# Manipulate Data With VB6's New Grid Control

## Use VB6's new Data Environment and Data Form Wizard with the Hierarchical FlexGrid control to view and change data.

by Stan Schultes and Chris Barlow

**WHAT YOU NEED**
**VB6 Professional or Enterprise Edition**

Before VB6, VB included very simple data controls, offering little capability for building "real" applications. Times have changed, and the provided controls are much more functional straight out of the box, including tight integration with the Visual Studio development environment. Don't believe us? Then watch this! This column focuses on the new VB6 control called MSHFlexGrid. This hierarchical grid control offers developers an easy means of implementing data viewing and manipulation similar to a spreadsheet.

VideoSoft developed the VB6 flex grid controls for Microsoft. The versions included with VB6, without some of the enhanced functionality of commercial versions (such as VideoSoft's VSFlexGrid or Janus Systems' GridEX), offer developers a powerful yet simple way to display and manipulate tabular data. MSFlexGrid, which comes with versions of VB prior to VB6, offers entry-level grid control capability. The new MSHFlexGrid integrates tightly with the VB6 development environment and can display hierarchical recordsets. A hierarchical recordset is an advanced concept that allows a recordset to contain another recordset within a field. Picture this by envisioning a data grid that contains another grid in one of its cells. We won't demonstrate hierarchical data display in this column, but instead will focus on how the MSHFlexGrid works within VB6's new data environment.

### We're Off to See the Wizard

The easiest way to begin with any new or unfamiliar technology is to let someone else do the work! The MSHFlexGrid control provides a wizard to get you started. Code generated with a wizard can be somewhat inflexible, but offers a good jump start to learning about the control. Start VB6 and create a Standard EXE project. Choose the Data Form Wizard from the Add-Ins menu. If the Data Form Wizard is not on the Add-Ins menu, choose Add-In Manager from the Add-Ins menu and select the VB6 Data Form Wizard. Check the Loaded/Unloaded box under Load Behavior. If you always want the Data Form Wizard available from the Add-Ins menu, also check Load on Startup.

Once the Data Form Wizard is running, step through the screens to create a simple data access project. Select None for the profile on the Introduction screen, and Access as the database format on the Database Type screen. On the Database screen, select the Northwind sample database that comes with Visual Basic (mine was found under C:\Program Files\Microsoft Visual Studio\VB98\Nwind.mdb). On the Form screen, choose the MSHFlexGrid option in the Form Layout listbox, and name your form. We called ours frmHFlex. Notice that the wizard either uses an ADO Data control or generates ADO code. Choose the ADO Data control; we will look at ADO code another time (you can use the ADO code option to create some useful ADO starter code to look at).

On the Record Source screen, choose the Orders table in the Record Source combobox, and choose any desired fields in the Selected Fields listbox. On the Select Grid Type screen, choose Outline. This sets the grid options to merge cells, enhancing the grid's readability. On the Set Appearance Style screen, set the grid style to suit your taste. Later, you'll be able to see from the generated code how to set the appearance of the grid. The Set Column Settings screen allows you to set the order of columns by dragging and dropping the column headings. And finally, the Set Application UI Options screen allows runtime drag-and-drop of columns to change your view of the displayed data. Choose this option for greater runtime flexibility. On the last screen, save the wizard settings as a profile if desired. Clicking on the Finish button creates the frmHFlex form, complete with code!

Remove the default Form1 form by right-clicking on it in the Project Explorer window on the right side of your screen and choosing Remove Form1. Then right-click on Project1 in the Project window and choose Project1 Properties. Change the Startup Object to frmHFlex, and set the Project Name to FlexTest. Notice that the wizard adds both the MSHFlexGrid and ADODC controls to the toolbox for you. Now run the project to see what the generated code does for you. You can see from the two left-most columns what the Merge Cells option does.

Admittedly, the generated application is simple, but it comes with a fair amount of code you can use simply to begin learning the control or as a starting point for your own application. When using a new control, the hardest part can be knowing how to get started. The wizard-generated code does this and more; particularly, it offers insight into how to implement column drag-and-drop operations. Be sure to browse through the code (or set a breakpoint in Form_Load and step through the code on startup) to see how it works. You need to set breakpoints manually in the drag-and-drop routines to see how they work. Not a bad start.

Note this caveat: Using the ADO Data control with MSHFlexGrid displays the data as read-only. No updates can be done to the underlying recordset through the grid. This is fine for reporting-type applications, but not for data update applications such as order-entry systems. VideoSoft's (and other companies') commercial controls offer read/write capability with the ADO Data control.

## Building Data Apps is No Longer a Drag

As if the Data Form Wizard weren't enough, MSHFlexGrid has yet more integration with the VB6 development environment. We've written a lot of data access code in VB, which often involved poring over the help files and spending time building SQL queries with Microsoft Access' nice Query By Example (QBE) visual query builder. No more! VB6 now sports a Data View window and Data Environment Designer within the development environment. Just wait until you get a taste of these tools—they'll spoil you beginning VB programmers.

To see how these new data design tools work, add another form to the project, and name it frmData. Choose View | Data View Window, or click on the Data View button on the Standard toolbar. It helps to dock the Data View window at the bottom of your screen by dragging it below the code viewing area. In the Data View window, right-click on the Data Links entry and select Add a Data Link… from the context menu. Choose the Microsoft Jet 3.51 OLE DB Provider on the Provider tab. You can also use the MS OLE DB provider for ODBC drivers, but you must create an ODBC data source name (DSN) through the ODBC Manager in the control panel. On the Connection tab, enter the path to your database. As a nice touch, you can click on the Test Connection button to see if VB can open the database successfully using your chosen provider. Click on the OK button, and name the Data Link Northwind in the Data View window.

