<< Back

Next >>

1 Finish

Cancel

Help

DATASNAP SERVIDOR página 3

New DataSnap Server

Port Numbers

Specify the ports that will be used by the DataSnap server to listen for client requests. Use "Test" button to make sure the port is not already in use on this computer.

Click a field for more information

211

Test Port

Find Open Port

Information

Test port succeeded

OK

5 of 6

<< Back

Next >>

Finish

Cancel

Help

New DataSnap Server

Port Numbers

Specify the ports that will be used by the DataSnap server to listen for client requests. Use "Test" button to make sure the port is not already in use on this computer.

Click a field for more information

211

Test Port

Find Open Port

5 of 6

<< Back

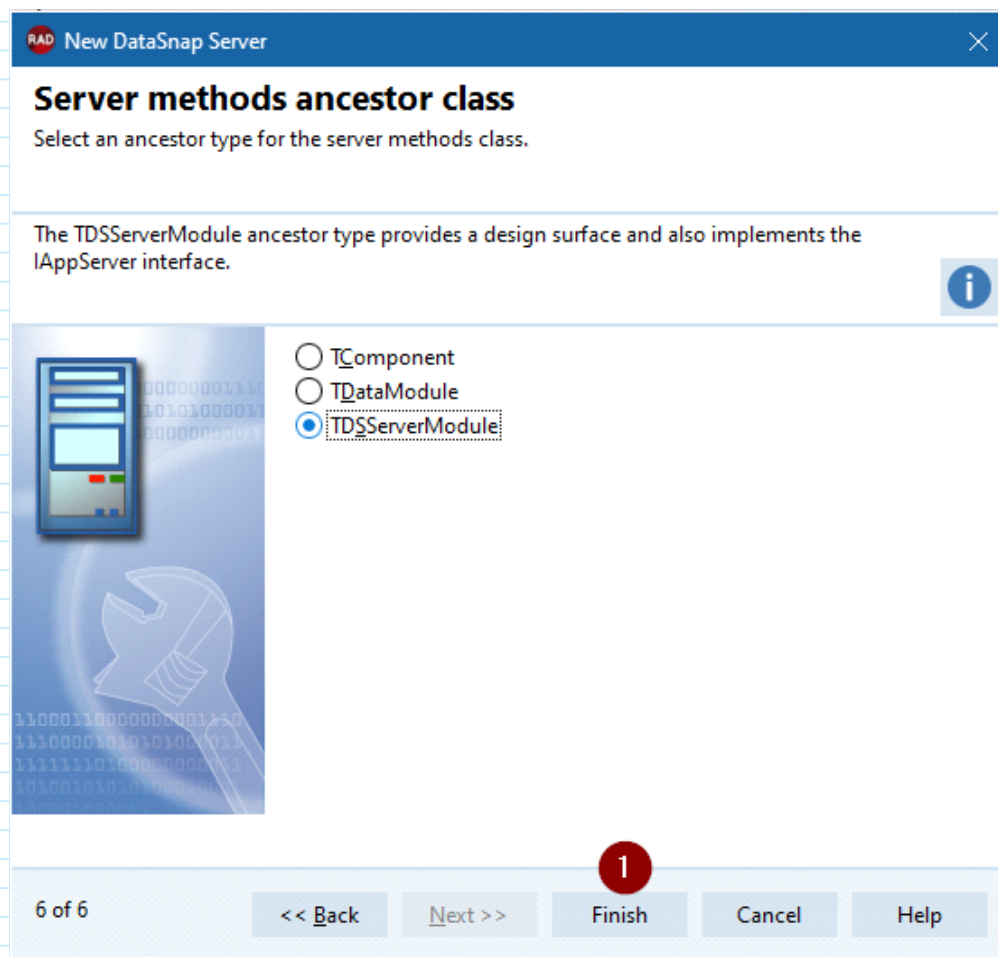
Next >>

1 Finish

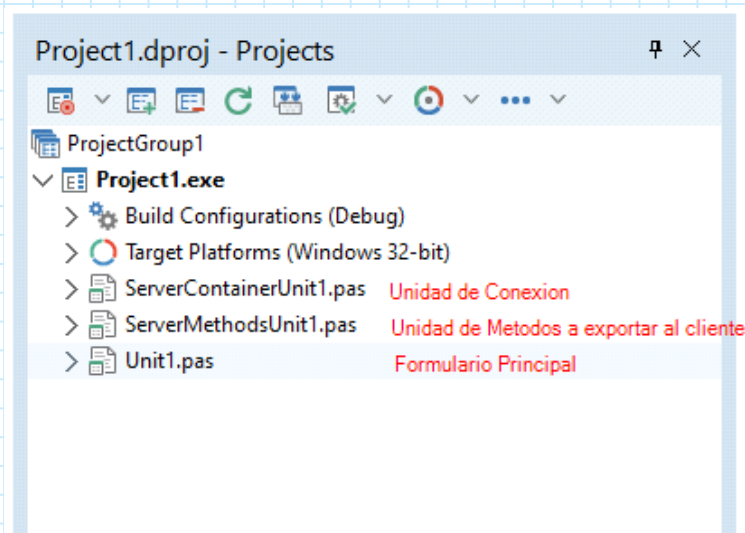
Cancel

Help

DATASNAP SERVIDOR página 4



2. Se crean las unidades por defecto. Renombrar según sea el caso

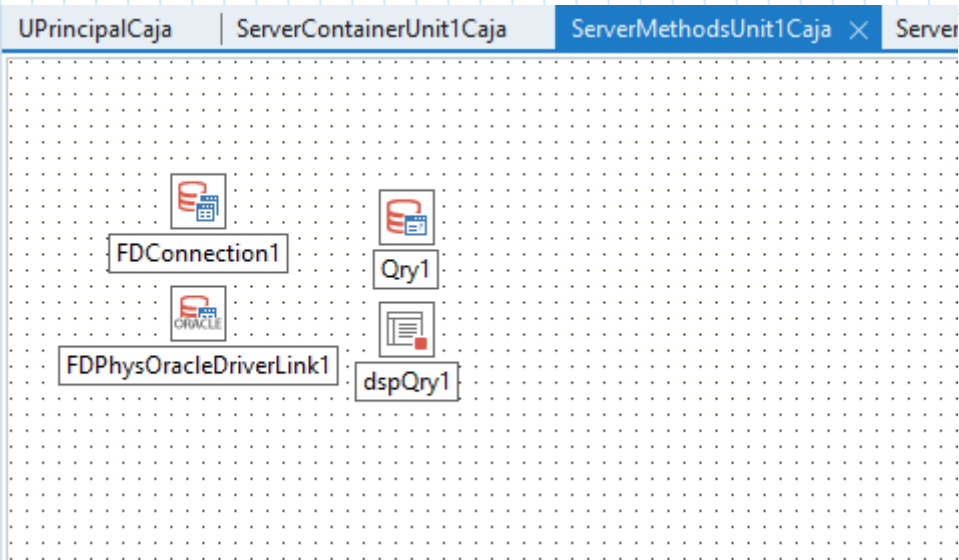




# PERSONALIZACION

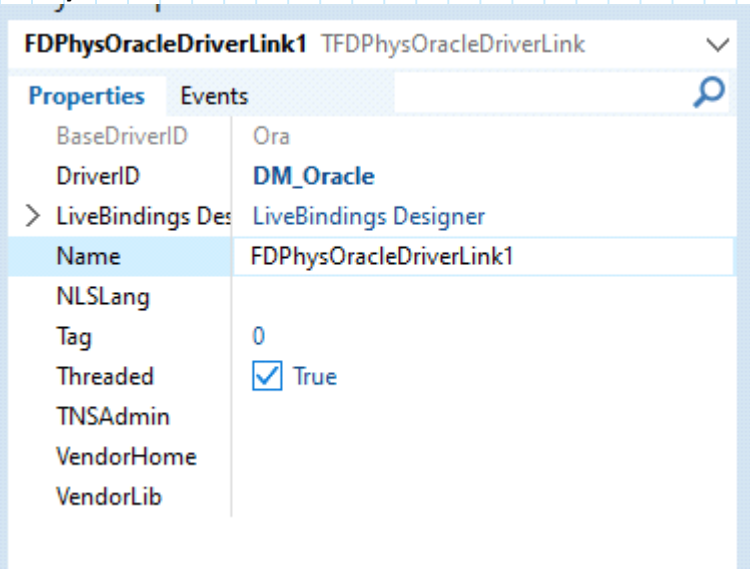
martes, 10 de setiembre de 2024 12:13

## Añadir componentes

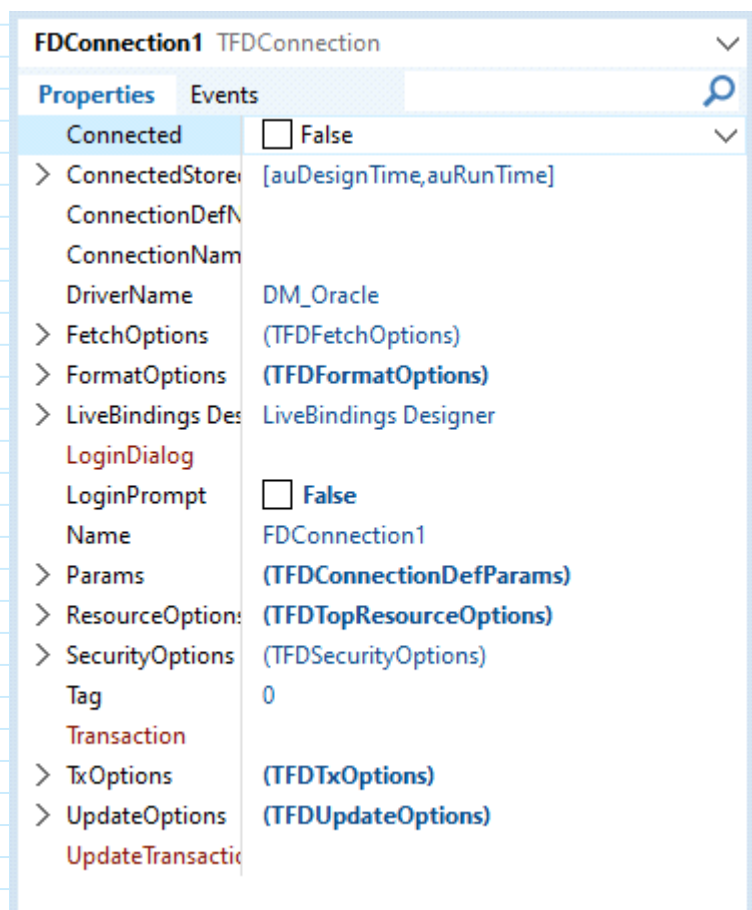


## Propiedades de los componentes

### FDPhysOracleDriverLink1



### FDConnection1



Clic derecho sobre el componente, Connection Editor



**FireDAC Connection Editor - [FDConnection1]**

Select driver or select connection definition name to override, then setup parameters

Definition Options Info SQL Script

Driver ID:

Connection Definition Name:

Test Wizard Revert To Defaults Help

Parameter	Value	Default
DriverID	DM_Orade	DM_Orade
Pooled	False	False
Database	DEMADBTEST01	
User_Name	DB2ADMIN	
Password	lider	
MonitorBy		
OSAuthent		
AuthMode	Normal	Normal
ReadTimeout		
WriteTimeout		
CharacterSet	<NLS_LANG>	<NLS_LANG>
NCharReplacement	True	True
StringFormat	Choose	Choose
BooleanFormat	Choose	Choose
ApplicationName		
OradeAdvanced		
MetaDefSchema		
MetaCurSchema		

OK Cancel

dspQry1

**Object Inspector**

dspQry1 TDataSetProvider

Properties Events

Constraints ☒ True

> DataSet Qry1

Exported ☒ True

> LiveBindings Designer LiveBindings Designer

Name dspQry1

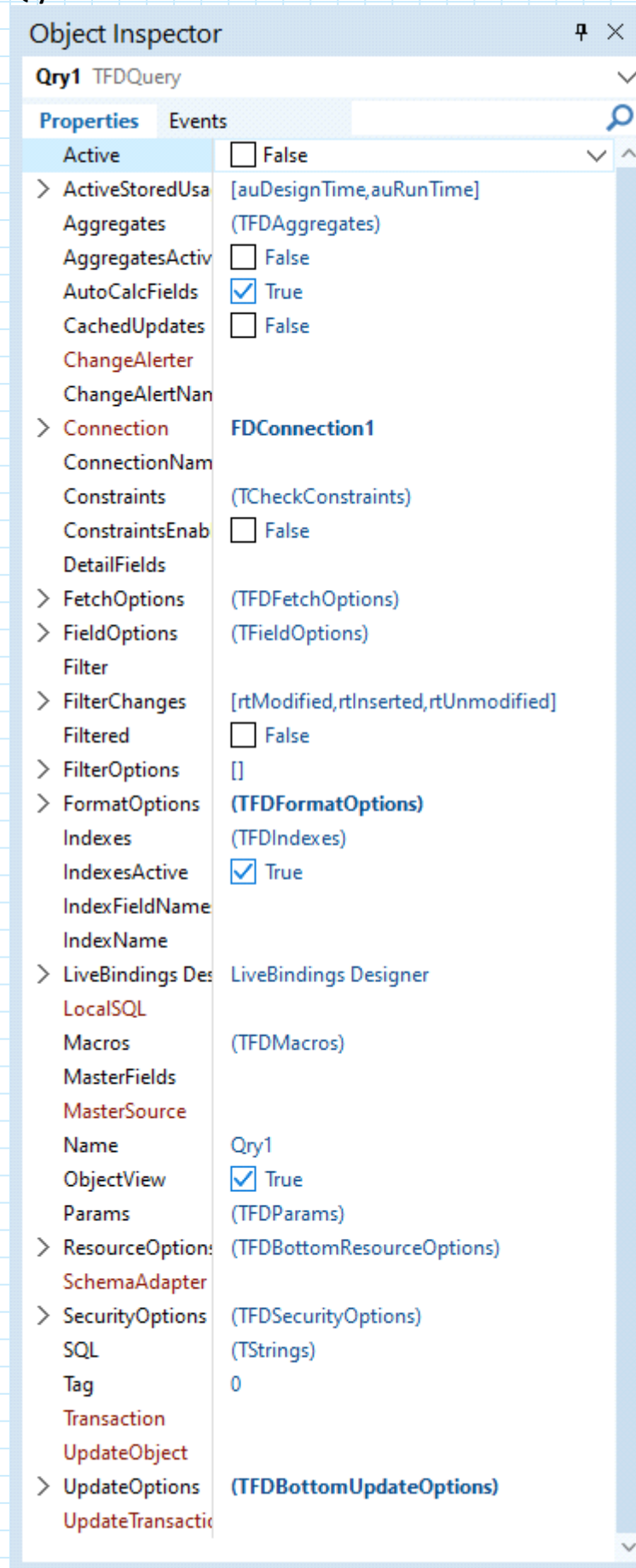
> Options [poUseQuoteChar]

