

VISUAL BASIC – Diseño de Formularios y manipulación de Datos - Proyecto DBGrid

Este trabajo está asociado a un proyecto de Visual Basic llamado DBGrid y su objetivo es mostrar en forma teórica y práctica cómo funciona el control DBGrid asociado al control Data y a otros controles habituales para captura de datos.

Para todos los formularios hay que tener habilitado el control DBGrid que es parte del componente **Microsoft Data Bound Grid Controls (6.0 SP6)**. Menú Herramientas/Componentes

El control DBGrid está ligado a datos y es similar al estándar Grid sólo que al estar ligado a datos tiene la propiedad DataSource que vincula al DBGrid a un Control Data, para que la cuadrícula se llene en forma automática tanto en cuanto a los registros como a los encabezados de columna en base al Recordset del control Data.

Sin embargo veremos también que este control puede no estar vinculado al control Data en tiempo de diseño y sí establecerse en tiempo de ejecución.

Este control sirve para mostrar un conjunto de datos, ya sea de una tabla o de una consulta. En la vista propiedades se puede establecer a verdadero que el control permita agregar registros y efectuar actualizaciones.

Iremos desarrollando los distintos ejemplos de menor a mayor complejidad.

Formulario DBGrid1

Este formulario consta de un control Data llamado **datCargos**, un control DBGrid llamado **DBGrid1** un array de controles de tipo cuadro de texto que hemos llamado **txtEntrada** y un array de etiquetas a las que hemos llamado **lblRotulos**.

Su objetivo es mostrar cómo están relacionados unos controles con otros. Al DBGrid1 lo hemos vinculado mediante la propiedad **DataSource** con el control Data lo mismo que a cada elemento del array txtEntrada. En este caso también hemos establecido el **DataField**, es decir la columna del control Data a la que se va a vincular cada cuadro de texto que forma el txtEntrada.

Entonces si nos desplazamos por el control datCargos veremos que se mueve el registro de puntero de DBGrid1 y que cambia el contenido del texto de txtEntrada. Si nos movemos por el DBGrid1 veremos que se modifica el caption del control Data (que ha sido realizado mediante un procedimiento) y también cambia el contenido de txtEntrada.

El código utilizado es mínimo: simplemente el necesario para mostrar el registro absoluto que representa un registro particular en la tabla Cargos:

Option Explicit

Private Sub Position()

'calcula el número de registro del cargo que es distinto al IdRegistro'

'Es equivalente a decir el IdCargo 30 es el registro número...

datCargos.Caption = Str(datCargos.Recordset.AbsolutePosition + 1) & "/" &
Str(datCargos.Recordset.RecordCount)

End Sub

Private Sub datCargos_Reposition()

Call Position

End Sub

Private Sub Form_Activate()

datCargos.Caption = "Inicio de Sesión"

End Sub

Formulario DBGrid2

Este formulario consta de los mismos controles que el ejemplo anterior. Su funcionalidad sin embargo es distinta ya que es un poco más compleja. Puede apreciarse en el código expuesto seguidamente que tiene dos procedimientos nuevos que consideran eventos sobre los rótulos.

Estos eventos, el clic del mouse y el doble clic permiten ordenar el DBGrid1 y el control Data datCargos en forma ascendente o descendente respectivamente en base a la selección que haga el usuario. Si se hace clic sobre el rótulo del salario mínimo los controles se ordenarán en forma ascendente en base al salario mínimo y si se hace doble clic sobre el mismo rótulo se ordenarán en forma descendente.

Para poder efectuar este ordenamiento tenemos que aplicar SQL sobre el conjunto de datos de origen. El control Data es el vincula los datos de la base de datos al formulario y aquí importa si tener muy en claro que lo que se va a modificar en tiempo de ejecución es la propiedad **RecordSource**, es decir el origen de los datos que muestran los controles.

