Off The Shelf: Resources For The Programmer's Library

By Jim Douglas

Editors note: This article launches a new feature for The BASIS Advantage*, the Advantage Resource Review. In the future, this column will cover books, websites, conferences, or classes that can provide helpful information and advice on a variety of programming topics.*

Customers want programming in the Windows environment. The problem is where to start. The Visual PRO/5™ manuals describe the SYSGUI device and the syntax for working with various GUI controls-BUTTON, CHECKBOX, RADIOBUTTON, LISTBOX, etc. That's all important stuff, but it doesn't help with more basic questions: How do I construct an effective GUI screen? How do I decide what kind of control to use in a given situation? All three of the books discussed in this article focus on how to design GUI applications with an emphasis on Windows 95.

Essential Guide is Encyclopedic

The Essential Guide to User Interface Design: An introduction to GUI design principles and techniques, by Wilbert O. Galitz, discusses how to design effective GUI screens and gives a lot of practical examples, not dry theory. Chapter 1, The Importance of Good Screen Design, highlights a few key points to consider. User-interface code makes up more than 50% of a typical modern application, and various studies have shown that well-designed user interfaces are more usable. In one study, redesigned screens enabled the users to work 25% faster with 25% fewer errors.

Paying attention to a few simple principles can make screens easier and more natural to use. For example, understanding how users read the screen-upper left to lower right, larger elements before smaller ones, color before black and white, and graphical elements before text-helps you to design screens that flow more naturally. Following these principles, the most important elements should be in the upper left, with [Update] and [Cancel] push-buttons towards the lower right.

The book is organized around twelve steps in the GUI screen design process. Step 7 includes a description

The GUI Screen Design Process

of all the controls available in Visual PRO/5™ 2.0, with the exception of INPUTE and INPUTN. Galitz goes into a lot of detail for each type of control: there are fourteen pages describing how to use push-buttons, ten pages on check boxes, and more than sixteen pages on various kinds of list boxes. Each section includes a description of the control, its purpose (why it is commonly used), advantages, disadvantages, and usage rules. A summary block is followed by extensive examples, both good and bad, showing how the control can be used, or misused, in the context of a complete GUI screen.

1. Know Your User or Client
2. Understand the Business Function
3. Understand the Principles of Good Screen Design
4. Select the Proper Kinds of Windows
5. Develop System Menus
6. Select the Proper Device-Based Controls
7. Select the Proper Screen-Based Controls
8. Organize and Lay Out Windows
9. Choose the Proper Colors
10. Create Meaningful Icons
11. Provide Effective Messages, Feedback, Guidance, and Language Translation
12. Test, Test, and Retest

Step 8 is particularly useful; it shows several GUI screens (some taken directly from Windows 3.1) and applies the principles described throughout the book to show how the screens could be improved, making them more readable and intuitive for the user.

The Essential Guide to User Interface Design, Wilbert O. Galitz, Wiley, 1997, ISBN 0-471-15755-4, 626 pages, US $49.95.

Windows Interface Guidelines Lays Down Microsoft Rules

When designing user interfaces for Windows 95, there's one rule that overrides all of the others: when in doubt, follow Microsoft. *The Windows Interface Guidelines for Software Design* lays out both the basic principles and the detailed rules that Microsoft has established for designing and implementing graphical user interfaces for Windows 95 and Windows NT. The basic principles are presented in the first two chapters.

Chapter 1, User-Centered Design, presents several principles:

- Remember that the user is always in control, not the application

- Be consistent with existing Windows standards, as well as within the application itself

- Be forgiving; allow the user to explore the application without being afraid that he'll break it

- Provide feedback to keep the user comfortable

- Keep aesthetic issues in mind. Chapter 13, Visual Design, explores this in detail
- Keep it simple

Chapter 2, Data-Centered Design, simply means that programmers should design applications with the so-called "Object-Action" model in mind. This means that the user first selects an object (typically a document), then selects an application to manipulate that document (or an application is chosen automatically based on the file extension). This differs from the more traditional "Action-Object" model, in which the user first launches an application, then selects the data to be acted on by that application.