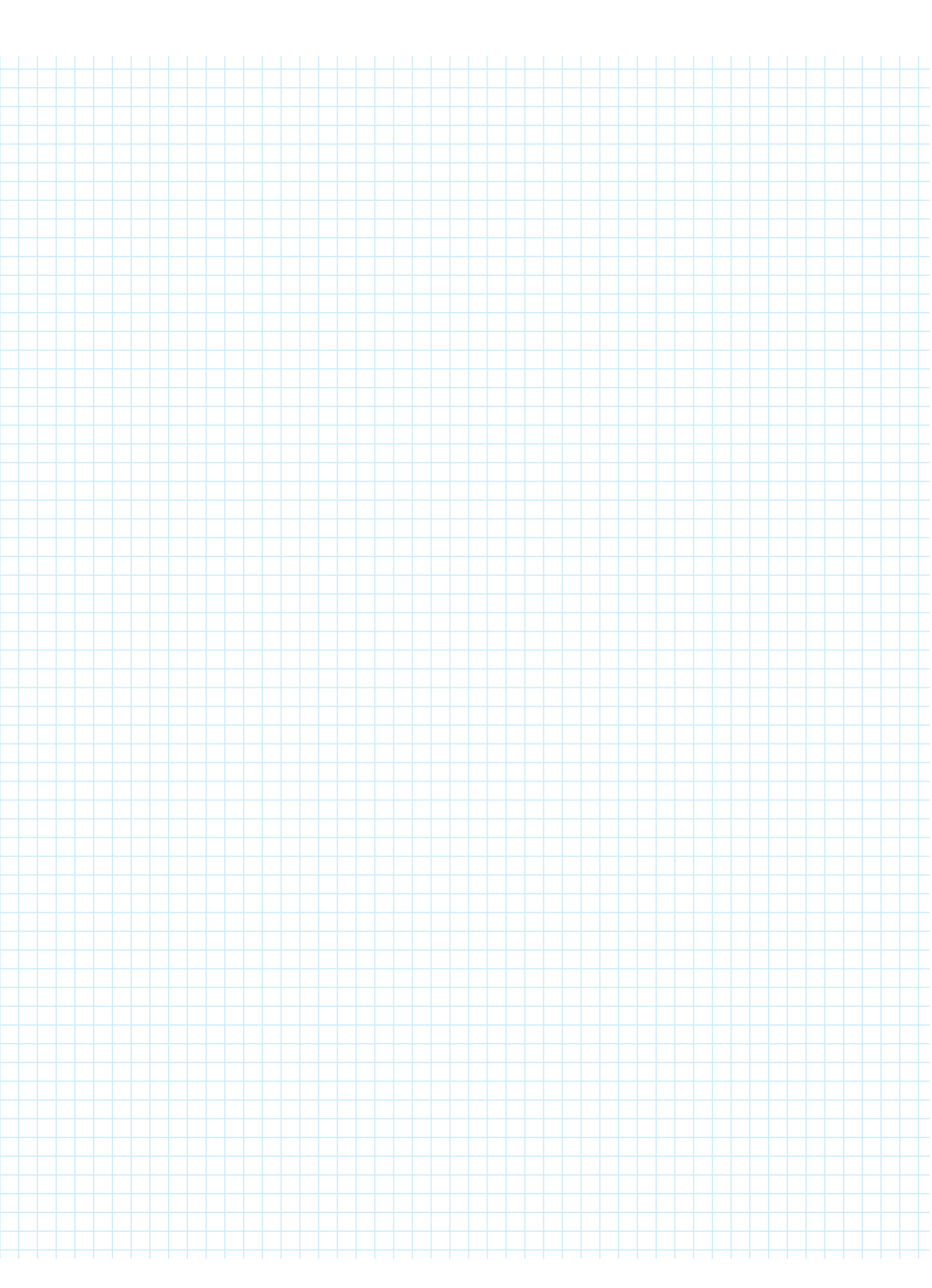
ResolveToDataSet ☐ False

Tag 0

UpdateMode upWhereAll

Qry1





# METODOS DEL SERVIDOR

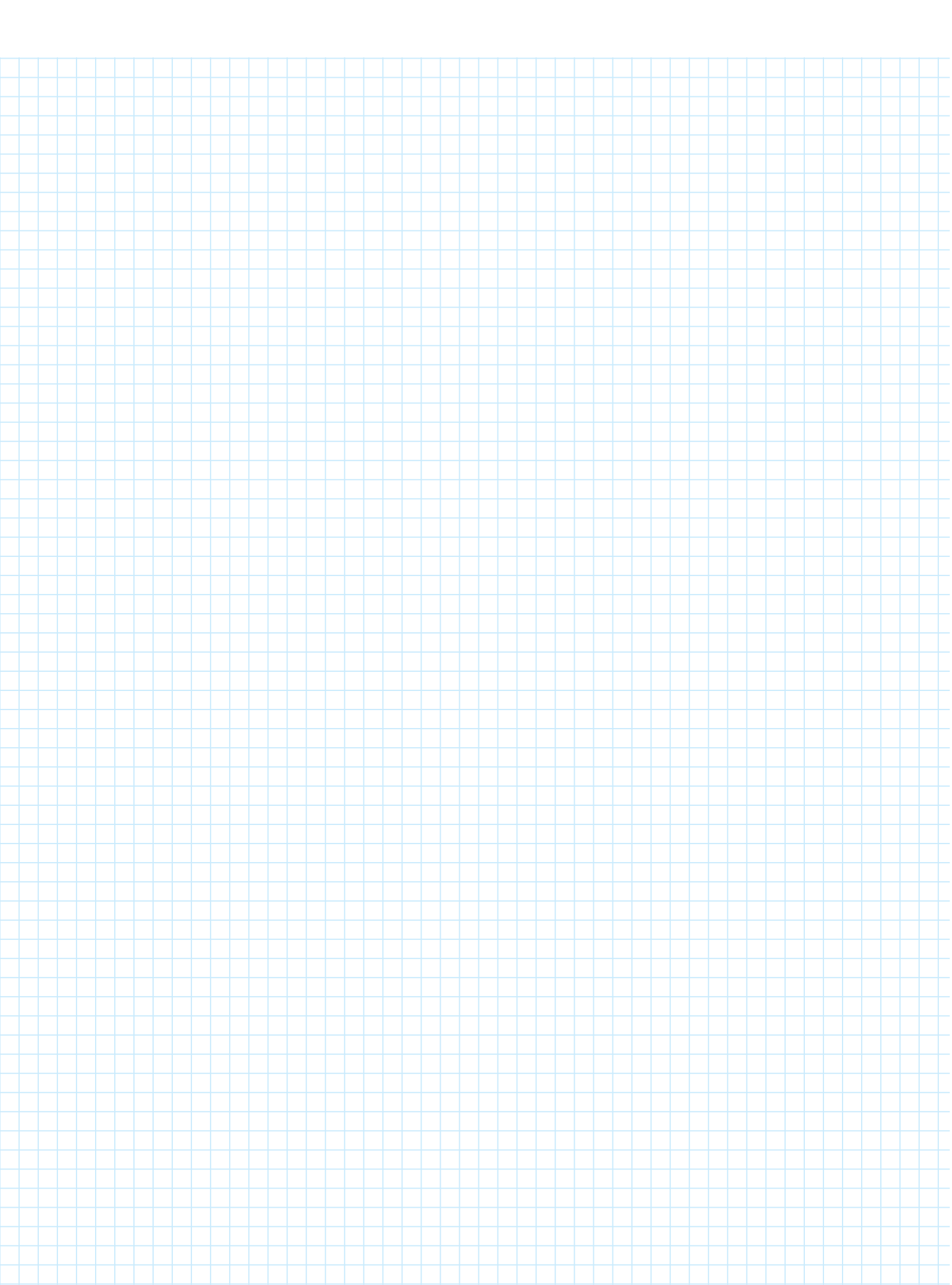
martes, 10 de setiembre de 2024 12:23

## Método DataRequest y EjecutaData

```

• unit ServerMethodsUnit1Caja;
•
• interface
•
• uses System.SysUtils, System.Classes, System.Json,
•       DataSnap.DSProviderDataModuleAdapter,
•       DataSnap.DSServer, DataSnap.DSAuth, FireDAC.Stan.Intf, FireDAC.Stan.Option,
•       FireDAC.Stan.Error, FireDAC.UI.Intf, FireDAC.Phys.Intf, FireDAC.Stan.Def,
•       FireDAC.Stan.Pool, FireDAC.Stan.Async, FireDAC.Phys, FireDAC.Phys.Oracle,
10  FireDAC.Phys.OracleDef, FireDAC.VCLUI.Wait, Data.DB, FireDAC.Comp.Client,
•       FireDAC.Stan.Param, FireDAC.DatS, FireDAC.DApt.Intf, FireDAC.DApt,
•       DataSnap.Provider, FireDAC.Comp.DataSet;
•
• type
•   TServerMethods1Caja = class(TDSServerModule)
•     FDConnection1: TFDConnection;
•     FDPhysOracleDriverLink1: TFDPhysOracleDriverLink;
•     Qry1: TFDQuery;
•     dspQry1: TDataSetProvider;
20  function DataRequest(Sender: TObject; Input: OleVariant): OleVariant;
•   private
•     { Private declarations }
•   public
•     { Public declarations }
•     function ReverseString(Value: string): string;
•     function EjecutaData(sSQL:String;var sError:string):Boolean;
•   end;
30  implementation
•
•   {$R *.dfm}
•
•   uses System.StrUtils;
•
•   function TServerMethods1Caja.DataRequest(Sender: TObject;
•     Input: OleVariant): OleVariant;
40  begin
•     TFDQuery(TDataSetProvider(sender).DataSet).SQL.Text := Input;
•   end;
•
•   function TServerMethods1Caja.EjecutaData(sSQL: String;
50  var sError: string): Boolean;
•   begin
•     Result:=True;
•     FDConnection1.StartTransaction;
•     try
•       Qry1.Close;
•       Qry1.SQL.Clear;
•       Qry1.SQL.Add(sSQL);
•       Qry1.ExecSQL;
•       FDConnection1.Commit;
60  except
•     on E: Exception do
•       begin
•         FDConnection1.Rollback;
•
•         Result:=false;
•         sError:=E.Message;
•       end;
•     end;
•   end;
• end.