Spend some time checking out your new Northwind data link in the Data View window. Click on the plus sign, and Tables and Views appear. Click on the plus sign next to Tables to see a list of the tables in the Northwind database. Double-clicking on one of the table names displays a "Run Table" showing live data. Careful—it's really live data you can change in the database! When doing data access software development, always develop against an offline copy of the database—not against the live mission-critical corporate data.

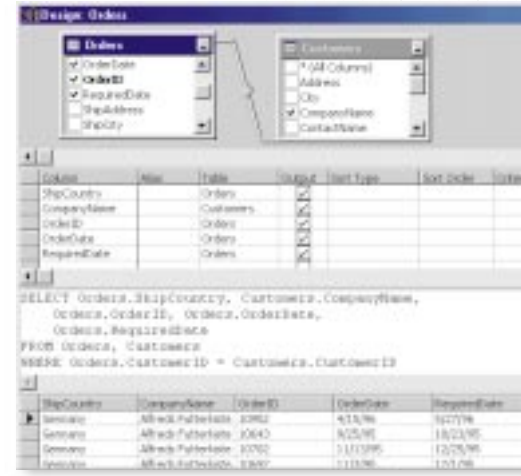

**Figure 1 Data Design the Way it Was Meant To Be.** Everything you need to build a data interface from your VB application—a visual builder, field and attribute selector, SQL view, and sample output when run in VB6's design mode. What more could you ask for?

Now add a Data Environment Designer to your project by right-clicking in the Project Explorer window and selecting Add | More ActiveX Designers | Data Environment. Name the added data environment denvNWind. Find the Orders table in the Data View window, then drag and drop it onto denvNWind. This creates a new ADO Connection object with the Orders table attached. Name the new Connection object connOrders, and delete the default Connection1 object. Experiment with the Data Environment window. See what happens when you double-click on a field name in the Orders table, and look at the right-click Properties dialog for a field.

Once you add the Orders table to the Data Environment, further refine the information to be returned in this view. Right-click on the Orders table and select Properties. On the General tab, click on the SQL Statement radio button, then the SQL Builder button. Lo and behold, a query designer similar to that in Microsoft Access comes up. Although this article uses the Northwind Access database, the designer would look the same if we were using a SQL Server database. Note, however, that not all OLE DB providers let you see the data layout as graphically as this example shows.

To create a specific view of the sample data, drag the Orders and Customers tables from the Data View window onto the SQL Design window. Note that VB knows the relationship between the tables. In the table selections, click on the checkboxes for desired fields, then drag and drop them in the Fields View to set their

**RESOURCES**

See the First Looks section in this issue for information on other grid controls for Visual Basic.

order. You can also see the SQL statement being built as you go. Right-click at the top of the designer, near the tables, and select Run from the context menu to see a view of the data selected by the query (see Figure 1). Close the query design box and save the query when you're satisfied with the results.

## The Grand Finale

In the Data Environment Designer, note that the icon for the Orders table has changed to a query icon. To display the data selection, make sure the frmData form is visible next to the Data Environment Designer. Right-click on the Orders query in the Data Environment Designer and drag it onto the form. From the context menu, select Hierarchical FlexGrid. Right-click on the grid and select Retrieve Structure from the menu. This fills in the column headings with the proper field names. Finally, move the grid up and resize it. Add a command button, make its caption Close, double-click on it, and add Unload Me as the lone line of code in the Command1_Click event. Change the FlexText project property to make frmData the startup form, then save and run your project.

It's amazing what you can do without having to write one line of code. Data access application construction ought to be done this way! However, generated code often creates a fairly vanilla application. To help make the grid easier on the eyes, add some code to format the grid at load time (download Listing 1 from the free, Registered Level of The Development Exchange; see the Code Online box for details). Set the grid to merge cells using the MergeCells and MergeCol properties of MSHFlexGrid:

```
' set grid's column merging and
' sorting
.MergeCells = flexMergeFree
For i = 0 To .Cols - 1
   .MergeCol(i) = True
Next I
```

The constant flexMergeFree allows merging across all rows and columns. Now running the project displays the data with merged columns and basic column formatting, making it more readable.

The Data Form Wizard, Data View window, and Data Environment Designer are very cool new features. Microsoft estimates that 95 percent of all VB programs access data, and these new tools make doing so much more convenient than ever before. Next time you have a fresh application design (you do design your applications before coding, don't you?), pull out VB6's new bound controls and data designer features, and you'll be on your way in no time. VBPJ

### About the Authors

Stan Schultes is the lead developer at SunOpTech, where he is responsible for development and worldwide support of the ObjectBank and ObjectOrder products. Stan has 20 years of experience in the computing field, most of that spent with a Fortune 200 company where he developed manufacturing systems and was a corporate technology consultant. Stan holds a degree in computer engineering from Purdue University. Reach Stan at Stan@VBExpert.com.

Chris Barlow is president and CEO of SunOpTech, a developer of manufacturing decision-support applications, including the ObjectBank and the ObjectJob systems. Chris holds degrees from Harvard Business School and Dartmouth College, where he worked with Drs. Kemeny and Kurtz on the Basic language. Reach Chris at Chris@VBExpert.com.

### CODE ONLINE

You can find all the code published in this issue of **VBPJ** on The Development Exchange (DevX) at http://www.vbpj.com. For details, please see "Get Extra Code in DevX's Premier Club" in Letters to the Editor.

**Manipulate Data With VB6's New Grid Control**
**Locator+ Codes**
Listings for the entire issue, plus the code files to use the MSHFlexGrid control, Data Environment Designer, and code generated by the Data Form Wizard (free Registered Level): VBPJ1198
Listings for this article only, plus the code files described above (subscriber Premier Level): GS1198