Esta propiedad en tiempo de diseño especifica que el origen de datos es la tabla Cargos. En tiempo de ejecución esta propiedad tiene como origen de datos una consulta ya que se está seleccionando toda la tabla cargos con un orden en particular:

```
datCargos.RecordSource = "select * from cargos order by " & lblRotulos(Index).Caption  
datCargos.Refresh
```

En la primera línea se establece la consulta de selección para todos los campos de la tabla Cargos ordenados en base al rótulo que presione el usuario para lo cual se tiene en cuenta el 'caption' de cada rótulo. El index sirve para identificar qué rótulo obtuvo el enfoque por parte del usuario y se pasa así ese número (0,1,2 ó 3) al procedimiento clic o al procedimiento doble clic.

La segunda línea sirve para que los cambios sean visibles y se actualice el formulario.

Option Explicit

Private Sub Position()

'calcula el número de registro del cargo que es distinto al IdRegistro'

'Es equivalente a decir el IdCargo 30 es el registro número...

```
    datCargos.Caption = Str(datCargos.Recordset.AbsolutePosition + 1) & "/" &  
    Str(datCargos.Recordset.RecordCount)
```

End Sub

Private Sub datCargos_Reposition()

Call Position

End Sub

Private Sub Form_Activate()

datCargos.Caption = "Inicio de Sesión"

End Sub

Private Sub lblRotulos_Click(Index As Integer)

datCargos.RecordSource = "select * from cargos order by " & lblRotulos(Index).Caption

datCargos.Refresh

End Sub

Private Sub lblRotulos_DblClick(Index As Integer)

datCargos.RecordSource = "select * from cargos order by " & lblRotulos(Index).Caption & " desc"

datCargos.Refresh

End Sub

Formulario DBGrid3

Este formulario permitirá ejecutar sentencias SQL del tipo DML y ver el resultado de las mismas en un DBGrid.

No contiene un analizador sintáctico SQL ni un manejador de errores, motivo por el cual hay que fijarse bien al ingresar las sentencias.

El formulario consta de los siguientes controles:

Tipo de Control	Nombre	Propiedades
Data	datConsu	Connect = Access DataBaseName=bdEmpresas.mdb RecordSetType=Dynaset RecordSource="" (vacío) Visible=false
DBGrid	gridConsu	AllowAddNew=false AllowUpdate=false DataMode=Bound DataSource=datConsu
TextBox	txtSQL	MultiLine=true ScrollBars=Both
Array de botones de comandos 1	cmdEjecuta	CausesValidation=true
Array de botones de comandos 2	cmdSQL	CausesValidation=true

El código es:

Option Explicit

Private Sub cmdEjecuta_Click(Index As Integer)

Dim intMSG

If (txtSQL.Text <> "") Then

Select Case (Index)

Case 0:

If (Left(txtSQL.Text, 6) = "SELECT") Then

datConsu.RecordSource = txtSQL

Else

intMSG = MsgBox("Para consultar debe utilizar la sentencia DML SELECT", vbCritical, "Error")

End If

Case 1:

If (Left(txtSQL.Text, 6) = "INSERT") Then

datConsu.Database.Execute (txtSQL)

Else

intMSG = MsgBox("Para insertar debe utilizar la sentencia DML INSERT", vbCritical, "Error")

End If

Case 2:

If (Left(txtSQL.Text, 6) = "UPDATE") Then

datConsu.Database.Execute (txtSQL)

Else

intMSG = MsgBox("Para actualizar debe utilizar la sentencia DML UPDATE", vbCritical, "Error")

End If

Case 3:

If (Left(txtSQL.Text, 6) = "DELETE") Then

datConsu.Database.Execute (txtSQL)

Else

intMSG = MsgBox("Para eliminar debe utilizar la sentencia DML DELETE", vbCritical, "Error")

End If

End Select

Else: intMSG = MsgBox("No puede ejecutarse una sentencia vacía", vbCritical, "Error")

End If

datConsu.Refresh

End Sub

Private Sub cmdSQL_Click(Index As Integer)

txtSQL.Text = cmdSQL(Index).Caption

txtSQL.Text = txtSQL.Text & " "

txtSQL.SelStart = Len(txtSQL.Text) + 1

txtSQL.SetFocus

End Sub

Como puede apreciarse sólo los arrays de botones de comando tienen procedimientos.