```

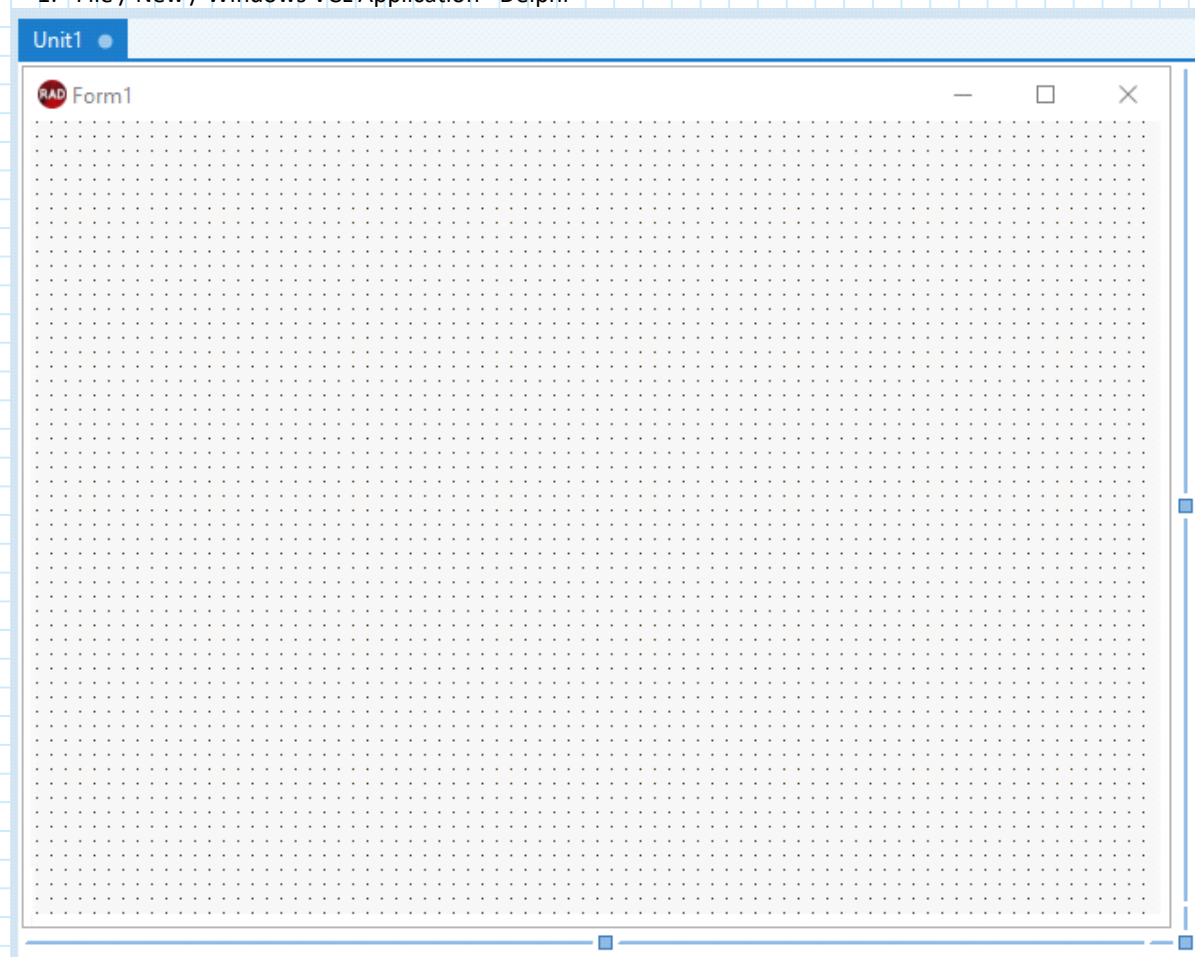


# CREACION

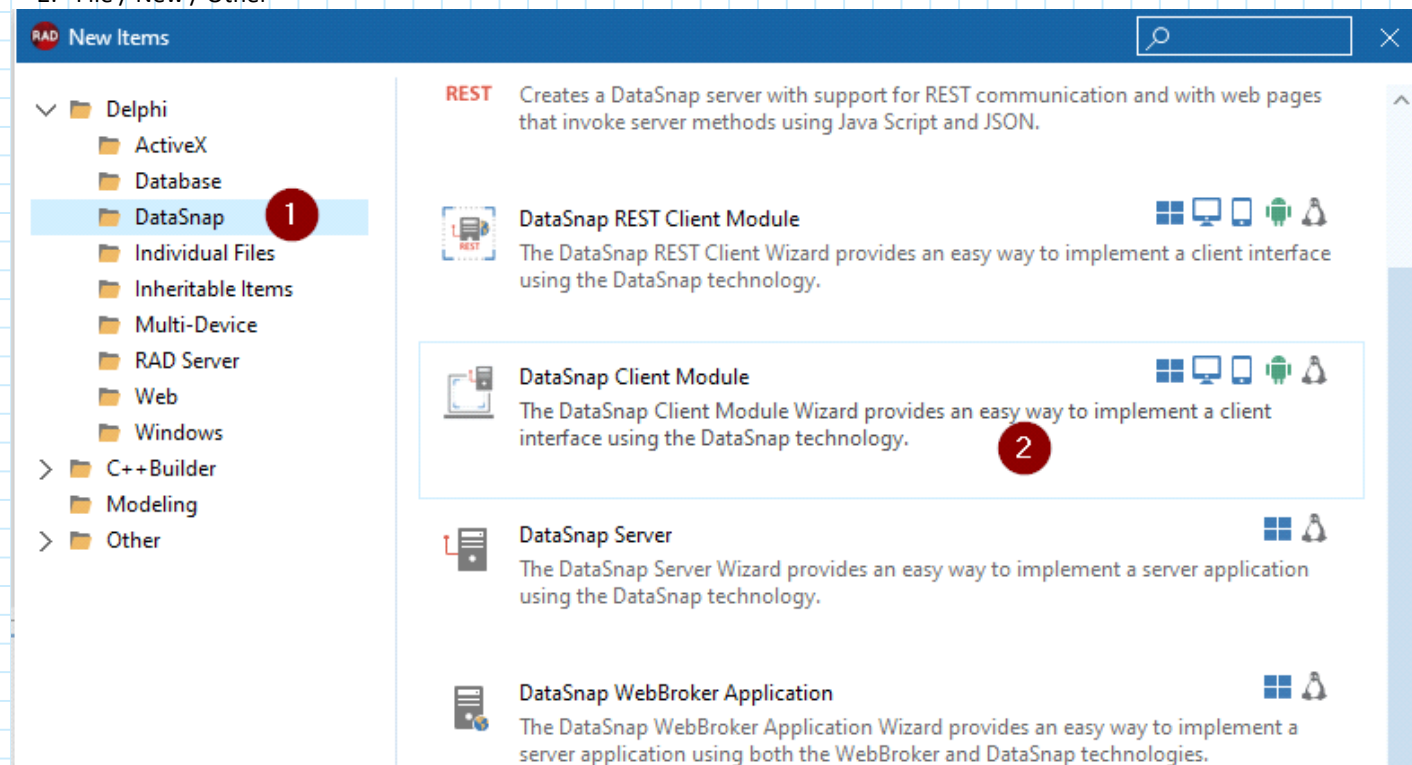
lunes, 9 de setiembre de 2024 09:36

## Pasos para Crear una aplicación DataSnap Cliente

1. File / New / Windows VCL Application - Delphi



2. File / New / Other





## DataSnap WebBroker Application



The DataSnap WebBroker Application Wizard provides an easy way to implement a server application using both the WebBroker and DataSnap technologies.

Template Libraries ▼

OK

Cancel

Help

### DataSnap Client Module

## DataSnap server location

Specify whether the DataSnap server is running on this computer or a remote computer. Note that the DataSnap server must be running to create a connection.

A remote DataSnap server runs on a different computer. Later, this wizard will prompt for the computer name or IP address.



☐ Local server

☒ Remote server

1

1 of 4

<< Back

Next >>

Finish

Cancel

Help



RAD

DataSnap Client Module

## Connection Protocol

Identify the protocol to connect to the DataSnap server

The TCP/IP protocol is selected. The server must have support for TCP/IP.

DBX

☒ TCP/IP

☐ HTTP

<< Back

Next >>

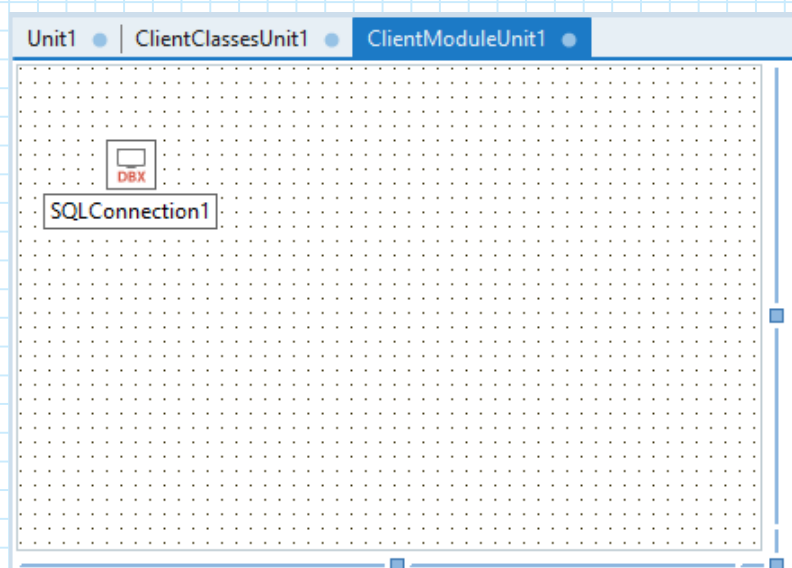
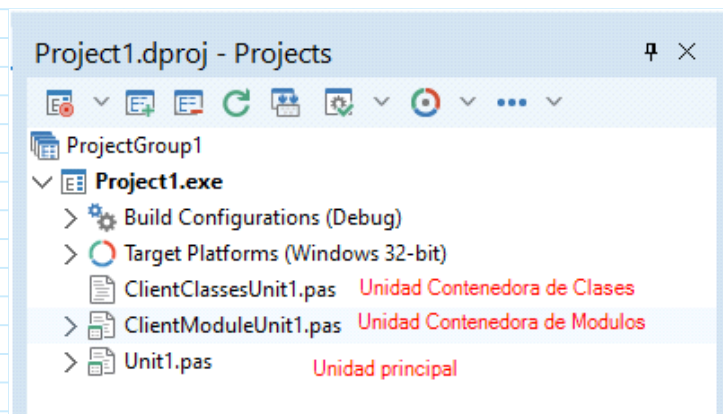
Finish

Cancel

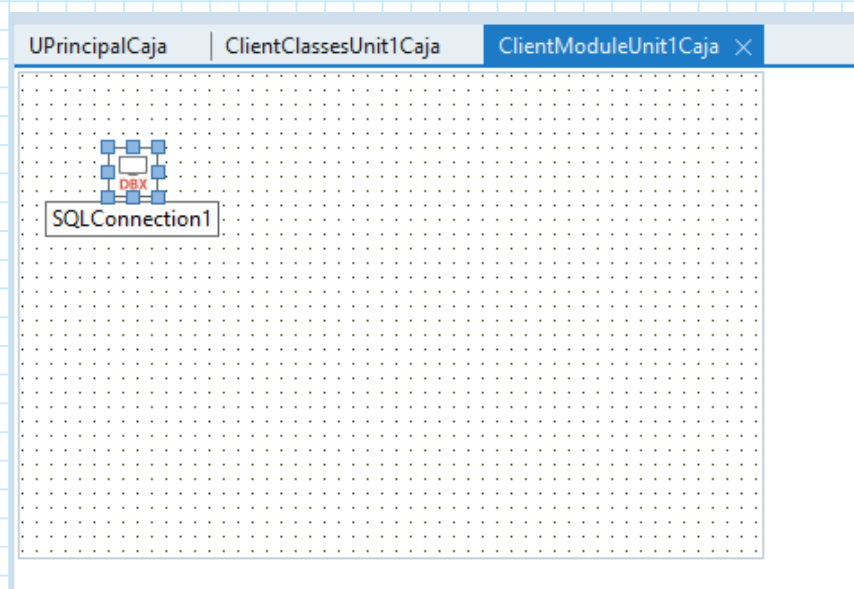
Help

DATASNAP CLIENTE página 17





3. Renombrar según sea el caso

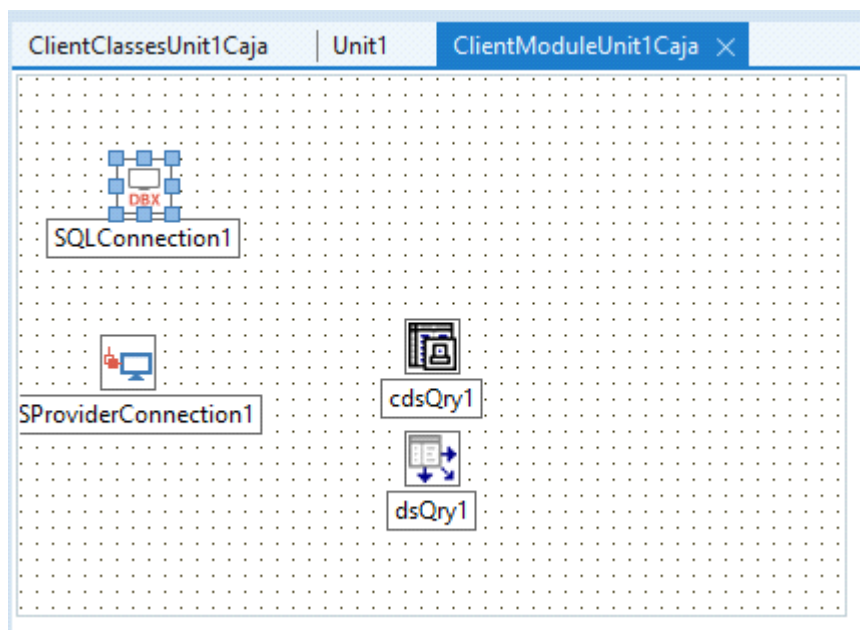




# PERSONALIZACION

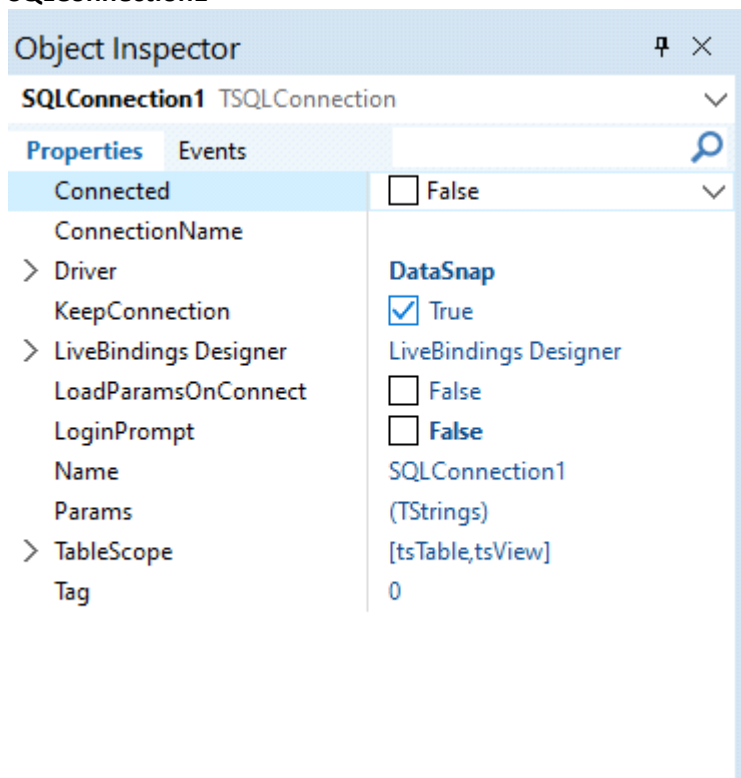
martes, 10 de setiembre de 2024 12:31

## Añadir Componentes



## Propiedades de los componentes

### SQLConnection1



### SQLConnection1.Params

**Value List Editor**

Key	Value
Port	213
HostName	10.250.2.14
CommunicationProtocol	tcp/ip
DatasnapContext	datasnap/

Code Editor... OK Cancel Help

### DSPProviderConnection1

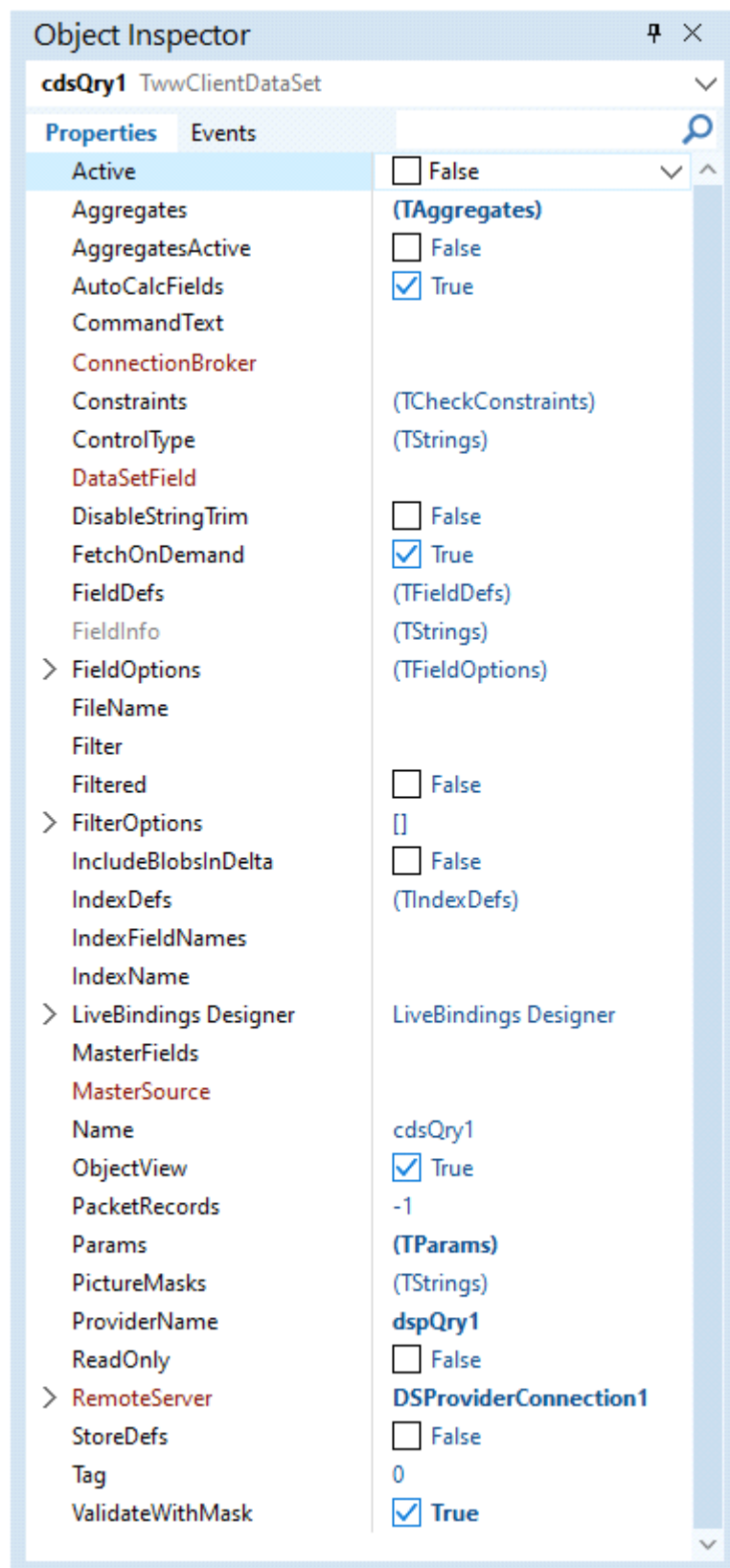
**Object Inspector**

**DSPProviderConnection1** TDSPProviderConnection

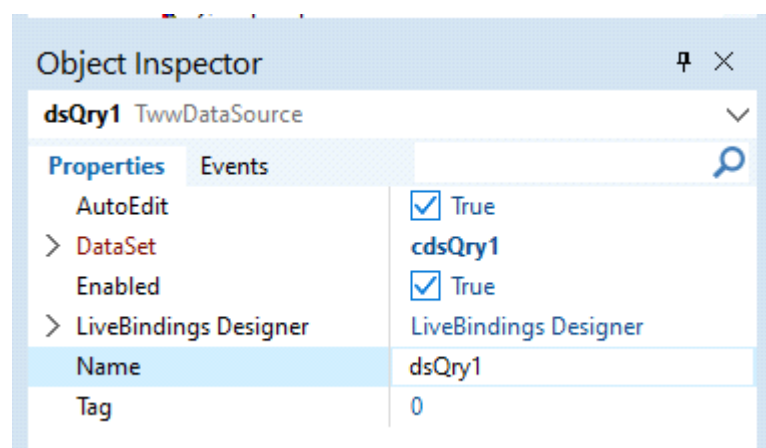
**Properties** Events

Connected	<input type="checkbox"/> False
> LiveBindings Designer	LiveBindings Designer
Name	DSPProviderConnection1
ServerClassName	TServerMethods1Caja
> SQLConnection	SQLConnection1
Tag	0