From the perspective of a Visual PRO/5 programmer, the most valuable sections of this book are Chapters 6, 7, and 8. Chapter 6 talks about main (application-level) windows, describing all of their elements, including the title bar and the minimize, maximize, and close buttons. It also describes the overall behavior of application-level windows. Chapter 7 discusses the Windows 95 controls, including all controls available in Visual PRO/5 2.0, with the exception of INPUTE and INPUTN. Chapter 8 details the use of child windows, dialogs, and message boxes-the 'ASK' mnemonic or the MSGBOX() function in Visual PRO/5 2.0.

The appendices contain a wealth of useful reference information: summaries of the mouse interface, the keyboard interface, and overall design guidelines; a detailed list of Windows-related words and phrases translated from English to twenty-six other languages; and a twelve-page glossary. Not everything in this book will be of use to a Visual PRO/5 programmer (for example, chapters on OLE integration and manipulating the registry), but there's enough useful information here to make the book a worthwhile addition to the bookshelf.

The Windows Interface Guidelines for Software Design, Microsoft Press, 1995, ISBN 1-55615-679-9, 445 pages, US $29.95.

Opinionated "About Face" Stresses User

Alan Cooper, the author of *About Face: The Essentials of User Interface Design*, does not think much of the software being written today:

> "If you examine most commercially available software today, you will find user interfaces that are particularly adept at several things:
>
> - Making the user look stupid
> - Causing the user to make big mistakes

- Slowing the user so he doesn't get an adequate amount of work done

- Preventing fun and boring the user

> "Most of that same software is equally bad at achieving its business purposes."

The problem, he says, is that programmers tend to focus on technology rather than the user. Programmers too often build the user interface as a thin veneer barely hiding the implementation details. Users, on the other hand, maintain a conceptual model of how a given tool works; this conceptual model doesn't always correspond with the internal implementation. We need to move the user interface closer to the user's conceptual model and away from the internal details, which are irrelevant to the user.

Programmers should tend towards using modeless feedback, Cooper says, which doesn't interrupt the user's work. Use modal feedback only in the event of a catastrophic error. For example, the Microsoft Word Spell Checker searches the current document in background and quietly underlines what it considers to be misspelled words with a distinctive ragged red underline, but it doesn't interrupt a user's thought process to point out a mistake. This is good. Internet Explorer, on the other hand, is not so considerate. When checking email, if it fails to connect to the mail server for whatever reason, it interrupts the current work with a modal dialog to inform the user of the fact. This is bad. In general, programmers should aim for as little user interface as possible.

Cooper is opinionated and readers may not agree with everything he has to say. For example, he makes the reasonable point that programmers confuse the concepts of *possible* and *probable*, asking the user to confirm actions even when the answers will be the same 999 times out of 1000. He gives something of a questionable example, though: When printing, it's *possible* that a user might want two copies, or that he or she might want to print in landscape mode, but it's *probable* that he or she will want a single copy in portrait mode. So when the user selects print, Cooper says, just do the most likely action. If the user wanted something else, he or she should have used a separate "Print Configuration" dialog. In this case, there's nothing wrong with a typical File-Print dialog that defaults to a single copy, but waits for a user to click "OK" to confirm this assumption and start printing. Cooper does, however, go on to make a very good point: wherever possible, programmers should build long-term memory into programs that learn how the user works and remember user preferences rather than constantly querying the user.

Early in the book, Cooper says that "the task of user interface design is fundamentally different from software engineering…. Programmers, deep in their thoughts of algorithms and coding arcana, design user interfaces the way miners design the landscape with their cavernous pits and enormous tailing piles. The software design process alternates between the accidental and the non-existent." Perhaps in the future, there will be a profession of "Software Designer," distinct from "Software Engineer." In the meantime, programmers and analysts who must take on the task of designing software for end users will find this book both useful and thought-provoking.

About Face: The Essentials of User Interface Design, Alan Cooper, IDG Books, 1995, ISBN 1-56884-322-4, 580 pages, US $29.99.

Jim Douglas, a thirteen-year Business Basic veteran, is currently working with BASIS on Volcano™.