FORMAT1.WRI

This example shows how to create some popular table formats in code. More than an example, you can drop the FORMAT1.TXT file into your project and quickly achieve the same formatting. For example:

1. Add FORMAT1.TXT to your project
2. In code, select the area you want to format
3. Call FormatCells with your Formula One object and a constant that describes the type of formatting desired.

About the Example:
The primary goal of this example is to show the code used to generate some attractive formatting. A useful side effect is the creation of a reusable module. VB is wonderful for rapidly creating prototype applications. Reusing simple code like this can add to the sophistication of your prototype with little extra development time. The Cell_Formatting module provides a higher level view to cell formatting. It uses one simple public routine that incorporates a set of assumptions that are general enough to apply to many cases. That routine uses sub-routines that are a higher level view of Formula One formatting so it can easily be extended. When the time comes to turn prototype into reality, you can simply prune away the code that is not used or incorporate the code that is used in you project.

How to use the module:

1. Start the demo program and push buttons until you find a suitable format.
2. Look up the function that does the formatting in the cmdFormat_Click Event Handler.
3. Add this module to your project.
4. Add the function that does the formatting to the applicable part of your code.
5. Modify as necessary to fit your needs.

How it works:
The data is broken into five ranges:

1. The Selection, which is all data to be formatted
2. The hdrRange, which is the top row and contains column header text
3. The ftrRange, which is the bottom row and usually contains a summary formula (like =sum(c2:c7)).
4. The col1Range, the left column that usually contains row header text.
5. The bodyRange, which is the Selection without the other 3 ranges - normally contains the data.

Most tables can be formatted using these ranges. The idea is to select the table area and then call FormatCells with the constant that specifies the format you want. FormatCells first breaks the selection into ranges 2, 3, 4, and 5 above. It then calls simplified versions of the Formula One formatting functions to create the desired format.

To extend this module, you should find a format that is similiar to the one you want, define a constant for it, copy and rename the case in FormatCells that handles it, and modify it as necessary.

To define a new pattern, start the Workbook Designer and select Format Pattern... from the Format menu. Select a pattern, foreground, and background that gives the desired effect in the pattern sample. Now, count by rows from the left to find the pattern and color index that you have selected. The patterns are zero based and the palette is 1 based. Use PaletteEntry(?) to set the color in the SetPattern method. Currently, all the OLE data types (like OLE_COLOR) are undefined in VB 4 so use a long instead.