### cdsQry



dsQry1



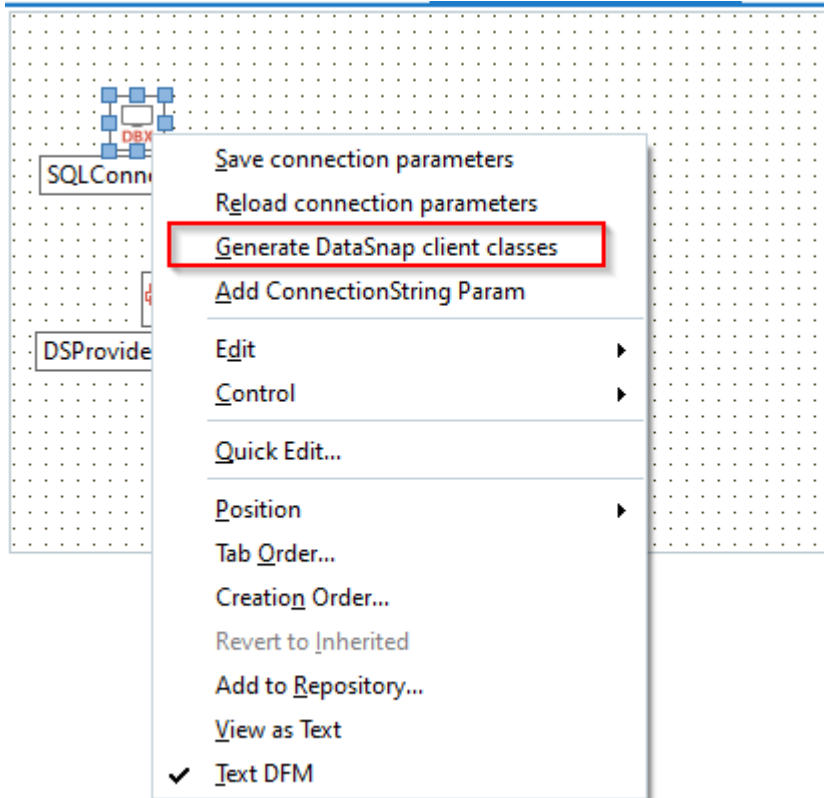


# ACTUALIZACION DE METODOS DEL SERVIDOR

martes, 10 de setiembre de 2024 12:36

Cada vez que se modifique los métodos de la aplicación servidor se debe generar nuevamente los métodos en la aplicación cliente.

Clic derecho sobre el componente SQLConecction



# CRUD

martes, 10 de setiembre de 2024 13:05

## Mantenimiento de Tabla Genérica

Form1

SELECT INSERT DELETE ACTUALIZAR

Ciaid

CiaDes

### SELECT

```
procedure TForm1.Button1Click(Sender: TObject);
var sSQL:string;
begin
  try
    sSQL:='SELECT * FROM TGE101 ORDER BY CIAID';
    ClientModule1Caja.cdsQry1.Close;
    ClientModule1Caja.cdsQry1.DataRequest(sSQL);
    ClientModule1Caja.cdsQry1.Open;
  except
    on E:exception do
      begin
        ShowMessage(E.Message);
      end;
    end;
  end;
end;
```

### INSERT

```

1 procedure TForm1.Button2Click(Sender: TObject);
2 var sError,sSQL:string;
3 begin
4
5     sSQL:='INSERT INTO TGE101 (CIAID,CIADES) VALUES ('+QuotedStr(edtCia.Text)+'+', '+QuotedStr(edtCiaDes.Text)+'')';
6
7
8     try
9     if ClientModule1Caja.ServerMethods1CajaClient.EjecutaData(sSQL,sError) then
10     begin
11         ShowMessage('Inserto ok');
12     end
13     else
14     begin
15         ShowMessage('No Inserto - '+sError);
16     end;
17 except
18     ShowMessage(sError);
19 end;
20
21 end;

```

#### DELETE

```

1 procedure TForm1.Button3Click(Sender: TObject);
2 var sError,sSQL:string;
3 begin
4     sSQL:='DELETE FROM TGE101 WHERE CIAID='+QuotedStr(edtCia.Text);
5
6     try
7     if ClientModule1Caja.ServerMethods1CajaClient.EjecutaData(sSQL,sError) then
8     begin
9         ShowMessage('Borro ok');
10     end
11     else
12     begin
13         ShowMessage('No Borro - '+sError);
14     end;
15 except
16     ShowMessage(sError);
17 end;
18
19 end;

```

#### ACTUALIZAR

```

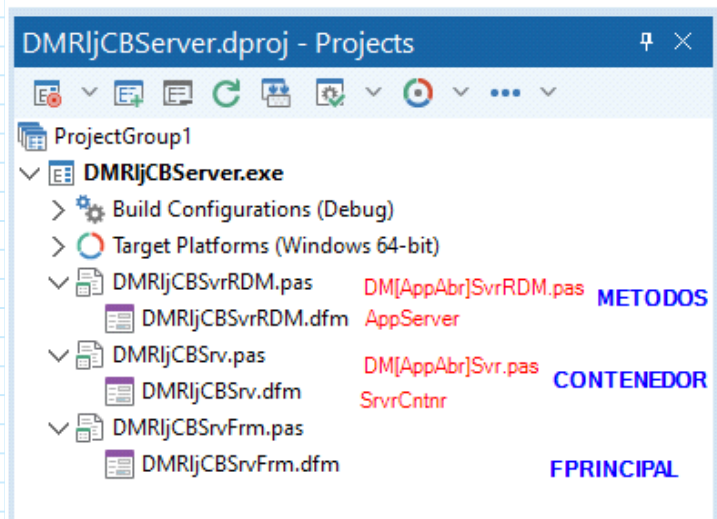
1 procedure TForm1.Button4Click(Sender: TObject);
2 var sError,sSQL:string;
3 begin
4     sSQL:='UPDATE TGE101 SET CIADES='+QuotedStr(edtCiaDes.text)+' WHERE CIAID='+QuotedStr(edtCia.Text);
5
6     try
7     if ClientModule1Caja.ServerMethods1CajaClient.EjecutaData(sSQL,sError) then
8     begin
9         ShowMessage('Actualizo ok');
10     end
11     else
12     begin
13         ShowMessage('No Actualizo - '+sError);
14     end;
15 except
16     ShowMessage(sError);
17 end;
18
19 end;

```

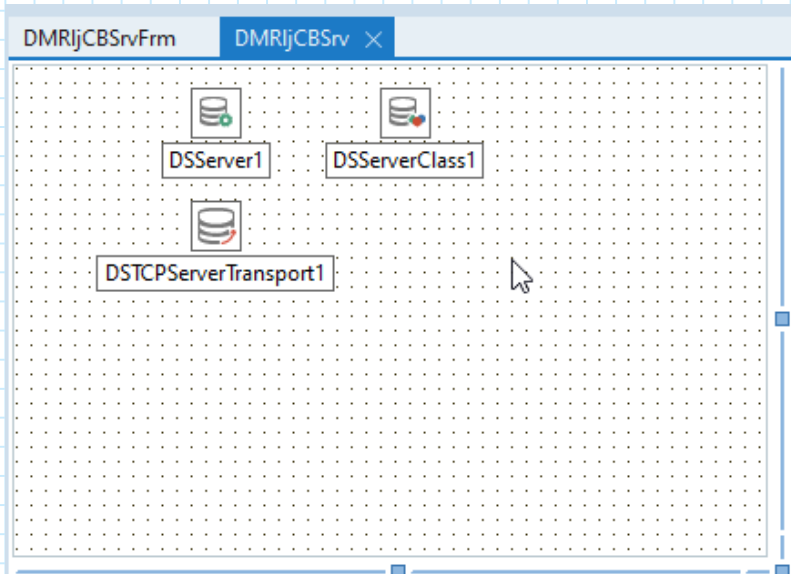
# SERVIDOR

jueves, 12 de setiembre de 2024 14:58

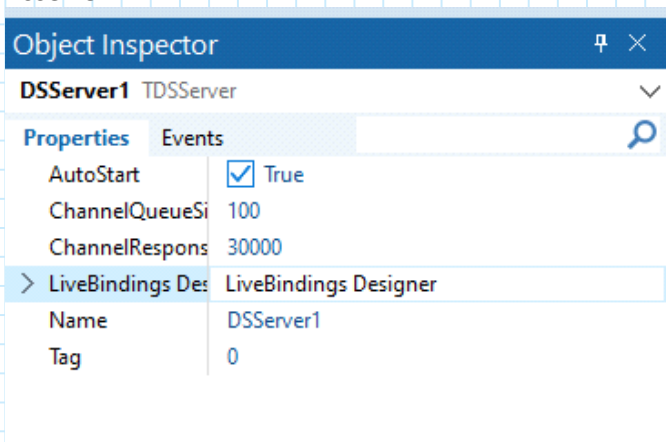
## Ficheros del Proyecto Servidor



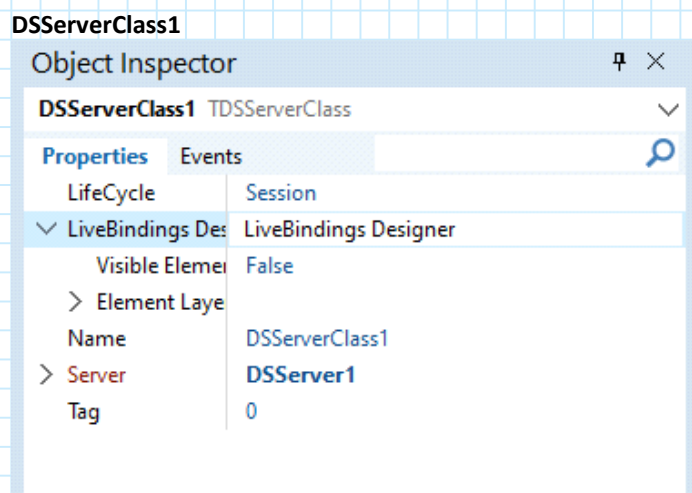
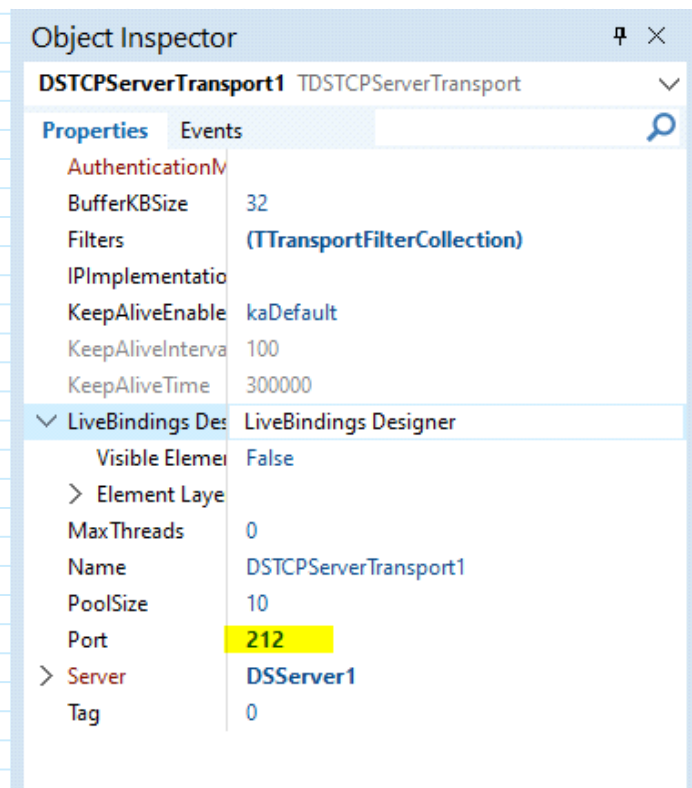
## Clases del Contenedor



