April, 2003

Advisor Answers

Displaying pictures in a grid

VFP 8/7/6

Q: I'm trying to do something which seems pretty simple: Display an icon (as an image object) inside a grid based on a cursor. I don't need much fancy functionality. The user won't have to change it at all and I'm simply using the small .BMP images from the samples of Visual Studio.

The simplest things (putting the image directly in the grid) don't seem to work at all. I tried making a container which contains an image and putting the container in the grid. That "sort of" worked, but all rows always have the first row's image.

Is there a way to do this?

–Dave Collins (via Advisor.COM)

A: This is one of those things that seems pretty straightforward, but turns out to be a little tricky. The first issue is how you're storing the pictures. The easier case is to store the pictures in a General field in the cursor on which the grid is based.

However, I prefer not to store data in General fields, because there's no easy way to get it back out when you want to. So I'm more likely to store a filename in a field to point to the picture. For this problem, that makes things more complicated.

Let's start with the easy case—having the pictures right in the cursor. In that situation, the solution is to use not the Image control, but the OLEBoundControl. It's designed for displaying data from outside applications, like pictures. Drop an OLEBoundControl onto the appropriate column. You don't need to set any properties of the control itself (though you may want to change its Name). You do need to set some column properties, however. Set the column's ControlSource to the name of the field containing the pictures, and set its Sparse property to .F.

The Sparse setting is a key issue here. VFP's grids are a case of smoke-and-mirrors at work. Although a grid looks like a spreadsheet, in fact, at any given time, only one row is live. To speed up display and to make the grid look less cluttered, by default, only the active row displays the specified control for each column; all the other rows display the default textbox. The Sparse property lets you control this behavior on a column-by-column basis. Setting Sparse to .F. for the column containing the OLEBoundControl ensures that the picture for each record is displayed all the time.

Figure 1 shows a form using this technique, based on the Category table from the TasTrade database. The form is included as GridWithPictures.SCX on this month's Professional Resource CD.

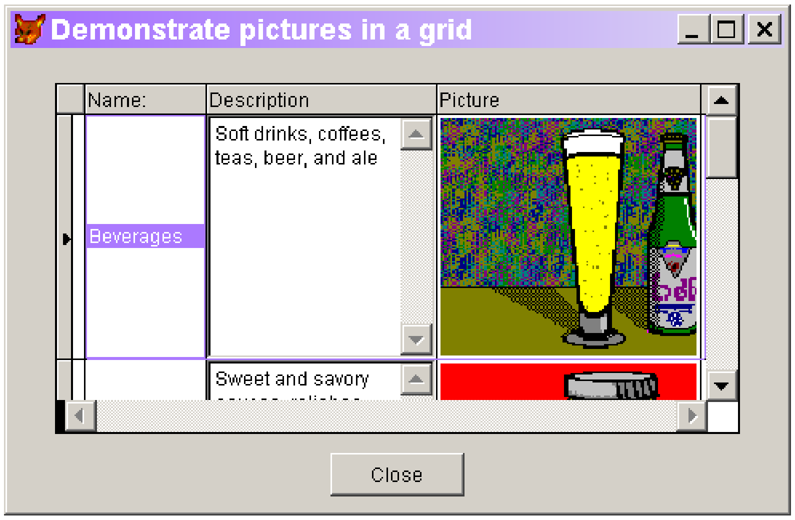
Demonstrate pictures in a grid
Name:
Description
Picture
Soft drinks, coffees, teas, beer, and ale
Beverages
Sweet and savory
Close

Figure 1: Using pictures in a grid—The OLEBoundControl control lets you display pictures stored in a General field.

Let's move on to the case of a table storing the name of a picture rather than the picture itself. Although the Image control lets you specify a field name as the source for its data, the same picture is displayed for each row in the grid, even if you set Sparse to .F. for the column.

In fact, I couldn't find a reasonable way to specify just the file name in a grid and have the right picture displayed. (You can do it by adding a control for each distinct picture and using the DynamicCurrentControl property to determine which is displayed for each record. This approach works well only if you have a limited number of pictures.) Instead, you need to throw the pictures into a cursor and then use the first technique. To associate each picture with the appropriate row of the original cursor, include the primary key for each record in the new cursor, and set a relation from the original cursor to the new cursor. (If the number of different pictures is limited, you could insert each picture only once and use the filename as a primary key.) The

TasTrade category table includes both the name of the picture file and the picture itself, so we can use it to demonstrate this approach, as well. This code shows how you'd create the separate cursor of pictures and relate it to the original, if you needed to.

```
* Open the Category table
USE HOME(2) + "TasTrade\Data\Category"
* Now create a cursor containing the pictures
CREATE CURSOR HoldPictures (cID C(6), gPicture G)
INDEX ON cID TAG cID
SELECT Category
SCAN
   SELECT HoldPictures
   INSERT INTO HoldPictures (cID) ;
      VALUES (Category.Category_ID)
   APPEND GENERAL gPicture ;
      FROM FORCEPATH(Category.Picture_File, ;
      HOME(2)+"TasTrade\Bitmaps")
ENDSCAN
GO TOP
SET RELATION TO Category_ID INTO HoldPictures
```

If your original table doesn't have a unique identifier or you'd rather not index the picture cursor, you can use a technique that goes back to the early days of FoxPro (in fact, FoxBase): relating two tables (or, in this case, a table and cursor) based on the record numbers. In that case, the code looks like this:

```
* Open the Category table
USE HOME(2) + "TasTrade\Data\Category"
* Now create a cursor containing the pictures
CREATE CURSOR HoldPictures (gPicture G)
SELECT Category
SCAN
   SELECT HoldPictures
   APPEND BLANK
   APPEND GENERAL gPicture ;
      FROM FORCEPATH(Category.Picture_File, ;
      HOME(2)+"TasTrade\Bitmaps")
ENDSCAN
GO TOP
SET RELATION TO RECNO() INTO HoldPictures
```

Once you've set up the relation, you can specify HoldPictures.gPicture as the ControlSource for the column containing the OLEBoundControl, and everything works just as if the pictures were in the original table.

This month's PRD includes GridWithPictureFiles.SCX, which demonstrates this technique using the primary key to link the two cursors. The Load method contains both versions for setting up the picture cursor, with the RECNO() version commented out. Just uncomment that code and comment the other to switch it.

–Tamar