Visual FoxPro
The Fox Hunt

By Whil Hentzen

One of the first things most programmers do after learning to create their own custom base classes is to put them on forms that they're creating. This is a good idea - that's the whole point, after all. But once you start actually developing an application, a number of questions will come up out of nowhere. Let's investigate some of the problems and needs you're likely to run into, and provide a couple of tricks and tips along the way.

First, how do you remember to use your base controls instead of Visual FoxPro's base classes? I wish I had a nickel every time someone showed me a form that had some controls that came straight from the base classes, but others that came from the developer's classes - yes, it happens all the time. Someone is very methodical about creating their own base classes, and creates all of their forms using these classes. A few weeks later, some changes come up, and one-two-three, the developer pulls up the form, grabs some new controls from that handy Form Controls toolbar, throws some code in, modifies some properties, and the next day realizes that he didn't use their own classes. This is one unhappy programmer.

By the way, if you're thinking you did this, but you don't remember how to determine "from whence" a control came from, look at the BaseClass, Class, and ClassLibrary properties of that control. (They're all conveniently grouped under the 'Other' tab in the Properties window.) And remember that you can select an object in a form, and the PEMs (properties, events, and methods) in the Property window will change to those of that object. The BaseClass tells you which VFP base class the object came from while the Class tells you from which class the object was instantiated. ClassLibrary, of course, contains the name of the VCX that holds the class.

As we discussed last month, you can temporarily add your own class library to the View Classes icon on the Form Controls toolbar with it's Add option, but that becomes a drag after you've done it two or three dozen times. Wouldn't it be nice if you could "permanently" add your own class libraries to the View Classes menu? Of course, you can!

Select the Tools, Options menu bar to bring forward the Options dialog. Select the Controls page frame, and use the Add command button to select the class library (or libraries) that you want to add. From now on, those class libraries will be on the View Classes menu.

However, you still have to remember to select your own class library so it replaces Visual FoxPro's base classes on the Form Controls toolbar. This may still be a problem, so here's an alternative method.

Open the Project Manager and select the Classes tab. Click on the Plus sign in front of the class library that you want to work with, so that the classes in that library are displayed. Then, when you need a new object, drag the appropriate class from the Project manager to your form.

I personally prefer this method over using the toolbar for several reasons. First, I never seem to have the right toolbar up, so I have to monkey around with them, moving them, shifting them around, opening new toolbars, rearranging them, and so on. But I always have the Project Manager open. Second, if you followed my instructions in previous articles, you'll see the description of that class in the bottom of the Project Manager window. (If you're a little weak on remembering where your classes are, you can also click on the class library itself and see the entire path of the VCX in the bottom of the Project Manager as well.) Finally, the drag and drop mechanism seems more natural to me - I keep trying to "drag" a toolbar icon onto a form.

The second problem you're liable to run into is having to create a new form and change a bunch of it's properties to suit your particular preferences. The good news is that you are not limited to creating classes from control objects. You can also subclass Visual FoxPro's base form, and derive a number of classes from it! Here's how.

Create a blank form, and use the Save As Class option, or use the Create Class command and select a Form as the control that the class is to be based on. In either case, you'll need to specify a class library name; as mentioned last month, I use a library called BASEFORM.VCX to hold all of my forms. You can change this form to suit your particular needs - make the background bright red, change the default title, even put some standard controls on it that you use on every form.

In order to use this new form class as the source for all of your forms from now on, select the Options dialog from the Tools menu pad, and change to the Forms page. Select the class you created as the Template Class Form (the text box will show you the name of your class and the class library it's located in.), and then Apply your changes. Every time you issue a Create Form command (or any of the other mechanisms used to create a new form), this class will be used as the source for the new form.

A third problem we’ve mentioned is that of dealing with a form that contains controls that come from Visual FoxPro’s base class as well as your own base classes. If you’ve scratched your head over trying to figure out how to change the class of an object based on VFP base classes, you’re not alone. Next month, we’ll show you how, and introduce you to a new tool in Visual FoxPro - builders.