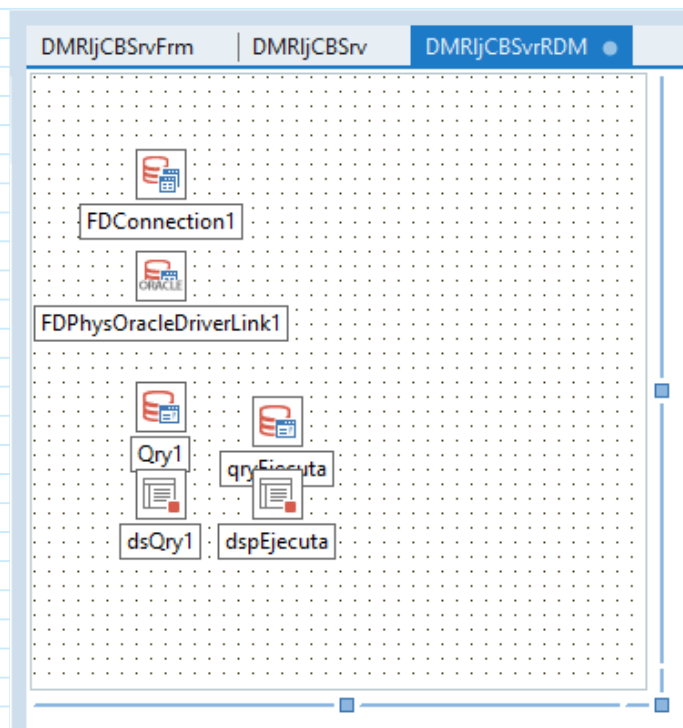
## DSServer1



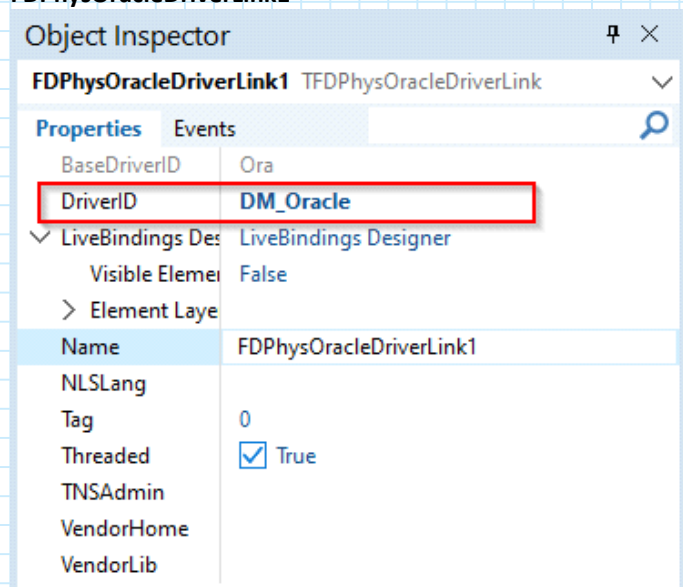
## DSTCPServerTransport1



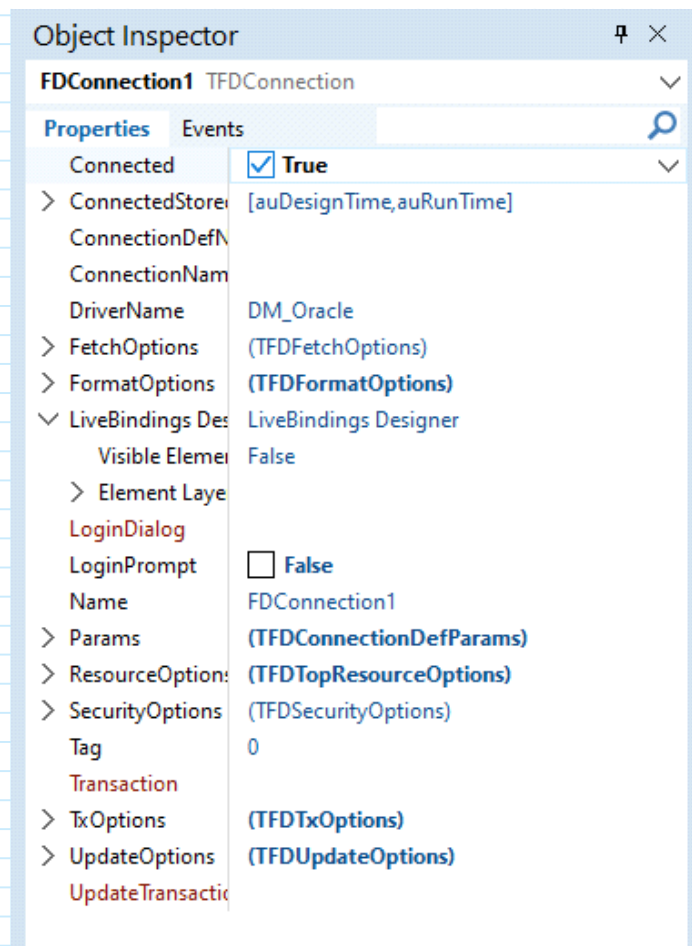
DMRIjCBSvrRDM



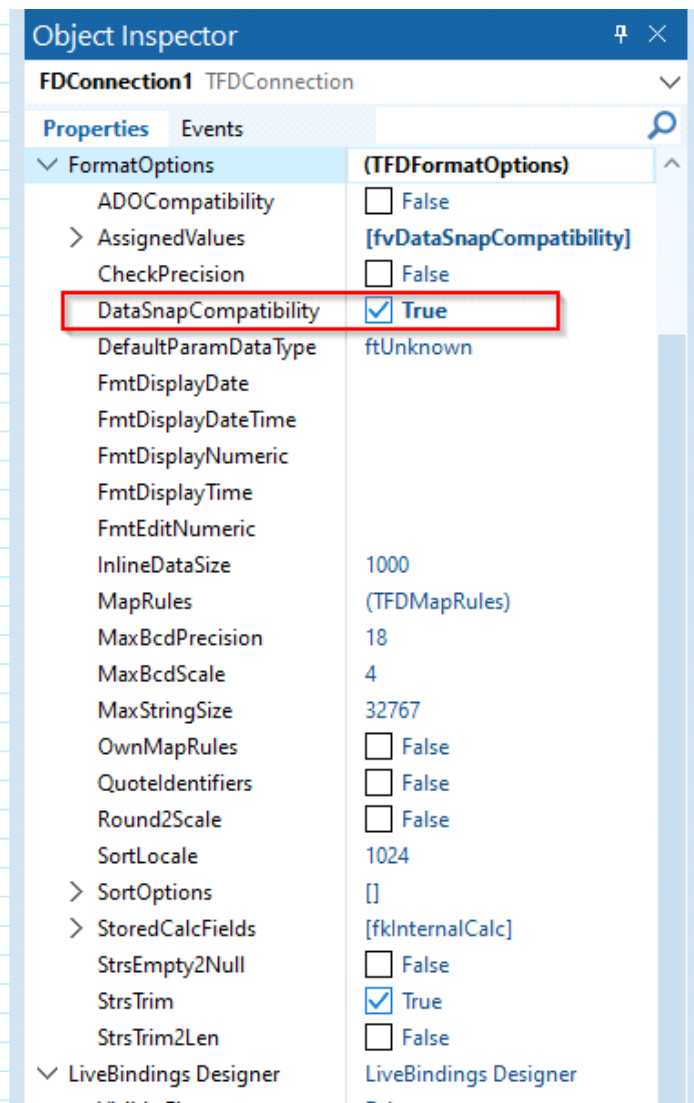
### FDPHysOracleDriverLink1



### FDConnection1



**FDCConnection1.FormatOptions**



FDCConnection1 Editor Connection



**FireDAC Connection Editor - [FDConnection1]**

Select driver or select connection definition name to override, then setup parameters

Definition Options Info SQL Script

Driver ID: DM\_Oracle

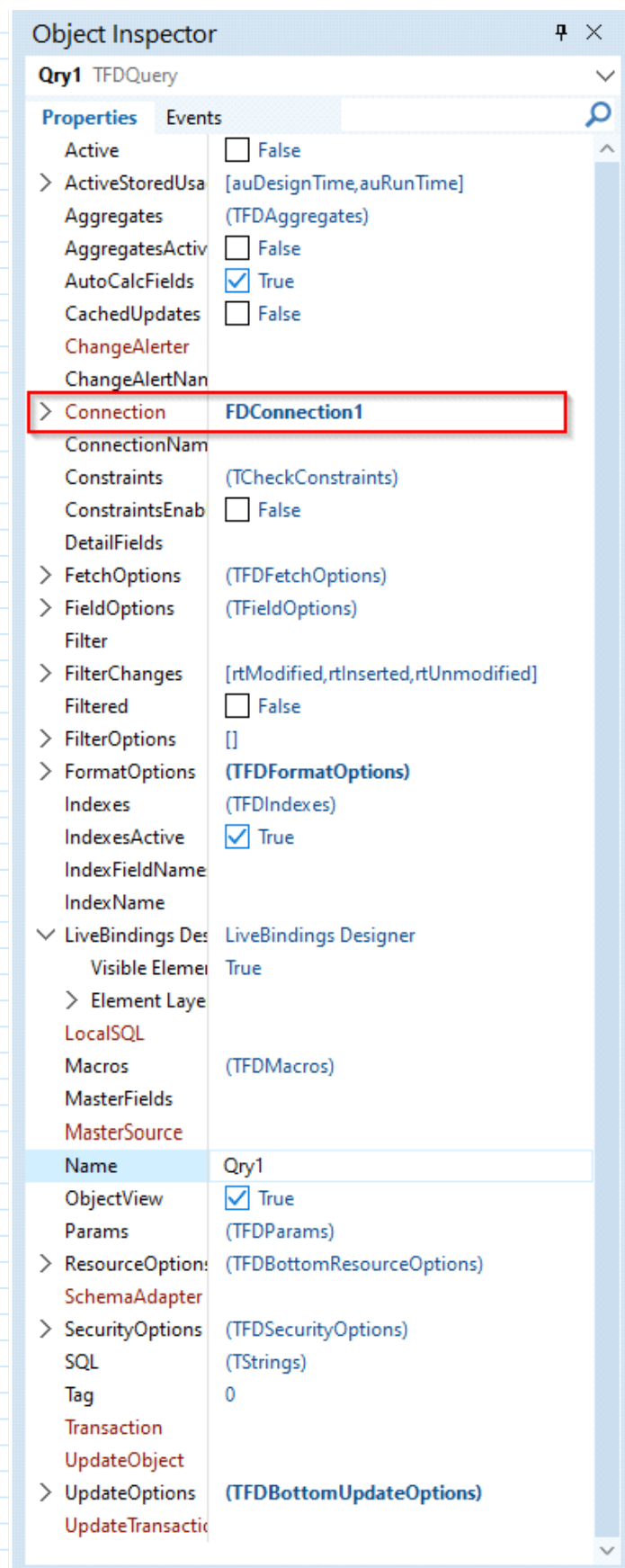
Connection Definition Name:

Test Wizard Revert To Defaults Help

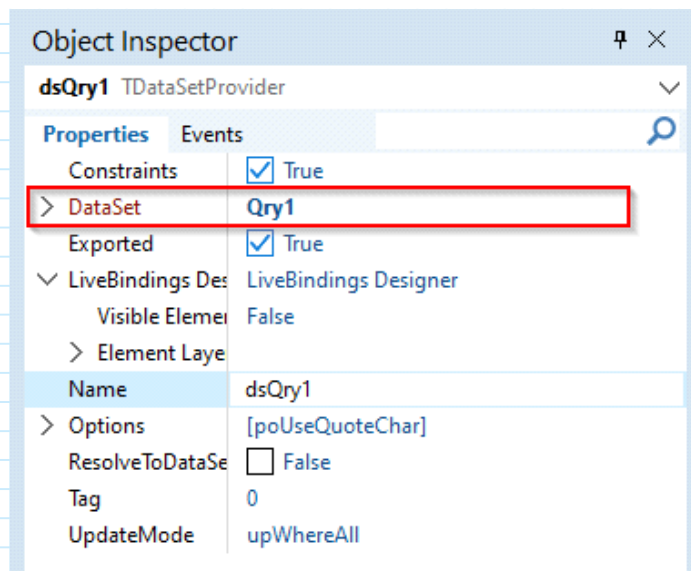
Parameter	Value	Default
DriverID	DM_Oracle	DM_Oracle
Pooled	False	False
Database	DEMADBTST01	
User_Name	DB2ADMIN	
Password	lider	
MonitorBy		
OSAuthent		
AuthMode	Normal	Normal
ReadTimeout		
WriteTimeout		
CharacterSet	<NLS_LANG>	<NLS_LANG>
NCharReplacement	True	True
StringFormat	Choose	Choose
BooleanFormat	Choose	Choose
ApplicationName		
OracleAdvanced		
MetaDefSchema		
MetaCurSchema		

OK Cancel

Qry1



dsQry1



## Métodos

Quitar los tipos de datos **widestring**

```

-
-   type
-   TAppServer = class(TDSServerModule)
-       FDCConnection1: TFDConnection;
-       FDFPhysOracleDriverLink1: TFDPhysOracleDriverLink;
20  Qry1: TFDQuery;
-       dsQry1: TDataSetProvider;
-       qryEjecuta: TFDQuery;
-       dspEjecuta: TDataSetProvider;
-       function DataRequest(Sender: TObject; Input: OleVariant): OleVariant;
-       procedure FDCConnection1AfterConnect(Sender: TObject);
-   private
-       { Private declarations }
-       xSuFijo: string;
-       xDB: string;
30  public
-       { Public declarations }
-       procedure ConexionOFF(var IUuario, IIdConex: String);
-       procedure ConexionON(var IUuario, IDirIP, IModulo, IIdConex, IFecHorCon, IIdPC: String);
-       procedure EjecutaSQL(var ISQL: String);
-       procedure GrabaTransaccion(var sSQL: string);
-       procedure IniciaTransaccion(var sSQL: String);
-       procedure RetornaTransaccion(var sSQL: String);
-       function setPassword(var IUuario, IPassword: String): String;
-       function Conectate(var IUuario, IPassword: String): String;
40  function EchoString(Value: string): string;
-       function ReverseString(Value: string): string;
-   end;
-

```

## Función Conectate

```

function TAppServer.Conectate(var IUsuario, IPasswor: String): String;
var
  IniFile: TIniFile;
begin
  try
    IniFile:=TIniFile.Create(ExtractFilePath(application.ExeName) + 'DM_Conf.ini');
    xDB := IniFile.ReadString('BD', 'BASE_DE_DATOS_RLJ', '');
    xSufijo := IniFile.ReadString('BD', 'SUFIJO', '');
    if Length(Trim(xDB)) <> 0 then
    begin
      try
        FDConnection1.Params.Values['USER_NAME'] := IUsuario;
        FDConnection1.Params.Values['PASSWORD'] := IPasswor + xSufijo;
        FDConnection1.Params.Values['DATABASE'] := xDB;
        FDConnection1.Params.Values['MULTIPLE TRANSACTION'] := 'True'; // para ejecutar Las transacciones
        FDConnection1.Connected := True;
        result := 'OK';
      except
        FDConnection1.Connected := False;
        result := 'ERROR';
      end;
    end
  else
  begin
    FDConnection1.Connected := False;
    result := 'ERROR';
    //ShowMessage('No existe ninguna Archivo INI que pueda reconocer');
  end;
except
  on E:Exception do
  begin
    result:=E.Message;
  end;
end;
end;

