ODBC1.WRI

Introduction to the new Formula One database functions

Database connectivity is dramatically different in this version of Formula One. The Data Control has been replaced with ODBC connectivity which offers an incredible speed difference along with great flexibility in populating your spreadsheet.

The ODBC1 Demo is provided to allow you to experiment with the three new functions but first you must have the newest ODBC drivers installed. In Visual Basic 4, there is a subdirectory of the main directory named ODBC32. You need to run the SETUP program in this directory to install the ODBC32 admin program. You can get other drivers from MSVC 2.2, Visual FoxPro, Access95 or other vendors. Refer to ODBCREAD.WRI for more information After that is done, step through the description below to learn how to use the new database functions.

Using the new tools consists of three steps:

ODBCConnect to connect to a data source
ODBCQuery to populate a spreadsheet
ODBCDisconnect to disconnect from the data source

ODBCConnect takes three arguments: a connect string, a boolean indicating whether to show the connection dialog, and an error return code. Below is the code that implements the connection in this demo.

```
  On Error GoTo ConnectError

  Dim returnCode%, pConnect$

  F1Book1.ODBCConnect pConnect, True, returnCode

  Exit Sub
ConnectError:
  MsgBox Error
```

The connect string and return code must be a variable. This allows you to let the user select the database at run time and returns the data source in the connect string. The return code indicates any error that occured. When a null string is passed in (in the above case) a dialog is show for the user to select a data source.

Connections are made on a per sheet basis. With Formula One 3.0, you can have many sheets in a workbook and populate each one from a different query or a different database. This is an exceptional capability, since you control alignment of the tables on each sheet. Why? Because Formula One functions will work across multiple sheets (just like Excel 5). As an example, you can load district sales, one district per sheet. On each sheet you can do any summary or calculation you wish to show statistics about that district. If you own First Impression, you can select a range of data and draw a chart on the spreadsheet representing that data. The link will be live so modifications to the data are shown instantly.

So now that you have a connection to a database, you will want to make a query. Below is the code that implements this in the demo.

```
  On Error GoTo FetchError

  Dim returnCode%, query$
  Dim setColNames As Boolean, setColFormats As Boolean
  Dim setColWidths As Boolean, setMaxRC As Boolean

  Let query = cboQueries.TEXT
  setColNames = chkSetColNames.Value
  setColFormats = chkSetColFormats.Value
  setColWidths = chkSetColWidths.Value
  setMaxRC = chkSetMaxRC.Value

  F1Book1.ODBCQuery query, Val(txtStartRow.TEXT), Val(txtStartCol.TEXT), _
     optShowDialog.Value, setColNames, setColFormats, setColWidths, setMaxRC, returnCode

  Exit Sub
FetchError:
  MsgBox Error
```

The ODBCQuery also wants a lot of variables. This is so you can set the query or the user can select it from a dialog at run time. Variables are used so you can tell what the user selected and take the appropriate action - the variables are set to the user's selections after the function returns. The amount of freedom you allow your user is up to you.

For the demo, I have several check boxes to let you easily experiment with the options. The values from these check boxes are placed in variables and then used in the query. If the query string is null or you set the optShowDialog value to true, the query dialog will be shown. This dialog allows the user to select how the sheet will be formatted after the query is run. It also shows information about the existing database connection, tables and fields, and allows the user to type in a SQL query.

After your query you should disconnect from the database. In the demo, this is done with

```
  F1Book1.ODBCDisconnect
```

Creating Data Sources

So what about these data sources. The ODBCConnect dialog allows you to create one on the fly. You will probably want to create them before hand to make life easier for your user. After installing the ODBC pack (as mentioned above), start the ODBC Admin program (odbcad32.exe) and setup a new datasource by clicking the Add... button. For this example we will use an Access database provided with VB4 and the Access 32 bit ODBC driver provided with MSVC 2.2. The beauty of ODBC is that it doesn't care what driver you use (theoretically), just that you have a data source.

In the Add Data Source dialog, select Access and click OK. In the ODBC Microsoft Access 2.0 Setup dialog click the Database Select... button and navigate to your VB4 directory and select BIBLIO.MDB and click OK. Fill in a data source name (I have used F1_VB4_Biblio_Test_Source in this example) and a description if desired. Click OK and F1_VB4_Biblio_Test_Source will appear as an option in the Data Sources dialog. Click Close. You now have a new data source - it will show up as an option when you use ODBCConnect.

How to use the ODBC1 demo

Start ODBC1 demo in VB4 if you want to look at the code, or just double click the exe. The menu holds three items: Connect!, Fetch!, and Disconnect!. When you click on these menu items, they read the options set below and take the appropriate action. The query options frame holds items that affect the query and the way the data is layed out on the sheet when it is fetched.

Set Col Names - Sets the grey column headers to the field names
Set Col Formats - Sets the column number formats for the type of data in that field
Set Col Widths - Adjusts the column width to the widest entry in that field
Set Max Row/Col - Sets the maximum row and column to the size of the table retrieved
Start Row - Determines the first row that data is placed in
Start Col - Determines the first column that data is placed in

Show Dialog option - If true, the query dialog is shown
Use Query option - enables the combo box with some predefined queries. You can also typ in your own.

The Sheet Formatting frame lets you reformat a sheet after a query. The column names, widths, formats, and the maximum row and column are all properties in the Formula One control. These are not changed when you perform a new query - there are some cases where you will not want them to be. You can address these as you define your needs. The three text boxes set the number of rows, columns, and sheets for the workbook. After typing in these values, click on the Set Formats button and they are applied. The Init Sheet initializes the active spreadsheet. This will disconnect from the database and return the spreadsheet to its default condition.

Limits:

To maintain Excel compatibility, each cell is limited to 256 characters. Text is truncated to fit in the cell if it contains more than 256 characters.

Additional Help

If Formula One is not on your tool palette, select the Menu Item Tools - Custom Controls... and scroll down to VCI Formula One Library. If this control does not appear in the list, it has not been properly registered. Click on the Browse... button and navigate to where you installed the Formula One OCX (VCF132.OCX) and select it. Click OK and then OK in the Custom Controls dialog. It is now registered and will appear on your tool palette.

If you want to know more about Formula One functions, press F2 and select the VCI Formula One Library in the Libraries/Projects: listbox. The first entry in the left listbox is F1Book and lists all the methods and properties in the right list box. The rest of the entries are Formula One Constants. If you select a method or property in the right listbox, syntax and a help string will appear at the bottom. You can press the question mark button and the Formula One Beta function reference will appear for that method or property.

This is very handy when you get a type mismatch error executing your code. The chances are that you passed a constant value when Formula One wanted a variable. Unfortunately, VB4 does not shows syntax in Basic form. The Formula One help file will describe the arguments and whether they are passed by reference or value. Note that if you pressed F2 when the Code Window was active, you can paste from the browser into the code window.