MacTech Review (formerly "MacTutor")

During the summer of 1992 we submitted ViewIt for review to MacTech™, the major programming journal for the Macintosh. MacTech asked Philip Borenstein, an experienced programmer and author of Symantec's THINK C and THINK Pascal manuals, to review ViewIt. The review appeared in the Nov/Dec 1992 issue of MacTech. Its complete text appears below, along with comments in brackets that help clarify some of the points made by the reviewer.

FaceWare's ViewIt by Philip Borenstein

What Is ViewIt?

The promotional literature that FaceWare puts out describes ViewIt as a window designer and manager. That description is accurate, but it makes ViewIt sound like yet another interface design kit. FaceWare also describes ViewIt as extensions to the Macintosh Dialog Manager and Control Manager. And that's true, too. What these descriptions don't convey is how different ViewIt is from other interface designers and toolbox extenders.

Like other interface-building tools, ViewIt lets you create windows and lay out controls in those windows. But instead of generating source code that you customize, ViewIt uses code resources that already know how to manage those windows and controls. This approach has some clear advantages and some drawbacks. Depending on the kinds of applications you write, the advantages may outweigh the drawbacks.

The main advantage is that the code that you write is remarkably simple. Most of what you do is set up a window, call the main event loop routine, and check for hits in your controls. In general, you don't need to concern yourself with the Macintosh Toolbox. The main disadvantage is that most of the behavior is locked away in code resources, so your application may be carrying around extra baggage that you don't really need. [You can eliminate code and other resources from ViewIt and other FaceWare modules that are not needed in finished applications. ViewIt also reduces the size of program code to such a degree that this quickly offsets what it adds in size to finished programs.]

What's In The Package?

ViewIt works with most popular Macintosh development environments. The commercial version comes with demo programs for THINK C, THINK Pascal (both of which can be used with MPW C and MPW Pascal), Absoft Fortran, Language Systems Fortran, and MacFortran. The THINK C and THINK Pascal programs come with ready-to-use projects. Two separate utilities let you use FaceWare modules with HyperCard and Prograph. [The HyperFace product is required to support calling FaceWare modules from HyperCard's HyperTalk. All FaceWare modules, however, can be called directly from C or Pascal-based external code resources that are often supported by high-level environments like HyperCard or Prograph.]

The ViewIt package consists of three parts: FaceIt, ViewIt, and UtilIt. ViewIt is the part of the system that lets you lay out controls in windows. FaceIt is the part of the system that deals with the main event loop, the menu bar, and other program-wide features. UtilIt consists of utility routines used by FaceIt, ViewIt, and by your own program.

You can use the ViewIt module without the FaceIt module to design modal dialog boxes [or any other type of modal window]. With FaceIt you get automatic support for modeless windows. The documentation does tell you how you can use ViewIt modeless windows in programs that have an existing event loop, but if you're starting from scratch, you may not want to go through the trouble.

All the modules that make up ViewIt live in a resource file called FaceWare. You can place this file in your System folder, so all your ViewIt-based applications can share it, or you can copy resources from the FaceWare file into your application to create stand-alone applications.

ViewIt costs $95. FaceWare distributes a shareware version that includes nearly everything that comes in the commercial package. You can use the shareware version for 30 days. After that, you either get rid of it, or you buy the commercial package.

How Does It Work?

Instead of customizing source code, you make calls to a dispatching routine called FaceIt() which loads the appropriate modules for program- wide behavior like cutting, pasting, and printing and for control-specific behavior. In addition [to] the standard Macintosh control manager, which uses CDEFs and CNTLs, ViewIt uses its own control drivers (stored in FCMD resources) and control descriptions (stored in FCTL resources). These FCMDs are the code resources that do all the work. [The custom control drivers and records overcome the limitations of the control manager without sacrificing backward compatibility with existing CDEF and CNTL resources.]

To create windows and lay out controls, you don't use a separate application. You don't even need to use ResEdit. Instead, you lay out the controls while your application is running. Once you've created a window, you press the

Command-Shift-Option keys to enter ViewIt's editing mode. In this mode, you can add or delete new controls and edit existing controls. To see how they'll work, you just press the Enter key and you're back in run mode again.

Since the edit mode is part of ViewIt (it's stored in the FaceWare file), you can fine tune your user interface without using your development environment. Suppose you're doing a demo for your users, and they say, "This button should be red. That menu should be over there." All you have to do is go into edit mode, make the changes, and try them out. ViewIt comes with a smaller FaceWare file without the edit mode that you can use when your application is finished.

The FaceWare file with the on-line editing support takes up about 1 MB, and the file without the on-line editing support is about 440K. [The FaceWare files being referred to by the author contain modules other than ViewIt and FaceIt, so these sizes are misleading. The use of both ViewIt and FaceIt adds about 100K to the size of finished programs, not the 440K-1MB implied by the author's statement!]

What Kinds Of Interface Elements Does ViewIt Handle?

In addition to the standard controls that the Macintosh Dialog Manager provides (buttons, check boxes, radio buttons, static text, and editable text) ViewIt also gives you pop-up menus, picture-based palettes, graphic buttons based on PICTs, ICONs, and SICNs, dials, and more esoteric controls like a help text viewer and a scrap viewer. [ViewIt's expanded set of control messages allows ViewIt controls to be much more sophisticated than standard Macintosh controls. The EditControls product, for example, includes complete text, array, and graphic editors in the form of controls that can be added to any ViewIt window. The QuickControl product provides complete support for editing and playing QuickTime movies, and CommControl provides complete support for the Macintosh Communications Toolbox.]