```

#### Función ConexionON

```

procedure TAppServer.ConexionON(var IUsuario, IDirIP, IModulo, IIIdConex,
  IFecHorCon, IIIdPC: String);
var
  vSQL: String;
begin
  vSQL := 'INSERT INTO ACCESOS_LOG(IDPC, IDEREG, APLCLIENTE, IPCLIENTE, USUCLIENTE, FECHORREGING) '
    + 'VALUES (' + QuotedStr(IIIdPc) + ',' + QuotedStr(IIIdConex) + ',' + QuotedStr(IModulo) + ','
    + QuotedStr(IDirIp) + ',' + QuotedStr(IUsuario) + ', SYSDATE )';
  EjecutaSQL(vSQL);

  FPrincipal.cdsAccesos.Insert;
  FPrincipal.cdsAccesos.FieldByName('USUCLIENTE').AsString := IUsuario;
  FPrincipal.cdsAccesos.FieldByName('IPCLIENTE').AsString := IDirIp;
  FPrincipal.cdsAccesos.FieldByName('FECHORREGING').AsString := IFecHorCon;
  FPrincipal.cdsAccesos.FieldByName('IDEREG').AsString := IIIdConex;
  FPrincipal.cdsAccesos.FieldByName('IDPC').AsString := IIIdPc;
  FPrincipal.cdsAccesos.Post;
  FPrincipal.dbgusuact.Refresh;
end;

```

#### Función ConexionOFF

```

procedure TAppServer.ConexionOFF(var IUusuario, IIdConex: String);
var
  vSQL: String;
begin
  vSQL := ' UPDATE ACCESOS_LOG '
    + ' SET FECHORREGOUT = SYSDATE '
    + ' WHERE USUCLIENTE = ' + QuotedStr(IUusuario)
    + ' AND IDEREG = ' + QuotedStr(IIdconex);

  EjecutaSQL(vSQL);

  IF FPrincipal.cdsAccesos.IndexFieldNames <> 'IDEREG;USUCLIENTE' THEN
    FPrincipal.cdsAccesos.IndexFieldNames := 'IDEREG;USUCLIENTE';
  FPrincipal.cdsAccesos.SetKey;
  FPrincipal.cdsAccesos.FieldName('IDEREG').AsString := IIdConex;
  FPrincipal.cdsAccesos.FieldName('USUCLIENTE').AsString := IUusuario;
  if FPrincipal.cdsAccesos.GotoKey then
    FPrincipal.cdsAccesos.Delete;
  FPrincipal.cdsAccesos.IndexFieldNames := '';
  FDConnection1.Connected := False;
end;

```

#### Función DataRequest (Para todos los TDataSetProvider)

```

function TAppServer.DataRequest(Sender: TObject; Input: OleVariant): OleVariant;
begin
  TFDQuery(TDataSetProvider(sender).DataSet).SQL.Text := Input;
end;

```

#### Función EjecutaSQL

```

150 procedure TAppServer.FDConnection1AfterConnect(Sender: TObject);
begin
  Qry1.ExecSQL('alter session set NLS_NUMERIC_CHARACTERS=','');
  Qry1.ExecSQL('alter session set nls_language= ''AMERICAN''');
  Qry1.ExecSQL('alter session set nls_territory= ''AMERICA''');
  Qry1.ExecSQL('alter session set nls_numeric_characters= ','');
  Qry1.ExecSQL('alter session set nls_date_format= ''DD-MM-YYYY''');
  Qry1.ExecSQL('alter session set nls_date_language= ''AMERICAN''');
  Qry1.ExecSQL('alter session set nls_time_format = ''HH.MI.SSXXFF AM''');
  Qry1.ExecSQL('alter session set nls_timestamp_format = ''DD-MON-RR HH.MI.SSXXFF AM''');
160 Qry1.ExecSQL('alter session set nls_time_tz_format = ''HH.MI.SSXXFF AM TZH:TZM''');
  Qry1.ExecSQL('alter session set nls_timestamp_tz_format = ''DD-MON-RR HH.MI.SSXXFF AM TZH:TZM''');
end;

```

#### Creación de Formularios

Project Options for DMRljCBServer.exe (Win64 - Debug)

Forms

Main form  
FPrincipal

Auto-create forms

FPrincipal  
SrvrCntnr

Available forms

AppServer

Save Cancel Help

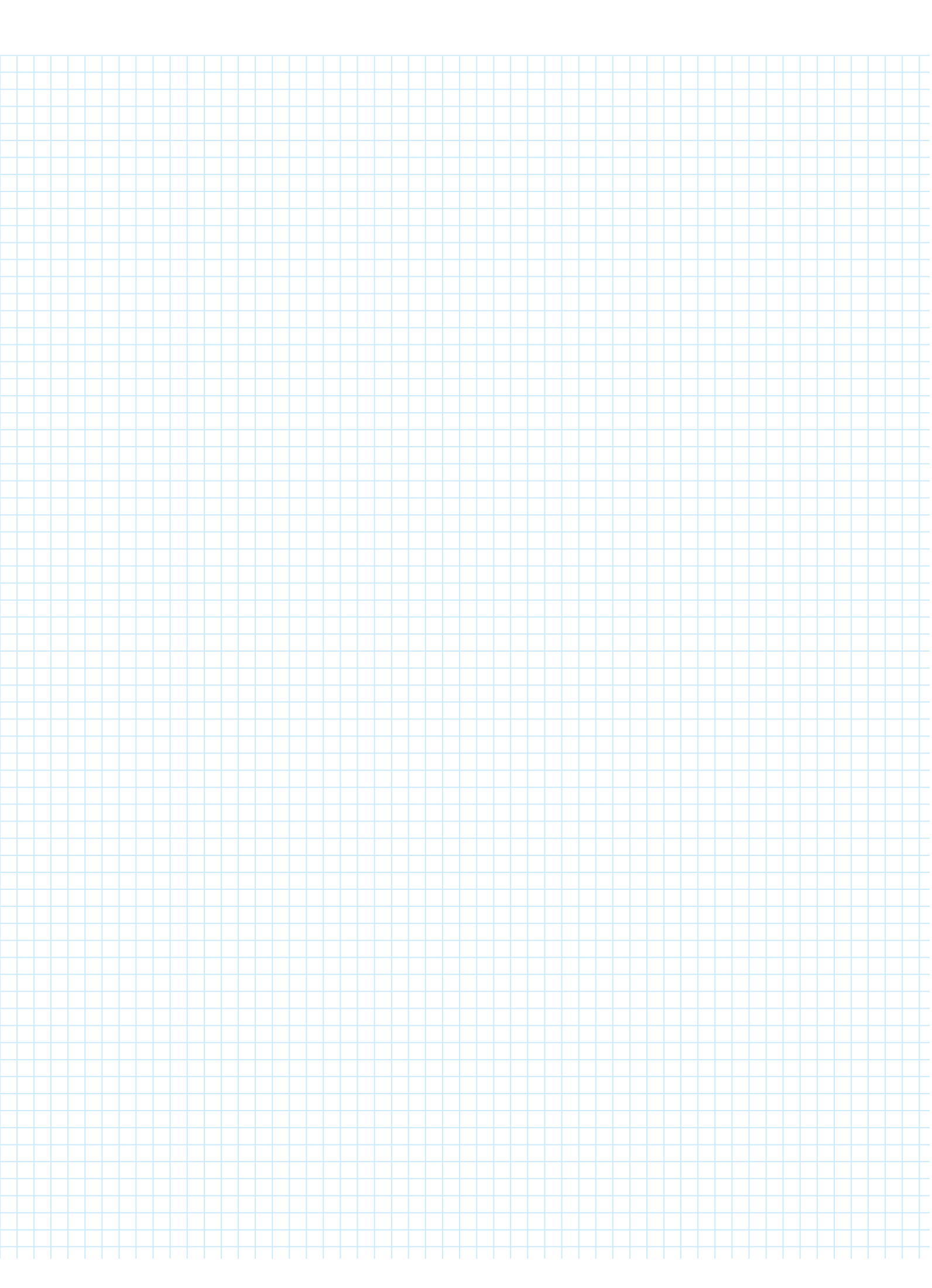
DMRljCBSrvFrm DMRljCBServer

Search for a type Search for a method

```

1 program DMRljCBServer;
.
.
. uses
.   Vcl.Forms,
.   Web.WebReq,
.   IdHTTPWebBrokerBridge,
.   DMRljCBSvrRDM in 'DMRljCBSvrRDM.pas' {AppServer: TDSServerModule},
.   DMRljCBSrv in 'DMRljCBSrv.pas' {SrvrCntnr: TDataModule},
.   DMRljCBSrvFrm in 'DMRljCBSrvFrm.pas' {FPrincipal};
10
.   {$R *.res}
.
. begin
.   Application.Initialize;
.   Application.MainFormOnTaskbar := True;
.   Application.CreateForm(TFPrincipal, FPrincipal);
.   Application.CreateForm(TSrvrCntnr, SrvrCntnr);
.   Application.Run;
. end.
20

```



# CLIENTE

jueves, 12 de setiembre de 2024 14:59



# CAMBIOS

miércoles, 18 de setiembre de 2024 10:23

## 1. Se añade Componente para Conexión DataSnap

```
TMant = class(TComponent)
private
    { Private declarations }
    FOnShow      : TNotifyEvent;
    FOnActivate  : TNotifyEvent;
    FOnInsert    : TNotifyEvent;
    FOnCierra    : TNotifyEvent;
    FOnCreateMant : TNotifyEvent;
    FOnDestroyMant : TNotifyEvent;

    FOnEdit      : TOnEdit;
    FOnDelete    : TOnDelete;

    FPopupMenu   : TPopupMenu;
    FModule      : String;
    FTableName   : String;
    FTabla2      : String;
    FTablaLLave  : String;
    FTabla2LLave : String;
    FTabla2Order : String;
    FAdmin       : String;
    FUser        : String;
    FTipo        : String;
    FFilter      : String;
    FTabla2Filter : String;
    FFilterObliga : Boolean;
    FInsertAutom : Boolean;
    FTitulo      : String;
    FColors      : TStrings;
    FNoVisible   : TStrings;
    FUserCheckB  : TStrings;
    FUsuarioSQL  : TStrings;
    FTituloFont  : TFont;
    FGrupoCols   : Integer;
    FNivel       : Integer;
    FRegistros   : Integer;

    FFileNameIni : string;
    FSectionName : string;

    FMultiColorRows,
    FMultiSelect : Boolean;
    FControlGridDisp : TControlGridDisp;
    FwwClientDataSet : TwwClientDataSet;
    FDComC        : TSQLConnection; TSocketConnection x TSQLConnection
    FDSPPCC       : TDSProviderConnection; Añadido

protected
    { Protected declarations }
    procedure SetPopupMenu ( value : TPopupMenu );
    procedure SetControlGridDisp(Value : TControlGridDisp);
    procedure SetwwClientDataSet(Value : TwwClientDataSet);
    procedure SetDComC(Value : TSQLConnection); TSocketConnection x TSQLConnection
    procedure SetDSPPCC(Value : TDSProviderConnection); Añadido
```

## 2. Propiedades del Componente

```

published
{ Published declarations }

property MultiColorRows: Boolean read FMultiColorRows write FMultiColorRows;
property Module : String read FModule write FModule;
property TableName : String read FTableName write FTableName;
property Admin : String read FAdmin write FAdmin;
property Filter : String read FFilter write FFilter;
property Tabla2Filter : String read FTabla2Filter write FTabla2Filter;
property FilterObliga : Boolean read FFilterObliga write FFilterObliga;
property InsertAutom : Boolean read FInsertAutom write FInsertAutom;
property Titulo : String read FTitulo write FTitulo;
property User : String read FUser write FUser;
property Tipo : String read FTipo write FTipo;
property GrupoCols : Integer read FGrupoCols write FGrupoCols;
property Registros : Integer read FRegistros write FRegistros;
property Nivel : Integer read FNivel write FNivel;
property TituloFont : TFont read GetTituloFont write SetTituloFont;
property Colors : TStrings read FColors write SetColors;
property NoVisible : TStrings read FNoVisible write SetNoVisible;
property UserCheckB : TStrings read FUserCheckB write SetUserCheckB;
property FileNameIni : string read FFileNameIni write FFileNameIni;
property SectionName : string read FSectionName write FSectionName;
property UsuarioSQL : TStrings read FUsuarioSQL write SetUsuarioSQL;
property TablaLLave : String read FTablaLLave write FTablaLLave;
property Tabla2 : String read FTabla2 write FTabla2;
property Tabla2LLave : String read FTabla2LLave write FTabla2LLave;
property Tabla2Order : String read FTabla2Order write FTabla2Order;

property PopupMenu : TPopupMenu read FPopupMenu write SetPopupMenu;
property MultiSelect : Boolean read FMultiSelect write FMultiSelect;
property ControlGridDisp: TControlGridDisp read FControlGridDisp write SetControlGridDisp;
property ClientDataSet : TwwClientDataSet read FwwClientDataSet write SetwwClientDataSet;
property DComC : TSQLConnection read FDComC write SetDComC; Modificado
property DSPCC : TDSProviderConnection read FDSPCC write SetDSPCC; Añadido
property OnInsert : TNotifyEvent read FOnInsert write FOnInsert;
property OnEdit : TOnEdit read FOnEdit write FOnEdit;
property OnCierra : TNotifyEvent read FOnCierra write FOnCierra;
property OnShow : TNotifyEvent read FOnShow write FOnShow;
property OnActivate : TNotifyEvent read FOnActivate write FOnActivate;
property OnDelete : TOnDelete read FOnDelete write FOnDelete;
property OnCreateMant : TNotifyEvent read FOnCreateMant write FOnCreateMant;
property OnDestroyMant : TNotifyEvent read FOnDestroyMant write FOnDestroyMant;

end;

```

### 3. Funciones de propiedades

```

- |
- | procedure TMant.SetDComC(Value : TSQLConnection);
- | begin
- |   FDComC := Value;
- |   if FDComC <> nil then
- |     begin
- |       FDComC.FreeNotification(Self);
440 |   end;
- | end;
- |
- | procedure TMant.SetDSPCC(Value : TDSProviderConnection);
- | begin
- |   FDSPCC := Value;
446 |   if FDSPCC <> nil then
- |     begin
- |       FDSPCC.FreeNotification(Self);
- |     end;
450 |   end;

```

### 4. Parámetro para el DataSetProviderConnection

```

result := FMant.Execute( FTableName, FModule,
                        FAdmin, FTipo, FUser,
                        FFilter, FTabla2Filter,
                        FFileNameIni, FSectionName,
                        FwwClientDataSet,
                        FDComC, FDSPCC ); Añadido