El **array cmdSQL** permite seleccionar una operación DML sobre las tablas de la base de datos dbEmpresa.mdb.

El **array cmdEjecuta** permite ejecutar la sentencia DML. El código varía si se trata de una sentencia DML que arroja un conjunto de resultado como es una consulta de selección (SELECT) a si se trata de una inserción, actualización o eliminación de un registro por cuanto en este caso la operación se realiza con el siguiente código:

```
datConsu.Database.Execute (txtSQL) [Ver breve descripción de los objetos Database y Recordset]
```

Es decir sobre la base de datos a la que hemos vinculado el control Data.

En cambio si se trata de un SELECT el código se aplica sobre el conjunto de registros.

Por ejemplo si pulsamos el botón SELECT se escribirá en el cuadro de texto txtSQL "SELECT " con un espacio. Allí podremos continuar con la instrucción SQL.

Suponiendo que la instrucción que queremos ingresar es: SELECT * FROM EMPLEADOS deberemos entonces presionar el botón cmdEjecuta(0), o sea, 'Consultar' y cuyo código hace lo siguiente:

```
If (Left(txtSQL.Text, 6) = "SELECT") Then
    datConsu.RecordSource = txtSQL
Else
    intMSG = MsgBox("Para consultar debe utilizar la sentencia DML SELECT", vbCritical, "Error")
End If
```

Primero verifica que la palabra ingresada sea SELECT, si es así actualizará el origen de datos con la sentencia. Si no es así se generará un mensaje de error.

Asimismo, cualquiera sea el tipo de instrucción elegida, al salir del Select Case del cmdEjecuta, se actualizarán los datos del DBGrid.

Este ejemplo permite aplicar sentencias SQL y ver su resultado por medio de un DBGrid. Sí cabe aclarar que no posee un analizador sintáctico ni un manejador de errores para evitar salidas intempestivas cuando el usuario ingrese una sentencia SQL no correcta sintácticamente hablando.

Sin embargo se puede realizar un analizador sintáctico de SQL y también se puede crear un manejador de eventos.

Sí conviene además ahondar en algunos conceptos que ya hemos estado viendo al fijar las propiedades de los controles ya que se puede manipular la base de datos mediante código solamente.

Breve descripción de los Objetos Database y Recordset

Un objeto **Database** es una representación lógica de una base de datos física que puede asignarse a una variable de tipo **Database** y ser manejada por ésta.

Una vez que se añade un **control Data** válido a un formulario ya se está trabajando con un objeto **Database**. El control Data tiene una propiedad Database asociada (que no es la misma que la propiedad **DatabaseName** que podemos ver en el cuadro de propiedades y con la que hemos estado trabajando.) y contiene una referencia al objeto Database asociado con el control actual.

Sin embargo pueden declararse también variables del tipo Database en los procedimientos como puede apreciarse en el código es un atributo del objeto Data ya que la línea **datConsu.DataBase.Execute()** lo que hace es ejecutar sentencias SQL sobre la base de datos a la que está vinculado el control Data llamado datConsu.

Un objeto **Recordset** es un conjunto lógico de registros asociados con una base de datos física; representa los registros en una tabla o los que resultan de hacer una consulta. Los objetos Recordset son los más necesarios para manipular los datos de la base de datos.

Cualquier objeto Recordset está formado o construido mediante el uso de filas o registros y columnas o atributos de las tablas de la base de datos a la que hace referencia.

En el cuadro de propiedades de un control Data habrá observado que se presentan tres tipos diferentes de RecordSet:

- **Tabla:** Es una representación de una tabla base que puede usar para añadir, cambiar o borrar registros de una sola tabla sencilla de base de datos.
- **Dynaset:** Es el resultado de una consulta que puede tener registros actualizables. Y la palabra en sí, Dynaset, nos está indicando que se trata de un conjunto de registros dinámico que puede usarse para añadir, cambiar o borrar registros desde una o varias tablas de base de datos subyacentes.
- **SnapShot:** Es una copia estática de un conjunto de registros que puede usar para encontrar datos o generar reportes. Un snapshot al ser una imagen estática obviamente puede servir para mostrar datos pero no puede alterarse.