May, 2002

Advisor Answers

Changing a form's class

VFP 7.0/6.0/5.0/3.0

Q: Based on a tip I read in FoxPro Advisor, I want to add some code to the Init method of each form in my application. I created a Form class with this code, which was easy. Then, I tried to change the class library of each form in my application to point to this new class to become based on it, but this wasn't allowed.

To update all the forms to base them on my template class, it seems like I would have to recreate every form. This can't be right. Is there a way I can automate the introduction of form classes into older projects?

–Frank Fallon (via Advisor.COM)

A: Most of us have been in the situation of having a form or control that's based on the wrong class, whether that's a VFP base class or another class that's just not the right one in the current situation. Fortunately, this is a case where FoxPro's open architecture makes a solution possible, even easy.

The key issue is that VFP forms and classes (as well as reports, labels, and projects) are stored in ordinary tables that use special extensions. A VFP form consists of an .SCX/.SCT pair, where the .SCX is really a .DBF and the .SCT is really an .FPT. A class library has a .VCX/.VCT pair, with the same division: the .VCX is a .DBF and the .VCT is an .FPT.

We can open a form or class as a table and modify the data stored there. For your problem, there are three relevant fields: Class, ClassLoc and BaseClass. There's one record in a form that describes the form itself. That's the one you want to change and it has "form" in the BaseClass field. The Class field contains the name of the class and the ClassLoc field lists the class library containing that class. So, to change the class a form is based on, you'd use code something like this:

```
USE MyForm.SCX
REPLACE Class WITH "myformclass", ;
        ClassLoc WITH "myclasslib.vcx" ;
```

```
   FOR UPPER(BaseClass) = "FORM"
```

ClassLoc should provide a relative path to the class library, if it's on the same drive. The example assumes that the class library is in the same directory as the form. Suppose you have a directory for the project with a Forms subdirectory and a Libs subdirectory. In that case, you'd put something like "..\libs\myclasslib.vcx" in the ClassLoc field. (Either way, use all lowercase; VFP doesn't like upper case there.)

So far, so good. But this still leaves you with the problem of changing every single form. You don't have to recreate them, but you do have to open each one and make this change. However, you don't have to do it manually.

The best solution depends on what version of VFP you're using. In all versions, you can open the project file (.PJX) as a table, and find the forms. Look for records with the Type field equal to "K". The Name field provides the file name. Once you have the Name, you can open that table and make the change.

However, in VFP 6 and later, there's a better answer. In those versions, open projects can be addressed programmatically. There's an object model that makes it easy to address and work with individual files.

The Project object has a Files collection, with one entry (reference to a File object) for each file in the project. The File object has Name and Type properties that let us find and address each form.

This code goes through the active project and changes the class of each form. It assumes that the name and class library (including path) of the new class are stored in variables cClass and cClassLib. The only tricky issue here is ensuring that the class library has a relative path, but the SYS(2014) function makes short work of that problem.

```
oProject = _VFP.ActiveProject
FOR EACH oFile IN oProject.Files
   WITH oFile
      IF .Type = "K"  && It's a form
         USE (.Name)
         cAdjustedClassLib = SYS(2014, cClassLib, ;
                                 ADDBS(JUSTPATH(.Name)))
         REPLACE Class WITH cClass, ;
                 ClassLoc WITH cAdjustedClassLib ;
            FOR UPPER(BaseClass) = "FORM"
         USE
      ENDIF
```

```
   ENDWITH
ENDFOR
```

Now you can automate changing every form in a project to use a particular form class. Of course, in reality, you might not want to have every form use the same class. There are various ways you can solve that problem. Perhaps you only want to change those forms that are based on VFP's Form baseclass. In that case, modify the REPLACE command to affect only forms whose ClassLoc field is empty:

```
REPLACE Class WITH cClass, ;
        ClassLoc WITH cAdjustedClassLib ;
   FOR UPPER(BaseClass) = "FORM" AND ;
       EMPTY(ClassLoc)
```

Or maybe you need to change from one form class to another. Again, modify the FOR clause of the REPLACE to affect only the right forms.

You can use the same strategy to change the class of controls, as well. Just be careful that you only change the records you mean to change.

As always, when you're hacking like this, make a copy of your project and form files before you start.

–Tamar