```

### 5. Definición de function TSolMantenimiento.Execute

```

- function TSolMantenimiento.Execute ( sTabla, sModulo, sAdmin, sTipo,
-                                     sUser, sFilter, sTabla2Filter,
-                                     sFile, sSection : String;
-                                     wcdsj : TwwClientDataSet;
-                                     wDCOM : TSQLConnection; Modificado
-                                     wDSPC : TDSPProviderConnection Añadido
680                                     ): Boolean;
-
- VAR
-     i, wExisteW, wGUID : Integer;
-     wsModulo, tSQL : String; wClassSrv : PChar; wResult : string;
685 begin
-     Screen.Cursor:= crHourGlass;
-     //cds1.RemoteServer:=wDCOM;
-     cds1.RemoteServer:=wDSPC; Modificado
690     wExisteW:=1;
-
-     wGUID:=1;
-
-     // if wsModulo='AD' then if wDCOM.ServerGUID='{454BF8F4-FDF3-4FF9-98A8-8B52FBAE1B10}' then else wGUID:=0;
-     // if wsModulo='IN' then if wDCOM.ServerGUID='{9E3C4406-FAAC-491D-95E1-08E1B5D1F85F}' then else wGUID:=0;
-     // if wsModulo='OP' then if wDCOM.ServerGUID='{685CCA99-C035-44DA-B56C-0B599F7B062A}' then else wGUID:=0;
-     // if sModulo='FAC' then if wDCOM.ServerGUID='{294DA083-D58B-11D3-B576-10005ABAB7A3}' then else wGUID:=0;
-     // if wsModulo='CC' then if wDCOM.ServerGUID='{0D9012AE-6188-49DA-8578-A6949CD9E92F}' then else wGUID:=0;
-     // if wsModulo='TE' then if wDCOM.ServerGUID='{BC18EC06-FCD4-42F7-B8C9-C8C0207C995D}' then else wGUID:=0;
750 // if wsModulo='CN' then if wDCOM.ServerGUID='{644F1B97-F4CE-4573-9621-AA90F16B4018}' then else wGUID:=0;
-     // if sModulo='ACF' then if wDCOM.ServerGUID='{294DA083-D58B-11D3-B576-10005ABAB7A3}' then else wGUID:=0;
-     // if sModulo='RRHH' then if wDCOM.ServerGUID='{294DA083-D58B-11D3-B576-10005ABAB7A3}' then else wGUID:=0;
-     // if sModulo='GRH' then if wDCOM.ServerGUID='{35946163-9315-11D4-A02F-0020AF63F8A3}' then else wGUID:=0;
-     // if wsModulo='PR' then if wDCOM.ServerGUID='{AA7DEC55-8A62-4140-9DFA-BD2AA1808129}' then else wGUID:=0;
-     // if sModulo='SEG' then if wDCOM.ServerGUID='{294DA083-D58B-11D3-B576-10005ABAB7A3}' then else wGUID:=0;
-     // if wsModulo='IG' then if wDCOM.ServerGUID='{1AA8B386-BFD7-49C3-9C1E-A886EB375070}' then else wGUID:=0;
-     //
-     // if wsModulo='RH' then if wDCOM.ServerGUID='{9D5F5C09-2F10-42D6-986D-3A1373AD3D66}' then else wGUID:=0;
-     // if wsModulo='APO' then if wDCOM.ServerGUID='{6893A1E6-2B7F-473E-8C71-1DE049589B23}' then else wGUID:=0;
760 // if wsModulo='CRE' then if wDCOM.ServerGUID='{EF8501A3-94AE-11D6-9A80-005004FC893C}' then else wGUID:=0;
-     // if wsModulo='COB' then if wDCOM.ServerGUID='{7A6C7183-94BF-11D6-9A80-005004FC893C}' then else wGUID:=0;
-     // if wsModulo='PRE' then if wDCOM.ServerGUID='{7A6C7188-94BF-11D6-9A80-005004FC893C}' then else wGUID:=0;
-     // if AnsiPos(wsModulo, 'AD,IN,OP,VE,CC,TE,CN,AF,PL,RH,PR,IG,APO,CRE,COB,PRE')=0 then wGUID:=0; Modificado
-
-     wDSPC.ServerClassName:='TAppServer';
-     wDSPC.SQLConnection:=wDCOM;
-
-     wGUID:=1;
-     if (wExisteW=0) or (wGUID=0) then
770 begin
-         ShowMessage( 'Consulte con su Proveedor OASIS');
-         Exit;
-     end;
-
-     ///
-     wTipoAp := sTipo;
-
-     if Length(sTabla)>0 then begin
-         { VHNXX Cambiado para que el filtro se abra con cds creados
780         cds2 := wcdsj;
-         cds2.Close;
-         cds2.IndexName:='';
-         }
-         cds2 := TwwClientDataSet.Create( Self );
-         cds2.ProviderName:='prvIndRef';
-         cds2.RemoteServer:= wDSPC; Modificado
-         cds2.OnReconcileError:=cdsRep2ReconcileError;
-         pprMant.Free;
-         ppdbMant.Free;
790     end
-

```

```

else begin
    if wTipoAp='HELP' then
    begin
        cds2:=wcdsj;
        cds2.Close;
        cds2.IndexName:='';
    end
    else
    begin
800         cds2 := TwwClientDataSet.Create( Self );
        cds2.ProviderName:='prvIndRef';
        cds2.RemoteServer:= wDSPC;
        cds2.OnReconcileError:=cdsRep2ReconcileError;
    end;
    scTablas:= TStrContainer.Create(self);
end;

cdsBusca.RemoteServer :=wDSPC; Modificado
cdsReporte.RemoteServer:=wDSPC; Modificado
810 cdsRep2.RemoteServer :=wDSPC; Modificado
CDSGrid.RemoteServer :=wDSPC; Modificado
CDSGrid2.RemoteServer :=wDSPC; Modificado
//DCOMT :=wDCOM;
DSPCT :=wDSPC; Añadido
cdsUsuSi.RemoteServer :=wDSPC; Modificado
cdsUsuNo.RemoteServer :=wDSPC; Modificado
cdsPlan.RemoteServer :=wDSPC; Modificado
cdsFiltro2.RemoteServer:=wDSPC; Modificado
cdsQryT.RemoteServer :=wDSPC; Modificado
820

```

```

// vhdema
// xxDirPub := Trim(cdsBusca.FieldByName('SRV_RUTA').AsString);
// xxDirPub := '\\'+wDCOM.ComputerName+'C';
xxDirPub := '\\'+wDCOM.Params.Values['HostName']; Modificado
xxDirLoc := 'C:';
xxServer := 'C:';
// xxDirect := 'SOL';
xxDirect := '\\oaUser';
xxDirMod := '\\'+wModulo;
xxDirPrn := '\\'+wSection;

wPacketData :=30;
wPacketCombo:=100;

```

#### 6. Llamada a componente

```

Var
    MRefe: TMant;
begin
    Try
        MRefe := TMant.Create(Application);
        MRefe.Module := 'CNT';
        MRefe.Admin := 'N';
        //MRefe.DComC := DMCNT.DCOM1;
        MRefe.DComC := DCOM1;
        MRefe.DSPCC := DSPProviderConnection1; Añadido
        MRefe.User := 'HUGO';
        MRefe.Execute;
    Except
    End;
end;

```

#### 7. Errores en tiempo de Ejecución

Buscar y cambiar todas las llamadas de este tipo `xTitLabel[ Pos('~',xTitLabel)] := ''`; que no tengan condicional antes

```

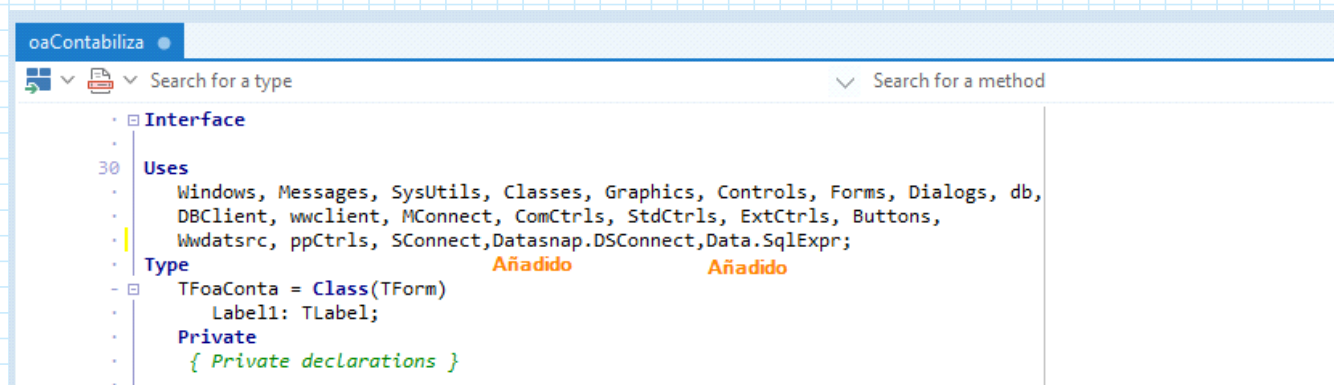
procedure TSoIMantenimiento.dbgFiltroTitleButtonClick(Sender: TObject; AFieldName: String);
var
    xTitLabel : String;
begin
77     pnlBusca.Visible := True;
    isBusca.SearchField := AFieldName;
    dbgFiltro.SetActiveField( AFieldName );
30     cds2.IndexFieldNames:=AFieldName;
    xTitLabel := dbgFiltro.SelectedField.DisplayLabel;
    if Pos('~',xTitLabel)>0 then Añadido
        xTitLabel[ Pos('~',xTitLabel)] := ' ';
    lblBusca2.caption := xTitLabel;
    isBusca.SetFocus;
end;

```

# CAMBIOS

miércoles, 18 de setiembre de 2024 12:19

## 1. Componentes de Conexion



### Var

```
FoaConta: TFoaConta;
s_vgUsuario: String;
s_vgPassword: String;
iOrden: integer;
wReplaCeros: String;
DCOM_C: TSQLConnection; Modificado
DSPC_C: TDSPProviderConnection; Añadido
cdsNivel_C: TwwClientDataSet;
cdsPresup_C: TwwClientDataSet;
```

## 2. Funciones

```
Function SOLDesConta(xxxCia, xxxDiario, xxxAnoMM, xxxNoComp, xSRV: String;
cdsNivelx: TwwClientDataSet;
DCOMx: TSQLConnection; DSPCx: TDSPProviderConnection; xForm_C: TForm): Boolean;
Var Modificado Añadido
xSQL: String;
Begin
160 SRV_C := xSRV;
CNTDet := 'CNT301';
Provider_C := 'dspTem6';
DCOM_C := DCOMx; Modificado
DSPC_C := DSPCx; Añadido
DSPC_C.SQLConnection:=DCOM_C; Añadido

cdsNivel_C := cdsNivelx;

cdsQry_C := TwwClientDataSet.Create(Nil);
170 cdsQry_C.RemoteServer := DSPCx; Modificado
cdsQry_C.ProviderName := Provider_C;
```

```
480
cdsCNT := TwwClientDataSet.Create(Nil);
cdsCNT.RemoteServer := DSPCx; Modificado
483 cdsCNT.ProviderName := Provider_C;
xSQL := 'select * '
+ ' from CNT311 '
+ ' where CIAID=' + quotedstr(xCia)
+ ' and TDIARID=' + quotedstr(xTDiario)
+ ' and CNTANOMM=' + quotedstr(xAnoMM)
+ ' and CNTCOMPROB=' + quotedstr(xNoComp)
490 + ' order BY CNTREG';
cdsCNT.Close;
cdsCNT.DataRequest(xSQL);
cdsCNT.Open;

xNoCompP := xNoComp;

cdsQry_G := TwwClientDataSet.Create(Nil);
cdsQry_G.RemoteServer := DSPCx; Modificado
cdsQry_G.ProviderName := Provider_C;
500
```

```

720 cdsClone := TwwClientDataSet.Create(Nil);
    cdsClone.RemoteServer := DSPC; Modificado
    cdsClone.ProviderName := Provider_C;
    cdsClone.Close;

```

```

    cdsClone := TwwClientDataSet.Create(Nil);
    cdsClone.RemoteServer := DSPC; Modificado
1349 cdsClone.ProviderName := Provider_C;
1350 cdsClone.Close;

```

```

Function SOLConta(xCia, xTDiario, xAnoMM, xNoComp, xSRV, xTipoC, xModulo: String;
    cdsMovCNT, cdsNivelx, cdsResultSetx: TwwClientDataSet;
    DCOMx: TSQLConnection; DSPC: TDSProviderConnection; xForm_C: TForm): Boolean;
Var Modificado Añadido
    sSQL, xNREG, xSQL, xCajaAut: String;
    xNumT, iOrdenx: Integer;
    sCIA, sCuenta, sDeHa: String;
    dDebeMN, dHabeMN, dDebeME, dHabeME: double;
    xTotDebeMN, xTotHabeMN, xTotDebeME, xTotHabeME, xDif: Double;
1740 cdsClone: TwwClientDataSet;
    xxModulo: String;
Begin
    If (xTipoC = 'P') Or (xTipoC = 'C') Or (xTipoC = 'BP') Or (xTipoC = 'CCNA') Or (xTipoC = 'PCNA') Then
    Begin
        CNTDet := 'CNT311';
        If xTipoC = 'P' Then
            CNTCab := 'CNT310'
        Else
            CNTCab := 'CNT300';
1750 End
    Else
    Begin
        // Para Mayorización
        CNTCab := 'CNT300';
        CNTDet := 'CNT301';
    End;

    FoaConta.CreaPanel(xForm_C, 'Contabilizando');
1760 DCOM_C := DCOMx;
    DSPC_C := DSPC; Añadido
    DSPC_C.SQLConnection := DCOM_C; Añadido

    SRV_C := xSRV;
    xTipoC_C := xTipoC;

    If (SRV_C = 'DB2NT') Or (SRV_C = 'DB2400') Then
    Begin
1770 wReplaCeros := 'COALESCE';
    End
    Else
    Begin
        If SRV_C = 'ORACLE' Then
        Begin
            wReplaCeros := 'NVL';
        End;
    End;

    Provider_C := 'dspTem6';

1780 cdsNivel_C := cdsNivelx;
    cdsResultSet_C := cdsResultSetx;

    cdsQry_C := TwwClientDataSet.Create(Nil);
    cdsQry_C.RemoteServer := DSPC; Modificado
    cdsQry_C.ProviderName := Provider_C;

    If copy(xAnoMM, 5, 2) <> '13' Then
    Begin
1940 FoaConta.PanelMsg('Generando Asientos Automaticos', 0);

        // GENERA ASIENTOS AUTOMATICOS PARA LA CUENTA 1

        cdsClone := TwwClientDataSet.Create(Nil);
        cdsClone.RemoteServer := DSPC; Modificado
        cdsClone.ProviderName := Provider_C;
        cdsClone.Close;

