December, 1997

Learning Visual FoxPro—The Top Ten List

So much is different from FoxPro 2.x that it's easy to feel overwhelmed.

By Tamar E. Granor, Editor, Andy Neil, Contributing Editor, Ted Roche, Contributing Editor, and Malcolm C. Rubel, Contributing Editor

You can look at the task of switching to Visual FoxPro from FoxPro 2.x as either an exciting path to the 21st Century or as a trip to the dentist. There's so much that's new and different it can seem overwhelming to even the most accomplished programmers. While making the change, we felt lost at sea on more than one occasion. Isn't a nice feeling, yet it's one that's understandable. "In FoxPro 2.6, I'm an expert, but in VFP I'm an idiot" is not a fun thing to contemplate.

We've all traveled down this road already, collecting a fair number of scars along the way. Based on our experiences, here's a list of the top ten things to learn as you transform yourself from a procedural Xbase programmer of the 90s to an object-oriented, buzzword-savvy, VFP programmer for the new millennium.

1. Variable scope

Realize that OOP requires encapsulation and public variables are not only no longer needed, but no longer valid. Global variables should be scoped to the form as properties of that form. Things that are specific to a control are properties of that control. Your application should have only one public variable-the application object. All other variables are either private or, better yet, local. (In fact, the application object can be private, too, if you prefer.) Local variables (and LPARAMETERS) are only visible to the procedure that created them. Remember that locals (and even privates) go out of scope when the method that declared them finishes. Private variables declared in the form's Init method are not available once the form has been instantiated. Once the Init method is finished running, the variables go away.

2. Buffering and private data sessions

Learn about buffering and private data sessions and forget about all the code you used to handle dirty GETs, sets of memory variables, and/or arrays to hold values while editing, and the programs you used to determine changes and write them to the table.

GETNEXTMODIFIED(), CURVAL(), OLDVAL(), TABLEUPDATE() and TABLEREVERT() handle this for you. Realize that you can have as many buffered views of the same set of data as you want. This opens up a whole new world for you to think about and program for, but the power of private data sessions is undeniable.

3. READ EVENTS

Happily, you can forget about the Foundation READ. All the code you used to control a program (which ended up being a major portion of the effort in getting things to work) is no longer necessary. READ EVENTS replaces it all. READ EVENTS is easier to understand and use. In fact, the change from one active form (screen) to another is handled easily with no external code. Yes, there are methods within the form to deal with housekeeping as you leave one form and enter another, but it takes only a short time to get familiar with the basics.

4. Database Container

VFP's database binds data into a container that includes persistent relationships, field, row, and table rules, and triggers that fire automatically when you either insert, update, or delete records. Code to handle referential integrity can be generated and then kicks in automatically. The Database Container also offers long names for tables and fields.

Along with the .DBC concept of a database, you should also start thinking of data as views, or result sets from a SQL select. With VFP, views are updateable as well. The change from looking at related tables and dealing with all the complexities this entails, to looking at a flat view, is actually a simplification of how you approach data, even though it does require a change of thinking.

The change of thinking actually involves the separation of data from interface. With FoxPro 2.x, we always thought of data as being an integral part of the screen. With VFP, this view of the world is no longer the best way to look at things.

5. Builders and wizards

Even though "real programmers don't use code generators," you should look at the builders and wizards that come with VFP. Although you don't have to start writing your own builders right away, you can actually do this without too many problems.

As all custom properties and methods of all controls are stored in special DBF files, it's now easy to write a builder (a program is all that this is) to add or modify existing properties and settings of all controls. Several are available from the VFOX forum on CompuServe.

6. Object-oriented programming and design

OOP isn't the same as structured programming. We are used to thinking of programs that run in a linear fashion with a beginning point, stuff in the middle that follows logically from point A to B to C, and so forth—even with conditional branching in the code and then ending at a logical end point. You can't approach object-oriented programming in the same manner. It just isn't structured in the same fashion.

You have to look at method code (like 2.x snippets) as stand-alone pieces of code that, when totaled up, make up the personality and behavior of the object that contains them. Looking at sections of code as "personality traits" is not something we're used to doing. It involves accepting that code either is or is not going to run based on user input and that its personality (either programmed or inherited) is what we should be concerned with, not so much where the code gets called from and what gets called next.

To really understand how code fires in an object-oriented program, you need to understand the personality of objects and how and when various methods fire, and in what order they fire. To help you in this, you need to learn more about the new FoxPro debugger.

7. The new FoxPro debugger

With VFP 5.0, Microsoft has finally done something to help us look at our programs as they run. The old (from FoxPro 1.0) debugger has been completely overhauled and expanded. Spend some time learning how to use it, especially the new Coverage and Event Tracking functions. Event Tracking will help you become familiar with what event methods fire under different circumstances.

8. Classes

Start thinking about modifying FoxPro's behavior by building a class library. Create your own set of base classes so you can begin to modify VFP's behavior and make it more like you need. Don't go any further than this right now. You don't have to. But you'll be glad you did because later, when you begin to realize you need VFP to do more than it was designed to do originally, you will already have the entry point (your base classes) to make the necessary modifications.

9. Grids

Grids are much more capable successors to the cranky BROWSE. Grids make one-to-many or master-detail forms a breeze. Columns, headers, and controls can be customized in much more detail than a BROWSE, with fonts, colors, and so forth. In addition to text, grid cells can display checkboxes, pictures, ActiveX controls or, in fact, any control at all.

10. ActiveX controls

FoxPro is now part of Visual Studio and makes use of ActiveX controls. Learn the basic behavior of the ones shipped with VFP. You're going to be using components, so you might as well get used to them.

But wait, there's more

Here are three more bonus ideas that didn't quite make it to the top 10, partly because they're more focused on you than on your code.

11. Tools and samples

Use the tools provided with the product, like the Class Browser and GENDBC, as well as the samples (especially the Solutions set), to make you more productive and to learn how things are done. The Solutions samples can answer almost all of the most frequently asked questions. GENDBC's output shows how to create databases and how to find out about existing databases.

12. KISS: Keep It Simple, Sam (or Sally, or even Stupid)

Start with one-table editing forms, and gradually add features like parent-child forms, grids, and so forth. VFP has many gnarly gotchas on complex forms. Start with simple modal forms, and gradually add complexity to the system. Start with a Start.PRG that issues a READ EVENTS and a menu option that issues CLEAR EVENTS. Check out the Application Wizard for a simple foundation to start with.

13. Don't go it alone

Hang out on CompuServe (GO VFOX), and check the libraries. Ditto for newsgroups and web sites.

14. It takes time

Finally, remember that you didn't learn to program overnight. Changing from procedural to object-oriented programming won't happen instantly either. We hope our list (see sidebar on page 32) helps you get productive in VFP quickly, even if you look back on your early code a year from now and blush.