You can specify the color and framing style of any control. For text-based controls, you can set the font, size, style, and text color. You can have buttons draw their text in bold, outline Palatino, or you can have right-justified check boxes with the check box on the right side of the text. Most controls have variations. Pop-up menus, for example, come in several varieties. Some menus behave like menus in the main menu bar. Some menus check only one item at a time while others have several items checked.

Every control belongs to a view, which is kind of a meta-control, and a window can have several views. Views can be longer than the window they belong to, and it's easy to add both horizontal and vertical scroll bars to them. Views let you group controls, so you can move them around as a unit. Since you can show and hide views, you can use them for dialogs that have multiple pages.

ViewIt makes a distinction between a control's appearance and its behavior. This distinction is useful for picture-based controls [and other types of controls]. For instance, you can have an icon behave like a button, so it highlights while the mouse button is pressed on it. Or you can have several icons behave like radio buttons where only one icon of a group is highlighted at a time. Of course, some combinations, like editable buttons, don't work because the CDEF or FCMD doesn't support a particular behavior, and other combinations, like check boxes that behave like radio buttons, will get you into trouble with the user interface police.

If you want to make a control behave a little (or a lot) differently, ViewIt gives you hooks that let you intercept virtually everything a control does. [A single override procedure can be set up to intercept all messages sent to a control. Source code for all of the controls shipped with ViewIt is also available as part of the Inside FaceWare product.]

In traditional dialog box programming, when you have a list of radio buttons, check boxes, or other controls, you need to set the controls' values from one of your data structures. When the user dismisses the dialog, you need to get the value of each control and set your data structure accordingly. ViewIt has a nice feature called Data Linking that does this for you automatically. When you create a window, you pass ViewIt the address of a record. When you design your controls, you give the offset into the record where the value is stored. ViewIt takes care of doing all the numeric to string conversions - even for real numbers. [Isolated variables can also be linked. Use of a record is not necessary.]

Of course, you don't have to use Data Linking. You can have controls report when they've been hit. ViewIt uses a mechanism similar to menu events to tell you which control has been hit. ViewIt variables tell you which control, in which view, and in which window received the click. The ViewIt routines let you get the state of the control so you can tell what's in an editable field, whether a check box is on or off, and so forth.

Documentation

All of ViewIt documentation is on-line, and it's always available when you're in edit mode. [The main ViewIt Help window can be kept open as a modeless window. You don't need to be in edit mode to access it.] When you're editing a specific control, you can click on a button to get its documentation. The documentation is right where you need it when you're working, and it doesn't clutter your desk. A printed manual, though, would have made learning ViewIt a bit easier.

With printed documentation you can flip through the pages to get a sense of what a piece of software is about, and even the manual's physical size gives you a hint about the complexity of the software. With on-line manuals a list of topics in a menu may refer to half a kilobyte of text or to half a megabyte text. There's no way to know. [There are many advantages to on-line documentation, and you can easily print all of the on-line documentation with the "Print

All" menu command.]

The demo programs that come with ViewIt are complete in the sense that they show off virtually every feature, but a little hand-holding would have been nice. You can copy controls from the ViewIt demos [or from any of ViewIt's built-in editing dialogs] and paste them into your own application, or you can try modifying them to see how different settings work.

Who Is ViewIt For?

ViewIt is ideal for in-house programmers and consultants who need to write Macintosh applications quickly. Often, these programmers need to respond quickly to requests for new features, and ViewIt's edit mode makes it easy for the programmer and the user to collaborate on the user interface. In-house programmers can take advantage of the fact that many ViewIt-based programs can share the same FaceWare file. This arrangement keeps the application size small.

ViewIt is also useful for programmers who have existing programs that they need to port to the Macintosh. ViewIt is much easier to learn to use than the Macintosh Toolbox. It's certainly easier to learn ViewIt than it is to learn object-oriented programming, and for vanilla Macintosh programs, ViewIt handles most of the Macintosh housekeeping tasks.

Commercial (and shareware) software vendors may find that ViewIt's overhead makes ViewIt-based applications larger than hand-rolled applications, particularly for small applications. It is possible to move only the resources your application uses from the FaceWare file. ViewIt comes with a utility program to do just that. [Also note that no license fees are required to distribute programs that use FaceWare modules.]

Summary

ViewIt is a surprisingly powerful interface-builder. Its approach to the problem is considerably different from the other tools on the market, so it may be difficult to get into at first. For non-Macintosh programmers who need to put together a Macintosh application in a hurry, ViewIt takes care of most of the standard housekeeping chores. ViewIt doesn't require you to learn object-oriented programming or a class library [but it does not prevent using these in combination with ViewIt].

The only way you'll be able to decide whether ViewIt will work for your application is to try it. You can use the shareware version of ViewIt for 30 days, and if you like it, you're obligated to buy the full version. If not, you can wipe it off your disk, and program like you programmed before.

Xplain Corporation
1617 Pontius Ave.
2nd Floor
Los Angeles, CA 90025
310-575-4343