```



```

• Procedure TFoaConta.GeneraEnLinea401(xxxCia, xxxDiario, xxxAnoMM, xxxNoComp, xSuma: String);
• Var
•   xCtaPrin, xClAux, xCuenta, xAuxDes, xAno, xMes, xDH, xSQL, xSQLn: String;
•   xMov, xAux, xCCos, xCCoDes, xCtaDes, xFLAux, xFLCCo, xNivel, xNREG: String;
•   xDigitos, xDigAnt, xNumT: Integer;
•   xImpMN, xImpME: Double;
•   cdsMovCNT2: TwwClientDataSet;
•   cdsQry2x: TwwClientDataSet;
2550  cdsTge202x: TwwClientDataSet;
•   cdsTge202xxx: TwwClientDataSet;
•   cAno, cMes, cMesA, fLDolar: String;
• Begin
•   xAno := Copy(xxxAnoMM, 1, 4);
•   xMes := Copy(xxxAnoMM, 5, 2);
•
•   FoaConta.PanelMsg('Actualizando Saldos...', 0);
•
•   cdsQry2x := TwwClientDataSet.Create(Nil);
2560  cdsQry2x.RemoteServer := DSPC_C; Modificado
•   cdsQry2x.ProviderName := Provider_C;
•
•   xSQL := 'Select CNTSOLODOLAR from TGE101 Where CIAID=' + quotedstr(xxxCia);
•   cdsQry2x.Close;
•   cdsQry2x.DataRequest(xSQL); // Llamada remota al proveedor del servidor
•   cdsQry2x.Open;
•
•   cdsMovCNT2 := TwwClientDataSet.Create(Nil);
•   cdsMovCNT2.RemoteServer := DSPC_C; Modificado
•   cdsMovCNT2.ProviderName := Provider_C;
•   cdsMovCNT2.Close;
•   cdsMovCNT2.DataRequest(xSQL);
•   cdsMovCNT2.Open;
2630  {
•
•   cdsTge202x := TwwClientDataSet.Create(Nil);
2640  cdsTge202x.RemoteServer := DSPC_C; Modificado
2642  cdsTge202x.ProviderName := Provider_C;
•   xSQL := 'Select CUENTAID, CTANIV, CTAABR, CTA_MOV, CTA_ME from TGE202 Where CIAID=' + quotedstr(xxxCia);
•   cdsTge202x.Close;
•   cdsTge202x.DataRequest(xSQL);
•   cdsTge202x.Open;
•   cdsTge202x.IndexFieldNames := 'CUENTAID;CTANIV';
•
• Procedure TFoaConta.GeneraAsientosGlobal(xCia, xDiario, xAnoMM, xNoComp, xTCP: String; cdsMovCNT: TwwClientDataSet);
• Var
•   cdsQryT: TwwClientDataSet;
•   xSQL, xWhere: String;
4340  xCtaCaja: String;
•   sDeHa: String;
•   dHabeMN, dHabeME, dDebeMN, dDebeME: Double;
•   nReg: Integer;
• Begin
•   // Números de Comprobantes Nuevos
•   // Verifica si ya tiene comprobantes
•   xWhere := 'SELECT ECPERREC FROM CAJA302 '
•           + 'WHERE CIAID=' + '' + xCia + ''
•           + ' and TDIARID=' + '' + xDiario + ''
4350  + ' and ECANOMM=' + '' + xAnoMM + ''
•           + ' and ECNOCOMP=' + '' + xNoComp + '';
•   cdsQryT := TwwClientDataSet.Create(Nil);
4353  cdsQryT.RemoteServer := DSPC_C; Modificado
•   cdsQryT.ProviderName := Provider_C;
•
•   cdsQryT.Close;
•   cdsQryT.DataRequest(xWhere);
•   cdsQryT.Open;

```

```

Procedure TfoaConta.GeneraAsientosComplementarios(xCia, xDiario, xAnoMM, xNoComp, xTCP: String; cdsMovCNT: Tw
Var
    cdsQryT: TwwClientDataSet;
    xSQL, xWhere: String;
    xCtaCaja: String;
5020 sDeHa: String;
    dHabeMN, dHabeME, dDebeMN, dDebeME: Double;
    nReg: Integer;
Begin
    // Números de Comprobantes Nuevos
    // Caja Autonoma
    If (SRV_C = 'DB2NT') Or (SRV_C = 'DB2400') Then
    Begin
        xWhere := 'SELECT COALESCE( MAX( CNTCOMPROB ), ''0'' ) AS NUMERO FROM CNT300 '
5030 + 'WHERE CIAID=' + '''' + xCiaOri + ''''
        + ' and TDIARID=' + '''' + xOrigen2 + ''''
        + ' and CNTANOMM=' + '''' + xAnoMM + '''';
    End;

    If SRV_C = 'ORACLE' Then
    Begin
        xWhere := 'SELECT NVL( MAX( CNTCOMPROB ), ''0'' ) AS NUMERO FROM CNT300 '
        + 'WHERE CIAID=' + '''' + xCiaOri + ''''
        + ' and TDIARID=' + '''' + xOrigen2 + ''''
5040 + ' and CNTANOMM=' + '''' + xAnoMM + '''';
    End;

    // Verifica si ya tiene comprobantes
    xWhere := 'SELECT ECPERREC FROM CAJA302 '
    + 'WHERE CIAID=' + '''' + xCia + ''''
    + ' and TDIARID=' + '''' + xDiario + ''''
    + ' and ECANOMM=' + '''' + xAnoMM + ''''
    + ' and ECNOCOMP=' + '''' + xNoComp + '''';

5050 cdsQryT := TwwClientDataSet.Create(nil);
5051 cdsQryT.RemoteServer := DSPC_C; Modificado
    cdsQryT.ProviderName := Provider_C;

```

```

5820 Function SOLPresupuesto(xCia, xUsuario, xNumero, xSRV, xModulo: String;
    cdsResultSetx: TwwClientDataSet;
    DCOMx: TSQLConnection; DSPCx:TDSProviderConnection; xForm_C: TForm; xTipoMay: String): Boolean;
    Var
        Modificado
        Añadido
        sSQL, xNREG, xSQL, xCajaAut, xSQL1: String;
        xNumT, iOrdenx: Integer;
        sCIA, sCuenta, sDeHa: String;
        dDebeMN, dHabeMN, dDebeME, dHabeME: double;
        cdsClone: TwwClientDataSet;
        cdsAsiento: TwwClientDataSet;
5830 Begin
    CNTDet := 'PPRES311';

    FoaConta.CreaPanel(xForm_C, 'Generando Presupuestos');

    DCOM_C := DCOMx;
    DSPC_C := DSPCx; Añadido
5837 DSPC_C.SQLConnection := DCOM_C; Añadido

    SRV_C := xSRV;

    cdsPresup_C := TwwClientDataSet.Create(nil);
    cdsPresup_C.RemoteServer := DSPCx; Modificado
5860 cdsPresup_C.ProviderName := 'dspTem5';

    cdsQry_C := TwwClientDataSet.Create(nil);
    cdsQry_C.RemoteServer := DSPCx; Modificado
    cdsQry_C.ProviderName := Provider_C;

    cdsClone := TwwClientDataSet.Create(nil);
5900 cdsClone.RemoteServer := DSPCx; Modificado
    cdsClone.ProviderName := Provider_C;
    cdsClone.Close;

    cdsAsiento := TwwClientDataSet.Create(nil);
    cdsAsiento.RemoteServer := DSPCx; Modificado
    cdsAsiento.ProviderName := 'dspTem4';

```



```

* Procedure TFoaConta.GeneraMayorPresupuestos(xxxCia, xxxUsuario, xxxNumero, xSuma: String);
* Var
*   xCtaPrin, xTipPres, xTipoCol, xAnoMM, xClAux, xCuenta, xAuxDes, xAno, xMes, xDH, xSQL: String;
6170 xMov, xAux, xCCos, xCCoDes, xCtaDes, xFLAux, xFLCCo, xNivel, xNREG: String;
*   xDigitos, xDigAnt, xNumT, xContr: Integer;
*   xImpMN, xImpME: Double;
*   cdsQry2x: TwwClientDataSet;
*   cdsNivel_C: TwwClientDataSet;
*   cAno: String;
*   cMes: String;
*   cMesA: String;
* Begin
*   FoaConta.PanelMsg('Actualizando Saldos...', 0);
6180
*   xSQL := 'Select * from PPRES103 Order by PARPRESNIV';
*   cdsNivel_C := TwwClientDataSet.Create(nil);
*   cdsNivel_C.RemoteServer := DSPC_C; Modificado
*   cdsNivel_C.ProviderName := Provider_C;
*   cdsNivel_C.Close;
*   cdsNivel_C.DataRequest(xSQL);
*   cdsNivel_C.Open;
*
*   cdsQry2x := TwwClientDataSet.Create(nil);
6190 cdsQry2x.RemoteServer := DSPC_C; Modificado
*   cdsQry2x.ProviderName := Provider_C;
*
*
*   cdsMovPRE2 := TwwClientDataSet.Create(nil);
6220 cdsMovPRE2.RemoteServer := DSPC_C; Modificado
*   cdsMovPRE2.ProviderName := Provider_C;
*   cdsMovPRE2.Close;
*   cdsMovPRE2.DataRequest(xSQL);
*   cdsMovPRE2.Open;

```

# MANTENIMIENTO

miércoles, 18 de setiembre de 2024 10:14

En function **TSolMantenimiento.Execute** variable **wRutaCds**

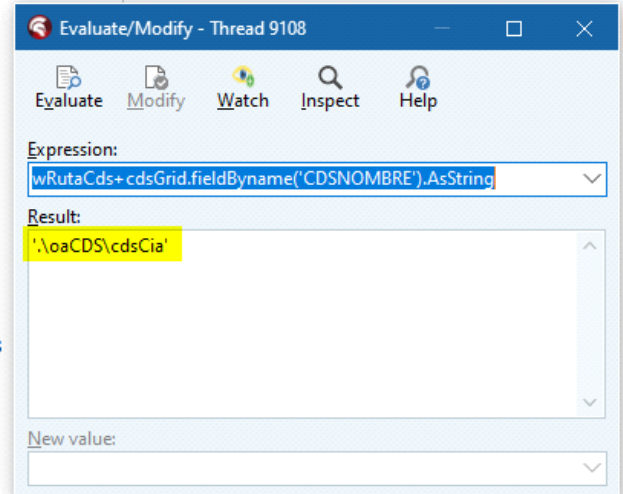
```
if DirectoryExists( 'C:\WINNT\SYSTEM32' ) then
  wRutaCds:='C:\WINNT\SYSTEM32\'
else
  wRutaCds:='C:\WINDOWS\SYSTEM\'
wRutaCds:='.\\oaCDS\';
```

```
procedure TSolMantenimiento.ActualizaTablas;
var
  i : Integer;
  tSQL, tSQL2 : String;
begin
  tSQL:='Select * from TGE008';
  cdsGrid.Close;
80  cdsGrid.IndexFieldNames:='';
  cdsGrid.Filter := '';
  cdsGrid.Datarequest( tSQL );
  cdsGrid.Open;

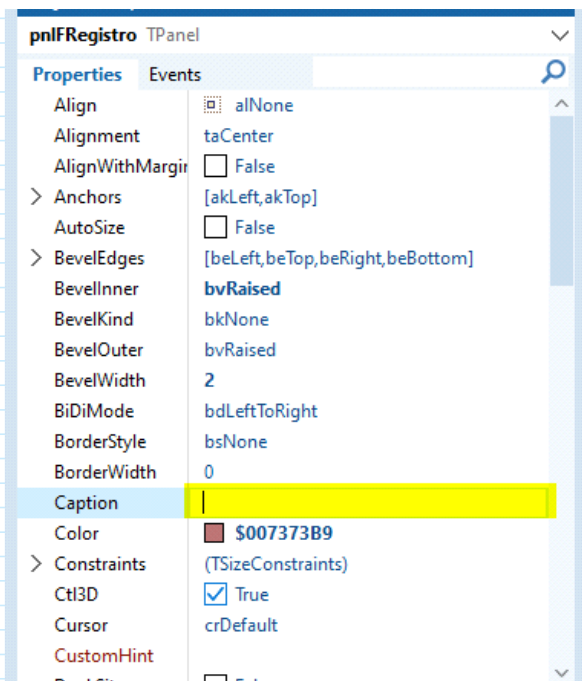
  if scTablas.Lines.Count>0 then
  begin
    for i:=0 to scTablas.Lines.Count-1 do
    begin
90      cdsGrid.Filtered:=False;
      cdsGrid.Filter := '';
      cdsGrid.Filter := 'TABLANOMBRE='' +scTablas.Lines[ i ]+'''';
      cdsGrid.Filtered:=True;

      if cdsGrid.RecordCount>0 then
      begin
        while not cdsGrid.EOF do
        begin
          cdsBusca.Close;
          cdsBusca.DataRequest( cdsGrid.fieldByName('SQLSTRING').AsString );
          cdsBusca.Open;
92      cdsBusca.SaveToFile( wRutaCds+cdsGrid.fieldByName('CDSNOMBRE').AsString );
          cdsGrid.Next;
        end;

        tSQL2:='Insert Into TGE009( USERID, TABLANOMBRE ) '
          + 'SELECT USERID, '''+scTablas.Lines[ i ]+''' FROM ( select USERID FROM TGE006 GROUP BY USERID ) A '
          + 'where not exists ( select USERID from TGE009 B WHERE A.USERID=B.USERID )';
        cdsBusca.Close;
10      cdsBusca.DataRequest( tSQL2 );
        cdsBusca.Execute;
      end;
    end;
  end;
```



1. Unit **SOLRef01** panel **pnlFRegistro**



# oaSISPk8.bpl

lunes, 23 de setiembre de 2024

08:22

- Quitar todos los componentes antiguos de Infopower y Reportbuilder y BDE
- Quitar la unit DBTables
- Quitar la unit DBXpress
- Wwtable -> Vcl.Wwtable
- TppCustomPageSetupDialog -> Añadir ppDesignerForms en el uses

# CNTPK00.BPL

lunes, 23 de setiembre de 2024

09:11

- Quitar referencias a units antiguos (Infopower, ReporBuilder, BDE)

# oaLIBConta.bpl

lunes, 23 de setiembre de 2024

09:17

- Quitar referencias a units antiguos (Infopower, ReporBuilder, BDE)
- Quitar unit de uses Excel2000, Excel\_2K\_SRVR;
- Quitar unit de uses Qt,
- Cambiar funcion ComputerName

```
function TDMPL.ComputerName: String;
begin
  var Size: Cardinal := 0;
  if (not GetComputerName(PChar(Result), Size)) and (GetLastError <> ERROR_BUFFER_OVERFLOW) then
    RaiseLastOSError;
  SetLength(Result, Size-1);
  if (not GetComputerName(PChar(Result), Size)) then
    RaiseLastOSError;
end;
```
- Cambiar PChar(string) -> PAnsiChar(AnsiString(string))
- Cambiar Bk: TBookmarkStr -> Bk: Tbookmark
- Cambiar TTransactionDesc -> TD: TDBXTransaction (uses Data.DBXCommon)
- IniciaTransaccion: TD := DBX\_Connect1.BeginTransaction(TDBXIsolations.ReadCommitted);
- GrabaTransaccion: DBX\_Connect1.CommitFreeAndNil(TD);
- RetornaTransaccion: DBX\_Connect1.RollbackFreeAndNil(TD);